软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, 2022,33(11):3917−3929 [doi: 10.13328/j.cnki.jos.006328]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

求解子集和问题的采样格归约算法*

曹金政[1,2], 程庆丰[1,2], 史闻博[3], 鲁 宁[4,5]

[1](战略支援部队信息工程大学, 河南 郑州 450001)

[2](数学工程与先进计算国家重点实验室, 河南 郑州 450001)

[3](东北大学 秦皇岛分校 计算机与通信工程学院, 河北 秦皇岛 066004)

[4](东北大学 计算机科学与工程学院, 辽宁 沈阳 110169)

[5](西安电子科技大学 计算机科学与技术学院, 陕西 西安 710126)

通信作者: 程庆丰, E-mail: qingfengc2008@sina.com

摘 要: 子集和问题是计算机科学中的重要问题, 也是构建多种公钥密码体制的基础. 提出了采样归约算法, 使用随机采样方法降低问题维度, 将原问题分解并归约为多个更小规模的格上最短向量, 降低了构造格的半径, 从而提高求解的效率, 得到原问题的精确解或提高近似解的逼近程度. 给出了理论上采样归约算法最差情况的成功率. 更进一步地, 在目标解重量较低的情况下, 可以进行分段采样, 对问题增加限定条件, 提高解题效率. 实验结果表明, 对于高维度的子集和问题, 与 CJLOSS 等已有的格归约子集和问题方法相比, 该算法可以更高效地求解出问题的精确解, 而且可以提高近似解的逼近程度, 输出近似解的平均长度达到了 CJLOSS 算法的 0.55 倍、DR 算法的 0.64 倍.

关键词: 子集和问题; 格归约方法; 降维算法; 近似解

中图法分类号: TP301

中文引用格式: 曹金政, 程庆丰, 史闻博, 鲁宁. 求解子集和问题的采样格归约算法. 软件学报, 2022, 33(11): 3917–3929. http://www.jos.org.cn/1000-9825/6328.htm

英文引用格式: Cao JZ, Cheng QF, Shi WB, Lu N. Sampling-based Lattice Reduction Algorithm for Subset Sum Problem. Ruan Jian Xue Bao/Journal of Software, 2022, 33(11): 3917−3929 (in Chinese). http://www.jos.org.cn/1000-9825/6328.htm

Sampling-based Lattice Reduction Algorithm for Subset Sum Problem

CAO Jin-Zheng[1,2], CHENG Qing-Feng[1,2], SHI Wen-Bo[3], LU Ning[4,5]

[1](Strategic Support Force Information Engineering University, Zhengzhou 450001, China)

[2](State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China)

[3](School of Computer and Communication Engineering, Northeastern University at Qinhuangdao, Qinhuangdao 066004, China)

[4](School of Computer Science and Engineering, Northeastern University, Shenyang 110169, China)

[5](School of Computer Science and Technology, Xidian University, Xi'an 710126, China)

Abstract: The subset sum problem is an important problem in computer science and the basis for constructing many public key cryptosystems. A random sampling method is proposed, so the dimension of the problem can be reduced by decomposing the original problem into multiple smaller subsets sum problems, reducing the radius of the constructed lattice, thereby improving the efficiency of solving SVP, and then reaching the solution. The theoretically worst-case success rate of the algorithm is given, and a possible method to improve the success rate of the algorithm is given based on the 0-1 distribution of the solution vector, when the weight of target solution is low, the solution vector is divided into sectors, thus applying restrictive conditions to the problem to improve the efficiency of problem-solving. Experimental results show that for high-dimensional subsets and problems, compared with the existing lattice reduction

* 基金项目: 国家自然科学基金(61872449, 62072092, 62072093)

收稿时间: 2020-12-07; 修改时间: 2021-02-08; 采用时间: 2021-03-03

© 中国科学院软件研究所 http://www.jos.org.cn

subsets and problem methods such as CJLOSS, the proposed algorithm can solve the exact solution of the problem more efficiently, and can improve the approximation degree of the approximate solution, the average length of the output approximate solution is 0.55 times that of the CJLOSS algorithm and 0.64 times that of the DR algorithm.

Key words: subset sum problem; lattice reduction method; dimension-reduce algorithm; approximate solution

子集和问题由 Merkle 等人在 1978 年提出[1], 作为一个形式简洁的 NP 困难问题, 是公钥密码学研究的一个重点问题[2]. 在密码学中, 子集和问题也称为背包问题. 1978 年, Merkle 和 Hellman 提出了 Merkle-Hellman 背包密码. Shamir 对 Merkle-Hellman 背包密码体制做了进一步优化[3], 在保证算法安全性的同时, 放宽了密码体制的条件. Chor 等人进一步提出了基于有限域上高密度子集和问题的 Chor-Rivest 密码体制[4]. 此后, Okamoto 等人结合 Merkle-Hellman 密码算法和 Chor-Rivest 密码算法提出了基于子集和问题的 OTU 密码方案[5], 能够抵抗对背包密码算法的密钥恢复攻击. Kate 等人[6]基于 OTU 密码体制提出了不需要 OTU 密码体制中离散对数群的密码体制. 目前, 子集和问题在公钥密码体制方面仍有重要应用, 如 2018 年 Yang 等人[7]提出的基于子集和的同态加密方案和 2019 年 Zhang 等人[8]提出的签名方案.

研究解决子集和问题的方法具有重要的意义: 一方面, 如果存在算法在多项式时间内解决所有情况下的子集和问题, 就能够证明 P=NP 问题; 另一方面, 在公钥密码学研究中, 基于子集和问题构造的密码描述简洁, 加解密速度快, 是在早期受到较多关注的密码算法种类. 子集和问题的求解算法是对多种基于子集和问题的密码算法进行攻击的主要工具. 到目前为止, 存在多种算法解决一般的子集和问题——遗传算法、蚁群算法[9−12]、动态规划算法[13]法等. 目前最新的成果有: 2019 年, Esser 等人[14]提出的基于搜索树的启发式求解算法, 时间复杂度为 $2^{0.255n}$; 2019 年, Delaplace[15]的算法降低了存储要求; 2020 年, 欧密会 Esser 等人[16]提出了新的算法, 时间复杂度为 $2^{0.65n}$, 存储复杂度为多项式级; 2020 年, 美密会 Corond 等人[17]发表了求解隐藏子集和问题的一种多项式时间算法, 适用于子集和问题的一种特殊形式. 子集和问题是 NP 困难问题, 然而除了一般性的子集和问题求解算法, 对于满足一定条件的问题实例, 用专门的特殊求解算法, 可以明显提高求解效率. 例如, 对于低密度子集和问题, 存在利用格归约求解的方法: 1992 年, Coster 等人[18]提出了 CJLOSS 算法, 证明了当密度 $d<0.9408$ 时, 子集和问题以极大的概率可以通过寻找某个格上的最短向量来求解; 2018 年, Ping 等人[19]提出了另一种确定性格归约方法, 证明了低重量、高密度背包密码中的子集和问题可以确定性地归约到格上的最短向量问题. 格归约方法将原问题转化为格上最短向量问题. 该问题是公认的格上困难问题, 也是一个 NP 困难问题. 其复杂度仍然取决于问题的维度, 目前效率最好的算法有 BKZ[20], BKZ 2.0[21]等格基约化算法和 2019 年提出的改进筛法[22]. 对于格上的最短向量问题, 160 维以上的格矩阵已经非常难以求出准确的问题解. 针对目前存在的问题, 降低问题的维度是求解子集和问题的一个重要方向, 具有重要的理论研究价值和现实意义.

本文针对子集和问题的格归约方法现阶段面临的问题, 设计了新的采样归约算法, 通过采样, 降低构造格矩阵的规模, 达到降维的目的, 缩短求解格上最短向量所需的时间. 理论分析表明: 原问题的解可以转化为一个更小维度的格上最短向量问题, 并可以通过多次采样求出, 缩短每次进行格基约化所需的时间. 进一步地, 本文给出了算法求得精确解的概率. 另外, 每次采样可以确定性地恢复出原问题的一个近似解向量. 由于格的规模减小, 格基约化的效率更高, 所以求得的向量长度明显小于未降维的格归约算法输出的近似解长度. 实验结果表明: 对于高维度的子集和问题, 与 CJLOSS 等已有的格归约子集和问题方法相比, 该算法可以在更短时间内高效地求解出问题的精确解, 而且可以提高近似解的逼近程度, 输出近似解的平均长度达到了 CJLOSS 算法的 0.55 倍, DR 算法的 0.64 倍.

本文主要工作如下:

(1) 在格归约方法的基础上, 讨论了由格上近似最短向量是否可以恢复出一个原子集和问题的近似解, 对精确解进行逼近. 以 CJLOSS 格构造方法为例, 给出了根据格向量求出近似向量的一般公式. 并采用降维方法, 设计采样归约算法提高解题效率. 主要步骤是使用随机采样的方法, 将原问题分解为若干个低维度局部子集和问题, 并通过格归约方法构造矩阵, 将局部子集和问题转化为一个较低

© 中国科学院软件研究所 http://www.jos.org.cn

维度格上的最短向量问题提高格上最短向量问题的求解效率, 进而寻找原问题的解.

(2) 经过理论分析, 给出采样归约算法在一般情况下找到精确解的概率, 进而得到了复杂度估计. 特别地, 对于已知具有一定特征(如已知解向量 0-1 分布规律)的子集和问题, 可以对采样方法附加条件, 提高找到精确解的概率. 实验结果表明: 采样归约算法相比未经降维直接归约求解的已有格归约算法, 可以提高求解子集和问题的效率, 尤其在高维度的子集和问题上, 可以得到明显更优的近似解.

本文第 1 节介绍基础知识和现有的结果. 第 2 节提出采样归约算法. 第 3 节说明采样归约算法的正确性和效率. 第 4 节进行总结.

1 预备知识

1.1 子集和问题基本定义

本节介绍子集和问题相关定义, 更多相关知识可以参考文献[23].

定义 1(子集和问题). 给定 n 个正整数 $a_1,a_2,\ldots,a_n$ 和正整数 s, 要求确定 n 元向量 $\boldsymbol{x}=[x_1x_2\ldots x_n]$, $x_i\in[0,1]$, $1\leqslant i\leqslant n$, 使得下式成立:

$$\sum_{i=1}^{n}a_ix_i=s,$$

$\boldsymbol{x}$ 称为问题的精确解或 0-1 解.

存在特殊的子集和问题类型, 即超递增背包问题. 所谓超递增序列是指序列 $a_1,a_2,\ldots,a_n$ 满足条件:

$$\sum_{j=1}^{i-1}a_j\leqslant a_i,\ i=1,2,\ldots,n.$$

实际上, 对于超递增序列子集和问题, 已经有很多算法能够在多项式时间内解决. 而一般的子集和问题是一个困难问题.

定义 2(子集和问题的近似解). 对于定义 1 描述的子集和问题, 如果有 n 元向量 $\boldsymbol{x}=[x_1x_2\ldots x_n]$, $x_i\in Z$, $1\leqslant i\leqslant n$, 使得下式成立:

$$\sum_{i=1}^{n}a_ix_i=s,$$

则称 $\boldsymbol{x}$ 为子集和问题的近似解. 这里的系数 x_i 可以是正数或负数.

定义 3(子集和问题的密度). 给定一个子集和问题 $\sum_{i=1}^{n}a_ix_i=s$, 定义子集和问题的密度为

$$d=\frac{n}{\log_2(\max\limits_{1\leqslant i\leqslant n}a_i)}.$$

通常情况下, 可以将子集和问题按照密度分为 $d\leqslant 1$ 和 $d>1$ 两种情况. 一般的研究对于 $d\leqslant 1$ 的情况更为关注, 因为当 $d>1$ 时, 对于同一个子集和问题的解不唯一, 导致不能应用于通信.

定义 4(汉明重量). 给定一个向量 $[e_1e_2\ldots e_n]$, 该向量的汉明重量 w 是指向量分量中不为 0 的个数. 特别地, 当向量是 0-1 向量时, 有向量的汉明重量为

$$w=|\{e_i|e_i=1,1\leqslant i\leqslant n\}|.$$

1.2 格的基本概念

本文主要讨论格归约方法求解子集和问题, 以下介绍格的基本知识, 包括格的定义、格上最短向量问题等, 关于格的相关理论可以参考文献[20,21].

定义 5(格). R^m 上 n 个线性无关向量 $\boldsymbol{v}_1,\boldsymbol{v}_2,\ldots,\boldsymbol{v}_n$ 的整数线性组合得到的集合, 称为 R^m 上的格 $\mathcal{L}$. 用数学符号表示如下:

$$\mathcal{L}(\boldsymbol{v}_1,\boldsymbol{v}_2,\ldots,\boldsymbol{v}_n)=\left\{\sum_{i=1}^{n}x_i\boldsymbol{v}_i:x_i\in Z\right\},$$

© 中国科学院软件研究所 http://www.jos.org.cn

其中, 称整数 n 为格$\mathcal{L}$的秩, 记为 $\dim(\mathcal{L})=n$; 称整数 m 为格的维. 若 $n=m$, 则称格$\mathcal{L}$是满秩的. 称向量组 $\boldsymbol{v}_1, \boldsymbol{v}_2,\ldots,\boldsymbol{v}_n$ 为格$\mathcal{L}$的一组基, 可以记为 $\boldsymbol{B}=[\boldsymbol{v}_1\boldsymbol{v}_2\ldots\boldsymbol{v}_n]^T\in R^{n\times m}$(即格基是矩阵的行向量). 利用此记号, 可以将格记为 $\mathcal{L}(\boldsymbol{B})=\{\boldsymbol{x}\boldsymbol{B}|\boldsymbol{x}\in Z^n\}$. 其中, $\boldsymbol{x}\boldsymbol{B}$ 表示向量和矩阵相乘.

定义 6(最短向量问题). 如果格$\mathcal{L}$中的向量 v 满足:

$$\|\boldsymbol{v}\|=\min\{\|\boldsymbol{b}\|\|\boldsymbol{b}\in\mathcal{L}\backslash\{0\}\},$$

即向量 $\boldsymbol{v}$ 的长度为格$\mathcal{L}$中所有向量长度的最小值, 那么称向量 $\boldsymbol{v}$ 为格$\mathcal{L}$中的最短向量. 用$\lambda_1(\mathcal{L})$表示最短向量的长度, 即$\|\boldsymbol{v}\|=\lambda_1(\mathcal{L})$. 给定一个格, 找到其中非零的最短向量 $\boldsymbol{v}$, 称为格上最短向量问题.

1.3 子集和问题格归约算法

格归约算法可以求解密度或重量具有一定特点的子集和问题. 主要步骤是通过构造格矩阵, 将子集和问题转化为格上的最短向量问题. 本节主要介绍 CJLOSS 算法、DR 归约算法以及一种基于丢番图逼近的启发式求解算法.

1992 年, Coster 等人提出了 CJLOSS 算法. 该算法在 LO 算法基础上做出了优化, 几乎能够解决所有密度小于 0.941 子集和问题. 根据 Coster 等人的证明[18], 设 A 是一个正整数, $a_1,a_2,\ldots,a_n$, 其中, $0<a_i\leqslant A$, $1\leqslant i\leqslant n$. 设向量 $\boldsymbol{e}=[e_1e_2\ldots e_n]\in\{0,1\}^n$, 且有 $s=\sum_{i=1}^{n}a_ie_i$, 即 $\boldsymbol{e}$ 是解向量. 如果子集和的密度 $d<d_0=0.9408\ldots$, 如果存在一个可在多项式时间内解决 SVP 问题的算法, 那么 CJLOSS 算法几乎可以解决所有用 $a_1,a_2,\ldots,a_n$ 和 s 定义的子集和问题.

2017 年, 王保仓等人提出了基于联立丢番图逼近的启发式求解算法[24]. 该算法通过建立子集和问题与联立丢番图逼近问题之间的关系, 由给定的子集和问题构造联立丢番图逼近问题, 使用格归约方法寻找该联立丢番图逼近问题的解, 由此构造与原始子集和问题的线性无关的新的子集和问题, 从而对原问题降维.

2018 年, Ping 等人[19]提出另一种确定性格归约方法, 称为 DR 算法. 他们证明了 Chor–Rivest 等低重量、高密度背包密码体制中使用的子集和问题中的子集和问题可以确定性地归约到格上的最短向量问题. 根据算法的格构造方法, 格中最短向量以 1 的概率对应原问题的解向量.

2 采样归约算法

本节介绍采样归约算法. 关于子集和问题, 一般的格归约算法的思想是使用子集和问题的数据构造格矩阵并约化求最短向量, 最终得到子集和问题的解. 但是随着问题维度增大, 格上 SVP 的求解越来越困难. 为了加快解题速度, 使用随机采样的方法以一定概率降低问题的维度.

2.1 近似解的恢复

在描述算法之前, 讨论由归约格约化后向量求出相应解向量(或近似解向量)的方法.

使用 CJLOSS 算法的格构造方法, 根据归约后初始格基向量的形式, 已知对于 $i=1,\ldots,n$ 初始的格基形式为 $b_i=[-1\ \underset{i}{-1}\ldots n+1\ldots-1\ Na_i]$, 由此可以推导出格向量的形式.

BKZ 约化后输出的格向量可以表示为

$$\boldsymbol{\omega}=\sum_{i=1}^{n+1}x_i\boldsymbol{b}_i=[\omega_1,\ldots,\omega_{n+2}].$$

对于 $n+2$:

$$\omega_{n+2}=N\left(\sum_{i=1}^{n}x_ia_i-x_{n+1}s\right).$$

若 $\omega_{n+2}=0$, 则有 $\sum_{i=1}^{n}x_ia_i-x_{n+1}s=0$, 则由 $\boldsymbol{\omega}$ 可以求出一个符合题目要求的解向量.

对于 $1\leqslant j\leqslant n+1$:

© 中国科学院软件研究所 http://www.jos.org.cn

$$\omega_j=(n+1)x_j-\sum_{\substack{i=1\\ i\neq j}}^{n+1}x_i=(n+2)x_j-\sum_{i=1}^{n+1}x_i.$$

所以有:

$$\sum_{i=1}^{n+1}\omega_i=(n+2)\sum_{i=1}^{n+1}x_i-(n+1)\sum_{i=1}^{n+1}x_i=\sum_{i=1}^{n+1}x_i,$$

$$x_j=\frac{\omega_j+\sum_{i=1}^{n+1}\omega_i}{n+2}.$$

若要求 $x_1,\ldots,x_n$ 是一个解, 还应有 $x_{n+1}\neq 0$. 即

$$x_{n+1}=\frac{\omega_{n+1}+\sum_{i=1}^{n+1}\omega_i}{n+2}\neq 0,$$

$$\omega_{n+1}+\sum_{i=1}^{n+1}\omega_i\neq 0.$$

这样, 由格向量$\boldsymbol{\omega}$可以求得原问题的解向量:

$$[x_1\ldots x_n]=\left[\frac{\omega_1+\sum_{i=1}^{n+1}\omega_i}{n+2}+\ldots.+\frac{\omega_n+\sum_{i=1}^{n+1}\omega_i}{n+2}\right]$$

和系数:

$$x_{n+1}=\frac{\omega_{n+1}+\sum_{i=1}^{n+1}\omega_i}{n+2}.$$

使得 $x_1a_1+x_2a_2+\ldots+x_na_n=x_{n+1}s$. 由此得到了由归约后的格向量恢复原问题的精确解或近似解的方法.

2.2 采样归约方法

在已知目标解的汉明重量的情况下, 以一定概率可以降低问题的维度. 对于一般情况的子集和问题, 可以假设基本解向量中非0位近似于均匀分布. 因此可以随机排除部分数据, 仅以剩余的题目数据进行求解, 再转化为格上 SVP 进行求解. 但是删除部分数据对问题进行降维处理可能导致无法求出问题的 0-1 解, 只能求出一个近似解. 因此需要使用随机采样的方法, 即进行多次上述降维过程, 每次选取不同的数据, 这样就得到若干个不同的低维度子集和问题, 其中有一定概率可以出现一个局部子集和问题, 恰好包含得到精确解所需的数据, 进而可以恢复出原问题的解向量. 根据第 3.1 节的讨论, 每一个生成的低维度问题一定可以得到原问题的一个近似解.

首先, 对采样归约算法所求解的子集和问题的解向量有了如下基本的假设.

1) 问题的 a_i 大小符合均匀分布;

2) 解向量中 0, 1 符合均匀分布;

3) 解向量的汉明重量与问题维度相比很低($\leqslant n/8$).

假设 1)说明了问题数据的随机性, 没有超递增序列等特殊的数值规律. 假设 2)说明目标的解向量中, 0-1 的位置是随机且均匀分布的, 不会出现 1 集中出现在某些位置的情况. 实际上, 对一个子集和问题实例, 可以将 a_i 随机重新排序, 使之符合假设 2)的情况. 假设 3)表明了: 如果在解向量中随机取某些位置, 取到 0 的概率更大. 根据以上假设, 新算法的思路是: 选定若干个 $m<n$, 在解向量中取 m 个位置, 假设在解向量中对应的位置全为 0, 其余的位置可能为 0 或 1. 在此基础上, 使用格归约方法求解向量剩余 $n-m$ 个位置上的值, 这样可以将 n 维的子集和问题降低至 $n-m$ 维. 采样所得新的子集和问题称为一个局部子集和问题, 参数记为 $a_{i_1},a_{i_2},\ldots,a_{i_{n-m}},s$. 其中, $a_{i_1},a_{i_2},\ldots,a_{i_{n-m}}$ 是从原问题的参数 $a_1,a_2,\ldots,a_n$ 中取得的, s 即为原问题的目标和 s. 多次重复采样过程, 获得多个不同的局部问题并求解, 直到得到精确解或达到近似解的逼近要求. 存在一定的概率,

© 中国科学院软件研究所 http://www.jos.org.cn

随机采样取得的 m 个位置 $u_{i_1},u_{i_2},u_{i_3},...,u_{i_m}$ 在问题目标解对应的位置 $x_{i_1},x_{i_2},x_{i_3},...,x_{i_m}$ 恰好全为 0, 即目标的精确解向量 $\boldsymbol{x}$ 中, $x_{i_1}=0,x_{i_2}=0,...,x_{i_m}=0$. 此时, 采样得到的局部子集和问题 $a_{i_1},a_{i_2},...,a_{i_{n-m}},s$ 包含了求得精确解的全部题目数据信息, 且维度由 n 降低到了 $n-m$, 与原问题相比更有可能求出归约格的最短向量.

具体步骤如下.

算法 1. 采样归约方法.

定义 $\boldsymbol{a}=[a_1a_2...a_n]$和 s.

初始化 $l=0$, 向量 $\boldsymbol{e}$, 阈值 thr, 整数 N.

1) 选择 $P\subset\{1,...,n\}$, $|P|=m$;

2) 取值 $[u_{i_1}...u_{i_m}]=[0\,0...0]$;

3) 对如下构造的 $n-m+1$ 格矩阵 B 进行 BKZ 约化:

$$\boldsymbol{b}_{i_1}=[n-m+1\ -1\ ...\ -1\ -1\ Na_{i_1}]$$
$$\boldsymbol{b}_{i_2}=[-1\ n-m+1\ ...\ -1\ -1\ Na_{i_2}]$$
$$...$$
$$\boldsymbol{b}_{i_{n-m}}=[-1\ -1\ ...\ n-m+1\ -1\ Na_{i_{n-m}}]$$
$$\boldsymbol{b}_{i_{n-m+1}}=[-1\ -1\ ...\ n-m+1\ Ns]$$

4) 取最后一个坐标是 0 的最短向量, 设 $[e'_{i_1}\ ...\ e'_{i_{n-k}}\ 0]$ 是返回值;

5) 令

$$[e_{i_1},...,e_{i_{n-m}}]=\left[\frac{e'_{i_1}+\sum_{l=1}^{n-m+1}e_{i_l}}{n-m+2}\ ...\ \frac{e'_{i_{n-m}}+\sum_{l=1}^{n-m+1}e_{i_l}}{n-m+2}\right],\ [e_{j_1}...e_{j_m}]=[u_1...u_m];$$

6) **if** $\sum_{i=1}^{n}a_ie_i=s$, $[e_1...e_n]\in\{0,1\}^n$

7) **then** 输出$[e_1...e_n]$;

8) **else**

9) **if** $[e_1...e_n]$长度小于 l

10) **then** 令 $l=[e_1...e_n]$长度, $e=[e_1...e_n]$;

11) **end**

12) **end**

13) **if** $l<thr$

14) **then** 输出 e.

根据对算法的假设, 本文所研究子集和问题解向量的重量较低, 且解向量的 0-1 基本符合均匀分布, 这说明如果将解向量均分为若干段, 则每一段中 0 与 1 的数目基本相同, 且在分段内 1 仍然是均匀分布的. 因此在选取 $[u_{i_1}...u_{i_m}]=[0\,0...0]$ 时, 可以将解向量划分为 r 段, 记为 $\boldsymbol{a}=[u_1u_2...u_n]=[\boldsymbol{u}^{(1)}\boldsymbol{u}^{(2)}...\boldsymbol{u}^{(r)}]$, 每一段的长度相同, 其中,

$$\boldsymbol{u}^{(i)}=[u_{i_1}\ u_{i_2}\ ...\ u_{i_s}],\ \sum_{i=1}^{r}\sum_{k=1}^{s}u_{i_k}=n.$$

在每一个分段 $\boldsymbol{u}^{(i)}$, $1\leqslant i\leqslant r$ 中选取一定数量的元素设为 0. 这样就将问题的维度减小了 m. 在具体优化过程中, 取 $m=n-90$.

每次采样后, 建立的格可能没有充分约化, 因而得不到最短向量. 这种情况下, 对每个矩阵进行约化, 并计算得到的近似子集和解向量, 从中取最短向量, 并选中其对应的矩阵, 认为这个矩阵有可能含有原问题的解, 继续约化求解 SVP. 当然, 只取一个最短结果可能不够精确, 或者错漏一些可以取到精确解的情况. 针对这个问题, 可以在算法过程中维持一个列表 L, 在采样并求近似解后更新列表, 使表中储存所得的较短的结

© 中国科学院软件研究所 http://www.jos.org.cn

果, 最后对表中记录的格矩阵做 SVP 求解, 选取最短者得到最终的解向量.

另外需要考虑采样后局部问题的密度. 原问题的密度定义为

$$d=\frac{n}{\log_2(\max\limits_{1\leqslant i\leqslant n}a_i)}.$$

经过采样降维后, 得到新的密度为

$$d'=\frac{n-m}{\log_2(\max\limits_{j\notin P,1\leqslant j\leqslant n}a_j)},$$

其中, $n-m<n$, $\max_{j\notin P,1\leqslant j\leqslant n}a_j\leqslant\max_{1\leqslant i\leqslant n}a_i$. 因此, 新产生局部子集和问题的密度可能会有变化. 为了使算法构造格矩阵仍然能符合格归约方法的条件, 需要令参数 N 满足 $N>(n-m)^2\cdot\max_{1\leqslant i\leqslant n}a_i$. 等价于令采样得到的局部子集和问题 $a_{i_1},a_{i_2},\ldots,a_{i_{n-m}},s$ 同乘以 $\max_{1\leqslant i\leqslant n}a_i$, 可以保证局部问题的密度 $d'\leqslant d$.

3 算法分析

3.1 正确性分析

首先证明采样后的局部子集和问题可以得出原问题的解.

定理 1. 采样归约方法在随机采样后, 如果取到的局部格矩阵中存在精确解, 则采样得到的格中最短向量对应其精确解. 在局部问题对应的格上, 解向量对应一个以$[t\ t\ldots t]$, $|t|\lesssim(n-m)/2$ 为中心、半径 $\lesssim\sqrt{n-m}/2$ 的短向量.

证明: 采样归约算法产生的局部问题, 归约后得到一个 $n-m+1$ 维的格, 用 $\boldsymbol{b}_1,\boldsymbol{b}_2,\ldots,\boldsymbol{b}_{n-m+1}$ 表示其基向量.

考虑格中的任一向量, 可以用基线性表示为 $\boldsymbol{\omega}=\sum_{i=1}^{n-m+1}x_i\boldsymbol{b}_i=[\omega_1\ldots\omega_{n-m+2}]$.

对于 $1\leqslant j\leqslant n-m+1$:

$$\omega_j=(n-m+1)x_j-\sum_{\substack{i=1\\ i\neq j}}^{n+1}x_i.$$

对于 $n-m+2$:

$$\omega_{n-m+2}=N\left(\sum_{i=1}^{n-m}x_ia_i-x_{n-m+1}s\right).$$

通过乘方可得$\boldsymbol{\omega}$的长度平方:

$$\|\boldsymbol{\omega}\|^2=(n-m+2)^2\sum_{i=1}^{n-m+1}x_i^2-(n-m+3)\left(\sum_{i=1}^{n-m+1}x_i\right)^2+N^2\cdot E^2,$$

其中, $E=\sum_{i=1}^{n-m}x_ia_i-x_{n-m+1}s$.

如果 $x_i=e_i$, $1\leqslant i\leqslant n-m$, $x_{n-m+1}=1$, 则 $E=0$, 且$\|\boldsymbol{\omega}\|^2$ 的上界约为$(n-m)^3/4$(而且如果 e_i 大多为 0, 上界可以进一步降低). 接下来讨论大多数情况下没有更短的向量. 如果 $E\neq0$, 则 $\|\boldsymbol{\omega}\|^2\geqslant N^2\geqslant(n-m)^4\cdot\max_{1\leqslant i\leqslant n}a_i^2$, 所以只需考虑 $E=0$ 的情况. 此时, $x\in Z^{n-m+1}$ 的上界估计值:

$$F(\boldsymbol{x})=(n-m+2)^2\sum_{i=1}^{n-m+1}x_i^2-(n-m+3)\left(\sum_{i=1}^{n-m+1}x_i\right)^2$$

满足 $F(\boldsymbol{x})\leqslant(n-m)^3/4$.

$F(\boldsymbol{x})$又可以表示为

$$F(\boldsymbol{x})=\sum_{i=1}^{n-m+1}x_i^2+(n-m+3)\sum_{1\leqslant i<j\leqslant n-m+1}(x_i-x_j)^2.$$

© 中国科学院软件研究所 http://www.jos.org.cn

设 t 的值:

$$t=\frac{1}{n-m+1}\sum_{i=1}^{n-m+1}x_i,$$

则

$$F(\boldsymbol{x})=\sum_{i=1}^{n-m+1}x_i^2+(n-m+1)(n-m+3)\sum_{i=1}^{n-m+1}(x_i-t)^2.$$

进而，如果 $F(\boldsymbol{x})\lesssim(n-m)^3/4$，那么，

$$\sum_{i=1}^{n-m+1}(x_i-t)^2\lesssim(n-m)/4.$$

所以，$\boldsymbol{\omega}$包含在$[t\ t\ldots t]$为中心、半径 $\sqrt{n-m}/2$ 的球内．接下来可以讨论 t 的取值个数．若 $F(\boldsymbol{x})\lesssim(n-m)^3/4$，显然，$\sum_{i=1}^{n-m+1}x_i^2\lesssim(n-m)^3/4$．根据 Cauchy-Schwartz 不等式:

$$\left(\sum_{i=1}^{n-m}x_i\right)^2\leqslant(n-m+1)\sum_{i=1}^{n-m+1}x_i^2\lesssim(n-m)^4/4,$$

所以，$|t|\lesssim(n-m)/2$．进一步地，$(n-m+1)t\in Z$，所以可知，t 的取值个数$\lesssim(n-m)^2$．由此可以计算出所需的满足 $F(\boldsymbol{x})\lesssim(n-m)^3/4$ 的向量 $\boldsymbol{x}\in Z^{n-m+1}$ 的个数． □

以上讨论说明了格中最短向量极大概率对应的就是原问题的解，这就将原子集和问题归约到了一个相应维度的最短向量求解问题．关于局部问题的格归约，更多具体细节可以参考 Coster 等人[15]的讨论．如果原问题不经降维直接构造格进行约化，给出的格所在空间半径为 $\sqrt{n}/2$．采样后，问题维度由 n 降为 $n-m$，新得到的矩阵为 $n-m+1$ 维．则解向量 $\boldsymbol{x}'$对应向量是以$[t\ t\ldots t]$为中心、半径 $\sqrt{n-m}/2$ 的球中的向量．这样，通过降低问题的维度，可以将半径减小到 $\sqrt{n-m}/2$，从而提高约化效率．

推论 1．随机采样后，如果取到的局部格矩阵中存在近似解 $\boldsymbol{x}=[c_1\ldots c_{n-m}]$，$|c_i|\leqslant 2$，$i=1,\ldots,n-m$，则采样得到局部问题对应的格上，近似解向量对应一个以$[t\ t\ldots t]$，$|t|\lesssim(n-m)/2$ 为中心、半径 $\lesssim\sqrt{2(n-m)}$ 的短向量．

实际上，近似解对应的向量$\boldsymbol{\omega}$长度平方为

$$\|\boldsymbol{\omega}\|^2=(n-m+2)^2\sum_{i=1}^{n-m+1}x_i^2-(n-m+3)\left(\sum_{i=1}^{n-m+1}x_i\right)^2+N^2\cdot E^2.$$

如果 $x_i=c_i$，$1\leqslant i\leqslant n-m$，$x_{n-m+1}=1$，则 $E=0$(前提已经说明$[c_1\ldots c_{n-m}]$是近似解)，且$\|\boldsymbol{\omega}\|^2$ 的上界约为$(n-m)^3$．此时，$\boldsymbol{x}\in Z^{n-m+1}$ 的上界估计值:

$$\begin{aligned}F(\boldsymbol{x})&=(n-m+2)^2\sum_{i=1}^{n-m+1}x_i^2-(n-m+3)\left(\sum_{i=1}^{n-m+1}x_i\right)^2\\&=\sum_{i=1}^{n-m+1}x_i^2+(n-m+3)\sum_{1\leqslant i<j\leqslant n-m+1}(x_i-x_j)^2\\&=\sum_{i=1}^{n-m+1}x_i^2+(n-m+1)(n-m+3)\sum_{i=1}^{n-m+1}(x_i-t)^2\\&\leqslant 2(n-m)^3.\end{aligned}$$

进而有:

$$\sum_{i=1}^{n-m+1}(x_i-t)^2\leqslant 2(n-m).$$

所以，$\boldsymbol{\omega}$包含在$[t\ t\ldots t]$为中心、半径 $\sqrt{2(n-m)}$ 的球内．

实际上，以上对向量$\boldsymbol{\omega}$长度估计的前提是问题目标解重量$\leqslant n/2$，对于低重量子集和问题，$\boldsymbol{\omega}$长度可以得到更低的长度上界．例如，对于目标解重量$\leqslant n/8$ 的问题，精确解对应的向量长度 $F(\boldsymbol{x})\lesssim 7(n-m)^3/64$，所以$\boldsymbol{\omega}$半径上界为 $\sqrt{7(n-m)}/8$．因此，子集和问题的重量越低，采样归约算法需要求解的格向量长度越短．这样，如果

© 中国科学院软件研究所 http://www.jos.org.cn

降维后取到的数据恰好包含原问题的解的数据, 则仍然能够通过求解最短向量问题得到原子集和问题的解.

3.2 复杂度分析

对于一个 n 维、重量为 k 的子集和问题, 如果不考虑对解向量的分段, 可以推导出该算法的成功率如下:

首先, 经过采样过程, 向量由 n 维减为 $n–m$ 维, 可能的情况共有 $\binom{n}{n-m}$ 种, 其中, $\binom{n-k}{n-m-k}$ 种情况有可能求出精确解的情况, 即采样得到的数据可以形成一个有解的 $n–m$ 维子集和问题.

根据本文对子集和问题的基本假设, 目标解向量的 0-1 均匀分布, 可以得到, 算法以如下概率在采样后可以求得原问题的解:

$$\frac{\binom{n-k}{n-m-k}}{\binom{n}{n-m}}=\frac{(n-k)!(n-m)!m!}{m!(n-m-k)!n!}=\frac{(n-k)!(n-m)!}{(n-m-k)!n!}=\frac{(n-m-k+1)(n-m-k+2)...(n-m)}{(n-k+1)(n-k+2)...n}>\left(\frac{n-m-k+1}{n-k+1}\right)^k.$$

即需要求解 $\left(\frac{n-k+1}{n-m-k+1}\right)^k$ 次格上最短向量, 可以得到原问题的一个精确解. 平均求解次数随 k 的增大而增大. 当 k 的值足够小, 可以近似认为 $n–k\approx n$, $n–m–k\approx n–m$, 此时正确概率为

$$\frac{\binom{n-k}{n-m-k}}{\binom{n}{n-m}}=\frac{(n-k)!(n-m)!m!}{m!(n-m-k)!n!}=\frac{(n-k)!(n-m)!}{(n-m-k)!n!}=\frac{(n-m-k+1)(n-m-k+2)...(n-m)}{(n-k+1)(n-k+2)...n}\approx\left(\frac{n-m}{n}\right)^k.$$

因此, 对于采样规模 m 和可忽略的重量 k, 平均要求解 $\left(\frac{n}{n-m}\right)^k$ 次格上最短向量, 可以得到原问题的一个精确解. 平均次数随重量指数增加.

下面考虑分段采样的情况. 假设将解向量分解为 r 段(为简化讨论, 可以假设 r 整除 n, m, k), 则目标向量中每个分段中应该约有 k/r 个 1.

与以上讨论类似, 每个小分段中恰好取得正确解的概率为 $\frac{\binom{n/r-k/r}{m/r}}{\binom{n/r}{m/r}}$. 根据假设, r 个分段之间互不相关, 所以可得算法成功率:

$$\frac{\binom{n/r-k/r}{m/r}^r}{\binom{n/r}{m/r}^r}=\frac{(n/r-k/r)!(n/r-m/r)!}{(n/r-k/r-m/r)!(n/r)!}.$$

根据斯特林公式, 可以近似估计概率:

$$\begin{aligned}\frac{\binom{n/r-k/r}{m/r}^r}{\binom{n/r}{m/r}^r}&=\frac{(n/r-k/r)!(n/r-m/r)!}{(n/r-k/r-m/r)!(n/r)!}\\&\approx\sqrt{\frac{(n/r-k/r)(n/r-m/r)}{(n/r-k/r-m/r)(n/r)}}\cdot\frac{(n/r-k/r)^{n/r-k/r}(n/r-m/r)^{n/r-m/r}}{(n/r-k/r-m/r)^{n/r-k/r-m/r}(n/r)^{n/r}}\\&=\frac{(n-k)^{1/2+n/r-k/r}(n-m)^{1/2+n/r-m/r}}{(n-k-m)^{1/2+n/r-k/r-m/r}n^{1/2+n/r}}\approx\frac{(n-k)^{n/r-k/r}(n-m)^{n/r-m/r}}{(n-k-m)^{n/r-k/r-m/r}n^{n/r}}.\end{aligned}$$

© 中国科学院软件研究所 http://www.jos.org.cn

由此可以得到取得原问题解所需进行 SVP 求解次数的期望值:

$$\frac{(n-k-m)^{n/r-k/r-m/r}n^{n/r}}{(n-k)^{n/r-k/r}(n-m)^{n/r-m/r}}.$$

当 k 与 n, $n-m$ 相比可以忽略时, 期望值可以进一步近似为

$$\frac{(n-k-m)^{n/r-k/r-m/r}n^{n/r}}{(n-k)^{n/r-k/r}(n-m)^{n/r-m/r}}\approx\frac{(n-m)^{n/r-k/r-m/r}n^{n/r}}{n^{n/r-k/r}(n-m)^{n/r-m/r}}\approx\frac{n^{k/r}}{(n-m)^{k/r}}.$$

根据估计, 正确求解子集和问题所需要的采样数随降维的幅度 m 指数增加. 增大分段数 r 可以减少所需的采样数, 提高理论成功率.

4 实验分析

本节实验有两个目的: 一是验证采样归约算法的正确性, 并展示其效率; 二是验证对算法成功率的估计值进行验证.

4.1 实验准备

使用的实验数据是随机生成的低密度子集和问题, 目标解的重量约为维度的 1/8. 问题的维度为 60–200 维, 以 20 的间隔递增. 确保实验所用子集和问题的数据在一定区间内均匀分布, 且问题密度约为 0.9. 以 200 维的子集和问题为例, 使用随机生成的数据, 数据最小为 173 比特, 最大 180 比特, 密度 0.9. 求解格上最短向量问题使用的算法是 BKZ 和 BKZ 2.0 算法. 求解格 SVP 问题时的矩阵运算作和格相关算法利用了 NTL 函数库[25]的实现. 测试计算机使用 2.70 GHz Intel Core i5 处理器和 4 GB 内存.

4.2 结果与分析

首先考察采样归约算法在较低维度(n=60, 80, 100, 120)的问题的输出结果. 在这种情况下, 问题求解的效率较高, 可以在较短时间内求得问题的精确解, 如表 1 所示.

表 1 求精确解的实验结果

维度	60	80	100	120	140	140
问题密度	0.9	0.9	0.9	0.9	0.9	1.1
目标解的重量	5	7	10	12	15	15
解的长度	$\sqrt{5}$ =2.2361	$\sqrt{7}$ =2.6458	$\sqrt{10}$ =3.1623	$\sqrt{12}$ =3.4641	$\sqrt{15}$ =3.8730	$\sqrt{15}$ =3.8730
用时(s)	0.5	0.6	25	3 043	5 854	4 763

经过以上实验, 在 60–140 维的子集和问题上验证了本文新提出的采样归约算法的正确性和可行性. 限制格归约方法效率的主要因素就是在高维度格上求解 SVP 的速度太慢, 难以得到需要的精确解向量, 只能得到一个近似解. 在较高维度上使用新算法与已有文献的算法做实验比较, 具体结果见表 2.

表 2 高维度问题求近似解的实验结果

算法	维度	精确解长度	求得长度	用时(h)
CJLOSS 算法	160	$\sqrt{20}$	10.954 5	9
	180	$\sqrt{22}$	15.329 7	39
	200	$\sqrt{25}$	23.685 4	70
DR 算法	160	$\sqrt{20}$	13.856 4	8
	180	$\sqrt{22}$	21.023 8	20
	200	$\sqrt{25}$	23.494 7	54
本文采样归约算法	160	$\sqrt{20}$	$\sqrt{20}$ (精确解)	6
	180	$\sqrt{22}$	12.124 4	8
	180	$\sqrt{22}$	9.643 7	9
	200	$\sqrt{25}$	19.364 9	11
	200	$\sqrt{25}$	12.083 0	12

© 中国科学院软件研究所 http://www.jos.org.cn

在求解 160 维问题时, 利用采样归约算法求出了精确解, 而原始算法近似的时间内只能得到一个近似解. 在 180 维和 200 维的问题上, 新算法也可以在更短时间内得到较好的近似结果. 根据 160–200 维实例上的实验结果, 采样归约算法输出近似解的平均长度达到了 CJLOSS 算法的 0.55 倍、DR 算法的 0.64 倍, 逼近程度明显优于已有格归约算法. 同时, 在求近似解的过程中, 采样归约算法耗时明显小于已有的算法, 平均用时达到了 CJLOSS 算法的 0.23 倍、DR 的 0.33 倍.

用得到解的长度与目标解长度的比作为衡量解近似程度的指标, 可以发现, 随着维度增大, 得到近似解的逼近程度逐渐下降, 其中, 本文提出的采样归约算法得到的结果明显优于已有算法, 如图 1 所示.

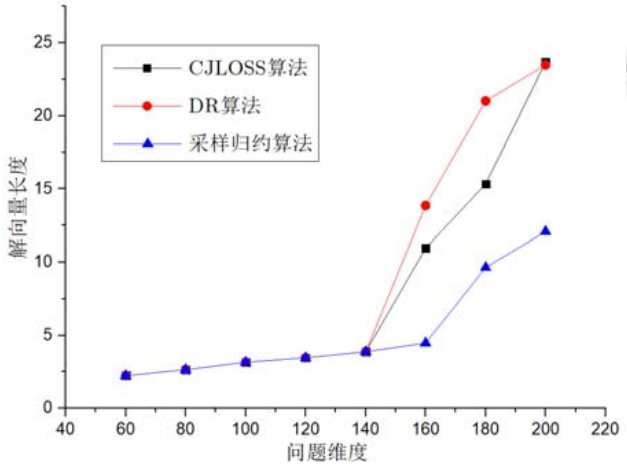

图 1 各维度求解近似程度

综合以上结果, 利用随机采样的降维算法, 在 160–200 维的较高维度上均得到了很好的优化. 并且, 对于 160 维, 用时 6 h 找到了该题的精确解, 而同一问题使用 CJLOSS 算法经过 9 h、DR 算法经过 8 h 仍然没有得到精确的 0-1 解. 同样地, 对于 180 维、200 维的数据, 新算法的使用也可以很好地减少归约格的 BKZ 约化耗时, 从而提高解题效率.

以求解 100 维的子集和问题为例, 通过实验估计随机采样得到问题精确解的概率, 并与第 4.2 节的理论估计做比较. 结果见表 3. 结果说明, 实验得到的精确解分布频率和理论估计概率基本相符, 验证了该算法的复杂度估计.

表 3 求精确解概率的实验结果

n	m	r	k	0-1 解分布	估计概率
100	10	1	12	0.200 0	0.260 8
100	10	4	12	0.281 3	0.266 8
100	20	1	12	0.009 2	0.057 4
100	20	4	12	0.062 5	0.060 4
100	20	10	12	0.119 6	0.107 4
100	40	1	12	0.004 6	0.001 3
100	40	4	11	0.281 3	0.002 7
100	50	5	10	0.002 7	0.000 7
100	50	10	10	0.005 2	0.000 9

经过以上讨论, 初步的理论分析和实验都验证了降维算法的正确性, 可以得出高维度子集和问题的近似解. 同时, 其速度的优化基本合理, 即使在较高的维度下也能够以较短的时间输出子集和问题的解, 且输出质量与已知算法相比有较大的提升.

© 中国科学院软件研究所 http://www.jos.org.cn

5 总 结

子集和问题的格归约算法是求解较低密度子集和问题的一种重要方法. 本文在已有格归约算法的基础上, 结合对已有算法的理论分析和对问题数据的研究, 提出了求解子集和问题的算法——采样归约算法. 主要思想是: 使用随机采样方法将原问题分解为多个局部子集和问题, 降低问题维度, 将原问题分解为多个更小规模的子集和问题, 降低了构造格的半径, 从而提高求解相应 SVP 的效率, 进而得到原问题的精确解或提高近似解的逼近程度. 与原有的格归约算法相比, 该算法使用了新的优化技术, 可以在一定成功率的前提下降低问题的维度, 使格基约化效率进一步提高.

在题目假设下进行理论分析, 给出了算法的成功率和计算出精确解需要的采样次数估计, 并提出对于目标解 0-1 均匀分布的情况可以使用分段采样的方法, 并给出了优化后的算法成功率. 通过在较高维度子集和问题的实验结果, 验证了该算法的实用性. 通过与未降维的格归约子集和算法比较, 可以验证该算法的性能具有较大的提高, 同时, 其速度的优化基本合理, 即使在较高维度的问题下, 解出最短向量的时间仍在可接受范围内, 且输出质量与已知算法相比有较大提升. 尤其在高维度子集和问题的求近似解过程中, 新算法用时远少于 CJLOSS 和 DR, 且求得的近似解质量更好, 在合适的参数下可以达到速度与质量的较好平衡. 下一步研究中, 算法在提高成功率和与其他方法结合方面需要继续改进.

References:

[1] Merkle RC, Hellman ME. Hiding information and signatures in trapdoor knapsacks. IEEE Trans. on Information Theory, 1978, 24(5): 525−530.

[2] Thomas HC, Charles EL, Ronald LR, Clifford S. Introduction to Algorithms. 3rd ed., New York: MIT Press, McGraw-Hill, 2009.

[3] Shamir A. Embedding cryptographic trapdoors in arbitrary knapsack systems. Information Processing Letters, 1983, 17(2): 77−79.

[4] Chor B, Rivest RL. A knapsack type public key cryptosystem based on arithmetic in finite fields. In: Proc. of the Advances in Cryptology—CRYPTO 1988. Santa Barbara, 1988. 901−908.

[5] Okamoto T, Tanaka K, Uchiyama S. Quantum public-key cryptosystems. In: Proc. of the Advances in Cryptology—CRYPTO 2000. Santa Barbara, 2000. 147−165.

[6] Kate A, Goldberg I. Generalizing cryptosystem based on the subset sum problem. Int'l Journal of Information Security, 2011, 10(3): 189−199.

[7] Yang J, Fan M, Wang G. A public key size homomorphic encryption scheme based on the sum of sparse subsets and integers. Cognitive Systems Research, 2018, 52: 543−549.

[8] Zhang X, Liu S, Pan J, Gu D. Tightly secure signature schemes from the LWE and subset sum assumptions. Theoretical Computer Science, 2019, 795: 326−344.

[9] Fidanova S. Ant colony optimization and multiple knapsack problem. In: Proc. of the Numerical Analysis and Its Applications—NAA 2004. Rousse, 2004. 280−287.

[10] Abd-Alsabour N. Binary ant colony optimization for subset sum problems. Multi-objective Swarm Intelligence, 2015, 592: 105−121.

[11] Maniezzo V, Roffilli M. Very strongly constrained problems: An ant colony optimization approach. Cybernetics and Systems: An Int'l Journal, 2008, 39(4): 395−424.

[12] Syslo MM. Discrete optimization algorithms. Microprocessors and Microsystems, 1985, 9(2): 118−165.

[13] Cao Z. New algorithms for subset sum problem. 2018. https://arxiv.org/pdf/1807.02611.pdf

[14] Esser A, May A. Better sample—Random subset sum in $2^{0.255n}$ and its impact on decoding random linear codes. 2019. https://arxiv.org/abs/1907.04295

[15] Delaplace C, Esser A. Improved low-memory subset sum and LPN algorithms via multiple collisions. In: Proc. of the IMACC 2019. Oxford, 2019. 78−101.

[16] Esser A, May A. Low weight discrete logarithms and subset sum in $2^{0.65n}$ with polynomial memory. In: Proc. of the EUROCRYPT 2020. Zagreb, 2020. 94−122.

© 中国科学院软件研究所 http://www.jos.org.cn

[17] Coron J, Gini A. A polynomial-time algorithm for solving the hidden subset sum problem. In: Proc. of the CRYPTO 2020. Santa Barbara, 2020. 3−31.

[18] Coster MJ, Joux A, Lamacchia B, *et al*. Improved low-density subset sum algorithms. Computational Complexity, 1992, 2(2): 111−128.

[19] Ping Y, Wang B, Tian SL, *et al*. Deterministic lattice reduction on knapsacks with collision-free properties. IET Information Security, 2018, 12(4): 375−380.

[20] Schnorr C, Euchner M. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. Mathematical Programming, 1994, 66(1−3): 181−199.

[21] Chen Y, Nguyen PQ. BKZ 2.0: Better lattice security estimates. In: Proc. of the ASIACRYPT 2011. Seoul, 2011. 1−20.

[22] Ducas L. Shortest vector from lattice sieving: A few dimensions for free. In: Proc. of the EUROCRYPT 2018. Tel Aviv, 2018. 125−145.

[23] Cohen H. A Course in Computational Algebraic Number Theory. Berlin: Springer, 1993.

[24] Wang BC, Lu K. Heuristic algorithm for the subset sum problem based on simultaneous Diophantine approximation. Journal of Cryptologic Research, 2017, 4(5): 498−505 (in Chinese with English abstract).

[25] Shoup V. NTL, number theory C++ library. 2020. http://www.shoup.net/ntl/

附中文参考文献:

[24] 王保仓, 卢珂. 基于联立丢番图逼近的子集和问题启发式求解算法. 密码学报, 2017, 4(5): 498−505.

曹金政(1998－), 男, 硕士生, 主要研究领域为公钥密码, 格密码.

史闻博(1980－), 男, 博士, 教授, 博士生导师, 主要研究领域为信息安全.

程庆丰(1979－), 男, 博士, 教授, 博士生导师, 主要研究领域为公钥密码, 密码协议.

鲁宁(1984－), 男, 博士, 副教授, 博士生导师, 主要研究领域为网络安全.

© 中国科学院软件研究所 http://www.jos.org.cn