doi:10.12068/j.issn.1005－3026.2024.11.003

基于非交互零知识证明的可验证全同态加密算法

孙劲桐[1]，周福才[1]，王 强[1]，边 澈[2]
（1.东北大学 软件学院，辽宁 沈阳 110169；2.中国医科大学 附属第四医院，辽宁 沈阳 110165）

摘 要：同态加密（homomorphic encryption，HE）由于低执行效率和无法保护数据完整性的问题严重限制了其在实际应用中的部署，尤其是在对延迟有严格要求的场景中，为此，提出了一种新的HE来解决这些问题并增强通用性．为了解决执行效率低的问题，设计了多线程矩阵乘法（multithreaded matrix multiplication，MMM）算法．利用MMM算法，可以将加密任务拆解分配给多个线程并行执行，达到加速的目的．针对恶意服务器场景下的数据篡改问题，设计了一个可验证加密机制，利用非交互零知识证明（zk-SNARK）技术保护外包计算中密文的完整性．结合MMM算法，设计了一种高效的基于零知识证明的可验证全同态加密算法（verifiable fully homomorphic encryption based on zk-SNARKs，zk-VFHE）．理论分析和实验结果表明，zk-VFHE比同类协议具有更快的执行速度和更高的安全性．

关 键 词：全同态加密；误差学习；非交互零知识证明；可验证计算；矩阵编码
中图分类号：TP 309.2 **文献标志码**：A **文章编号**：1005-3026(2024)11-1537-10

Verifiable Fully Homomorphic Encryption Based on Zero-Knowledge Succinct Non-interactive Arguments of Knowledge

SUN Jin-tong[1]，*ZHOU Fu-cai*[1]，*WANG Qiang*[1]，*BIAN Che*[2]
（1. School of Software，Northeastern University，Shenyang 110169，China；2. The Fourth Affiliated Hospital，China Medical University，Shenyang 110165，China. Corresponding author：ZHOU Fu-cai，E-mail：fczhou@mail.neu.edu.cn）

Abstract：Homomorphic encryption（HE）is severely limited in its practical deployment due to low execution efficiency and the inability to ensure data integrity，particularly in scenarios with strict latency requirements. To address such issues and enhance general applicability，a new HE scheme is proposed. To improve execution efficiency，a multithreaded matrix multiplication（MMM）algorithm is designed. With the MMM algorithm，encryption tasks can be decomposed and distributed across multiple threads for parallel execution，thus achieving acceleration. To tackle data tampering in malicious server environments，a verifiable encryption mechanism is designed using zk-SNARK techniques to protect the integrity of ciphertext in outsourced computations. By combining MMM，an efficient verifiable fully homomorphic encryption based on zk-SNARK（zk-VFHE）was developed. Theoretical analysis and experimental results demonstrate that zk-VFHE outperforms similar protocols in terms of both execution speed and security.

Key words：fully homomorphic encryption；learning with errors；zero-knowledge succinct non-interactive arguments of knowledge（zk-SNARK）；verifiable computation；matrix codes

近年来，随着信息技术的快速发展，人们对 数据安全的关注度越来越高．无论是企业、政府

收稿日期：2023-06-05
基金项目：国家自然科学基金资助项目（62072090，62202090，62173101）；辽宁省自然科学基金医工交叉联合基金资助项目（2022-YGJC-24）；辽宁省博士科研基金资助项目（2022-BS-077）；中央高校基本科研业务费专项资金资助项目（N2217009）．
作者简介：孙劲桐（1995-），男，辽宁鞍山人，东北大学博士研究生；周福才（1964-），男，辽宁长春人，东北大学教授，博士生导师．

还是个人,都在不断加强对数据的保护.各种保护数据安全的手段应运而生,其中可以在加密数据上直接进行计算的HE算法是在外包计算场景下有效保护数据安全的重要技术之一,被广泛地应用于工业界.

传统的HE方案[1-2]是基于半可信的恶意模型设计的,即外包计算服务器会诚实地执行密文运算算法,但是会根据其得到信息来推测对应明文的相关信息.然而这种恶意模型不符合现实应用场景,在现实场景中,外包服务器是完全恶意的,其可以执行任何恶意操作,甚至可以不忠实地执行密文运算算法,因此传统的HE方案在实际部署上十分受限.在恶意模型中强行部署传统HE方案,不仅无法保证计算结果的正确性,甚至连加密算法最基本的机密性也无法保障.恶意的服务器可以发起密钥恢复攻击来构建解密预言机,从而破坏传统的HE方案.已经出现了很多密钥恢复攻击的具体方案[3],在这些方案中服务器可以将客户端作为预言机来生成指定私钥才能解密的密文.HE方案通常被使用在外包计算的场景下对被外包的数据和计算的结果进行保护,而密文的伪造会对此类应用带来严重的信誉危机,基于HE的外包计算结果的完整性将面临被破坏的风险,例如:外包计算服务提供商为了节约算力,可以跳过计算任务,直接伪造一个计算委托方可以解密的密文作为计算结果并返回.因此研究在完全恶意的威胁模型下构建HE具有重要意义.

另一类研究则侧重于保护HE的完整性[4-8],即在不失机密性的前提下,保证函数在密文上的正确执行.这类方案也被称为可验证同态加密(verifiable homomorphic encryption,VHE),但是上述方案存在明显的缺陷,只能抵御可以使用验证预言机敌手的攻击,无法抵御可以使用解密预言机敌手的攻击.因此研究在解密预言机敌手的威胁下有效保护HE完整性的方法具有重要意义.

综上所述,现有的HE和VHE方案存在的问题:无法在恶意模型中保护密文的机密性;加密算法的执行效率较低;无法在解密预言机敌手的威胁下保护密文运算的完整性.

针对上述问题,本文提出了一种基于非交互零知识证明的可验证全同态加密算法(verifiable fully homomorphic encryption based on zk-SNARKs,zk-VFHE).本文的主要贡献包括3个方面:

1)提出了一种基于容错学习(learning with errors,LWE)的全同态加密算法,通过简单的矩阵乘法和模运算来生成密文,具有较高的执行效率.

2)提出了一种多线程矩阵乘法算法,通过将加密算法中的矩阵乘法操作分解并分配给多个线程同时执行,进一步提高了加密算法的执行效率.

3)提出了一种基于零知识证明的验证机制,保证外包函数在密文上计算结果完整性的同时,也可以保护外包服务器私有输入的机密性.

1 预备知识

1.1 同态加密

同态加密是一种加密技术,它允许对加密数据进行计算而不需要解密,同时保持数据的安全性和完整性.同态加密算法可以表示为由4个多项式算法组成的密码学原语,HE=(KeyGen,Enc,Dec,Eval),其中KeyGen是密钥生成算法,由可信的密钥中心调用,生成用于参与加密的公钥pk和用于参与解密的私钥sk;Enc是加密算法,以pk和明文$\boldsymbol{m}$为输入,输出密文$\boldsymbol{c}$;Dec是解密算法,由私钥持有者调用,以sk和$\boldsymbol{c}$为输入,输出$\boldsymbol{m}$;Eval是密文运算算法,以操作符op和两个操作数o_1和o_2为输入,其中o_1和o_2至少有一个是密文,输出密文计算结果$\boldsymbol{c}_e$.

密文的同态性有两种,分别为加法同态和乘法同态.设有两个明文$\boldsymbol{m}_1$和$\boldsymbol{m}_2$,它们对应的密文分别为$\boldsymbol{c}_1$和$\boldsymbol{c}_2$,即$\boldsymbol{c}_1=\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{m}_1)$,$\boldsymbol{c}_2=\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{m}_2)$,满足式(1)的密文具有加法同态性,满足式(2)的密文具有乘法同态性.

$$\mathrm{Eval}(+,\boldsymbol{c}_1,\boldsymbol{c}_2)=\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{m}_1+\boldsymbol{m}_2),\tag{1}$$

$$\mathrm{Eval}(\times,\boldsymbol{c}_1,\boldsymbol{c}_2)=\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{m}_1\times\boldsymbol{m}_2).\tag{2}$$

同态加密可以分为三种类型:全同态加密[9](fully homomorphic encryption,FHE)、半同态加密[10](partially homomorphic encryption,PHE)和部分同态加密(somewhat homomorphic encryption,SWHE)[11].FHE的密文同时具有加法同态性和乘法同态性;PHE的密文只具有一种同态性;SWHE的密文具有一种同态性的同时也有限地具有另一种同态性.

1.2 双线性映射

双线性映射指的是循环群之间相对应的线

性映射关系.设$\boldsymbol{G}$和$\boldsymbol{G}_{\mathrm{T}}$均为$p$阶循环群,$g$为群$\boldsymbol{G}$的生成元,双线性映射$e:\boldsymbol{G}\times\boldsymbol{G}\to\boldsymbol{G}_{\mathrm{T}}$满足如下属性:

1)双线性:对于任意的$a,b\in\boldsymbol{Z}_p$和$g_1,g_2\in\boldsymbol{G}$,均满足$e\left(g_1{}^a,g_2{}^b\right)=e(g_1,g_2)^{ab}$,其中$\boldsymbol{Z}_p$是模整数$p$的剩余类集;

2)非退化性:$e\left(g,g\right)\neq 1\boldsymbol{G}_{\mathrm{T}}$,其中$1\boldsymbol{G}_{\mathrm{T}}$为群$\boldsymbol{G}_{\mathrm{T}}$的单位元;

3)可计算性:对于任意的$g_1,g_2\in\boldsymbol{G}$,都存在有效的算法计算出$e(g_1,g_2)$.

1.3　二次算术程序

二次算术程序(quadratic arithmetic program,QAP)是算术电路(arithmetic circuit,AC)的变体.设电路中的门为$R=\{r_1,r_2,\cdots,r_m\}$,首先依次为电路中的门赋值,$r_1=1$, $r_2=2,\cdots,r_m=m$,构造函数$t(x)=\prod_{q=1}^{m}\left(x-r_q\right)$.由于AC的性质[12],一定能使用拉格朗日插值法找到一组$m-1$次的函数$U=\{u_i(x)\}$, $V=\{v_i(x)\}$, $W=\{w_i(x)\}$,$i\in[1,n]$,使得当$1\leqslant q\leqslant m$时,下列等式成立.

$$u_i\left(r_q\right)=u_{i,q},v_i\left(r_q\right)=v_{i,q},w_i\left(r_q\right)=w_{i,q}.$$

则一个m门n线的QAP同样定义了S中每个值的关系,如式(3)所示,其中等式右侧表示对多项式$t(x)$执行模运算结果是0的多项式剩余类.那么一定存在函数$h(x)$使得式(4)成立.

$$\sum_{i=1}^{n}s_iu_i(x)\sum_{i=1}^{n}s_iv_i(x)-\sum_{i=1}^{n}s_iw_i(x)\equiv 0(\mathrm{mod}\ t(x)),\quad(3)$$

$$\sum_{i=1}^{n}s_iu_i(x)\sum_{i=1}^{n}s_iv_i(x)-\sum_{i=1}^{n}s_iw_i(x)=h(x)t(x).\quad(4)$$

1.4　LWE难题

本小节的符号定义如表1所示.

LWE预言机[13].存在一个秘密向量$\boldsymbol{S}\in\boldsymbol{Z}_q^n$,随机选取$\boldsymbol{X}\xleftarrow{s}\boldsymbol{Z}_q^n$,则LWE预言机$\chi_{S,\alpha}$表示为$\beta=(\boldsymbol{X},<\boldsymbol{X},\boldsymbol{S}>+e)\leftarrow\chi_{S,\alpha}(\boldsymbol{X})$,其中误差$e\xleftarrow{s}\Psi_{\sigma,\mu,\Lambda}(x)$.

LWE难题[14].设存在一个挑战者A'可以向$\chi_{S,\alpha}$发起质询,并得到输出$(\boldsymbol{X},<\boldsymbol{X},\boldsymbol{S}>+e)$.现有一个多项式时间敌手$A$,$A$选择两个$\boldsymbol{X}_0,\boldsymbol{X}_1\xleftarrow{s}\boldsymbol{Z}_q^n,\boldsymbol{X}_0\neq\boldsymbol{X}_1$,并将$\boldsymbol{X}_0,\boldsymbol{X}_1$发送给$C$,$C$随机选择$\boldsymbol{X}_b,b\in\{0,1\}$,向$\chi_{S,\alpha}$发起质询,得到$\boldsymbol{c}_b=<\boldsymbol{X}_b,\boldsymbol{S}>+e$,并将$\boldsymbol{c}_b$返回给$A$,则$A$无法以不可忽略的优势$\varepsilon$分辨出$\boldsymbol{c}_b$. LWE难题表示为$P\left[b\leftarrow A\left(n,q,\alpha,\boldsymbol{X}_0,\boldsymbol{X}_1,\boldsymbol{c}_b\right)\right]<\frac{1}{2}+\varepsilon$.

表1　LWE难题的符号定义

Table 1　Symbols of LWE problem

符号	定义
n	正整数
q	正整数,$q\geqslant 2$
Λ	格,$\Lambda\in\mathbf{R}^q$
α	误差参数,$\alpha\in\mathbf{Z}^+$
$\rho_{\sigma,\mu}(x)$	高斯函数,$\rho_{\sigma,\mu}(x)=\exp\left(\frac{-\pi\lvert\lvert x-\boldsymbol{c}\rvert\rvert^2}{\sigma^2}\right)$,其中$\sigma=\frac{\alpha}{\sqrt{2\pi}},\mu=0$
$\rho_{\sigma,\mu}(\Lambda)$	格Λ上的离散积分,$\rho_{\sigma,\mu}(\Lambda)=\sum_{x\in\Lambda}\rho_{\sigma,\mu}(x)$
$\Psi_{\sigma,\mu,\Lambda}(x)$	离散高斯分布,$\Psi_{\sigma,\mu,\Lambda}(x)=\frac{\rho_{\sigma,\mu}(x)}{\rho_{\sigma,\mu}(\Lambda)}$

2　模型与定义

2.1　系统模型架构

本文提出的同态加密方案VFHE可以适用于如下场景:算力有限的客户端C向云服务器S外包公共计算函数$f(\boldsymbol{b},\boldsymbol{w})$,其中$\boldsymbol{b}$是$C$的私有输入,$\boldsymbol{w}$是$S$的私有输入,$S$不希望$C$获取$\boldsymbol{w}$,$C$不希望$S$获取$\boldsymbol{b}$以及计算结果$\boldsymbol{b}'=f(\boldsymbol{b},\boldsymbol{w})$,在服务器运算并返回计算结果之后,$C$还能够验证$\boldsymbol{b}'$的正确性.方案架构如图1所示.

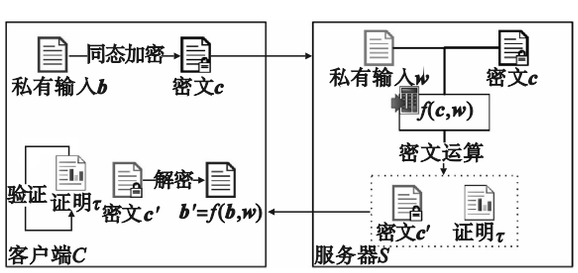

图1　隐私保护的可验证外包计算

Fig. 1　Privacy-preserving verifiably outsourced computing

为了保护客户端数据的隐私性,同时又可以使得服务器可以执行计算f,客户端先将自己的私有输入使用HE进行计算,并得到密文$\boldsymbol{c}$,服务器将自己的私有输入$\boldsymbol{w}$和$\boldsymbol{c}$作为输入计算$\boldsymbol{c}'=f(\boldsymbol{c},\boldsymbol{w})$,并将计算结果$\boldsymbol{c}'$和证明$\tau$一起返回给客户端,客户端可以根据$\tau$验证计算结果的正确性,若验证成功,则客户端解密$\boldsymbol{c}'$得到$\boldsymbol{b}'=f(\boldsymbol{b},\boldsymbol{w})$.

2.2　形式化定义

本方案各阶段主要包括5个概率多项式时间

算法,其形式化定义如下.

1) KeyGen$(\lambda,f)\to(\mathrm{pk},\mathrm{sk},\mathrm{param})$:密钥生成算法,概率性算法.输入安全参数$\lambda$,外包函数$f$,输出公共参数param和一对公私钥$(\mathrm{pk},\mathrm{sk})$.

2) $\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{b})\to\boldsymbol{c}$:加密算法,概率性算法.输入公钥pk和明文$\boldsymbol{b}\in\boldsymbol{Z}_N^{l_2}$,输出密文$\boldsymbol{c}$.

3) $\mathrm{Eval}_{\mathrm{pk}}(\boldsymbol{c},\boldsymbol{w},\mathrm{param})\to(\boldsymbol{c}',\tau')$:密文运算算法,概率性算法.输入公钥pk,密文$\boldsymbol{c}$,外包运算者的私有输入$\boldsymbol{w}\in\boldsymbol{Z}_N^{l_2}$和公共参数param,输出运算结果$\boldsymbol{c}'$和证明$\tau'$.

4) $\mathrm{Ver}_{\mathrm{sk}}(\boldsymbol{c},\tau)\to\mathrm{acc}$:验证算法,确定性算法.输入私钥sk,密文$\boldsymbol{c}$和证明$\tau$,输出验证结果acc.

5) $\mathrm{Dec}_{\mathrm{sk}}(\boldsymbol{c})\to\boldsymbol{b}$:解密算法,确定性算法.输入私钥sk和密文$\boldsymbol{c}$,输出明文$\boldsymbol{b}$.

2.3 安全性定义

若zk-VFHE被忠实地执行,那么其将具有以下4个性质.

定义1 正确性:设有一个明文$\boldsymbol{b}$,执行$\boldsymbol{c}\leftarrow\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{b})$,那么必定有

$$\boldsymbol{b}\leftarrow\mathrm{Dec}_{\mathrm{sk}}(\boldsymbol{c}).$$

定义2 全同态性:取任意数量的明文$\boldsymbol{B}=\{\boldsymbol{b}_1,\boldsymbol{b}_2,\cdots,\boldsymbol{b}_n\}$,其对应的密文为$\boldsymbol{C}_{\mathrm{m}}=\{\boldsymbol{c}_1,\boldsymbol{c}_2,\cdots,\boldsymbol{c}_n\}$,其中$\boldsymbol{c}_i\leftarrow\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{b}_i),i\in[1,n]$,那么必定有

$$\sum_{i=1}^{n}\boldsymbol{b}_i\leftarrow\mathrm{Dec}_{\mathrm{sk}}\Big(\sum_{i=1}^{n}\boldsymbol{c}_i\Big),$$

$$\prod_{i=1}^{n}\boldsymbol{b}_i\leftarrow\mathrm{Dec}_{\mathrm{sk}}\Big(\prod_{i=1}^{n}\boldsymbol{c}_i\Big).$$

注意,本文中的密文是向量,另外本文指的向量乘法并不是通常意义上的向量乘法,设有两个向量$\boldsymbol{c}_1=(x_1,x_2,\cdots,x_n)$,$\boldsymbol{c}_2=(y_1,y_2,\cdots,y_n)$,则$\boldsymbol{c}_1\boldsymbol{c}_2=(x_1y_1,x_2y_2,\cdots,x_ny_n)$.

定义3 完整性:设有一个明文$\boldsymbol{b}$,执行$\boldsymbol{c}\leftarrow\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{b})$,$(\boldsymbol{c}',\tau')\leftarrow\mathrm{Eval}_{\mathrm{pk}}(\boldsymbol{c},\boldsymbol{w},\mathrm{param})$,那么必定有

$$1\leftarrow\mathrm{Ver}_{\mathrm{sk}}(\boldsymbol{c}',\tau').$$

定义4 可靠性:设存在一个解密预言机$O_{\mathrm{Dec}}(\boldsymbol{c},\tau)\to\boldsymbol{b}$.当$\mathrm{Ver}_{\mathrm{sk}}(\boldsymbol{c},\tau)=0$时,$\boldsymbol{b}=\perp$;当$\mathrm{Ver}_{\mathrm{sk}}(\boldsymbol{c},\tau)=1$时,$\boldsymbol{b}=\mathrm{Dec}_{\mathrm{sk}}(\boldsymbol{c})$.对于一个可以访问$O_{\mathrm{Dec}}$的多项式时间敌手$A$,他不能以不可忽略的概率$\varepsilon$伪造出一个证据来欺骗用户接收一个错误的计算结果.设有一个明文$\boldsymbol{b}$,执行$\boldsymbol{c}\leftarrow\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{b})$,$(\boldsymbol{c}',\tau')\leftarrow\mathrm{Eval}_{\mathrm{pk}}(\boldsymbol{c},\boldsymbol{w},\mathrm{param})$,$(\boldsymbol{c}'',\tau'')\leftarrow A(\boldsymbol{c},\boldsymbol{c}',\tau',\mathrm{pk},O_{\mathrm{Dec}})$,则必定有

$$P\left[1\leftarrow\mathrm{Ver}_{\mathrm{sk}}(\boldsymbol{c}'',\tau'')\,|\,\boldsymbol{c}''\neq\boldsymbol{c}'\vee\tau''\neq\tau'\right]<\varepsilon.$$

3 方案的描述

本节先介绍了构建zk-VFHE所需的多线程矩阵乘法,然后详细地介绍了zk-VFHE加密算法的具体构造.

3.1 多线程矩阵乘法算法

多线程矩阵乘法(multithreaded matrix multiplication,MMM)算法是一种高效的矩阵乘法算法,以两个矩阵为输入,输出两个矩阵的乘法运算结果,设现有两个矩阵$\boldsymbol{A}$和$\boldsymbol{B}$,则$\mathrm{MMM}(\boldsymbol{A}^{\mathrm{T}},\boldsymbol{B})=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}$.下面将详细介绍MMM的构建.本小节的符号定义如表2所示.

表2 LWE难题的符号定义

Table 2 Symbols of MMM

符号	定义
$\boldsymbol{A}$	$\boldsymbol{A}\in\boldsymbol{Z}_q^{s\times r}$,矩阵乘法的左因数
$\boldsymbol{B}$	$\boldsymbol{B}\in\boldsymbol{Z}_q^{s\times t}$,矩阵乘法的右因数
m	$\boldsymbol{A}$的分片数
n	$\boldsymbol{B}$的分片数
$\boldsymbol{A}[\cdot]$	$\boldsymbol{A}$分片后得到的子矩阵数组
$\boldsymbol{B}[\cdot]$	$\boldsymbol{B}$分片后得到的子矩阵数组
$T[\cdot]$	线程序列
t	一个线程
N	线程的数量
id	线程的标识符
$(x_{\mathrm{id}},y_{\mathrm{id}})$	线程id生成的用于拉格朗日插值的点坐标

设$\boldsymbol{A}\in\boldsymbol{Z}_q^{s\times r}$是一个$s$行$r$列的矩阵,$\boldsymbol{B}\in\boldsymbol{Z}_q^{s\times t}$是一个$s$行$t$列的矩阵.首先将$\boldsymbol{A}$分割成$m$个子矩阵,将$\boldsymbol{B}$分割成$n$个子矩阵,如式(5)所示.其中每个$\boldsymbol{A}_i,i\in[0,m-1]$都是一个$s$行$r/m$列的矩阵,每个$\boldsymbol{B}_j,j\in[0,n-1]$都是一个$s$行$t/n$列的矩阵.此时$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}$可以表示成式(6)的形式,若能计算出每个$\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{B}_j$,就相当于计算出了$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}$.

$$\boldsymbol{A}=\left[\boldsymbol{A}_0,\boldsymbol{A}_1,\cdots,\boldsymbol{A}_{m-1}\right],\ \boldsymbol{B}=\left[\boldsymbol{B}_0,\boldsymbol{B}_1,\cdots,\boldsymbol{B}_{n-1}\right],\tag{5}$$

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}=\begin{bmatrix}\boldsymbol{A}_0^{\mathrm{T}}\boldsymbol{B}_0 & \boldsymbol{A}_0^{\mathrm{T}}\boldsymbol{B}_1 & \cdots & \boldsymbol{A}_0^{\mathrm{T}}\boldsymbol{B}_{n-1}\\ \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{B}_0 & \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{B}_1 & \cdots & \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{B}_{n-1}\\ \vdots & \vdots & & \vdots\\ \boldsymbol{A}_{m-1}^{\mathrm{T}}\boldsymbol{B}_0 & \boldsymbol{A}_{m-1}^{\mathrm{T}}\boldsymbol{B}_1 & \cdots & \boldsymbol{A}_{m-1}^{\mathrm{T}}\boldsymbol{B}_{n-1}\end{bmatrix}.\tag{6}$$

设现有$N\geqslant mn$个可以并行计算的线程,在$\boldsymbol{Z}_q$域中为每个线程$t\in\{0,1,\cdots,N-1\}$分配一个数字,用x_t表示,并确保每个x_t都不相等.每个线程t分别计算:

$$\widetilde{\boldsymbol{A}}_t=\sum_{i=0}^{m-1}\boldsymbol{A}_i x_t^i,\ \widetilde{\boldsymbol{B}}_t=\sum_{j=0}^{n-1}\boldsymbol{B}_j x_t^{jm}.$$

然后计算:

$$\boldsymbol{C}_t=\widetilde{\boldsymbol{A}}_t^{\mathrm{T}}\widetilde{\boldsymbol{B}}_t=\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{B}_j x_t^{i+jm},\ t\in[0,N-1].$$

那么每个线程的x_t和$\boldsymbol{C}_t$构成的点$(x_t,\boldsymbol{C}_t)$必定是多项式$g(x)=\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{B}_j x^{i+jm}$上的点.由于$g(x)$是一个$mn-1$次多项式,所以当最快的$mn$个线程返回结果之后,就可以使用拉格朗日插值法或更高效的多项式恢复算法来计算每个$\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{B}_j$,从而得到$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}$.

MMM的伪代码如表3所示.其中Split函数负责将矩阵分片,对应式(5)的操作.TO函数用于为每个线程生成各自的点坐标,TO算法的伪代码如表4所示.LI函数是拉格朗日插值函数,其输出是一个数组,数组的元素是拉格朗日插值得到的多项式自变量的系数,元素的顺序按照自变量的指数由小到大排列.注意,因为$\boldsymbol{A}$是一个s行r列的矩阵,$\boldsymbol{B}$是一个s行t列的矩阵,所以$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}$是一个$r$行$t$列的矩阵,但是MMM返回的$\boldsymbol{R}[][]$是一个$m$行$n$列的矩阵,这是因为$\boldsymbol{R}$中的每个元素都是一个$r/m$行,$t/n$列的矩阵,也就是说$\boldsymbol{R}$本身就是一个$r$行$t$列的矩阵.

表3　MMM算法

Table 3　MMM algorithm

$\mathrm{MMM}(\boldsymbol{A},\boldsymbol{B})\rightarrow\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}$
输入:$\boldsymbol{A},\boldsymbol{B}$
输出:$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}$
1. $\boldsymbol{A}[m]\leftarrow\mathrm{Split}(\boldsymbol{A},m)$
2. $\boldsymbol{B}[n]\leftarrow\mathrm{Split}(\boldsymbol{B},n)$
3. 调用N个线程并行执行 TO($T[i],i,\boldsymbol{A}[],\boldsymbol{B}[]$)
4. 接收最先返回的mn个结果记为:$\mathrm{TO}_0,\cdots,\mathrm{TO}_{mn-1}$
5. $\boldsymbol{P}[mn]=\mathrm{Null}$
6. **for** $i=0$ to $mn-1$ **do**
7. $\quad\boldsymbol{P}[i]=\mathrm{TO}_i$
8. **end for**
9. $\mathbf{COE}[mn]=\mathrm{LI}(\boldsymbol{P}[])$
10. $\boldsymbol{R}[m][n]=\mathrm{Null}$
11. **for** $i=0$ to $n-1$ **do**
12. $\quad$**for** $j=0$ to $m-1$ **do**
13. $\quad\quad\boldsymbol{R}[j][i]=\mathbf{COE}[j+i\times m]$
14. $\quad$**end for**
15. **end for**
16. **Return** $\boldsymbol{R}[][]$

下面分析MMM的执行效率.对一个k次多项式进行插值运算需要的时间复杂度是$O(k\cdot\lg^2(k)\lg(\lg(k)))$[15],$g(x)$是一个矩阵多项式,一共包含$rt/mn$个多项式,其中每个多项式的次数(多项式中出现的自变量的最高次)是$mn-1$,所以MMM的时间复杂度是$O\left(rt\lg^2(mn)\lg\left(\lg\left(mn\right)\right)\right)$.而直接对矩阵乘法进行计算需要的时间复杂度是$O(rts)$.因此只有当$mn\gg s$时,MMM的效率才会低于直接计算,而这是不可能的,所以MMM和直接计算相比具有很大的效率优势.

表4　TO算法

Table 4　TO algorithm

$\mathrm{TO}(T,\mathrm{id},\boldsymbol{A}[],\boldsymbol{B}[])\rightarrow(x_{\mathrm{id}},y_{\mathrm{id}})$
输入:$T,\mathrm{id},\boldsymbol{A}[],\boldsymbol{B}[]$
输出:$(x_{\mathrm{id}},y_{\mathrm{id}})$
1. $x_{\mathrm{id}}=\mathrm{id}+1$
2. $y_{\mathrm{id}}=0$
3. $\boldsymbol{A}'=0$
4. $\boldsymbol{B}'=0$
5. **for** $i=0$ to $m-1$ **do**
6. $\quad\boldsymbol{A}'+=\boldsymbol{A}[i]\times x_{\mathrm{id}}{}^{i}$
7. **end for**
8. **for** $i=0$ to $n-1$ **do**
9. $\quad\boldsymbol{B}'+=\boldsymbol{B}[i]\times x_{\mathrm{id}}{}^{im}$
10. **end for**
11. $y_{\mathrm{id}}=\boldsymbol{A}'^{\mathrm{T}}\boldsymbol{B}'$
12. **Return** $(x_{\mathrm{id}},y_{\mathrm{id}})$

3.2　zk-VFHE算法详细设计

本小节将构造的多线程矩阵乘法算法与LWE同态加密算法相结合,实现了zk-VFHE方案.其中各具体算法详细描述如下.

1)密钥生成算法.

① 根据λ生成参数$\alpha,\beta_u,\beta_v,\beta_w,\gamma\in\boldsymbol{Z}_N^+$,向量$\boldsymbol{z}\in\boldsymbol{Z}_N^l$,其中$l=l_1+l_2$,$l_1$是密文向量的长度,$l_2$是明文向量的长度.随机生成矩阵$\boldsymbol{A}_1\in\boldsymbol{Z}_q^{l_1\times r}$,$\boldsymbol{R}\in\{-1,0,1\}^{l_2\times l_1}$,生成行置换rp和其逆过程$\mathrm{rp}^{-1}$.实例化双线性映射$e:\boldsymbol{G}\times\boldsymbol{G}\rightarrow\boldsymbol{G}_{\mathrm{T}}$,其中$\boldsymbol{G}$是一个$q$阶循环群.实例化同态编码$E(x)=g^x$,其中$g$是$\boldsymbol{G}$的一个生成元;

② 将f转换成有着m个门、n个线路的算术电路AC,将AC转换成相应的QAP,$Q=(t(x),h(x),U,V,W)$,详见1.4;

③ 计算:

$$\boldsymbol{A}=\begin{bmatrix}\mathrm{MMM}(\boldsymbol{R},\boldsymbol{A}_1)\\-\boldsymbol{A}_1\end{bmatrix},\tag{7}$$

$$\boldsymbol{T}=\begin{bmatrix}\boldsymbol{I}&\boldsymbol{R}\\\boldsymbol{0}&\boldsymbol{0}\end{bmatrix}.\tag{8}$$

其中:$\boldsymbol{I}\in\{0,1\}^{l_2\times l_2}$是单位矩阵;$\boldsymbol{T}$为私钥中的参数,$\boldsymbol{T}\in\boldsymbol{Z}_q^{l\times l}$.

④ 生成公共计算密钥ek:

$$\mathrm{ek}=\Big(\{E(z^i)\}_{i=1}^n,\{E(\alpha z^i)\}_{i=1}^n,\{E(u_i(z))\}_{i=1}^n,$$
$$\{E(\alpha u_i(z))\}_{i=1}^n,\{E(\beta_u u_i(z))\}_{i=1}^n,\{E(v_i(z))\}_{i=1}^n,$$
$$\{E(\alpha v_i(z))\}_{i=1}^n,\{E(\beta_v v_i(z))\}_{i=1}^n,\{E(w_i(z))\}_{i=1}^n,$$
$$\{E(\alpha w_i(z))\}_{i=1}^n,\{E(\beta_w w_i(z))\}_{i=1}^n,$$
$$E(t(z)),E(\alpha t(z)),$$
$$E(\beta_u t(z)),E(\beta_v t(z)),E(\beta_w t(z))\Big).$$

⑤ 生成验证密钥 vk:

$$\mathrm{vk}=(E(1),E(\alpha),E(\gamma),E(\beta_u\gamma),E(\beta_v\gamma),$$
$$E(\beta_w\gamma),E(h(z)),E(\alpha h(z))).$$

⑥ 输出公共参数 $\mathrm{param}=(f,\mathrm{AC},Q,E,e)$, 公钥 $\mathrm{pk}=(\boldsymbol{A},\mathrm{rp},\mathrm{ek})$, 私钥 $\mathrm{sk}=(\boldsymbol{T},\mathrm{rp}^{-1},\mathrm{vk})$.

密钥生成算法的伪代码如表 5 所示. 其中 KeyParGen 函数负责步骤①中的参数生成. ACGen 函数负责将多项式 f 转换成相应的算术电路 AC. QAPGen 函数负责将算术电路 AC 转换成相应的二次算术程序 Q. MatrixGen 函数负责组装矩阵, 用来生成式(7)和式(8)所示的矩阵.

表 5 密钥生成算法

Table 5 KeyGen algorithm

$\mathrm{KeyGen}(\lambda,f)\rightarrow(\mathrm{pk},\mathrm{sk},\mathrm{param})$
输入: λ,f
输出: pk, sk, param
1. $\alpha,\beta_u,\beta_v,\beta_w,\gamma,z,\boldsymbol{A}_1,\boldsymbol{R},\mathrm{rp},\mathrm{rp}^{-1},e,E(x)\leftarrow\mathrm{KeyParGen}(\lambda)$
2. $\mathrm{AC}\leftarrow\mathrm{ACGen}(f)$
3. $Q\leftarrow\mathrm{QAPGen}(Ci)$
4. $\boldsymbol{A}\leftarrow\mathrm{MatrixGen}(\mathrm{MMM}(\boldsymbol{R},\boldsymbol{A}_1),-\boldsymbol{A}_1)$
5. $\boldsymbol{T}\leftarrow\mathrm{MatrixGen}(\boldsymbol{I},\boldsymbol{R})$
6. ek = Null
7. vk = Null
8. **for** $i=1$ to n **do**
9. ek. add($E(z^i),E(\alpha z^i),E(Q.u_i(z)),E(\alpha Q.u_i(z)),$ $E(\beta_u Q.u_i(z)),E(Q.v_i(z)),E(\alpha Q.v_i(z)),E(\beta_v Q.v_i(z)),$ $E(Q.w_i(z)),E(\alpha Q.w_i(z)),E(\beta_w Q.w_i(z))$)
10. **end for**
11. ek. add($E(Q.t(z)),E(\alpha Q.t(z)),E(\beta_u Q.t(z)),E(\beta_v Q.t(z)),$ $E(\beta_w Q.t(z))$)
12. vk. add($E(1),E(\alpha),E(\gamma),E(\beta_u\gamma),E(\beta_v\gamma),E(\beta_w\gamma),$ $E(\beta_w\gamma),E(Q.h(z)),E(\alpha Q.h(z))$)
13. $\mathrm{param}=(f,Ci,Q,E,e)$
14. $\mathrm{pk}=(\boldsymbol{A},\mathrm{rp},\mathrm{ek})$
15. $\mathrm{sk}=(\boldsymbol{T},\mathrm{rp}^{-1},\mathrm{vk})$
16. **Return** pk, sk, param

2) 加密算法.

① 生成长度为 l_1 的零向量 $\boldsymbol{b}'$, 随机生成向量 $\boldsymbol{s}\in\boldsymbol{Z}_N^r,\boldsymbol{e}\in\{0,1\}^l$;

② 将 $\boldsymbol{b}$ 填充为 $\boldsymbol{m}=[\boldsymbol{b}\quad\boldsymbol{b}']$;

③ 计算:

$$\boldsymbol{c}^{\mathrm{inv}}=\mathrm{MMM}(\boldsymbol{A},\boldsymbol{s}^{\mathrm{T}})+N\boldsymbol{e}^{\mathrm{T}}+\boldsymbol{m}^{\mathrm{T}}(\mathrm{mod}\ q);$$

④ 输出密文 $\boldsymbol{c}=\mathrm{rp}(\boldsymbol{c}^{\mathrm{inv}})$.

加密算法的伪代码如表 6 所示. 其中 EncParGen 负责步骤①中的参数生成.

表 6 加密算法

Table 6 Enc algorithm

$\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{b})\rightarrow\boldsymbol{c}$
输入: $\boldsymbol{b}$
输出: $\boldsymbol{c}$
1. $\boldsymbol{b}',\boldsymbol{s},\boldsymbol{e}\leftarrow\mathrm{EncParGen}(l,l_1,\boldsymbol{Z}_N^r)$
2. $\boldsymbol{m}=\boldsymbol{b}\|\boldsymbol{b}'$
3. $\boldsymbol{c}^{\mathrm{inv}}=\mathrm{MMM}(\mathrm{pk}.\boldsymbol{A},\boldsymbol{s}^{\mathrm{T}})+N\boldsymbol{e}^{\mathrm{T}}+\boldsymbol{m}^{\mathrm{T}}\bmod q$
4. $\boldsymbol{c}=\mathrm{pk}.\mathrm{rp}(\boldsymbol{c}^{\mathrm{inv}})$
5. **Return** $\boldsymbol{c}$

3) 密文运算算法.

① 随机选取 $\delta_1,\delta_2,\delta_3\in\boldsymbol{Z}_q$;

② 使用算术电路 AC 计算运算结果 $\boldsymbol{c}'=f(\boldsymbol{c},\boldsymbol{w})$, 并记录下每条线路上传输的值 $S=\{s_1,s_2,\cdots,s_n\}$;

③ 设:

$$u'(z)=u(z)+\delta_1 t(z)=\sum_{i=1}^n s_i u_i(z)+\delta_1 t(z),$$
$$v'(z)=v(z)+\delta_2 t(z)=\sum_{i=1}^n s_i v_i(z)+\delta_2 t(z),$$
$$w'(z)=w(z)+\delta_3 t(z)=\sum_{i=1}^n s_i w_i(z)+\delta_3 t(z).$$

输出运算结果 $\boldsymbol{c}'$, 证明 $\tau'=(E(u'(z)),E(\alpha u'(z)),E(v'(z)),E(\alpha v'(z)),E(w'(z)),E(\alpha w'(z)),E(\beta_u u'(z)+\beta_v v'(z)+\beta_w w'(z)),E(\delta_1),E(\delta_2),E(\delta_3),E(\delta_1 v(z)),E(\delta_2 u(z)))$.

密文运算算法的伪代码如表 7 所示. 其中 EvalParGen 负责步骤①中的参数生成. AC.eval 函数负责调用算术电路 AC 来计算 $f(\boldsymbol{c},\boldsymbol{w})$.

4) 验证算法.

① 随机选择 $i,j,k\in\boldsymbol{Z}_q^+$;

② 验证式(9)—(14)是否成立;

③ 若全部成立, 输出 acc = 1; 否则输出 acc = 0.

$$e(E(u'(z)),E(\alpha u_l(z)))=e(E(\alpha u'(z)),E(u_l(z))),\quad(9)$$

$$e(E(v'(z)),E(\alpha v_j(z)))=e(E(\alpha v'(z)),E(v_j(z))),\tag{10}$$

$$e(E(w'(z)),E(\alpha w_k(z)))=e(E(\alpha w'(z)),E(w_k(z))),\tag{11}$$

$$e(E(h(z)),E(\alpha))=e(E(\alpha h(z)),E(1)),\tag{12}$$

$$\begin{aligned}&e(E(\beta_u u'(z)+\beta_v v'(z)+\beta_w w'(z)),E(\gamma))=\\&e(E(u'(z)),E(\beta_u\gamma))\cdot e(E(v'(z)),E(\beta_v\gamma))\cdot\\&e(E(w'(z)),E(\beta_w\gamma)),\end{aligned}\tag{13}$$

$$\begin{aligned}&e(E(u'(z)),E(v'(z)))\cdot e(E(\delta_3),E(1))=\\&e(E(w'(z)),E(1))\cdot e(E(t(z)),E(h(z)))\cdot e(E(\delta_1),E(\delta_2))\cdot\\&e(E(\delta_1\sum_{i'=1}^{n}s_i v_i(z))E(\delta_2\sum_{i'=1}^{n}s_i u_i(z)),E(1)).\end{aligned}\tag{14}$$

表 7　Eval运算算法

Table 7　Eval calculation algorithm

$\mathrm{Eval}_{\mathrm{pk}}(\boldsymbol{c},\boldsymbol{w},\mathrm{param})\rightarrow(\boldsymbol{c}',\tau')$

输入：$\boldsymbol{c},\boldsymbol{w},\mathrm{param}$

输出：$\boldsymbol{c}',\tau'$

1. $\delta_1,\delta_2,\delta_3\leftarrow\mathrm{EvalParGen}(\boldsymbol{Z}_q)$
2. $\boldsymbol{c}',S\leftarrow$ AC. eval(param. $f,\boldsymbol{c},\boldsymbol{w}$)
3. $u=0,\ v=0,\ w=0,\ \alpha u=0,\ \alpha v=0,\ \alpha w=0,uu=0,\ vv=0,$ $ww=0$
4. $\tau'=\mathrm{Null}$
5. **for** $i=1$ to n **do**
6. $\quad u\cdot=\mathrm{ek}.\ E(u_i(z))^{s_i}$
7. $\quad v\cdot=\mathrm{ek}.\ E(v_i(z))^{s_i}$
8. $\quad w\cdot=\mathrm{ek}.\ E(w_i(z))^{s_i}$
9. $\quad \alpha u\cdot=\mathrm{ek}.\ E(\alpha u_i(z))^{s_i}$
10. $\quad \alpha v\cdot=\mathrm{ek}.\ E(\alpha v_i(z))^{s_i}$
11. $\quad \alpha w\cdot=\mathrm{ek}.\ E(\alpha w_i(z))^{s_i}$
12. $\quad uu\cdot=\mathrm{ek}.\ E(\beta_u u_i(z))^{s_i}$
13. $\quad vv\cdot=\mathrm{ek}.\ E(\beta_v v_i(z))^{s_i}$
14. $\quad ww\cdot=\mathrm{ek}.\ E(\beta_w w_i(z))^{s_i}$
15. **end for**
16. $\tau'.\mathrm{add}\Big(u\cdot\mathrm{ek}.\ E(t(z))^{\delta_1},v\cdot\mathrm{ek}.\ E(t(z))^{\delta_2},$ $w\cdot\mathrm{ek}.\ E(t(z))^{\delta_3},\alpha u\cdot\mathrm{ek}.\ E(\alpha t(z))^{\delta_1},\alpha v\cdot\mathrm{ek}.\ E(\alpha t(z))^{\delta_2},$ $\alpha w\cdot\mathrm{ek}.\ E(\alpha t(z))^{\delta_3},E(\delta_1),E(\delta_2),E(\delta_3),v^{\delta_1},u^{\delta_2},$ $uu\cdot vv\cdot ww\cdot\mathrm{ek}.\ E(\beta_u t(z))^{\delta_1}\cdot\mathrm{ek}.\ E(\beta_v t(z))^{\delta_2}\cdot$ $\mathrm{ek}.\ E(\beta_w t(z))^{\delta_3}\Big)$
17. **Return** $\boldsymbol{c}',\tau'$

验证算法的伪代码如表 8 所示．其中 $\mathrm{constrain}_{1-6}$ 函数是约束判断函数，分别对应式(9)~式(14)的约束．

5）解密算法．

① 构建 $\boldsymbol{T}^*=[\boldsymbol{I}\quad 0]\in\{0,1\}^{l_2\times l}$；

② 计算：

$\boldsymbol{c}^{\mathrm{inv}}=\mathrm{rp}^{-1}(\boldsymbol{c}),\boldsymbol{y}=\mathrm{MMM}(\boldsymbol{T},\boldsymbol{c}^{\mathrm{inv}})\ (\mathrm{mod}\ q)$；

③ 输出明文 $\boldsymbol{b}=\mathrm{MMM}(\boldsymbol{T}^*,\boldsymbol{y})\ (\mathrm{mod}\ N)$.

解密算法的伪代码如表 9 所示．其中 DecParGen 函数负责生成步骤①中的辅助矩阵 $\boldsymbol{T}^*$.

表 8　Ver验证算法

Table 8　Ver verification algorithm

$\mathrm{Ver}_{\mathrm{sk}}(\boldsymbol{c},\tau)\rightarrow\mathrm{acc}$

输入：$\boldsymbol{c},\tau$

输出：acc

1. 生成 $i,j,k\in\boldsymbol{Z}_q^+$
2. 初始化 acc = 0
3. **if** $(\mathrm{constrain}_1(i,\tau)\wedge\mathrm{constrain}_2(j,\tau)\wedge$ $\mathrm{constrain}_3(k,\tau)\wedge\mathrm{constrain}_4(\mathrm{sk.\ vk},\tau)\wedge$ $\mathrm{constrain}_5(\mathrm{sk.\ vk},\tau)\wedge\mathrm{constrain}_6(\mathrm{sk.\ vk},\tau))$
4. $\quad$ acc = 1
5. **Return** acc

表 9　Dec解密算法

Table 9　Dec decryption algorithm

$\mathrm{Dec}_{\mathrm{sk}}(\boldsymbol{c})\rightarrow\boldsymbol{b}$

输入：$\boldsymbol{c}$

输出：$\boldsymbol{b}$

1. $\boldsymbol{T}^*\leftarrow\mathrm{DecParGen}(l,l_2)$
2. $\boldsymbol{c}^{\mathrm{inv}}=\mathrm{sk.\ rp}^{-1}(\boldsymbol{c})$
3. $\boldsymbol{y}=\mathrm{MMM}(\boldsymbol{T},\boldsymbol{c}^{\mathrm{inv}})\ (\mathrm{mod}\ q)$
4. $\boldsymbol{b}=\mathrm{MMM}(\boldsymbol{T}^*,\boldsymbol{y})\ (\mathrm{mod}\ N)$
5. **Return** $\boldsymbol{b}$

4　安全性分析

4.1　正确性

定理 1　zk-VFHE具有正确性.

证明　设有一个明文 $\boldsymbol{b}$，执行 $\boldsymbol{c}\leftarrow\mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{b})$，有 $\boldsymbol{c}=\mathrm{rp}(\boldsymbol{c}^{\mathrm{inv}})=\mathrm{rp}\left(\boldsymbol{A}\boldsymbol{s}^{\mathrm{T}}+N\boldsymbol{e}^{\mathrm{T}}+\boldsymbol{m}^{\mathrm{T}}(\mathrm{mod}\ q)\right)$.

再执行 $\mathrm{Dec}_{\mathrm{sk}}(\boldsymbol{c})$，则有

$$\mathrm{MMM}(\boldsymbol{T}^*,\boldsymbol{y})\ (\mathrm{mod}\ N)=(\boldsymbol{T}^*\boldsymbol{y})\ (\mathrm{mod}\ N)=$$

$$[\boldsymbol{I}\quad\boldsymbol{R}]\begin{bmatrix}\boldsymbol{I}&\boldsymbol{R}\\\boldsymbol{0}&\boldsymbol{0}\end{bmatrix}\boldsymbol{m}^{\mathrm{T}}=\boldsymbol{b}.$$

所以zk-VFHE具有正确性.

4.2 全同态性

定理2 zk-VFHE具有全同态性.

1) zk-VFHE具有加法同态性.

证明 设有任意数量的明文 $\boldsymbol{B}=\{\boldsymbol{b}_1,\boldsymbol{b}_2,\cdots,\boldsymbol{b}_n\}$,执行 $\boldsymbol{c}_i\leftarrow \mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{b}_i),i\in[1,n]$,则有 $\boldsymbol{c}_i=\mathrm{rp}\left(\mathrm{MMM}(\boldsymbol{A},\boldsymbol{s}_i^{\mathrm{T}})+N\boldsymbol{e}_i^{\mathrm{T}}+\boldsymbol{m}_i^{\mathrm{T}}\left(\mathrm{mod}\ q\right)\right)$,那么有

$$\sum_{i=1}^{n}\boldsymbol{c}_i=\mathrm{rp}\left(\boldsymbol{A}\sum_{i=1}^{n}\boldsymbol{s}_i^{\mathrm{T}}+N\sum_{i=1}^{n}\boldsymbol{e}_i^{\mathrm{T}}+\sum_{i=1}^{n}\boldsymbol{m}_i^{\mathrm{T}}\left(\mathrm{mod}\ q\right)\right).$$

又因为

$$\mathrm{MMM}(\boldsymbol{T}^*,\boldsymbol{y})\left(\mathrm{mod}\ N\right)=[\boldsymbol{T}^*\boldsymbol{y}]\left(\mathrm{mod}\ N\right)=$$

$$[\boldsymbol{I}\quad\boldsymbol{R}]\begin{bmatrix}\boldsymbol{I}&\boldsymbol{R}\\\boldsymbol{0}&\boldsymbol{0}\end{bmatrix}\sum_{i=1}^{n}\boldsymbol{m}_i^{\mathrm{T}}=\sum_{i=1}^{n}\boldsymbol{b}_i.$$

所以zk-VFHE具有加法同态性.

2) zk-VFHE具有乘法同态性.

证明 设有任意数量的明文 $\boldsymbol{B}=\{\boldsymbol{b}_1,\boldsymbol{b}_2,\cdots,\boldsymbol{b}_n\}$,执行 $\boldsymbol{c}_i\leftarrow \mathrm{Enc}_{\mathrm{pk}}(\boldsymbol{b}_i),i\in[1,n]$,则有 $\boldsymbol{c}_i=\mathrm{rp}(\mathrm{MMM}(\boldsymbol{A},\boldsymbol{s}_i^{\mathrm{T}})+N\boldsymbol{e}_i^{\mathrm{T}}+\boldsymbol{m}_i^{\mathrm{T}}(\mathrm{mod}\ q))$,任取其中的两个密文 $\boldsymbol{c}_j$ 和 $\boldsymbol{c}_k$,那么有

$$\boldsymbol{c}_j\cdot\boldsymbol{c}_k=\mathrm{rp}\left(\boldsymbol{A}(\boldsymbol{s}_j^{\mathrm{T}}\boldsymbol{A}\boldsymbol{s}_j^{\mathrm{T}}+\boldsymbol{s}_j^{\mathrm{T}}N\boldsymbol{e}_j^{\mathrm{T}}+\boldsymbol{s}_j^{\mathrm{T}}\boldsymbol{m}_k^{\mathrm{T}}+\boldsymbol{s}_k^{\mathrm{T}}N\boldsymbol{e}_j^{\mathrm{T}}+\boldsymbol{s}_k^{\mathrm{T}}\boldsymbol{m}_j^{\mathrm{T}})+N(\boldsymbol{e}_j^{\mathrm{T}}N\boldsymbol{e}_k^{\mathrm{T}}+\boldsymbol{e}_j^{\mathrm{T}}\boldsymbol{m}_k^{\mathrm{T}}+\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{m}_j^{\mathrm{T}})+\boldsymbol{m}_j^{\mathrm{T}}\boldsymbol{m}_k^{\mathrm{T}}\left(\mathrm{mod}\ q\right)\right).\quad(15)$$

设 $\boldsymbol{s}'^{\mathrm{T}}=\boldsymbol{s}_j^{\mathrm{T}}\boldsymbol{A}\boldsymbol{s}_j^{\mathrm{T}}+\boldsymbol{s}_j^{\mathrm{T}}N\boldsymbol{e}_j^{\mathrm{T}}+\boldsymbol{s}_j^{\mathrm{T}}\boldsymbol{m}_k^{\mathrm{T}}+\boldsymbol{s}_k^{\mathrm{T}}N\boldsymbol{e}_j^{\mathrm{T}}+\boldsymbol{s}_k^{\mathrm{T}}\boldsymbol{m}_j^{\mathrm{T}}$,$\boldsymbol{e}'^{\mathrm{T}}=\boldsymbol{e}_j^{\mathrm{T}}N\boldsymbol{e}_k^{\mathrm{T}}+\boldsymbol{e}_j^{\mathrm{T}}\boldsymbol{m}_k^{\mathrm{T}}+\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{m}_j^{\mathrm{T}}$,则可将式(15)改写成式(16),即 $\boldsymbol{c}_j\boldsymbol{c}_k$ 相当于是 $\boldsymbol{b}_j\boldsymbol{b}_k$ 对应的密文.

$$\boldsymbol{c}_j\boldsymbol{c}_k=\mathrm{rp}\left(\boldsymbol{A}\boldsymbol{s}'^{\mathrm{T}}+N\boldsymbol{e}'^{\mathrm{T}}+\boldsymbol{m}_1^{\mathrm{T}}\boldsymbol{m}_2^{\mathrm{T}}\left(\mathrm{mod}\ q\right)\right).\quad(16)$$

递归地执行上述操作,再执行 $\mathrm{Dec}_{\mathrm{sk}}\left(\prod_{i=1}^{n}\boldsymbol{c}_i\right)$,则有

$$\mathrm{MMM}\left(\boldsymbol{T}^*,\boldsymbol{y}\right)\left(\mathrm{mod}\ N\right)=(\boldsymbol{T}^*\boldsymbol{y})\left(\mathrm{mod}\ N\right)=$$

$$[\boldsymbol{I}\quad\boldsymbol{R}]\begin{bmatrix}\boldsymbol{I}&\boldsymbol{R}\\\boldsymbol{0}&\boldsymbol{0}\end{bmatrix}\prod_{i=1}^{n}\boldsymbol{m}_i^{\mathrm{T}}=\prod_{i=1}^{n}\boldsymbol{b}_i.$$

所以zk-VFHE具有乘法同态性.

4.3 完整性

定理3 zk-VFHE具有完整性.

证明 如果式(9)~式(12)成立,则说明服务器(证明者)诚实地使用公共计算密钥ek来生成多项式 $u(z),v(z),w(z)$;如果式(13)成立,则说明服务器生成并使用了正确的 $S=\{s_1,s_2,\cdots,s_n\}$ 作为其多项式的线性系数;如果式(14)成立,则说明服务器执行了正确的函数计算,因为其生成并执行的函数满足了式(3)的约束.综上,zk-VFHE具有完整性.

4.4 可靠性

定理4 zk-VFHE具有可靠性.

证明 首先证明 τ' 是由随机数 α 和 $\boldsymbol{z}$ 生成的,这两个随机数不参与密文的生成,所以解密预言机不会为服务器伪造证据带来任何的优势.如果服务器(证明者)没有忠实地执行计算,那么其构建的 $u(x),v(x),w(x),h(x)$ 将不会满足式(10)的约束,即 $u(x)v(x)-w(x)\neq h(x)t(x)$. 在完整性验证时,无法使得式(14)成立,客户端(验证者)将会发现并拒绝服务器的计算结果.

首先,正确的 $u(x)$ 和 $v(x)$ 的次数为 $m-1$,是由确定的 m 个点通过拉格朗日插值法确定的,又因为 $t(x)$ 的次数为 m,所以 $h(x)$ 的次数是 $m-2$. $u(x)v(x)-w(x)$ 和 $h(x)t(x)$ 就是两个次数为 $2m-2$ 的多项式.根据Schwartz-Zippel引理可知,不同的次数为 n 的多项式,最多有 n 个交点,因此 $u(x)v(x)-w(x)$ 和 $h(x)t(x)$ 最多有 $2m-2$ 个交点.又因为 $\boldsymbol{z}\in\boldsymbol{Z}_N^l$,一共有 N^l 个可能的取值,所以服务器伪造成功的可能性为 $\frac{2m-2}{N^l}$,当 N^l 足够大时,服务器伪造成功的概率趋近于0,所以zk-VFHE具有可靠性.

5 性能分析

首先将zk-VFHE的特性与其他一些LWE加密算法进行比较.然后根据渐近复杂度和实际表现来评估zk-VFHE的性能.本节中使用的符号如表10所示.

表10 性能分析中的符号定义

Table 10 Symbol definition in performance analysis

符号	定义
n'	函数转换成算术电路后线路的数量
l	密文的长度和密钥矩阵的行数
r	密钥矩阵的列数
s	MMM函数分割矩阵的子矩阵个数
poly(f)	外包函数的执行时间
t_{e}	双线性映射的执行时间
t_{E}	同态编码的执行时间
t_G	$\boldsymbol{G}$群运算的执行时间
$t_{G_{\mathrm{T}}}$	$\boldsymbol{G}_{\mathrm{T}}$群运算的执行时间

5.1 特性比较

通过将zk-VFHE的特点与其他同态加密方案的特点进行比较来分析zk-VFHE在功能上的优势,如表11所示.

表 11　方案对比

Table 11　Comparison with prior schemes

方案	GHV[2]	VFHE[6]	SL-CP-ABE[16]	zk-VFHE
明文结构	矩阵	矩阵	向量	向量
密文结构	矩阵	矩阵	向量集合	向量
验证支持	不支持	不支持	不支持	支持
同态性	全同态	部分同态	部分同态	全同态
加密效率	$O(lr)$	$O(lr)$	$O(lr)$	$O(l\cdot \lg^2(s)\lg(\lg(s)))$
解密效率	$O(lr)$	$O(lr)$	$O(lr)$	$O(l\cdot \lg^2(s)\lg(\lg(s)))$
密文运算效率	$O(1)$	$O(1)$	$O(1)$	$O(1)$
外包运算效率	$O(\text{poly}(f))$	$O(\text{poly}(f))$	$O(\text{poly}(f))$	$O(\text{poly}(f))+O(t_E)$
验证效率	—	$O(lr)$	—	$O(t_e)+O(t_G)+O(t_{G_T})$

5.2　效率分析

表 12 显示了 zk-VFHE 各个函数执行的渐进时间复杂度,主要的衡量因素有:外包函数的执行时间,双线性映射的执行时间,同态编码的执行时间,$\boldsymbol{G}$ 群运算的执行时间和 $\boldsymbol{G}_T$ 群运算的执行时间.

表 12　zk-VFHE 的渐进时间复杂度

Table 12　Asymptotic time complexity of zk-VFHE

函数	渐进复杂度
KeyGen	$(11n+7)t_E$
Enc	$O(l\cdot \lg^2(s)\lg(\lg(s)))$
Eval	$2\text{poly}(f)+14t_E$
Ver	$O(l\cdot \lg^2(s)\lg(\lg(s)))+18t_e+6t_{G_T}+t_G$

KeyGen 的执行效率主要取决于函数转换成算术电路后线路的数量.本文测试了当 $n\in\{2^7,2^8,\cdots,2^{14}\}$ 时,密钥生成算法的执行时间,如图 2a 所示.

Enc 的执行效率主要取决于公钥矩阵的行数和 MMM 函数分割矩阵的子矩阵个数.本文测试了当 $l\in\{2^7,2^8,\cdots,2^{14}\}$, $s\in\{2^2,2^3,\cdots,2^6\}$ 时,加密算法的执行时间,如图 2b 所示.

Ver 的执行效率主要取决于公钥矩阵的行数和 MMM 函数分割矩阵的子矩阵个数.本文测试了当 $l\in\{2^7,2^8,\cdots,2^{14}\}$, $s\in\{2^2,2^3,\cdots,2^6\}$ 时,验证算法的执行时间,如图 2c 所示.

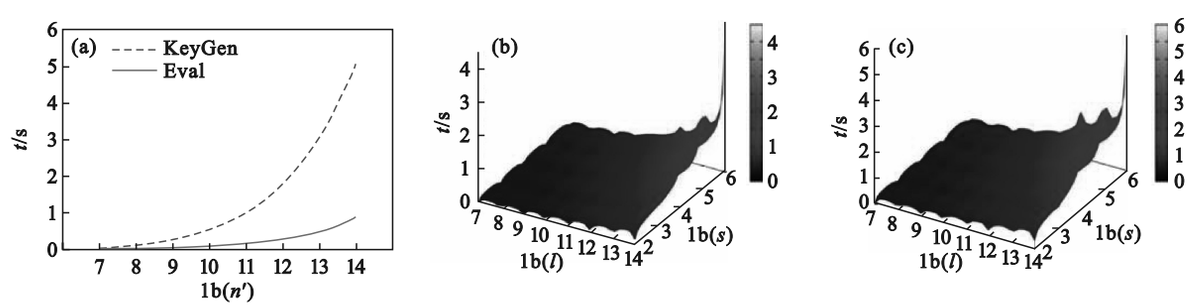

图 2　zk-VFHE 的运行时间

Fig. 2　Running time of zk-VFHE

(a)—密钥生成算法和密文运算算法执行效率;(b)—加密算法执行效率;(c)—验证算法执行效率.

5.3　效率对比

将 zk-VFHE 与 GHV, VFHE 的加密效率和解密效率进行对比.为了公平性将待加密的明文设置成一个向量(行数为 1 的矩阵),将密钥矩阵的列数设置为 $r=2^7$,将 MMM 函数分割矩阵的子矩阵个数设置为 $s=2^3$,分别测试了 3 个方案在明文长度 $l\in\{2^7,2^8,\cdots,2^{14}\}$ 时的运行效率,加密算法的效率对比结果如图 3a 所示,解密算法的效率对比结果如图 3b 所示.其中图 3 的横坐标中 $|l|$ 是密文的长度(密钥矩阵的行数).实验结果表明,zk-VFHE 的加密算法和解密算法在运行效率方面和同类方案相比具有优势.因为 MMM 算法的引入加速了加密和解密过程中的矩阵乘法操作,随着明文长度的增加,提速的效果将越来越显著.

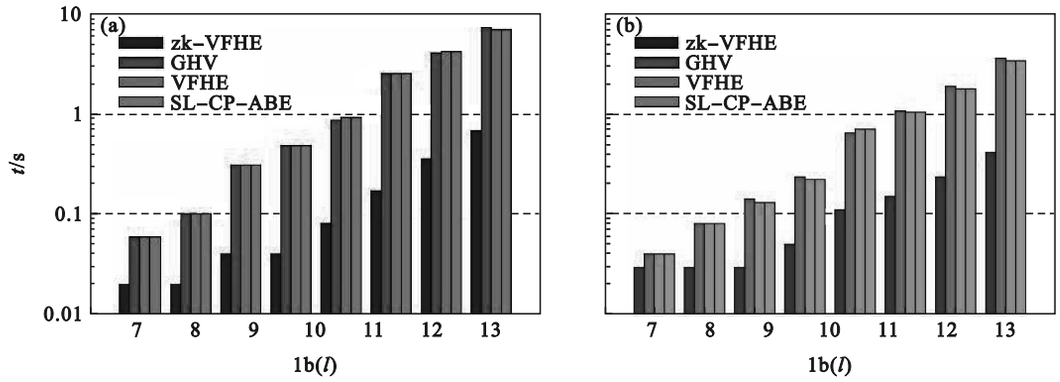

图3 同态加密算法的时间效率对比

Fig. 3 Time efficiency comparison of homomorphic encryptions

(a)—加密算法效率对比;(b)—解密算法效率对比.

6 结 语

针对现有的同态加密算法存在的执行效率低、无法保护数据完整性的问题,深入研究了国内外关于同态加密的相关方案和可验证计算的相关方案,并总结各方案的优缺点.结合可验证同态加密方案的密码学原语,提出了一个基于非交互零知识证明的可验证全同态加密算法,该算法具有如下特性:①正确性,使用该算法加密得到的密文均可由该算法的解密算法解密;②全同态性,使用该算法加密得到的密文均同时具备加法同态性和乘法同态性;③完整性,使用该算法生成的证据可以成功验证密文的数据完整性;④可靠性,多项式时间敌手无法为任何密文伪造一个可以通过验证的证据.本文证明了方案的安全性,理论分析与仿真实验表明zk-VFHE具有完备的安全性和优秀的效率.

参考文献:

[1] Brakerski Z, Gentry C, Vaikuntanathan V.(Leveled) fully homomorphic encryption without bootstrapping [J]. *ACM Transactions on Computation Theory* (*TOCT*), 2014, 6(3): 1-36.

[2] Gentry C, Halevi S, Vaikuntanathan V. A simple BGN-type cryptosystem from LWE [C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Heidelberg: Springer, 2010: 506-522.

[3] Zhang Z F, Plantard T, Susilo W. Reaction attack on outsourced computing with fully homomorphic encryption schemes [C]//International Conference on Information Security and Cryptology. Heidelberg: Springer, 2012: 419-436.

[4] Gennaro R, Gentry C, Parno B. Non-interactive verifiable computing: outsourcing computation to untrusted workers [C]//Annual Cryptology Conference. Heidelberg: Springer, 2010: 465-482.

[5] Gennaro R, Wichs D. Fully homomorphic message authenticators [C]//International Conference on the Theory and Application of Cryptology and Information Security. Heidelberg: Springer, 2013: 301-320.

[6] Huang R, Li Z, Zhao J. A verifiable fully homomorphic encryption scheme [C]//Security, Privacy, and Anonymity in Computation, Communication, and Storage: 12th International Conference, SpaCCS 2019. Atlanta, 2019: 412-426.

[7] Fiore D, Nitulescu A, Pointcheval D. Boosting verifiable computation on encrypted data [C]//IACR International Conference on Public-Key Cryptography. Cham: Springer, 2020: 124-154.

[8] Ganesh C, Nitulescu A, Soria-Vazquez E. Rinocchio: SNARKs for ring arithmetic [J]. *Journal of Cryptology*, 2023, 36(4): 41-93.

[9] Brakerski Z, Vaikuntanathan V. Fully homomorphic encryption from ring-LWE and security for key dependent messages [C]//Annual Cryptology Conference. Heidelberg: Springer, 2011: 505-524.

[10] Lu C F, Shieh S P. Secure key-evolving for public key cryptosystems based on the discrete logarithm problem [J]. *Journal of Information Science and Engineering*, 2004, 20(2): 391-400.

[11] Fiore D, Gennaro R, Pastro V. Efficiently verifiable computation on encrypted data [C]//Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. Scottsdale, 2014: 844-855.

[12] Pinto A M. An introduction to the use of zk-SNARKs in blockchains [C]//Mathematical Research for Blockchain Economy: 1st International Conference MARBLE 2019. Santorini: Springer International Publishing, 2020: 233-249.

[13] El-Yahyaoui A, El Kettani M D E C. A verifiable fully homomorphic encryption scheme to secure big data in cloud computing [C]//International Conference on Wireless Networks and Mobile Communications. Rabat: IEEE, 2017: 1-5.

[14] Regev O. On lattices, learning with errors, random linear codes, and cryptography [J]. *Journal of the ACM*, 2009, 56(6): 1-40.

[15] Kedlaya K S, Umans C. Fast polynomial factorization and modular composition [J]. *SIAM Journal on Computing*, 2011, 40(6): 1767-1802.

[16] Sravya G, Kumar P S, Padmavathy R. Secure lattice-based ciphertext-policy attribute-based encryption from module-LWE for cloud storage [C]//2023 IEEE 16th International Conference on Cloud Computing (CLOUD). Chicago: IEEE, 2023: 554-556.