计 算 机 研 究 与 发 展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202221052
61(12): 3069−3087, 2024

全同态加密研究进展

白利芳[1,2,3] 祝跃飞[1] 李勇军[1,4] 王 帅[1] 杨晓琪[2]

[1](信息工程大学网络空间安全学院 郑州 450001)

[2](中国软件评测中心网络空间安全测评工程技术中心 北京 100048)

[3](数据安全关键技术与产业应用评价工业和信息化部重点实验室(中国电子信息产业发展研究院) 北京 100048)

[4](中原工学院 郑州 450007)

(bailifang@cstc.org.cn)

Research Progress of Fully Homomorphic Encryption

Bai Lifang[1,2,3], Zhu Yuefei[1], Li Yongjun[1,4], Wang Shuai[1], and Yang Xiaoqi[2]

[1](*School of Cyberspace Security*, *Information Engineering University*, *Zhengzhou* 450001)

[2](*Cybersecurity Testing Engineering Technology Center*, *China Software Testing Center*, *Beijing* 100048)

[3](*Key Laboratory of Data Security Technology and Industrial Application Evaluation of the Ministry of Industry and Information Technology* (*China Center for Information Industry Development*), *Beijing* 100048)

[4](*Zhongyuan University of Technology*, *Zhengzhou* 450007)

Abstract With the acceleration of the digitization process, the problem of data security and privacy protection has attracted much attention. Data encryption has always been an important means to solve this problem. However, it is common to store and transfer data in encrypted form. Once calculation is involved, it is often necessary to decrypt the ciphertext, perform the calculation in plaintext, and then encrypt the calculation result. Full homomorphic encryption (FHE) extends encryption to the computing, which can perform meaningful calculations in ciphertext without decryption, and the calculation process and result are encrypted, so it has the natural characteristics of protecting data security and user privacy. The first FHE scheme was proposed by Gentry in 2009, and then FHE has always attracted the attention of the industry and academia. After more than ten years of research, FHE has developed to the fourth stage, and substantial progress has been made. We review and analyze the research progress of FHE from the aspects of the construction idea, different research stages and problems faced, introduce the application progress of FHE from the aspects of algorithm library, standardization progress and typical application scenarios, and put forward suggestions for future research direction.

Key words fully homomorphic encryption (FHE); lattice; LWE problem; bootstrapping; homomorphic decryption; algorithms library

摘 要 随着数字化进程的加速推进,数据安全和隐私保护问题备受关注.数据加密一直是解决该问题的重要手段,但加密存储和传输较为常见,一旦涉及计算往往需要先解密,以明文形式计算后再加密.全同态加密(fully homomorphic encryption, FHE)将加密延展到计算层面,无需解密即可以完成密文的处理任务,有保护数据安全和用户隐私的天然特性.首个FHE方案于2009年由Gentry提出,自此FHE方案一直

收稿日期: 2022-12-30; **修回日期**: 2023-10-30

基金项目: 国家重点研发计划项目(2019QY1300); 国家自然科学基金青年基金项目(62102447)

This work was supported by the National Key Research and Development Program of China (2019QY1300) and the National Natural Science Foundation of China for Young Scientists (62102447).

通信作者: 祝跃飞(mompidan@163.com)

备受业界和学界的关注.从FHE方案的构造思想、不同研究阶段及面临的问题等方面梳理分析了FHE 10余年的研究进展,从算法库实践、标准化进展以及典型应用场景等方面介绍了FHE的应用进展,并提出未来研究的方向建议.

关键词 全同态加密;格;LWE问题;自举;同态解密;算法库

中图法分类号 TP309.7

全同态加密(fully homomorphic encryption, FHE)支持对密态数据进行任意计算而无需事先解密,计算后的结果仍是加密状态,且解密后就是用户想要的结果,即与“对明文数据进行对应计算”的结果相同.可以理解为,FHE方案能够为任意数据处理功能构建一个“程序”,该“程序”无需解密输入数据,密态输入且密态输出,运行过程全密态,因此可在不受信任的一方运行.这一特性很大程度上解决了各应用场景下,互不信任的2方或多方之间进行数据处理时的数据安全和隐私保护问题,使得FHE具有很强的通用性.首个真正的FHE方案由Gentry于2009年提出[1],但由于该方案的实现效率远远达不到应用要求,给大家留下了FHE方案性能低的刻板印象.经过十几年的努力,目前FHE研究及应用取得了重大突破,已从最初的概念验证变为可实际应用的工具.

本文首先介绍相关概念定义,并从自举(bootstraping)、同态解密切入阐述FHE方案的一般构造思想,其次分析讨论不同阶段FHE的研究进展及局限性,随后从算法库实践、标准化进展、典型应用等方面介绍FHE的应用进展,最后总结并展望FHE亟待解决问题和发展方向.

1 概念定义

1.1 概念理解

1)同态.2个代数结构间的一种映射$\delta: A \to B$,满足$\delta(x *_A y) = \delta(x) *_B \delta(y)$.由于$x *_A y = z \Rightarrow \delta(x) *_B \delta(y) = \delta(z)$,所以代数结构$\langle A, *_A \rangle$和$\langle B, *_B \rangle$在该映射下相关结构保持不变,可以理解为同态是2个代数结构(如群、环、域或向量空间)之间的保持结构不变的映射.可以用交换图表述同态的概念,如图1所示,从左上角出发,“先右再下”和“先下再右”2条途径都可以得到右下角的结果,即对任意$x, y \in A$,“先$*_A$运算再δ映射”与“先δ映射再$*_B$运算”的结果相同,即满足交换性.例如,取$\delta(x) = e^x$时,交换图可具体表述为如图2所示.此时A为$\mathbb{R}$,B为$\mathbb{R}^+$(大于0的实数集合),$*_A$为“+”,$*_B$为“×”.

2)同态加密.同态加密是指满足同态性质的加密算法,即该加密算法的密文空间和明文空间之间的映射(解密函数)对特定运算满足同态性质.因此同态加密支持对密文空间进行特定功能的运算(用函数f表示),且解密后的结果与直接对明文空间进行对应运算的结果相同.若f只能由加法构成,即仅支持加法运算,称为加同态,如Paillier算法[2];若f只能由乘法构成,即仅支持乘法运算,称为乘同态,如RSA算法[3].一般称仅支持加法或乘法运算的加密方案为部分同态加密.若f可由加法和乘法构成,但支持的运算次数有限,则称为近似同态加密(somewhat homomorphic encryption, SHE).

3)FHE. FHE是指支持密文域上任意运算的同态加密算法.而同态加密算法定义在域上的任意运算均可用加法和乘法构造,因此称同时满足加同态和乘同态的加密算法为FHE算法.即FHE应能同时支持不限次的加法和乘法运算,可实现对密态数据任意有意义的操作,其形式化定义详见1.2节.

需要注意的是,此处的加法和乘法不等同于加减乘除中的加法和乘法,而是一种一般性表达,如布尔函数中的与(AND)运算和异或(XOR)分别对应此处的乘法和加法.事实上,任意逻辑运算均可通过AND和XOR运算组合表达,任意算术运算均可通过加法和乘法运算组合表达,即任意运算均可通过加法和乘法运算的组合来实现.

4)层级全同态加密(leveled FHE, LFHE).若支持一定复杂度范围内的任意运算,则称为LFHE.此处复杂度理解为将该运算用电路形式表示时其电路深

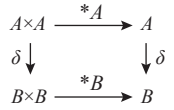


Fig. 1 Schematic diagram of homomorphic exchange

图1 同态交换示意图

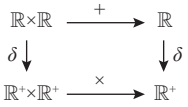


Fig. 2 Example of homomorphic exchange

图2 同态交换示例

度的大小，即电路深度应不超过某个值L. 而任意运算可理解为由加法和乘法任意构成.

事实上，现有同态加密方案中经常用运算电路C来表示f. 因为不管是布尔电路还是算术电路，其包含的操作仅仅是电路门所对应的逻辑操作或算术操作，没有选择、判断、循环等复杂操作，因此应用于同态加密研究时非常便利. 在密码学研究尤其是在计算复杂性研究中，经常使用电路来表示运算或算法，如此，一个函数可以等效为一个电路模型，电路模型下可通过电路深度和门电路个数来衡量计算复杂度. 本文将使用电路来表示运算或函数.

5）格与基. 格是欧氏空间$\mathbb{R}^m$中$n(m \geqslant n)$个线性无关向量组$\boldsymbol{b}_1,\boldsymbol{b}_2,\cdots,\boldsymbol{b}_n$所有整系数线性组合的集合：

$$\Lambda = L(\boldsymbol{B}) = \left\{ \boldsymbol{Bx} = \sum_{i=1}^{n} \boldsymbol{b}_i x_i :\ \boldsymbol{x} \in \mathbb{Z}^n \right\},$$

即格是$\mathbb{R}^m$中一类具有周期性结构离散点的集合. 称$\boldsymbol{b}_1,\boldsymbol{b}_2,\cdots,\boldsymbol{b}_n$为$n$维格$\Lambda$的一组基（即格基），$\boldsymbol{B} \in \mathbb{R}^{m\times n}$是格基的矩阵表示. 事实上，格基可以不唯一，且不同格基之间可通过一个行列式为± 1的矩阵进行转换.

6）容错学习（learning with errors, LWE）问题. 给定矩阵$\boldsymbol{A} \in \mathbb{R}_q^{m\times n}$和向量$\boldsymbol{b} = \boldsymbol{As} + \boldsymbol{e} (\mathrm{mod}\, q)$，求解随机的秘密向量$\boldsymbol{s} \in \mathbb{Z}_q^n$是困难的. 其中$q$是模数（通常取较大的素数），$\boldsymbol{e} \in \mathbb{Z}_q^m$是一个小的误差向量. LWE 问题可看作是求解带噪声的线性方程组问题，由 Regev[4] 于 2005 年提出，是一个被充分研究的密码学假设，由于目前尚无有效的量子算法求解 LWE 问题，因此基于 LWE 问题构造的加密方案被认为是抗量子攻击的. 若$\boldsymbol{A},\boldsymbol{s},\boldsymbol{e}$取自特定环上，则转化成 LWE 问题的变体，称为 R-LWE 问题. 其主要特点是：一般情形下（average-case）解决 LWE 问题并不比在最坏情形下（worst-case）解决一些著名格近似问题简单.

1.2 FHE 定义

1）核心算法. 目前 FHE 方案均基于公钥密码体制，一个 FHE 方案除了包含公钥密码体制中的 3 个基础算法外，还包含 1 个实现同态计算功能的算法，即包含 4 个概率多项式算法：密钥生成算法$KeyGen()$、加密算法$Enc()$、解密算法$Dec()$、同态计算算法$Eval()$.

①密钥生成算法. 该算法为随机算法，$(\boldsymbol{pk},\boldsymbol{sk}) \leftarrow KeyGen(1^\lambda)$，以安全参数$1^\lambda$为输入，输出公钥$\boldsymbol{pk}$和私钥$\boldsymbol{sk}$.

② 加密算法. 该算法为随机算法，$\boldsymbol{c} \leftarrow Enc(\boldsymbol{pk},m)$，以公钥$\boldsymbol{pk}$和消息$m \in \{0,1\}$为输入，输出密文$\boldsymbol{c}$.

③解密算法. 该算法为确定性算法，$m \leftarrow Dec(\boldsymbol{sk},\boldsymbol{c})$，以私钥$\boldsymbol{sk}$和密文$\boldsymbol{c}$为输入，输出消息$m \in \{0,1\}$.

④同态计算算法. 该算法为随机算法，$\boldsymbol{c}^* \leftarrow Eval(\boldsymbol{pk}, C, \boldsymbol{c}_1,\boldsymbol{c}_2,\cdots,\boldsymbol{c}_t)$，以公钥$\boldsymbol{pk}$、运算电路$C \in \mathcal{C}$和$\boldsymbol{c}_1,\boldsymbol{c}_2,\cdots,\boldsymbol{c}_t$为输入，输出密文$\boldsymbol{c}^*$. 其中，$\mathcal{C}$为一个电路族，$\boldsymbol{c}_i \leftarrow Enc(\boldsymbol{pk},m_i)$，$i = 1,2,\cdots,t$.

2）正确性. 若对$\forall m_1,m_2,\cdots,m_t \in \{0,1\}$，及$\forall C \in \mathcal{C}$，有$Dec(\boldsymbol{sk},\boldsymbol{c}^*) = C(m_1,m_2,\cdots,m_t)$，则称该加密方案关于同态电路族$\mathcal{C}$中的任意运算函数$C$是正确的. 可用交换图表述如图 3 所示.

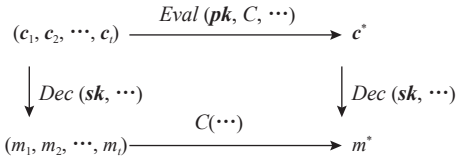


Fig. 3 FHE exchange diagram

图 3 FHE 交换图

每个同态加密方案均对应一个电路族$\mathcal{C}$，只有函数$C \in \mathcal{C}$时，先调用C再解密和先解密再调用C是可交换的，即该方案关于C是同态的. 若$\mathcal{C}$是任意的，即构造的同态加密方案对任意运算均满足正确性（即图 3 所示的可交换性），则该方案是全同态的.

3）紧凑性. 对于任意的安全参数λ，若存在一个多项式P，使得同态加密方案的解密算法能够用一个规模至多为$P(\lambda)$的电路D来表示，那么称该同态加密方案是紧凑的，即满足紧凑性意味着该方案的解密电路独立于t和C.

4）安全性. 同态加密的安全性通过语义安全来定义. 用基于游戏的范式可表述为：若对于任意多项式时间敌手$\mathcal{A}$，式（1）在λ上可忽略，称该加密方案是不可区分选择明文攻击安全的（IND-CPA 安全）.

$$\left| Pr\left[\mathcal{A}(\boldsymbol{pk}, Enc(\boldsymbol{pk},0)) = 1\right] - Pr\left[\mathcal{A}(\boldsymbol{pk}, Enc(\boldsymbol{pk},1)) = 1\right] \right| = neg(\lambda), \quad (1)$$

其中$(\boldsymbol{pk},\boldsymbol{sk}) \leftarrow KeyGen(1^\lambda)$. 即选择明文攻击（chosen plaintext attacks, CPA）是安全的，意味着 0 与 1 的密文对于敌手$\mathcal{A}$是不可区分的，所以安全的同态加密算法应为概率算法（每个明文将对应多个密文）.

2 构造思想

同态加密方案的构造总体上分为同态性满足和安全性满足两大实现. 构造实践中，可以先寻求同态性的满足，使得解密函数具有同态性，而后寻求方案的安全性满足. 也可以先建立一个密码学安全假设，然后在此假设下构建公钥加密方案，而后通过引入其他操作（即构造函数$Eval$）使方案满足同态性. 二者

的区别在于前者可以选择更多的数学结构用于同态加密方案的构造,后者则基于相对完备的密码学假设提前保障安全性,只是可能很难快速构造出满足同态性的方案.

Rothblum[5]给出了一种通用方法,可将明文空间为{0,1}的任意满足语义安全的私钥同态加密方案转化为满足语义安全性的公钥同态加密方案.该方法简化了方案的安全性证明,为基于同态映射来构建语义安全的同态加密方案提供了思路.

FHE方案的构造重点在于实现密文域上任意次数的加法和乘法同态运算.一般构造的初始方案均为SHE方案,因在加密和密文运算过程中会引入"噪声"用于掩饰明文信息,随着密文同态运算次数越多、噪声值越大,当噪声达到某一"临界"就会解密失败,因此能支持的同态运算次数有限.如何将SHE方案转化为FHE方案是关键,目前大多借助自举技术来实现.

2.1 自举技术

自举技术本质上是通过同态解密运算,即执行参数电路C为解密电路D(函数Dec的电路表现形式)的函数$Eval$,将密文刷新为一个噪声值更低的"新鲜"密文,如此不会因为噪声干扰导致解密失败,即可继续进行同态运算,从而实现不限次同态运算,自举过程示意图如图4所示.

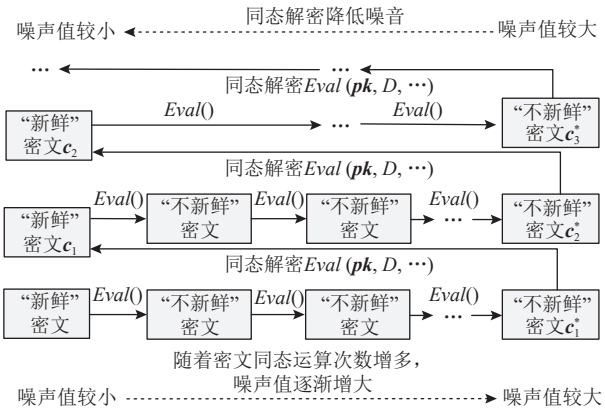


Fig. 4 Illustration of bootstraping process

图4 自举过程示意图

图4的同态解密过程中,解密电路D将作为函数$Eval$的执行参数,同态解密最右侧的"不新鲜"密文,得到最左侧噪声值更低的"新鲜"密文,且每次同态解密运算前后的密文对应同一个明文,即$\boldsymbol{c}_1$和$\boldsymbol{c}_1^*$对应相同的明文,$\boldsymbol{c}_2$和$\boldsymbol{c}_2^*$对应相同的明文.

值得注意的是,同态解密过程可以消除解密电路D对应的加密运算当初引入的噪声,但由于函数$Eval$是在密文空间执行的,因此还会引入新的噪声.因此需要保证新引入的噪声大小允许方案继续执行密文同态运算,这就要求解密电路的深度要小于起初构造的SHE方案所允许的密文计算深度.也就是说,方案实现自举过程的前提是函数$Eval$可运行方案自身的解密电路.当不能运行时,就要采取措施使其解密电路的深度变小,Gentry09[1]采用了一个"压缩(squashing)"技术去实现,代价是增加一个安全性假设——稀疏子集和问题(sparse subset sum problem, SSSP),不过目前的方案已能够运行自身解密电路,如BGV[6]方案、Bra12[7]方案等.

2.2 同态解密

自举的关键环节是同态解密,即在密文空间执行解密函数,该函数用加密的私钥去解密加密后的密文,输出一个噪声值更低的"新鲜"密文,即明文对应的另一个密文.实现原理以Gentry09[1]为例,说明如下:

明文空间为{0,1},即明文是单个比特,且有2个不同的公私钥对$(\boldsymbol{pk}_1,\boldsymbol{sk}_1)$和$(\boldsymbol{pk}_2,\boldsymbol{sk}_2)$,私钥由用户自行保管,公钥公开发布.

1)用户将明文消息m的密文$\boldsymbol{c}\leftarrow Enc(\boldsymbol{pk}_1,m)$发送至服务器;

2)用户用另一个公钥$\boldsymbol{pk}_2$对私钥$\boldsymbol{sk}_1$进行逐比特加密,得到密文序列$\overline{\boldsymbol{sk}_1}$(即加密的密钥),并发送至服务器;

3)服务器用公钥$\boldsymbol{pk}_2$对密文$\boldsymbol{c}$进行逐比特加密,得到密文序列$\bar{\boldsymbol{c}}$(即双重加密状态的密文);

4)服务器用加密的私钥$\overline{\boldsymbol{sk}_1}$同态解密双重加密的密文$\bar{\boldsymbol{c}}$,由同态加密的正确性定义得:

$$\begin{aligned}\boldsymbol{c}^*\leftarrow Eval\left(\boldsymbol{pk}_2,D,\overline{\boldsymbol{sk}_1},\bar{\boldsymbol{c}}\right)=\\ Enc_{\boldsymbol{pk}_2}(D(\boldsymbol{sk}_1,\boldsymbol{c}))=\\ Enc_{\boldsymbol{pk}_2}(m),\end{aligned}$$

即经过同态解密,密文$\boldsymbol{c}^*$是明文消息m用另一个公钥$\boldsymbol{pk}_2$加密的结果,和$\boldsymbol{c}$一样是"新鲜"密文,均具有较低的噪声值.该过程可扩展至$\boldsymbol{c}$为"不新鲜"密文时的情形,即可对经过多次同态运算的高噪声密文经同态解密后刷新为低噪声的"新鲜"密文,如图4所示.

注意到,同态加密过程中私钥被加密后作为公共参数予以公开,并作为函数$Eval$的输入,因此被称为计算公钥;另外出现了明文被双重加密的状态,其中第1次加密(看作"内层加密")的结果是第2次加密(看作"外层加密")的明文输入.同态解密的过程可看作是在双重加密状态下解密内层保留外层加密的过程,如图5所示.

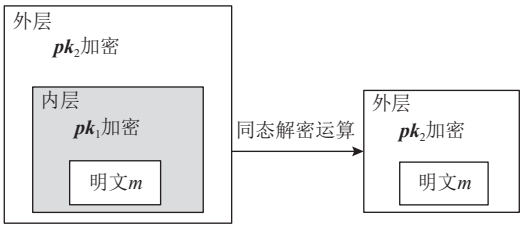


Fig. 5 Homomorphic decryption schematic diagram

图 5 同态解密示意图

3 FHE 研究进展

现有主流 FHE 方案的困难假设主要基于 2 种困难问题：一种是基于格困难问题，另一种是整数上基于近似最大公约数(approximate greatest common divisor, AGCD)困难问题. 其中基于格困难问题的 FHE 方案发展迅速，目前已发展至第 4 代，尤其以基于 R-LWE 问题假设的方案发展最为迅速(如第 2, 3, 4 代)，另外，还有一个基于 NTRU 的研究分支. 本节首先介绍 Gentry09[1] 提出前仅支撑部分同态运算的典型加密方案，再对基于格困难问题的 4 代典型 FHE 方案进行研究分析，最后对基于整数、NTRU 的 FHE 方案进行简要分析. 经典 FHE 方案的分类及继承关系如图 6 所示，其中虚线框内的方案基于格困难问题.

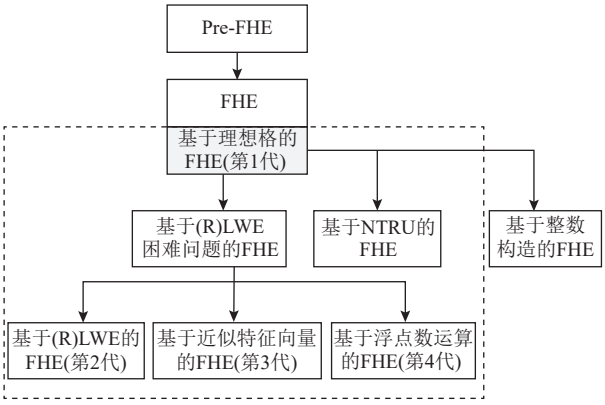


Fig. 6 FHE research venation schematic diagram

图 6 FHE 研究脉络示意图

3.1 支持部分同态的 Pre-FHE 方案

本文将只满足加同态或乘同态的加密方案，以及 SHE 方案统称为不完全同态加密方案. 自 1978 年 Rivest 等人提出隐私同态加密的概念(即 FHE 构想)以来，FHE 方案的探索研究历程可总结为对密文域上的任意多项式函数实现同态运算的过程. 在 2009 年 Gentry 构造出第 1 个真正的 FHE 方案之前，所有的同态加密方案均为不完全同态加密方案，如 RSA[3], Rabin[8], EIGamal[9] 算法是仅支持不限次乘法运算的乘同态加密算法，即支持任意多元单项式函数的同态运算；Paillier[2] 算法和文献 [10–19] 是仅支持不限次加法运算的加同态算法，即支持任意多元一次多项式函数的同态运算；SYY99[20] 支持任意 NC^1 复杂类电路(Fan-in 度为 $O(1)$，电路深度为 $O(\log n)$)，IP07[21] 支持任意有限长度的分支程序(branching programs)，包括 NC^1 中的电路(因任意 NC^1 类复杂度计算均可被多项式大小的分支程序执行)，文献 [20–21] 是仅支持有限次加法和乘法的 SHE 方案，即支持的多项式函数项数和次数有限. BGN05[22] 支持不限次的加法和至多 1 次乘法运算，即支持任意多元二次多项式函数的同态运算，且具备同态运算后密文大小保持不变的优点. MGH08[23] 支持不限次加法和有限次乘法运算，即支持任意多元 d 次多项式函数的同态运算；此外，Polly Cracker 方案[24] 和 GP06[25] 等曾尝试构建一个 FHE 方案，但均被证实存在严重安全漏洞[26-27]. 还有基于对称加密机制的探索[28-29] 但很快也被指出有安全漏洞[30-31].

由于对密文同态运算的支持能力不足(详见表 1)或安全性的问题，Pre-FHE 方案均未实现对密文数据进行任意多项式函数的同态运算，但此阶段的同态

Table 1 Imperfect Homomorphic Encryption Scheme

表 1 不完全同态加密方案

同态加密方案	支持的加、乘运算次数	支持的多项式运算
RSA[3]	不限次乘法	任意多元单项式
文献 [8]	不限次乘法	任意多元单项式
文献 [9]	不限次乘法	任意多元单项式
文献 [2]	不限次加法	任意多元一次多项式
GM82[10]	$\mathbb{Z}_2$ 上的不限次加法	任意多元一次多项式
文献 [11]	不限次加法	任意多元一次多项式
文献 [12]	不限次加法	任意多元一次多项式
文献 [13]	不限次加法	任意多元一次多项式
文献 [14]	不限次加法	任意多元一次多项式
GK05[15]	不限次加法	任意多元一次多项式
AKK07[16]	不限次加法	任意多元一次多项式
CGP08[17]	不限次加法	任意多元一次多项式
CB08[18]	不限次加法	任意多元一次多项式
AS08[19]	不限次加法	任意多元一次多项式
SYY99[20]	有限次加法和乘法	有限项多项式
IP07[21]	有限次加法和乘法	有限项多项式
BGN05[22]	不限次加法，至多 1 次乘法	任意多元二次多项式
MGH08[23]	不限次加法，有限次乘法	任意多元 d 次多项式

加密研究已然为首次成功构建 FHE 方案奠定了坚实基础,其中不乏中有基于格的不完全同态加密方案[15-19]. 尤其是首个 FHE 方案提出前的近几年,但多为加法同态加密方案.

3.2 基于理想格的第 1 代 FHE 方案

第 1 代 FHE 方案是指以 Gentry09[1] 为基础基于理想格的 FHE 方案,其构造过程为:首先基于理想格构建一个 SHE 方案,其安全性基于固定环上理想格的最近向量问题(closest vector problem, CVP);其次通过压缩解密电路使得方案可自举,安全性依赖于 SSSP 假设;而后基于压缩后的解密电路通过同态解密实现自举,从而实现全同态,其中自举过程是实现全同态的关键,同时也极大地增加了计算量.

该方案实现了 FHE 理论性的重大突破,但在实际应用中存在许多瓶颈:1)该方案在密钥生成算法中有个遗留问题,即如何生成拥有良好格基的理想格;2)压缩解密电路的过程中引入了一个未被充分研究的安全性假设(即 SSSP);3)计算量过大导致性能太低的问题,这也是最典型问题. 因此,后续有许多基于该方案的改进研究:Gentry10 给出了一个新的密钥生成算法[32],并通过量子最坏情况/平均情况归约提高 SSSP 假设的安全性;同年 SS10[33] 进一步完善了 Gentry10 提出的密钥生成算法,并对 SSSP 假设做了更深入的分析,并提出一种解密电路深度更小(为原方案的平方根)的概率解密算法以降低方案计算复杂度;OYKU10[34], SS11[35], GH11a[36] 等均对 Gentry09[1] 的密钥生成算法进行了优化.GH11b[37] 和 GHS12[38] 则专注于对自举过程的性能优化,但由于私钥向量由 n 维约减到 1 维,增加了遭受 Key Recovery 攻击的概率. 此外, Smart 等人[39] 在 Gentry09 方案的基础上,提出具有相对较小尺寸密钥和密文的 FHE 方案 SV10,但密钥生成性能很差且后续不再生成密钥对,意味着该方案是个 SHE 方案. SV10 首次在 FHE 语境下正式分析批处理的可能性,并指出只要参数选择合理即可支持单指令多数据(single instruction multiple data, SIMD)操作,但当时此方案的参数并不支持 SIMD 操作. 通过调整 SV10[39] 和 GH11a[36] 中的参数,利用中国剩余定义提出支持 SIMD 操作的 FHE 方案 SV11[40],该方案沿用了 GH11a 的密钥生成算法,并通过 SIMD 操作实现了重加密的并行执行. M12[41] 提出了一种具有更大明文空间的 SHE 方案,但密钥生成算法复杂度稍有增加.

文献 [32–41] 方案均针对 Gentry09 提出的 FHE 构造框架进行优化,直到 2011 年基于 LWE 假设的 FHE 方案被提出,至此, FHE 的研究进入第 2 个阶段.

第 1 代 FHE 方案的局限性在于:1)构造较为复杂且不易理解;2)噪声增长速度和密文膨胀较快,且自举程序性能太差,导致 FHE 方案的研究更多停留在理论研究阶段,不具备实际操作意义;3)为了使 FHE 方案可自举,需压缩解密电路、降低方案的性能和引入了未被充分研究的 SSSP 安全假设.

3.3 基于 LWE 的第 2 代 FHE 方案

第 2 代 FHE 方案首次基于 LWE 问题假设构造 FHE 方案, 2011 年, CRYPTO2011 和 FOCS2011 先后提出基于 R-LWE 问题和标准 LWE 问题假设的 FHE 方案. BV11a[42] 的 SHE 方案的构造基于 LPR10[43] 提出的公钥加密方案实现,密钥生成时无需生成格基,其安全性量子规约到理想格上的最坏情形困难问题,方案后续关于解密电路的压缩方法和自举实现仍沿用了 Gentry09[1] 的做法. BV11b[44] 的 SHE 方案采用的重线性化(relinearization)技术基于 Regev05[4] 构造,其安全性归约到任意格上的最坏情况下短向量问题(short vector problem, SVP)的困难性[45],而在此之前所有基于格的 FHE 方案构造均依赖于各种环中的理想. 此外, BV11b 不再采用 Gentry09 的解密电路压缩方法,提出一种新的维-模约减(dimension-modulus reduction)技术使得方案可自举,而无需引入额外的安全性假设;并提出一种模交换(modulus switching)技术来控制密文噪声的膨胀,且“降噪”过程无需第 1 代方案中的“计算公钥”,实现更简单、高效. 不久, BGV 方案[6] 在 ITCS2012 被提出,该方案的参数大小与同态运算电路的深度而非阶数(degree)相关,性能上有了极大提升,此外提出密钥切换(key switching)技术来控制密文维数的膨胀,并证明了 BV11b[44] 提出的模交换技术可在不借助自举程序的情形下实现密文“降噪”,使得 LFHE 方案的构造无需依赖 Gentry09 的自举程序,一定程度上缓解了 FHE 方案构造对自举程序的依赖.

BGV 方案中密钥切换作为关键环节,需借助 2 个子算法:比特分解算法 $BitDecomp(\cdot)$ 和 2 的幂次算法 $Powerof2(\cdot)$ 来实现,伪代码描述如算法 1 和算法 2[6,46] 所示.

算法 1. $BitDecomp(\boldsymbol{x},q)\rightarrow\boldsymbol{u}$ 将 n 维模 q 整数向量 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)$ 分解为二进制表示形式.

```
{BitDecomp(x, q)
  for (i = 1; i ⩽ n; i++)
    将 x_i 表示为二进制;
    将 x_i 的第 j 比特写入 u_j 的第 i 分量;
```

end for

$\boldsymbol{u} \leftarrow \left(\boldsymbol{u}_0, \boldsymbol{u}_1, \cdots, \boldsymbol{u}_{\lceil \mathrm{lb}q \rceil - 1}\right)$;

return $\boldsymbol{u}$;

}

算法 2. $Powerof2(\boldsymbol{x}, q) \rightarrow \boldsymbol{v}$的输出为$n$维模$q$整数向量$\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$乘以 2 的不同次幂.

$Powerof2(\boldsymbol{x}, q)$

{

for ($i = 0$; $i \leqslant \lceil \mathrm{lb}q \rceil - 1$; $i++$)

$\boldsymbol{v}_i = \boldsymbol{x} \cdot 2^i$;

end for

$\boldsymbol{v} \leftarrow \left(\boldsymbol{v}_0, \boldsymbol{v}_1, \cdots, \boldsymbol{v}_{\lceil \mathrm{lb}q \rceil - 1}\right) \bmod q$;

return $\boldsymbol{v}$;

}

对于等长的向量$\boldsymbol{c}$和$\boldsymbol{s}$, 有性质 1:

性质 1. $\langle BitDecomp(\boldsymbol{c}, q), Powerof2(\boldsymbol{s}, q)\rangle = \langle \boldsymbol{c}, \boldsymbol{s}\rangle \bmod q$.

性质 1 可以应用到密钥切换过程: 首先利用$SwithKeyGen()$算法, 输出一个附加信息$\boldsymbol{\tau}_{s_1 \rightarrow s_2}$用于密钥切换算法$SwitchKey()$, 伪代码描述如算法 3 和算法 4 所示.

算法 3. $SwitchKeyGen(\boldsymbol{s}_1, \boldsymbol{s}_2) \rightarrow \boldsymbol{\tau}_{s_1 \rightarrow s_2}$输入 2 个不同长度的密钥$\boldsymbol{s}_1$($n_1$维)和$\boldsymbol{s}_2$($n_2$维), 输出的附加信息$\boldsymbol{\tau}_{s_1 \rightarrow s_2}$是一个矩阵.

$SwitchKeyGen(\boldsymbol{s}_1, \boldsymbol{s}_2)$

{

$\boldsymbol{A} \leftarrow PubKeyGen(\boldsymbol{params}, \boldsymbol{s}_2)$; /*$\boldsymbol{s}_2$对应的公钥$\boldsymbol{A}$*/

$\boldsymbol{v} \leftarrow Powerof2(\boldsymbol{s}_1, q)$;

将向量$\boldsymbol{v}$(按列)加到矩阵$\boldsymbol{A}$的首列, 得到矩阵$\boldsymbol{B}$;

$\boldsymbol{\tau}_{s_1 \rightarrow s_2} \leftarrow \boldsymbol{B}^{\mathrm{T}}$;

return $\boldsymbol{\tau}_{s_1 \rightarrow s_2}$.

}

算法 4. $\boldsymbol{c}_2 \leftarrow SwitchKey\left(\boldsymbol{\tau}_{s_1 \rightarrow s_2}, \boldsymbol{c}_1\right)$将可用密钥$\boldsymbol{s}_1$解密的密文$\boldsymbol{c}_1$切换为可用$\boldsymbol{s}_2$解密的密文$\boldsymbol{c}_2$.

$SwitchKey\left(\boldsymbol{\tau}_{s_1 \rightarrow s_2}, \boldsymbol{c}_1\right)$

{

$\boldsymbol{c}_2 \leftarrow \tau_{s_1 \rightarrow s_2} \times BitDecomp(\boldsymbol{c}_1)$;

return $\boldsymbol{c}_2$.

}

结合算法 3、算法 4 和性质 1, 得到性质 2.

性质 2. $\langle \boldsymbol{c}_2, \boldsymbol{s}_2\rangle = (2\langle BitDecomp(\boldsymbol{c}_1, q), \boldsymbol{e}_2\rangle + \langle \boldsymbol{c}_1, \boldsymbol{s}_1\rangle) \bmod q$, 其中错误向量$\boldsymbol{e}_2$在生成公钥$\boldsymbol{A}$过程中随机抽样而得, 且满足$\boldsymbol{A}\boldsymbol{s}_2 = 2\boldsymbol{e}_2$. 由于第 1 项的内积较$q$非常小(因为$\boldsymbol{e}_2$很小), 因此有$\langle \boldsymbol{c}_2, \boldsymbol{s}_2\rangle = \langle \boldsymbol{c}_1, \boldsymbol{s}_1\rangle \bmod q$, 结合 BGV 方案的解密算法, 性质 2 意味着经过密钥切换, $\boldsymbol{c}_2$经过$\boldsymbol{s}_2$解密后的结果与$\boldsymbol{c}_1$经过$\boldsymbol{s}_1$解密后的结果相同. 如此, 可将同态运算后维度扩张的密文经密钥切换后刷新至低维度.

BGV 被提出的当年, Bra12[7] 利用张量乘积技术构造了一个标度不变(scale-invariant)的 FHE 方案, 即无需利用复杂的模交换技术即可使得密文噪声随同态计算呈线性增长而非指数级增长, 其安全性基于 GapSVP 的困难性. FV12[47](即 BFV 方案)对 Bra12[7] 进行了优化, 将其推广到基于 R-LWE 问题假设, 通过分析各种同态运算所引入噪声的严格最坏边界, 确定 SHE 方案应能处理的最小电路深度, 确保自举过程结束后还能处理 1 次乘法运算, 从而实现全同态; 此外还提出 2 个更优的重线性化版本使得密钥更小且运算速度更快. 随后也有一些针对 Bra12[7] 的优化方案[48-50]. 2016 年, BEH16[51] 提出一个基于 BFV 方案的 Full-RNS (residue number systems)变体, 该方案能与乘法和解密过程中涉及的逐系数除法和舍入操作更好地兼容. 2020 年, 赵秀凤等人[52] 基于 BV11a[41] 提出密钥独立消息(key dependent message, KDM)安全的非对称版本 FHE 方案, 即循环安全的公钥 FHE 方案. 在 EUROCRYPT2023, GIK23[53] 通过利用模p^e零多项式的代数结构特点, 将影响 FHE 自举过程关键性能的数字提取函数进行稀疏表示, 实现方案 BGV 和 BFV 的自举加速, 比当前最先进 BGV 的自举速度提高 2 倍以上(仅限于特定参数下的对比).

第 2 代 FHE 方案除了上述安全性和性能方面的提升, 还有一个重大突破即支持 SIMD 操作. SV11[39] 首次提出基于 R-LWE 的 FHE 方案同样支持 SIMD 操作. GHS12[38] 采用 SV11[39] 的密文封装技术实现了基于 BGV[6] 的 SIMD 操作, 即可通过将多个明文比特消息封装在每个密文的多个槽(slots)中进行批处理, 并可利用自同构性质对各个槽进行置换, 如此各个槽中的数据也可互相运算, 且方案的性能开销与安全参数优化呈多重对数关系. BGH13[54] 提出利用 PVW08 的封装方法将多个明文比特封装到一个 Regev 加密方案[4] 的密文中获得一个封装的密文, 进而实现对打包密文的 SIMD 同态操作, PVW08[55] 相比 SV11[39] 的密文封装技术, 不只可用于基于 R-LWE 的 FHE 方案, 还可用于基于标准 LWE 的 FHE 方案, 虽然性能上有一定牺牲, 从此将 SIMD 同态操作拓展至任意基于 LWE 的 FHE 方案.

第 2 代 FHE 方案的构造仍需先获得一个 SHE 方

案, 而后通过自举技术实现完全同态, 但构造过程较第 1 代 FHE 方案主要有 4 点优势: 1)SHE 方案的构造基于 LWE 问题假设, 不直接依赖于格构造(密钥生成算法显然无需考虑格基的生成), 但其安全性可量子规约到任意格上的最坏情形困难问题; 2)函数 $Eval$可执行方案自身的解密电路, 实现自举无需引入额外的安全性假设, 即无需对解密电路进行压缩; 3)可不依赖自举程序获得 LFHE 方案; 4)可以处理封装的密文而不只是 1 个明文比特的密文. 总体上较第 1 代 FHE 方案安全性更强、构造更简单、高效且支持 SIMD 操作. 但其局限性在于: 1)增加了存储开销, 因密钥切换技术的实现需借助 $BitDecomp(\cdot)$和 $Powerof2(\cdot)$两个子模块来实现, 导致密钥尺寸的膨胀; 2)自举实现要求方案底层的格问题在近似因子为超多项式增长时仍保持困难性, 安全假设强度过大[56]; 3)方案若要支持任意次同态运算, 仍需借助自举技术.

3.4 基于近似特征向量的第 3 代 FHE 方案

2013 年, Gentry 等人[57]在 CRYPTO2013 提出基于近似特征向量的 FHE 方案(即 GSW 方案[57]), 并实现了基于身份的 FHE 方案和基于属性的 FHE 方案, 标志着第 3 代 FHE 方案的诞生. GSW 中的同态运算均为简单的矩阵加法和乘法运算, 本质上解决了密文维数膨胀的问题, 且通过对噪声项中的高范数矩阵分解为低范数二进制比特矩阵, 有效控制噪声增长, 又因运行函数 $Eval$时无需计算公钥, 所以该方案更简单、更高效, 但要实现全同态仍需借助自举程序. 2014 年, BV14[58]指出基于 GSW 可用更短的参数实现 FHE, 因此噪声增长更慢, 且安全性与基于格的常规 PKE 相当, 随后, Brakerski 等人[58]在 ITCS2014 提出利用 Barrington 定理[59]来改进 GSW 中的自举程序, 即利用分支程序(branching program)来实现同态运算. 然而同年 AP14[60]便指出该方法非常低效, 并通过将解密电路转换为算术电路(非 GSW 的布尔电路), 高效构造深度较小的解密电路, 且无需用 Barrington 定理将其转换为分支程序, 从而得到一种性能更优且误差增长更慢的自举方法; 同时利用 MP12[61]提出的工具矩阵(gadget matrix)给出一个技术层面更简单的 GSW 变体, 后续关于 GSW 的典型优化方案均基于该变体, 且自举愈加高效. 现结合 AP14, 采用工具矩阵 $\boldsymbol{G}$(而非原始 GSW 采用的 $Flatten(\cdot)$, $BitDecomp(\cdot)$, $Powerof2(\cdot)$等算法), 给出层级 GSW 方案的简要描述为:

1)参数生成. $G.Init\left(1^{\lambda},1^{l}\right)\rightarrow(q,n,N,\chi)$, 其中 λ为安全参数, l为层级数. 输出模数 q, 维数 n, 向量个数 N(一般大于 $2n\mathrm{lb}q$), $\mathbb{Z}$上的噪声分布高斯函数 χ, 令 $\boldsymbol{params}=(q,n,N,\chi)$.

2)密钥生成. $G.KeyGen(\boldsymbol{params})\rightarrow(\boldsymbol{pk},\boldsymbol{sk})$:

①随机选取 n维向量 $\boldsymbol{t}\leftarrow\mathbb{Z}_q^n$并计算 $\boldsymbol{sk}=\boldsymbol{s}\leftarrow(1,-\boldsymbol{t})$;

②随机选取矩阵 $\boldsymbol{B}\leftarrow\mathbb{Z}_q^{N\times n}$, 噪声向量 $\boldsymbol{e}\leftarrow\chi^N$, 设 $\boldsymbol{b}\leftarrow\boldsymbol{Bt}+\boldsymbol{e}\in\mathbb{Z}_q^N$, 令 $\boldsymbol{b}$右连接矩阵 $\boldsymbol{B}$记为 $\boldsymbol{A}$, 即 $\boldsymbol{A}\leftarrow[\boldsymbol{b}|\boldsymbol{B}]\in\mathbb{Z}_q^{N\times(n+1)}$, 输出 $\boldsymbol{pk}\leftarrow\boldsymbol{A}$(这里有 $\boldsymbol{As}=\boldsymbol{e}$).

3)加密. $G.Enc(\boldsymbol{params},\boldsymbol{pk},m)\rightarrow\boldsymbol{C},m\in\{0,1\}$:

①令 $\boldsymbol{G}$为 $(n+1)\times M$阶工具矩阵, 并随机选取矩阵 $\boldsymbol{R}\leftarrow\{0,1\}^{N\times M}$;

②计算并生成密文 $\boldsymbol{C}\leftarrow m\boldsymbol{G}+\boldsymbol{A}^{\mathrm{T}}\boldsymbol{R}(\mathrm{mod}\,q)\in\mathbb{Z}_q^{(n+1)\times M}$.

4)解密. $G.Dec(\boldsymbol{params},\boldsymbol{sk},\boldsymbol{C})\rightarrow m$:

①令 i满足 $q/4<2^{i-1}\leqslant q/2$, $\boldsymbol{C}_i$为密文 $\boldsymbol{C}$的第 i列, 在 $(-q/2,\,q/2]$范围内计算 $x\leftarrow\langle\boldsymbol{C}_i,\boldsymbol{s}\rangle(\mathrm{mod}\,q)$, 请注意 $\langle\boldsymbol{C}_i,\boldsymbol{s}\rangle=\boldsymbol{C}_i\boldsymbol{s}=m\boldsymbol{G}_i\boldsymbol{s}+\boldsymbol{R}_i\boldsymbol{A}\boldsymbol{s}=2^{i-1}m+\boldsymbol{R}_i^{\mathrm{T}}\boldsymbol{e}$;

②输出 $\lfloor x/2^{i-1}\rceil$. 注意此处如果 $|x|<2^{i-2}\leqslant q/4$则返回 0, 如果 $|x|>2^{i-2}$则返回 1.

5)同态加法运算. $G.Add(\boldsymbol{C}_1,\boldsymbol{C}_2)$:

$$\boldsymbol{C}_1+\boldsymbol{C}_2=(m_1+m_2)\boldsymbol{G}+\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{R}_1+\boldsymbol{R}_2)\in\mathbb{Z}_q^{(n+1)\times M}.$$

6)同态乘法运算. $G.Mult(\boldsymbol{C}_1,\boldsymbol{C}_2)$:

计算 $\boldsymbol{G}^{-1}(\boldsymbol{C}_2)\in\{0,1\}^{M\times M}$, 并输出 $\boldsymbol{C}_1\boldsymbol{G}^{-1}(\boldsymbol{C}_2)$. 注意此处有

$$\begin{aligned}\boldsymbol{C}_1\boldsymbol{G}^{-1}(\boldsymbol{C}_2)&=\left(m_1\boldsymbol{G}+\boldsymbol{A}^{\mathrm{T}}\boldsymbol{R}_1\right)\boldsymbol{G}^{-1}(\boldsymbol{C}_2)=\\&m_1\boldsymbol{C}_2+\boldsymbol{A}^{\mathrm{T}}\boldsymbol{R}_1\boldsymbol{G}^{-1}(\boldsymbol{C}_2)=\\&m_1m_2\boldsymbol{G}+\boldsymbol{A}^{\mathrm{T}}\left(m_1\boldsymbol{R}_2+\boldsymbol{R}_1\boldsymbol{G}^{-1}(\boldsymbol{C}_2)\right)\in\mathbb{Z}_q^{(n+1)\times M}.\end{aligned}$$

2014 年, DM14[62](即 FHEW 方案)将 AP14[60]提出的自举方法有效实例化, 自举时间降到 1 s 以下[63]. 2016 年, GINX16[64]提出内存需求更小的自举方法, 随后基于环面(torus)的变体方案 CGGI16[65-67](即 TFHE 方案)将自举方法实例化, 自举时间更是降到 0.1 s 以下, 将第 3 代 FHE 方案研究推进高潮. 自此, AP 自举[60]和 GINX[64]自举成为第 3 代 FHE 方案的主流自举方法, 前者支持任意密钥分发, 后者只需要更小的同态计算密钥. MP21[68]对这 2 种方法进行了详细的比较, 指出当 LWE 密钥服从高斯(或均匀)分布时, AP 方法更快, 而 GINX 方法在二元 LWE 密钥的情形下领先. 随后, KDE21[69]和 BIP22[70]通过降低一半的计算量, 对 GINX 进行了改进, 但只针对三元密钥, 无法扩展至更大的密钥, 其中 KDE21 证明了此前只适用第 3 代 FHE 方案的盲旋转(rotation)技术, 也可用于第 2 代的 BGV, BFV 等方案. JP22[71]提出了一个 GINX 变种, 它是 MP21[68]和 KDE21[69]中自举方法的推广, 而

且仍对二元和三元密钥最有效. 在 CRYPTO2023, XZD23[72] 从 NTRU 假设出发, 基于类 GSW 方案提出一个新的盲旋转算法, 进而构造出一个性能更优(耗时 112 ms), 且同态计算密钥更小的自举算法. 在 EUROCRYPT2023, YDA23[73] 针对类 FHEW 方案提出的新自举方法兼具 AP 和 GINX 这 2 种主流自举方法的优点, 在运行时间和密钥大小方面较已知类 FHEW 自举方法效果更佳(PALISADE/OpenFHE 实现), 且噪声增长更小.

第 3 代 FHE 方案延续了第 2 代 FHE 方案基于 (R)LWE 的安全性假设, 但较第 2 代 FHE 方案有 3 点改进: 1)同态运算更加高效(为矩阵的加、乘运算), 且不会引起密文维度的膨胀, 无需借助其他技术来控制密文维度的增长; 2)密文噪声的控制更加简洁、有效, 无需借助模交换技术实现; 3)摆脱了对计算公钥的依赖, 且自举性能总体上较第 2 代 FHE 方案更加简洁且有极大提升. 但第 3 代 FHE 方案的局限性在于: 1)不支持 SIMD 同态操作; 2)自举实现依赖的安全假设强度较第 2 代有所弱化, 但困难问题的近似因子仅为维度 n 的多项式[56]; 3)和第 2 代 FHE 方案一样, 若要实现全同态仍依赖自举技术.

3.5 支持浮点数运算的第 4 代 FHE 方案

现实中大多运算涉及实数或复数(计算机中用浮点数表示), 其运算结果往往只需保留一部分有效数字来满足精度要求即可, 前 3 代 FHE 方案建立在离散空间的精确算术运算上, 在处理浮点数运算及带有舍入的运算时并不适用. 第 4 代 FHE 方案的主要特点即支持浮点数同态运算. 2017 年, Cheon 等人[74] 在 ASIACRYPT2017 上提出能处理浮点数的 FHE 方案(即 CKKS 方案), 此方案加密时对明文进行舍入, 解密时输出满足精度要求的近似值, 即解密结果无需和明文完全一致; 在乘法同态运算时, 通过重缩放(rescaling)技术, 在保证计算精度的同时将密文模数增长从指数增长变为线性增长; 并通过使用批处理技术提高方案计算效率. 但该方案为 SHE 方案, 于是 Cheon 等人[75] 在 EUROCRYPT2018 基于 Gentry09[1] 的自举技术提出一种新的刷新密文的技术, 将上述的 SHE 方案扩展为 FHE 方案. 该技术用正弦函数作为模约减运算的近似, 并提出一个有效的同态计算策略使得每一次迭代只需 1 次乘法运算, 因此计算开销随解密电路的深度增大呈线性增长, 并以 SIMD 的方式实现了对封装密文的并行计算. 同年, 基于 CKKS 方案[74] 的 Full-RNS 变体(简称 RNS-CKKS)[76] 被提出, 但其采用的是文献 [75] 中的自举方法, 自举精度远达不到实际需求. 2019 年, Chen 等人[77] 又将方案每个明文槽的均摊(amortized)自举时间提升了 2 个数量级(从 1 s 降到 0.01 s). 2020 年, Han 等人[78] 基于文献 [77] 中改进的自举方法对 RNS-CKKS 方案的自举进行了改进, 通过减少密钥切换过程中临时模的数量, 实现了同等安全级别且不借助自举技术的情况下支持更深电路的同态计算; 同时提出一种新的多项式逼近方法求解加密状态下的正弦函数以实现 CKKS 的自举程序; 并基于 SHE 实现了 Full-RNS 变体的自举 FINAL 程序. KPP20[79] 提出更低误差的 RNS-CKKS 方案, 消除了加密和密钥切换操作中由于噪声引起的近似误差. 在 EUROCRYPT2021, LLLK21[80] 以及在文献 [78] 的基础上利用最优极小极大多项式逼近和反正弦函数实现了 RNS-CKKS 自举精度的提升; BMPH21[81] 则提出一个新自举方法, 针对出现的稀疏 R-LWE 密钥攻击问题, 提出一种不需要稀疏密钥的 RNS-CKKS 自举方法, 该方法的效率、准确度更优. 同时, BD21[82] 针对 CKKS 发起密钥恢复攻击, 指出 CKKS 不能充分捕获被动攻击, 攻击者可通过对解密消息进行噪声分析从而泄露密钥信息, 因此需要并提出一个比传统 IND-CPA 安全性更强的模型 ($\text{IND-CPA}^{\text{D}}$), 同时指出 CKKS 不满足 $\text{IND-CPA}^{\text{D}}$ 安全. 随后 KDE21[69] 通过引入小的重缩放误差而非 CKKS 自举过程引入的近似误差, 针对上述被动攻击提出一个性能更优更安全的自举算法. 在 ASIACRYPT-2021, Cho 等人[83] 提出一种支持近似计算的密文转换框架(即 RtF 框架), 在服务器端将对称密文转换为基于 CKKS 的同态密文, 减少了客户端的计算和通信负载(在客户端实现了 1.6 μs 的延迟和 21.7 MBps 的吞吐量, 分别比 CKKS 提升 9 085 倍和 17.8 倍)且有着更小的密文膨胀率(1.23~1.54, 至少降为 CKKS 的 1/23). 次年 EUROCRYPT2022, Ha 等人[84] 提出一种新的密文转换框架 Rubato, 该框架将噪声引入到对称密码中, 显著降低了乘法同态运算的复杂度, 同等对比 RtF 框架, 客户端和服务器端吞吐量分别提高了 22.9% 和 32.2%, 但增加了约 1.6% 的密文膨胀率; JM22[85] 通过对模函数进行一种新的低误差正弦级数逼近以实现高精度的自举程序, 并提出用泰勒级数逼近该正弦函数从而逼近模函数, 即可实现基于 CKKS 任意精度的自举; LLK22[86] 提出通过将误差方差最小化实现 SHE 的高精度自举方法. CRYPTO2022, BDM22[87] 在 BD21[82] 研究的基础上, 提出利用差分隐私保护机制对任意支持浮点运算的 FHE 方案实现 $\text{IND-CPA}^{\text{D}}$ 安全, 并基于 CKKS 方案进行了实现.

事实上, CKKS 的构造基于第 2 代的 BGV[6] 方案, 其本质的不同在于是否支持浮点数运算, 所以也有研究者认为 CKKS 属于第 2 代 FHE 方案. 第 4 代 FHE 方案由于对准确性允许一定误差, 使得 CKKS 比其他基于 (R)LWE 的 FHE 方案, 构造更加简单且性能也有很大提升. 近期的研究大多基于 CKKS 进行优化, 尤其是在自举程序的性能和精确度方面, 以及降低计算和通信开销方面. 目前, 最新的自举性能表现为每比特微秒级, 较第 1 代 FHE 方案的每比特千秒级性能提升了 9 个数量级.

3.6 其他 FHE 方案

1)基于整数的 FHE 方案

基于格的 FHE 方案理论性较强且不易被理解, 因此基于简单代数结构上的 FHE 研究似乎是必然的. 2010 年, 即首个 FHE 方案提出后的第 2 年, 基于整数的 FHE 方案[88](即 DGHV 方案)被提出, 用整数模运算代替了格上复杂的矩阵和向量运算, 该方案完全遵循 Gentry09[1] 构造思路, 其 SHE 阶段的构造依赖于 AGCD 困难假设, 及时证明了复杂的 FHE 方案也可以基于简单的代数结构构造. 随后基于整数的 DHE 方案大多基于 DGHV 构造. 2011 年, CMNT11[89] 通过利用二次型加密(quadratic encryption)代替线性加密对 DGHV 进行了优化, 其主要贡献在于将公钥尺寸由$\tilde{O}(\lambda^{10})$降低至$\tilde{O}(\lambda^{7})$. 2012 年, CNT12[90] 在 CMNT-11 的基础上进一步优化二次型加密技术, 将公钥尺寸由$\tilde{O}(\lambda^{7})$降低至$\tilde{O}(\lambda^{5})$[91], 且对 BGV[6] 中的模交换技术进行调整使其能应用于 DGHV, 使得基于整数的 SHE 构造也可不依赖自举过程. 随后, YXW12[92] 又将公钥尺寸由$\tilde{O}(\lambda^{5})$优化至$\tilde{O}(\lambda^{3})$. 2013 年, CCK13[93] 采用 GHS12[38] 中的批处理技术提出整数域上支持批处理的方案, 使得$l=\tilde{O}(\lambda^{3})$个明文比特消息可以封装单个密文中进行处理, 从而提升计算效率. 2014 年, CLT14[94] 基于 Bra12[7] 构造了整数上的标度不变 FHE 方案. 2015 年, CS15[95] 同样基于 AGCD 提出了一种安全性不再依赖 SSSP 假设的方案, 该方案性能优于此前所有的基于 AGCD 的 FHE 方案. 同年, NK15[96] 将方案的明文空间从比特空间扩展至Z_q(其中q为任意素数), 同时降低了明文空间为比特空间时同态乘的算法复杂度, 且该方案可扩展为支持批处理的方案. 2016 年, 熊婉君等人[97] 在 DGHV 方案基础上, 利用 Gentry09 的 FHE 构造基本思路, 提出一个公钥更短、效率更高的 FHE 方案, 该方案将明文空间由$\{0,1\}$扩展到$\{0,1\}^{l}$, 但仍依赖 SSSP 假设. 2018 年, 王彩芬等人[98] 将方案的参与者由“1 对 1”推广至“多对 1”, 即“多方加密, 1 方解密”, 与已有方案相比扩展了数据传输量且提高了效率. CHK19[99] 将自举过程中同态乘的执行次数从$O(\lambda^{4})$减少至$O(\log^{2}\lambda)$, 使自举过程执行效率更高. 此外, 还有一些其他的基于整数的 FHE 方案, 如文献 [100–104].

2)基于 NTRU 的 FHE 方案

由于 NTRU 加密算法密钥生成较容易, 加密、解密的速度比 RSA, ECC 等著名算法快很多, 且安全性依赖于格中 SVP 的困难性, 因此, 也被用来构造 FHE 方案(以下称为 NTRU-FHE), 但在安全性方面颇有争议. 2012 年, LTV12[105] 构建 NTRU-FHE 方案, 由于 NTRU 加密体制本身的安全性可证明要求参数的选择符合一定分布条件以实现公钥统计意义上的均匀性[106], 而该 NTRU-FHE 方案要实现同态运算且安全性可证明, 需要引入一个非标准的安全性假设, 即决策小多项式比(decisional small polynomial ratio, DSPR)假设. 随后, YASHE 方案[107] 基于 Bra12[7] 的构造思想, 消除了 LTV12 中的 DSPR 假设, 构造了一个标度不变、无需模交换操作的变体方案. 文献 [108–109] 对 LTV12 进行了优化实现. 然而, 文献 [110–111] 实现了针对 LTV12[105] 和 YASHE[107] 的密钥恢复攻击; ABD16[112] 实现了对 LTV12 和 YASHE 的子域格攻击(subfield lattice attacks). 随后, KF17[113] 指出子域格攻击是基于 NTRU 格内密集子格(sublattice)的存在而实现的, 即这类攻击基于 NTRU 格自身的结构, 不能通过将代数结构切换到另一个多项式环来解决, 且难以评估这类攻击对 NTRU 安全性的影响, 因此, 很难对基于 NTRU 的 FHE 方案的安全级别进行正确估计. 因此在 ASIACRYPT2019, GGH19[114] 针对自动机提出将多项式环切换为一个矩阵形式的环, 这个基于矩阵 NTRU 假设的 FHE 方案很快就在 IACR2020 被指出容易受到子格攻击[115]. 2020 年, 车小亮等[116] 提出一种可更好抵御子域格攻击、效率更优的 NTRU-FHE 方案, 并将现有方案底层 2 的幂次分圆多项式环扩展应用到素数次分圆多项式环上. 文献 [105–116] 方案均需要依赖一个过度延伸(overstretched)的 NTRU 参数, 因此易受子域格攻击, 在 ASIACRYPT2021, DW21[117] 针对过度延伸的 NTRU 假设, 较 KF17[113] 进行了更详细的分析, 在 ASIACRYPT2022, 在 FINAL22[118] 基于 DW21[117] 的分析, 构建了 2 个 NTRU 参数落在过度延伸范围外的 FHE 方案, 第 1 个方案完全基于 NTRU, 与包括 TFHE[64] 在内的 FHE 方案相比性能相当; 第 2 种方案基于 NTRU 和 LWE 假设, 比 TFHE[64] 的自举速度快 28%. 虽然 FINAL22[118] 对是否可基于 NTRU 构造安

全、高效的 FHE 方案给出了较为正向的回应，但“过度延伸”的 NTRU 假设仍需进一步深入分析直至被充分研究.

3.7 主流 FHE 方案对比分析

该节主要从安全性（含依赖的数学困难问题）和性能等方面对当前主流 FHE 方案进行对比分析，如表 2 所示. 其中性能部分选取了 FHE 方案中计算复杂度最高的自举过程，并分别选取自举运行时间和每比特的均摊耗时，即自举运行时间/(明文槽数量×剩余可用层级数)，为统一度量. 而第 2 代和第 4 代由于自举过程中明文槽数量和剩余可用层级数不一定相同，所以每次自举耗时（即倒数第 2 列）并不能真正衡量其性能优劣，相对而言以每比特的均摊耗时（即最后 1 列）作为衡量更为合理. 表 2 中所列实验数据基于当前公开文献整理，其中第 4 代的数据均是基于 CKKS 方案的实验数据.

Table 2 Comparative Analysis of Main FHE Schemes

表 2 主流 FHE 方案对比分析

阶段	FHE 方案	安全性	依赖困难问题	自举性能	
				单次自举耗时/s	均摊耗时/(ms/bit)
第 1 代	Gentry09[1]	不可证明	CVP 问题和 SSSP 问题	1 800	/
第 2 代	BGV12[6,119]	CPA	R-LWE 问题	11	0.9
	GIK23[53]	CPA	R-LWE 问题	70	1.1
第 3 代	FHEW14[62]	CPA	R-LWE 问题	<1	/
	TFHE16[67]	CPA	R-LWE 问题	<0.1	/
	XZD23[72]	CPA	R-LWE 问题	0.112	/
	YDA23[71]	CPA	R-LWE 问题	0.08	/
第 4 代	CCS19[77,120]	CPA	R-LWE 问题	78.7	0.95
	HK20[121]	CPA	R-LWE 问题	52.8	0.46
	JKA21[122]	CPA	R-LWE 问题	135.4	0.11(1-CPU)
				0.527	423×10^{-6}(1-GPU)

注：/表示不支持 SIMD.

当前，在安全性方面，大多数 FHE 方案依赖于格上的困难问题，由于 FHE 具备密文同态运算的属性，同态特性意味着延展性（malleability），因此 FHE 体制不可能抵抗适应性选择密文攻击，即不可能达到 IND-CCA2 安全[46]. 目前大多数 FHE 方案只给出了抵抗适应性选择明文攻击的证明，即达到 IND-CPA 安全. 在性能方面，较第 1 代 FHE 方案已有质的提升，尤其是基于 GPU 的实现，但相较于明文运算，基于 FHE 的密文运算在大规模推广应用过程中性能还需进一步提升.

4 全同态加密应用进展

4.1 算法库实现

从第 2 代 FHE 方案开始，即有了相应的算法实现库，主流算法库及其支持的 FHE 方案如表 3 所示.

表 3 中，cuFHE 库支持 GPU 加速，可比使用 CPU（实现 TFHE）快 20 多倍；Concrete 库是实现了 TFHE 方案[64]的一个变体方案（增强版），而 TFHE 库是对 TFHE 原始方案的实现；Pyfhel 是一个 Python 版本的同态算法库，其基于 SEAL/PALISADE/HElib 等 C++库开发，当前版本只支持 SEAL 库；PALISADE 由美国防部高级研究计划局（DARPA）资助建立，由政府注册的非盈利组织 NumFOCUS 于 2019 年开始赞助，后逐步迁移至 OpenFHE 开源项目. OpenFHE 提供更简单的 API、模块化实现、跨平台支持和硬件加速器集成，整合了多个现有 FHE 算法库的设计理念，同时不断引入新的设计思想，由来自 PALISADE, HElib, HEAAN, FHEW 库的作者设计，目前已支持所有主流的 FHE 方案，包括 BGV[6], BFV[47], CKKS[74], FHEW[62], TFHE[64] 方案，且支持多种自举程序设计；OpenFHE 项目得到了 DARPA 的支持，由 NumFOCUS 直接赞助，且定期举办关于 FHE 和 OpenFHE 的网络研讨会. Google 公司更是开源了首个通用 FHE 的转译器（transpiler），可以将普通的 C++程序转译为基于 TFHE 同态库的同态程序，将明文运算转换为了同态密文运算.

Table 3 Main FHE Algorithm Libraries and Their Supported Schemes of FHE

表 3 主流 FHE 算法库及其支持的 FHE 方案

算法库	方案				
	BGV	BFV	FHEW	TFHE	CKKS
TFHE[123]		√	√	√	√
FHEW[124]			√		
HElib[125]	√				√
SEAL[126]	√	√			√
cuFHE[127]				√	
HEAAN[128]					√
Lattigo[129]	√	√			√
PALISADE[130]	√	√	√	√	√
Concrete[131]				√	
Pyfhel[132]	√	√			√
OpenFHE[133]	√	√	√	√	√
HEhub[134]	√			√	√

此外，还有诸如基于 GPGPU 实现 YASHE[107] 方案的 cuYASHE、基于 FINAL22[118] 实现的 FINAL 库、基于大整数实现的 libScarab、基于 GPU 实现 TFHE 的 NuFHE，以及提供 FHE 算法分布式数据流的 Spark-FHE 等开源实践.

现有 FHE 算法库大多基于 C++，可在 Linux，MacOS 或 Windows 操作系统环境执行；大多支持第 4 代的 CKKS(及其变体)方案；且基本由欧美国家主导实现，如表 4 所示.

4.2 标准化进展

2017 年 7 月，首个专注于 FHE 标准化的自发性组织——同态加密标准化开放联盟①成立，该组织由来自企业、学术界和政界的相关领域研究人员共同发起. 目前，该联盟已举办 5 届 FHE 标准化会议，参与成员包括微软、三星 SDS、英特尔、Duality、IBM、谷歌、万事达卡、阿里巴巴集团等企业，以及 NIST，ITU 等机构的代表和波士顿大学、麻省理工学院等各大高校的学者. 其中，首届 FHE 标准化研讨会于联盟成立之初在微软研究院举办，开启了 FHE 标准草案的编写工作，并发布了全同态加密安全标准、API 标准、应用标准 3 份白皮书，这 3 份白皮书也是后来发布的 FHE 标准草案的主要参考依据.

FHE 标准草案于 2018 年 3 月发布，并于 11 月更新，目前最新版本为 1.1，即 11 月的更新版本. 草案的内容主要包括同态加密方案的标准化和安全参数 2 个部分：第 1 部分主要对 BGV[6]，BFV[47] 和 GSW[57] 推荐的 3 个 FHE 方案进行了规范描述，其中还描述了一些基本功能函数(如参数生成、公私密钥及计算密钥生成、加密、解密、同态加、同态乘、密文刷新、有效性校验等)的定义，以及语义安全(IND-CPA)、紧致性、有效解密等算法性质，并对分布式加密、主动攻击、同态计算隐私、密钥演化、侧信道攻击、基于身份的加密也有一定的讨论；第 2 部分主要针对 LWE 和 R-LWE 困难问题给出安全参数建议，其中还分析了针对这 2 个困难问题的多种攻击算法，包括著名的 BKZ 算法[135].

Table 4 Comparison Table of Other Information of Each FHE Algorithm Library

表 4 各 FHE 算法库其他信息对照表

算法库	开发语言	OS 环境	发布年份	开发者/组织
TFHE	C++，C	Linux，MacOS	2016	Nicolas，Gama
FHEW	C++	Linux，MacOS，Windows	2017	Leo，Ducas 等
HElib	C++	Ubuntu，MacOS	2018	IBM
SEAL	C++	Linux，MacOS，Windows，FreeBSD，Android	2018	微软
cuFHE	Cuda，C++，python	Ubuntu，MacOS，Windows	2018	Gizemscetin 等
HEAAN	C++	Ubuntu	2018	三星
Lattigo	Go	Linux，FreeBSD，MacOS，Windows	2019	EPFL 实验室
PALISADE	C++	Ubuntu，CentOS，Windows	2019	Duality
Concrete	Rust	Linux，MacOS，WSL	2020	Zama
Pyfhel	Python，Cython	Linux，WSL	2021	Ibarrondo 等
OpenFHE	C++	Linux，MacOS，Windows	2022	Duality 主导
HEhub	C++	Linux，MacOS	2022	原语科技

4.3 典型应用场景

1) 技术赋能应用

① FHE 在云计算中的应用. 在云服务模式下，加密数据很难甚至无法被云应用或程序处理，所以目前大多数据并未加密存储，其安全性和隐私问题面

① https://homomorphicencryption.org/

临巨大挑战. FHE 技术可在不泄露敏感信息的前提下完成对云端密态数据的处理, 如私有信息检索、隐私保护数据挖掘、加密数据检索等, 而云服务商等不受信任的一方则无法获得明文或用户隐私信息, 即云计算的对象及结果全程加密, 即便 IT 基础设施遭遇攻击, 也可在不依赖外围防御的情况下保障数据不被泄露. 2020 年 DARPA 启动基于 FHE 硬件加速的“虚拟环境数据保护(data protection in virtual environments)”计划 DPRIVE, 英特尔公司、微软公司等参与其中; 12 月, IBM Security 宣布推出一项基于 FHE 的云计算新服务, 为用户设计、提供基于 FHE 的云计算安全解决方案. 此外, 亚马逊 AWS 云服务利用 FHE 对 DynamoDB 中存储的数据进行密态处理.

② FHE 在大数据中的应用. 在数据要素市场化配置趋势下, 多来源数据的计算场景增多, 如何在保证数据机密性的基础上实现数据的流通共享和合作开发利用, 成为大数据应用面临的基本挑战. 此外, 大数据中的计算形式复杂、多样, 特定计算场景下的隐私保护算法通常不能满足各类大数据应用需求, 因此需要一个较为通用的方案从本质上解决大数据应用隐私保护的问题. FHE 的出现为解决以上需求提供了有效思路, 如加密数据在进行多方安全计算时, 需要在聚合节点先解密, 计算完成后再加密传输, 实现了数据传输的安全, 但是不能保证数据计算的安全. 引入 FHE 后, 可实现明文数据及其衍生计算结果全程以密态形式呈现, 同时又能得到大数据分析得来的有价值的结果.

③ FHE 在人工智能中的应用. 人工智能以数据为“燃料”, 如何实现在保护数据安全和隐私的基础上, 尽可能释放人工智能所有功能是很有价值的问题. 如不同行业、部门甚至组织内部不同业务线之间往往因数据安全或隐私顾虑形成数据壁垒, 所以只能依赖有限数量或者较单一的数据进行独立训练, 导致无法建模或模型无法趋于全局最优. 而 FHE 的引入可以缓解这种尴尬, 如在多方参与的机器学习中, 支持对多来源的基础数据和模型加密后, 一直以密文形态参与学习或其他处理, 且处理结果仍然保密, 由此实现基于 FHE 的隐私保护机器学习. 英特尔公司是这方面的主要实践者之一, 并与微软公司合作于 2018 年 12 月发布基于 SEAL 算法库的开源版同态加密转换器(HE-transformer), 支持人工智能系统对敏感数据进行操作, 可应用于如谷歌公司 TensorFlow, Facebook PyTorch 等开源框架上的神经网络实现, 也是英特尔神经网络编译器 nGraph 的同态加密后端.

④ FHE 在区块链中的应用. 由于调用智能合约时使用的参数全集需明文公开, 否则其他节点无法验证合约执行结果是否正确, 如此不能保证合约调用者合约信息的隐私性, 而许多智能合约的应用场景都有隐私保护的需求, 如电子竞选/选举投票、多方参与的竞拍竞价、医疗健康合约等, 信息持有者通过调用智能合约提供相关信息时会有顾虑. 引入 FHE 后, 可在数据上链前调用 FHE 算法库, 将数据加密后再上链, 链上的密文数据也可通过调用 FHE 算法库实现支撑同态运算的智能合约, 合约执行结果仍是密文, 返至业务层后再通过 FHE 算法库进行解密[56].

2) 行业场景应用

①金融-信贷风控. 信贷风控可能需要来自央行征信数据、信用卡数据、人社部参保数据、教育部学籍数据、房管局房屋不动产信息以及法院黑名单信息等外部数据进行综合评估, 而外部合作方和信贷风控方因数据合规要求或其他安全考虑, 一般不会把自己掌握的数据传送给对方. 传统的处理方式即合作方独立建模, 并将建模结果进行融合, 此方式虽然保护了隐私, 但建模效果较差可能导致评估并不准确, FHE 的引入或可促进合作方之间的数据共享流通. 此外, 还可采用纵向联邦学习进行共同建模从而实现更精准的风控, 但要实现联邦学习过程中中间结果或参数的安全传输和计算, 可引入 FHE 保障联邦学习过程的数据隐私性, 从而保障信贷风控中多个参与方的数据安全.

② 互联网-私有信息处理. 首先是私有信息检索(PIR), 当进行一些敏感信息检索时, 若不想让服务器知道自己想从数据库中检索什么信息, 可借助 FHE 实现私有信息检索, 即存储和查询都使用加密数据, 检索结果也是加密的, 由服务器返回用户后用户解密获得检索结果; 其次是隐私集合求交(PSI), 多个参与方各自持有私有数据, 可通过 FHE 实现多个参与方数据求交, 而不泄露交集外的其他任何信息, 已在纵向联邦学习、好友发现、计算广告投放实际效果等方面有实际应用. 此外, 人脸、指纹、视网膜等个人生物特征数据等具有唯一性且不可再生的个人数据, 在互联网中常被用于身份识别和认证, 一旦泄露无法追回并变更, 将会带来不可逆的隐私保护问题, 可借助 FHE 实现加密存储和密态处理, 在不泄露隐私的情况下依然可以进行生物特征识别与认证.

③医疗-基因数据分析. 大多国家或地区对基因

数据共享有着较为严格的管控,而在进行生物学、复杂疾病、流行病学等方面的研究时往往需要大规模的基因数据样本,而 FHE 支持对密态基因数据进行有意义的操作,一方面可以促进基因数据的开放共享,另一方面可用于实现不同参与方之间的联邦学习、多方安全计算等,促进基因组学领域的合作与研究突破. 2014 年美国国立卫生研究院布局基因组学领域的数据隐私计算,并发起世界顶级隐私计算赛事 iDASH,每年均有 FHE 相关基因数据安全分析赛题.

④能源-电网数据分析. 电网数据被篡改可能导致电网控制和配电失灵,造成大规模停电事件. 专门的监控机构将监控智能电网中每个节点产生的数据,且可能会将监控和计算工作外包给云平台,但前提是进行电网数据监控和计算的云平台本身应是安全、可信的. 为对电网数据实现基于云平台的安全监控和计算,可利用 FHE 将智能电网中每个节点的测量值加密后发至云平台,云平台直接对密文数据进行处理而无需提前解密,处理结果以密文形式发送至电网决策中心(解密后)进行分析决策,攻击者即便窃取了数据也无法知道内容是什么,即便对数据进行篡改、伪造等操作也可被检测发现.

⑤政府-电子选举. 基于 FHE 设计的电子选举方案允许计票方在不掌握投票者投票内容的前提下对投票结果进行统计,也允许参与投票的公民确认他们的投票已被计数且未被更改,既保护了投票者的隐私,又保证了投票结果的公正. 2019 年,微软公司于 Build 开发者大会上发布了一款基于同态加密的开源电子投票开发包 Electionguard[①],支持对选举过程进行端到端验证,投票结果可在网上公布,也支持第三方机构进行验证,并允许选民确认他们的选票被正确计算. 此外,FHE 也可应用于政务数据的开放共享,以支持政务数据参与要素化配置,释放价值潜能.

当然,FHE 在这些行业领域还有其他应用,如金融行业基于客户多维信息的精准营销,医疗行业基于个人医疗及相关数据的个性化治疗等,此外在教育、工业、电信等行业领域也有应用场景. 无需回避的是,FHE 在应用推广中还面临一些问题,如虽然 FHE 发展迅速,但性能问题暂时依然是影响其大规模推广应用的主要因素;部分应用场景可能只涉及简单的数据操作,此时或许半同态(加同态或乘同态)是一种更好的选择,前提是需要事先了解涉及哪些类型的计算.

5 总 结

综上所述,针对 FHE 的研究发展迅速,效率方面有了质的突破,在各行业领域的应用也在逐步拓展. 首先介绍相关概念定义,并从自举、同态解密切入阐述 FHE 方案的一般构造思想. 其次分析讨论不同阶段 FHE 的研究进展及局限性. 最后从算法库实践、标准化进展、典型应用等方面介绍 FHE 的应用进展. 随着数据安全和隐私保护需求的多样化、专业化,对相关技术各方面的能力也提出了更高要求,结合 FHE 方案的研究进展,建议未来相关的主要研究方向关注 4 个方面:

1) 自举程序性能进一步优化

自举技术是当前将 SHE 方案转变为 FHE 方案的关键技术,也是唯一途径. 但由于过程中频繁调用同态解密,其成为制约现有 FHE 方案性能的主要因素之一,虽然经过 10 余年的发展,自举程序的执行效率较首次提出已有质的提升,但在市场推广应用过程中,性能问题往往是用户选择其他替代方案的主要原因. 当然,FHE 的性能还有其他影响因素,如是否可以进一步减缓密文噪声膨胀的速度以实现更低频度的自举程序调用,以及每次自举程序调用可以“支撑”更深电路的同态运算;是否支持多比特处理以及利用 FHE 专用硬件加速等.

2) FHE 自身安全性可证明

现代密码学以可证明安全为基石,由于 FHE 具备密文同态运算的属性,同态特性意味着延展性(malleability),因此 FHE 体制不可能达到 IND-CCA2 安全. 目前,大多 FHE 方案只能给出 IND-CPA 级别的安全性证明,能证明达到 IND-CCA1 级别的 FHE 方案少见[46]. 此外,在 FHE 方案发展的过程中,也出现了针对现有方案的攻击,如密钥恢复攻击、侧信道攻击. 尤其是基于 NTRU 的 FHE,在实现全同态的过程中,其“过度延伸”的安全假设还需进一步深入分析.

3)FHE 方案间互相切换支持

如用户当前使用第 3 代 FHE 方案逐比特处理数据,由于需求变化需要采用支持浮点数运算的第 4 代 FHE 方案,如何透明、安全、高效地实现 2 个方案间的切换是个值得深入研究的问题. 事实上,不只在

① https://github.com/ microsoft/electionguard

不同应用阶段、不同应用场景，应用方都可能会部署应用不同的 FHE 方案，可能会面临频繁的不同方案下的密文间互操作，可以探索研究支持不同方案间的互切换的方法，即支持在密文上使用同态解密函数，将某个方案的密文转换为另一个方案的密文，从而实现 2 个方案间的切换.

4）全新的 FHE 构造思路

距首个 FHE 方案提出仅有 10 余年，因此 FHE 研究大有可为，可以跳出现有构造思想，基于不同的困难问题假设、代数结构、数学性质等提出全新的构造思路，虽然难度非同一般.

作者贡献声明：白利芳提出了论文整体架构和思路，并负责主要研究内容的撰写和修订；祝跃飞审阅论文的整体架构和思路，并提出指导意见；李勇军负责论文部分内容的撰写和修订；王帅负责论文图表、文献的整理，以及论文整体的检查和修订；杨晓琪负责论文部分内容的撰写和修订.

参 考 文 献

[1] Gentry C. Fully homomorphic encryption using ideal lattices[C]// Proc of the 41st Annual ACM Symp on Theory of Computing. New York: ACM, 2009: 169–178

[2] Paillier P. Public-key cryptosystems based on composite degree residuosity classes[G]//LNCS 1592: Advances in Cryptology (EUROCRYPT'99). Berlin: Springer, 1999: 223–238

[3] Rivest R L, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems[J]. Communications of the ACM, 1978, 21(2): 120−126

[4] Regev O. On lattices, learning with errors, random linear codes, and cryptography[C]//Proc of the 37th ACM Symp on Theory of Computing. New York: ACM, 2005: 84–93

[5] Rothblum R. Homomorphic encryption: From private-key to public-key[G]//LNCS 6597: Proc of the 8th Conf on Theory of Cryptography. Berlin: Springer, 2011: 219–234

[6] Brakerski Z, Gentry C, Vaikuntanathan V. (Leveled) Fully homomorphic encryption without bootstrapping[C]//Proc of the 3rd Innovations in Theoretical Computer Science Conf. New York: ACM, 2012: 309–325

[7] Brakerski Z. Fully homomorphic encryption without modulus switching from classical GapSVP[G]//LNCS 7417: Advances in Cryptology (CRYPTO 2012). Berlin: Springer, 2012: 868–886

[8] Rabin M O. Digitalized signatures and public-key functions as intractable as factorization, MIT-LCS-TR-212[R]. Cambridge, MA: MIT, 1979

[9] Elgamal T. A public key cryptosystem and a signature scheme based on discrete logarithms[J]. IEEE Transactions on Information Theory, 1985, 31(4): 469−472

[10] Goldwasser S, Micali S. Probabilistic encryption and how to play mental poker keeping secret all partial information[C]//Proc of the 14th Annual ACM Symp on Theory of Computing. New York: ACM, 1982: 365–377

[11] Benaloh J. Verifiable secret-ballot elections[D]. New Haven, CT: Yale University, 1987

[12] Okamoto T, Uchiyama S. A new public-key cryptosystem as secure as factoring[G]//LNCS 1403: Advances in Cryptology (EUROCRYPT'98). Berlin: Springer, 1998: 308–318

[13] Naccache D, Stern J. A new public key cryptosystem based on higher residues[C]//Proc of the 5th ACM Conf on Computer and Communications Security. New York: ACM, 1998: 56−66

[14] Damgård I, Jurik M. A length-flexible threshold cryptosystem with applications[C]//Proc of the 8th Australasian Conf on Information Security and Privacy. Berlin: Springer, 2003: 350–364

[15] Goldwasser S, Kharchenko D. Proof of plaintext knowledge for the Ajtai-Dwork cryptosystem[C]//Proc of the 2nd Int Conf on Theory of Cryptography. Berlin: Springer, 2005: 529–555

[16] Kawachi A, Tanaka K, Xagawa K. Multi-bit cryptosystems based on lattice problems[C]//Proc of the 10th Int Conf on Practice and Theory in public-Key Cryptography. Berlin: Springer, 2007: 315–329

[17] Melchor C A, Castagnos G, Gaborit P. Lattice-based homomorphic encryption of vector spaces[C]//Proc of the 2008 IEEE Int Symp on Information Theory. Piscataway, NJ: IEEE, 2008: 1858–1862

[18] Peikert C, Waters B. Lossy trapdoor functions and their applications[J]. SIAM Journal on Computing, 2007, 40(6): 1803−1844

[19] Armknecht F, Sadeghi A R. A new approach for algebraically homomorphic encryption[EB/OL]. 2008[2022-07-21].https://eprint.iacr.org/2008/422

[20] Sander T, Young A, Yung M, et al. Non-interactive crypto computing for NC1[C]//Proc of the 40th Annual Symp on Foundations of Computer Science. Piscataway, NJ: IEEE, 2001: 554−566

[21] Ishai Y, Paskin A. Evaluating branching programs on encrypted data[C]//Proc of the 4th Theory of Cryptography Conf. Berlin: Springer, 2007: 575–594

[22] Boneh D. Evaluating 2-DNF formulas on ciphertexts[G]//LNSC 3378: Proc of the 2nd Theory of Cryptography Conf. Berlin: Springer, 2005: 325–341

[23] Melchor C A, Gaborit P, Herranz J. Additively homomorphic encryption with t-operand multiplications[G]//LNSC 6223: Advances in Cryptology (CRYPTO 2010). Berlin: Springer, 2010: 138–154

[24] Fellows M, Koblitz N. Combinatorial cryptosystems galore[J]. Contemporary Mathematics, 1994, 168: 51−61

[25] Grigoriev D, Ponomarenko I. Homomorphic public-key cryptosystems and encrypting Boolean circuits[J]. Applicable Algebra in Engineering, Communication and Computing, 2006, 17(3): 239−255

[26] Steinwandt R, Geiselmann W. Cryptanalysis of polly cracker[J].

IEEE Transactions on Information Theory, 2002, 48(11): 2990−2991

[27] Choi S J, Blackburn S R, Wild P R. Cryptanalysis of a homomorphic public-key cryptosystem[J]. Journal of Mathematical Cryptology, 2007, 1(4): 351−358

[28] Ferrer J. A new privacy homomorphism and applications[J]. Information Processing Letters, 1996, 60(5): 277−282

[29] Domingoferrer J. A provably secure additive and multiplicative privacy homomorphism[G]//LNCS 2433: Proc of the 6th Int Conf on Information Security. Berlin: Springer, 2002: 471−483

[30] Wagner D. Cryptanalysis of an algebraic privacy homomorphism[G]//LNCS 2851: Proc of the 6th Int Conf on Information Security. Berlin: Springer, 2003: 234–239

[31] Cheon J H, Kim W H, Nam H S. Known-plaintext cryptanalysis of the Domingo-Ferrer algebraic privacy homomorphism scheme[J]. Information Processing Letters, 2006, 97(3): 118−123

[32] Gentry C. Toward basing fully homomorphic encryption on worst-case hardness[G]//LNCS 6223: Advances in Cryptology (CRYPTO 2010). Berlin: Springer, 2010: 116–137

[33] Stehlé D, Steinfeld R. Faster fully homomorphic encryption[G]//LNCS 6477: Advances in Cryptology (ASIACRYPT 2010). Berlin: Springer, 2010: 377–394

[34] Ogura N, Yamamoto G, Kobayashi T, et al. An improvement of key generation algorithm for Gentry's homomorphic encryption scheme[G]//LNCS 6434: Proc of the 5th Int Workshop on Security. Berlin: Springer, 2010: 70−83

[35] Scholl P, Smart N P. Improved key generation for Gentry's fully homomorphic encryption scheme[G]//LNCS 7089: Proc of the 13th IMA Int Conf on Cryptography and Coding. Berlin: Springer, 2011: 10−22

[36] Gentry C, Halevi S. Implementing Gentry's fully-homomorphic encryption scheme[G]//LNCS 6632: Advances in Cryptology (EUROCRYPTO 2011), Berlin: Springer, 2011: 129–148

[37] Gentry C, Halevi S. Fully homomorphic encryption without squashing using depth-3 arithmetic circuits[C]//Proc of the 52nd IEEE Symp on Foundations of Computer Science. Piscataway, NJ: IEEE, 2011: 107−109

[38] Gentry C, Halevi S, Smart N P. Better bootstrapping in fully homomorphic encryption[G/OL]//LNCS 7293: Proc of the 15th Int Conf on Practice and Theory in Public Key Cryptography. Berlin: Springer, 2012[2022-09-02].https://doi.org/10.1007/978-3-642-30057-8_1

[39] Smart N P, Vercauteren F. Fully homomorphic encryption with relatively small key and ciphertext sizes[G]//LNCS 6056: Proc of the 13th Int Conf on Practice and Theory in Public Key Cryptography. Berlin: Springer, 2010: 420−443

[40] Smart N P, Vercauteren F. Fully homomorphic SIMD operations [EB/OL]. 2011[2022-08-01].https://eprint.iacr.org/2011/133

[41] Mikuš M. Experiments with the plaintext space in Gentry's somewhat homomorphic scheme[J]. Tatra Mountains Mathematical Publications, 2012, 53(1): 147−154

[42] Brakerski Z, Vaikuntanathan V. Fully homomorphic encryption from ring-LWE and security for key dependent messages[G]//LNCS 6841: Advances in Cryptology(CRYPTO 2011). Berlin: Springer, 2011: 505–524

[43] Lyubashevsky V, Peikert C, Regev O. On ideal lattices and learning with errors over rings[G/OL]//LNCS 6110: Advances in Cryptology (EUROCRYPT 2010). Berlin: Springer, 2010[2022-09-02].https://doi.org/10.1007/978-3-642-13190-5_1

[44] Brakerski Z, Vaikuntanathan V. Efficient fully homomorphic encryption from (standard) LWE[C]//Proc of the 52nd IEEE Annual Symp on Foundations of Computer Science. Piscataway, NJ: IEEE, 2011: 97−106

[45] Brakerski Z, Langlois A, Peikert C, et al. Classical hardness of learning with errors[C]//Proc of the 45th ACM Symp on Theory of Computing. New York: ACM, 2013: 575−584

[46] Li Zengpeng, Ma Chunguang, Zhou Hongsheng. Overview on fully homomorphic encryption[J]. Journal of Cryptologic Research, 2017, 4(6): 561−578 (in Chinese)
(李增鹏, 马春光, 周红生. 全同态加密研究[J]. 密码学报, 2017, 4(6): 561−578)

[47] Fan J, Vercauteren F. Somewhat practical fully homomorphic encryption[EB/OL]. 2012[2022-08-01].https://eprint.iacr.org/2012/144

[48] Wu Ting, Wang Hui, Liu Youping. et al. Optimizations of Brakerski's fully homomorphic encryption scheme[C]//Proc of the 2nd Int Conf on Computer Science and Network Technology. Piscataway, NJ: IEEE, 2012: 2000−2005

[49] Zhang Xiaojin, Xu Chunxiang, Jin Chunhua, et al. Efficient fully homomorphic encryption from RLWE with an extension to a threshold encryption scheme[J]. Future Generation Computer Systems, 2014, 36: 180−186

[50] Li Zengpeng, Ma Chunguang, Gang D U, et al. Dual LWE-based fully homomorphic encryption with errorless key switching[C]//Proc of the 22nd IEEE Int Conf on Parallel and Distributed Systems. Piscataway, NJ: IEEE, 2016: 1169−1174

[51] Bajard J C, Eynard J, Hasan M A, et al. A full RNS variant of FV like somewhat homomorphic encryption schemes[G]//LNCS 10532: Proc of the 23rd Int Conf on Selected Areas in Cryptography. Berlin: Springer, 2016: 423−446

[52] Zhao Xiufeng, Fu Yu, Song Weitao. Circular secure homomorphic encryption scheme[J]. Journal of Computer Research and Development, 2020, 57(10): 2117−2124 (in Chinese)
(赵秀凤, 付雨, 宋巍涛. 循环安全的同态加密方案[J]. 计算机研究与发展, 2020, 57(10): 2117−2124)

[53] Geelen R, Iliashenko I, Kang J, et al. On polynomial functions modulo PE and faster bootstrapping for homomorphic encryption [G]//LNCS 14006: Advances in Cryptology (EUROCRYPT 2023). Berlin: Springer, 2023: 257−286

[54] Brakerski Z, Gentry C, Halevi S. Packed ciphertexts in LWE-based homomorphic encryption[G/OL]//LNCS 7778: Proc of the 16th Int Conf on Practice and Theory in Public-key Cryptography. Berlin: Springer, 2012[2022-09-02].https://doi.org/10.1007/978-3-642-36362-7_1

[55] Peikert C, Vaikuntanathan V, Waters B. A framework for efficient and composable oblivious transfer[G]//LNCS 5157: Advances in Cryptology (CRYPTO 2008). Berlin: Springer, 2008: 554–571

[56] Zhong Yantao, Jiang Lin, Fang Junbin, et al. The Principle and Algorithm of homomorphic Cryptography[M]. Beijing: China Machine Press, 2022 (in Chinese)
(钟焰涛, 蒋琳, 方俊彬, 等. 同态密码学原理及算法[M]. 北京: 机械工业出版社, 2022)

[57] Gentry C, Sahai A, Waters B. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based[G]//LNCS 8042: Advances in Cryptology (CRYPTO 2013). Berlin: Springer, 2013: 75−92

[58] Brakerski Z, Vaikuntanathan V. Lattice-based FHE as secure as PKE[G/OL]//Proc of the 5th Conf on Innovations in Theoretical Computer Science. New York: ACM, 2014[2022-09-02].https://doi.org/10.1145/2554797.2554799

[59] Barrington D A. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1[J]. Journal of Computer and System Sciences, 1989, 38(1): 150−164

[60] Alperin-Sheriff J, Peikert C. Faster bootstrapping with polynomial error[G]//LNCS 8616: Advances in Cryptology (CRYPTO 2014). Berlin: Springer, 2014: 297−314

[61] Micciancio D, Peikert C. Trapdoors for lattices: Simpler, tighter, faster, smaller[G]//LNCS 7237: Advances in Cryptology (EUROCRYPT 2012). Berlin: Springer, 2012: 700−718

[62] Garg S, Gupta D. Efficient round optimal blind signatures[G]//LNCS 8441: Advances in Cryptology (EUROCRYPT 2014). Berlin: Springer, 2014: 477–495

[63] Ducas L, Micciancio D. FHEW: Bootstrapping homomorphic encryption in less than a second[G]//LNCS 9056: Advances in Cryptology (EUROCRYPT 2015). Berlin: Springer, 2015: 617−640

[64] Gama N, Izabachène M, Nguyen P Q, et al. Structural lattice reduction: Generalized worst-case to average-case reductions and homomorphic cryptosystems[G]//LNCS 9666: Advances in Cryptology (EUROCRYPT 2016). Berlin: Springer, 2016: 528−558

[65] Chillotti I, Gama N, Georgieva M, et al. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds[G]//LNCS 10031: Advances in Cryptology (ASIACRYPT 2016). Berlin: Springer, 2016: 3−33

[66] Chillotti I, Gama N, Georgieva M, et al. Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE[G]//LNCS 10624: Advances in Cryptology (ASIACRYPT 2017). Berlin: Springer, 2017: 377−408

[67] Chillotti I, Gama N, Georgieva M, et al. TFHE: Fast fully homomorphic encryption over the torus[J]. Journal of Cryptology, 2020, 33(1): 34−91

[68] Micciancio D, Polyakov Y. Bootstrapping in FHEW-like cryptosystems[C]//Proc of the 9th Workshop on Encrypted Computing & Applied homomorphic Cryptography, New York: ACM, 2021: 17−28

[69] Kim A, Deryabin M, Eom J, et al. General bootstrapping approach for RLWE-based homomorphic encryption[EB/OL]. 2021[2023-07-21].https://eprint.iacr.org/2021/691

[70] Bonte C, Iliashenko I, Park J, et al. FINAL: Faster FHE instantiated with NTRU and LWE[G]//LNCS 13792: Advances in Cryptology (ASIACRYPT 2022). Berlin: Springer, 2022: 188−215

[71] Joye M, Paillier P. Blind rotation in fully homomorphic encryption with extended keys[C/OL]//Proc of the 2022 Int Symp on Cyber Security, Cryptology, and Machine Learning. Berlin: Springer, 2022[2023-07-02].https://doi.org/10.1007/978-3-031-07689-3_1

[72] Xiang Binwu, Zhang Jiang, Deng Yi, et al. Fast blind rotation for bootstrapping FHEs[EB/OL]. 2023[2023-07-21].https://iacr.org/cryptodb//data/paper.php?pubkey=33119

[73] Lee Y, Micciancio D, Kim A, et al. Efficient FHEW bootstrapping with small evaluation keys, and applications to threshold homomorphic encryption[G]//LNCS 14006: Advances in Cryptology (EUROCRYPT 2023). Berlin: Springer, 2023: 227−256

[74] Cheon J H, Kim A, Kim M, et al. Homomorphic encryption for arithmetic of approximate numbers[G]//LNCS 10624: Advances in Cryptology (ASIACRYPT 2017). Berlin: Springer, 2017: 409−437

[75] Cheon J H, Hao K, Kim A, et al. Bootstrapping for approximate homomorphic encryption[G]//LNCS 10820: Advances in Cryptology (EUROCRYPT 2018). Berlin: Springer, 2018: 360–384

[76] Cheon J H, Han K, Kim A, et al. A full RNS variant of approximate homomorphic encryption[G]//LNCS 11349: Proc of the 25th Int Conf Selected on areas in cryptography. Berlin: Springer, 2018: 347–368

[77] Chen Hao, Chillotti I, Song Yongzhu. Improved bootstrapping for approximate homomorphic encryption[G]//LNCS 11477: Advances in Cryptology (EUROCRYPT 2019). Berlin: Springer, 2019: 34−54

[78] Han K, Ki D. Better bootstrapping for approximate homomorphic encryption[G]//LNCS 12006: Topics in Cryptology (CT-RSA 2020). Berlin: Springer, 2020: 364–390

[79] Kim A, Papadimitriou A, Polyakov Y. Approximate homomorphic encryption with reduced approximation error[EB/OL]. 2020[2022-08-17].https://eprint.iacr.org/2020/1118

[80] Lee J W, Lee E, Lee Y, et al. High-precision bootstrapping of RNS-CKKS homomorphic encryption using optimal minimax polynomial approximation and inverse sine function[C]//LNCS 12696: Advances in Cryptology (EUROCRYPT 2021). Berlin: Springer, 2021: 618–647

[81] Bossuat J P, Mouchet C, Troncoso-Pastoriza J R, et al. Efficient bootstrapping for approximate homomorphic encryption with non-sparse keys[G]//LNCS 12696: Advances in Cryptology (EUROCRYPT 2021). Berlin: Springer, 2021: 618–647

[82] Li Baiyu, Micciancio D. On the security of homomorphic encryption on approximate numbers[G]//LNCS 13696: Advances in Cryptology (EUROCRYPT 2021). Berlin: Springer, 2021: 648−677

[83] Cho J, Ha J, Kim S, et al. Transciphering framework for approximate homomorphic encryption[G]//LNCS 13092: Advances in Cryptology (ASIACRYPT 2021). Berlin: Springer, 2021: 640–669

[84] Ha J, Kim S, Lee B, et al. Rubato: Noisy ciphers for approximate homomorphic encryption[G]//LNCS 13275: Advances in Cryptology (EUROCRYPT 2022). Berlin: Springer, 2022: 581–610

[85] Jutla C S, Manohar N. Sine series approximation of the mod function for bootstrapping of approximate HE[G]//LNCS 13275: Advances in Cryptology (EUROCRYPT 2022). Berlin: Springer, 2022: 491–520

[86] Lee Y, Lee J W, Kim Y S, et al. High-precision bootstrapping for approximate homomorphic encryption by error variance minimi-

zation[G]//LNCS 13275: Advances in Cryptology (EUROCRYPT 2022). Berlin: Springer, 2022: 551–580

[87] Li Baiyu, Micciancio D, Schultz M, et al. Securing approximate homomorphic encryption using differential privacy[G]//LNCS 13507: Advances in Cryptology (CRYPTO 2022). Berlin: Springer, 2022: 560−589

[88] Dijk M V, Gentry C, Halevi S, et al. Fully homomorphic encryption over the integers[G]//LNCS 6110: Advances in Cryptology (EUROCRYPT 2010). Berlin: Springer, 2010: 24–43

[89] Coron J S, Mandal A, Naccache D, et al. Fully homomorphic encryption over the integers with shorter public keys[G]//LNCS 6841: Advances in Cryptology (CRYPTO 2011). Berlin: Springer, 2011: 487–504

[90] Coron J S, Naccache D, Tibouchi M. Public key compression and modulus switching for fully homomorphic encryption over the integers[G]//LNCS 7237: Advances in Cryptology (EUROCRYPT 2012). Berlin: Springer, 2012: 446–464

[91] Coron. Coron/fhe[EB/OL].(2012-04-10)[2022-04-10]. https//github.com/Coron/fhe

[92] Yang Haomiao, Xia Qi, Wang Xiaofen, et al. A new somewhat homomorphic encryption scheme over integers[C]//Proc of the 2012 Int Conf on Computer Distributed Control and Intelligent Environmental Monitoring. Piscataway, NJ: IEEE, 2012: 61−64

[93] Cheon J H, Coron J S, Kim J. Batch fully homomorphic encryption over the integers[G]//LNCS 7881: Advances in Cryptology (EUROCRYPT 2013). Berlin: Springer, 2013: 315–335

[94] Coron J S, Lepoint T, Tibouchi M. Scale-invariant fully homomorphic encryption over the integers[G]//LNCS 8383: Proc of the 17th Int Conf on Practice and Theory in Public-key Cryptography. Berlin: Springer, 2014: 311–328

[95] Cheon J H, Stehlé D. Fully homomorphic encryption over the integers revisited[G]//LNCS 9056: Advances in Cryptology (EUROCRYPT 2015). Berlin: Springer, 2015: 513–536

[96] Nuida K, Kurosawa K. (Batch) Fully homomorphic encryption over integers for non-binary message spaces[G]//LNCS 9056: Advances in Cryptology (EUROCRYPT 2015). Berlin: Springer, 2015: 537–555

[97] Xiong Wanjun, Wei Yongzhuang, Wang Huiyong. An improved fully homomorphic encryption scheme over the integers[J]. Journal of Cryptologic Research, 2016, 3(1): 67−78 (in Chinese)
(熊婉君, 韦永壮, 王会勇. 一个基于整数的全同态加密改进方案[J]. 密码学报, 2016, 3(1): 67−78)

[98] Wang Caifen, Cheng Yudan, Liu Chao, et al. Multiple to one fully homomorphic encryption scheme over the integers[J]. Journal of Electronics & Information Technology, 2018, 40(9): 2119−2126 (in Chinese)
(王彩芬, 成玉丹, 刘超, 等. 基于整数的多对一全同态加密方案[J]. 电子与信息学报, 2018, 40(9): 2119−2126)

[99] Cheon J H, Han K, Kim D. Faster bootstrapping of FHE over the integers[G]//LNCS 11975: Proc of the 22nd Int Conf Information Security and Cryptology. Berlin: Springer, 2019: 242–259

[100] Pisa P S, Abdalla M, Duarte O. Somewhat homomorphic encryption scheme for arithmetic operations on large integers[C/OL]//Proc of the 2012 Global Information Infrastructure and Networking Symp. Piscataway, NJ: IEEE, 2012[2022-09-02].https://doi.org/10.1109/GIIS.2012.6466769

[101] Aggarwal N, Gupta C P, Sharma I. Fully homomorphic symmetric scheme without bootstrapping[C]//Proc of the Int Conf on Cloud Computing and Internet of Things. Piscataway, NJ: IEEE, 2014: 14−17

[102] Nuida K, Kurosawa K. (Batch) Fully homomorphic encryption over integers for non-binary message space[G]//LNCS 9056: Advances in Cryptology (EUROCRYPT 2015). Berlin: Springer, 2015: 537–555

[103] Cheon J H, Kim J, Lee M S, et al. CRT-based fully homomorphic encryption over the integers[J]. Information Sciences, 2015, 310(C): 149−162

[104] Benarroch D, Brakerski Z, Lepoint T. FHE over the integers: Decomposed and batched in the post-quantum regime[G]//LNCS 10175: Proc of the 20th IACR Int Conf on Practice and Theory in Public-key Cryptography. Berlin: Springer, 2017: 271–301

[105] López-Alt A, Tromer E, Vaikuntanathan V. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption[C]//Proc of the 44th ACM Symp on Theory of Computing. New York: ACM, 2012: 1219–1234

[106] Stehlé1 D, Steinfeld R. Making NTRU as secure as worst-case problems over ideal lattices[G]//LNCS 6632: Advances in Cryptology (EUROCRYPT 2011). Berlin: Springer, 2011: 27–47

[107] Bos J W, Lauter K, Loftus J, et al. Improved security for a ring-based fully homomorphic encryption scheme[G]//LNCS 8308: Proc of the 14th IMA Int Conf on Cryptography and Coding. Berlin: Springer, 2013: 45–64

[108] Doröz Y, Hu Yin, Sunar B. Homomorphic AES evaluation using the modified LTV scheme[J]. Designs, Codes and Cryptography, 2016, 80(2): 333−358

[109] Li Zichen, Zhang Juanmei, Yang Yatao, et al. A fully homomorphic encryption scheme based on NTRU[J]. Acta Electronica Sinica, 2018, 46(4): 938−944 (in Chinese)
(李子臣, 张卷美, 杨亚涛, 等. 基于 NTRU 的全同态加密方案[J]. 电子学报, 2018, 46(4): 938−944)

[110] Dahab R, Galbraith S, Morais E. Adaptive key recovery attacks on NTRU-based somewhat homomorphic encryption schemes[G]//LNCS 9063: Proc of the 8th Int Conf on Information Theoretic Security. Berlin: Springer, 2015: 283–296

[111] Chenal M, Tang Qiang. Key recovery attacks against NTRU-based somewhat homomorphic encryption schemes[C]//LNCS 9290: Proc of the 18th Int Conf on Information Security. Berlin: Springer, 2015: 397–418

[112] Albrecht M, Bai Shi, Ducas L. A subfield lattice attack on overstretched NTRU assumptions: Cryptanalysis of some FHE and graded encoding schemes[G]//LNCS 9814: Advances in Cryptology (CRYPTO 2016). Berlin: Springer, 2016: 153−178

[113] Kirchner P, Fouque P A. Revisiting lattice attacks on overstretched NTRU parameters[G]//LNCS 10210: Advances in Cryptology (EUROCRYPT 2017). Berlin: Springer, 2017: 3−26

[114] Genise N, Gentry C, Halevi S, Homomorphic encryption for finite automata[G]//LNCS 11922: Advances in Cryptology (ASIACRYPT

2019). Berlin: Springer, 2019: 473–502

[115] Lee C, Wallet A. Lattice analysis on MiNTRU problem[EB/OL]. 2020[2022-09-02].https://eprint.iacr.org/2020/230

[116] Che Xiaoliang, Zhou Tanping, Li Ningbo, et al. Optimization of NTRU-type multi-key fully homomorphic encryption scheme[J]. Advanced Engineering Sciences, 2020, 52(5): 186−193 (in Chinese)
(车小亮, 周潭平, 李宁波, 等. NTRU 型多密钥全同态加密方案的优化[J]. 工程科学与技术, 2020, 52(5): 186−193)

[117] Ducas L, Woerden W V. NTRU Fatigue: How stretched is overstretched?[G]//LNCS 13093: Advances in Cryptology (ASIACRYPT 2021). Berlin: Springer, 2021: 3−32

[118] Bonte C, Iliashenko I, Park J, et al. Revisiting lattice attacks on overstretched NTRU parameters[EB/OL]. 2022[2022-11-02].https://eprint.iacr.org/2022/074

[119] Halevi S, Shoup V. Bootstrapping for HElib[J]. Journal of Cryptology, 2021, 34(1): 1−44

[120] Kang H, Lee J, Lee Y, et al. Bootstrapping on SEAL[EB/OL]. 2020[2023-07-21].https://eprint.iacr.org/2020/1594

[121] Han K, Ki D. Better bootstrapping for approximate homomorphic encryption[C]//LNCS 12006: Topics in Cryptology (CT-RSA 2020). Berlin: Springer, 2020: 364–390

[122] Jung W, Kim W, Ahn J H, et al. Over 100x faster bootstrapping in fully homomorphic encryption through memory-centric optimization with GPUs[EB/OL]. 2021[2023-07-20].https://eprint.iacr.org/2021/508

[123] Gama N. TFHE v1.0. 1[EB/OL]. (2017-08-16)[2022-12-16]. https://github.com/tfhe/tfhe

[124] Ducas L. FHEW v 2.0-alpha[EB/OL]. (2017-05-30)[2022-11-06]. https://github.com/lducas/FHEW

[125] Bergamaschi F. HElib v2.2. 2[EB/OL]. (2022-12-06)[2022-12-16]. https://github.com/homenc/HElib

[126] Wei Dai. SEAL v4[EB/OL]. (2022-03-18)[2022-11-06]. https://github.com/microsoft/SEAL

[127] Wei Dai. cuFHE v1.0_beta[EB/OL]. (2018-03-14)[2022-11-06]. https://github.com/vernamlab/cuFHE

[128] Kim A. HEAAN V2.1[EB/OL]. (2018-09-05)[2022-11-06]. https://github.com/snucrypto/HEAAN

[129] Bossuat J P. Lattigo v4.1. 0[EB/OL]. (2022-12-01)[2022-12-16].https://github.com/tuneinsight/lattigo

[130] Polyakov Y. PALISADE v1.11. 9[EB/OL]. (2022-12-02)[2022-12-16]. https://gitlab.com/palisade/palisade-release

[131] Tmontaigu. Concrete v0.2. 0[EB/OL]. (2022-10-19)[2022-12-16]. https://github. com/zama-ai/concrete

[132] Ibarrond. Pyfhel v3.3. 2[EB/OL]. (2022-12-08)[2022-12-16]. https://github.com/ibarrond/Pyfhel

[133] Yspolyakov. OPenFFHE v1.0. 1[EB/OL]. (2022-12-01)[2022-12-06]. https://github. com/openfheorg/openfhe-development

[134] Ppppbamzy. HEhub v0.1. 0-alpha[EB/OL]. (2022-10-24)[2022-11-06]. https://github.com/primihub/HEhub

[135] Albrech M, Chase M, Chen Hao, et al. Homomorphic encryption Standard[S/OL]. (2018-11-21)[2022-10-06]. http://homomorphicencryption.org/wp-content/uplo-ads/2018/11/HomomorphicEncryptionStandardv1.1.pdf



Bai Lifang, born in 1990. PhD candidate, senior engineer. Student member of CCF. Her main research interests include fully homomorphic encryption and privacy computing.

白利芳, 1990 年生. 博士研究生, 高级工程师. CCF 学生会员. 主要研究方向为全同态加密、隐私计算.



Zhu Yuefei, born in 1962. PhD, professor, PhD supervisor. His main research interests include cryptography and cyberspace security.

祝跃飞, 1962 年生. 博士, 教授, 博士生导师. 主要研究方向为密码学、网络空间安全.



Li Yongjun, born in 1983. PhD candidate, lecturer. Her main research interests include privacy computing and mobile computing.

李勇军, 1983 年生. 博士研究生, 讲师. 主要研究方向为隐私计算、移动计算.



Wang Shuai, born in 1988. Master. His main research interest includes big data security.

王　帅, 1988 年生. 硕士. 主要研究方向为大数据安全.



Yang Xiaoqi, born in 1993. Master. His main research interests include data security and privacy computing.

杨晓琪, 1993 年生. 硕士. 主要研究方向为数据安全、隐私计算.