公平可验证的多用户同态代理重加密方案

侯金秋[1]，彭长根[2]，丁红发[3]，杨训[4]
（1.安徽财经大学管理科学与工程学院，安徽 蚌埠 233030；2.贵州大学计算机科学与技术学院，贵州 贵阳 550025；
3.贵州财经大学贵州省高等学校区块链与金融科技重点实验室，贵州 贵阳 550025；
4.贵州工程应用技术学院理学院非线性分析与高性能计算实验室，贵州 毕节 551700）

摘　要：针对不可信云环境下传统全同态加密算法难以支持多用户密文共享及安全公平计算问题，提出了一种基于身份的可验证多用户同态代理重加密方案。首先，构建非合作博弈模型，提出动态资源分配算法，实现第三方云服务器的公平资源分配，确保资源使用的公平性与高效性。其次，通过代理重加密技术实现密文的转换，支持多用户同态运算，满足细粒度密文共享需求。最后，使用授权函数同态签名验证计算结果和云服务器行为，并在标准模型下给出所提方案的安全性证明。理论分析和实验结果表明，所提方案在保证重加密可验证性的同时，能够实现多用户同态运算。

关键词：代理重加密；同态加密；博弈论；可证明安全
中图分类号：TP309
文献标志码：A
DOI：10.11959/j.issn.1000–436x.2025040

Fairly verifiable multi-user homomorphic proxy re-encryption scheme

HOU Jinqiu[1], PENG Changgen[2], DING Hongfa[3], YANG Xun[4]
1. College of Management Science and Engineering, Anhui University of Finance & Economics, Bengbu 233030, China
2. School of Computer Science and Technology, Guizhou University, Guiyang 550025, China
3. Key Laboratory of Blockchain and Fintech of Department of Education of Guizhou Province, Guizhou University of Finance and Economics, Guiyang 550025, China
4. Nonlinear Analysis and High-Performance Computing Laboratory, School of Science, Guizhou University of Engineering Science, Bijie 551700, China

Abstract: To address the challenges of multi-user ciphertext sharing and secure fair computation in untrusted cloud environments where traditional fully homomorphic encryption algorithms prove inadequate, a verifiable identity-based multi-user homomorphic proxy re-encryption scheme was proposed. Firstly, a non-cooperative game model was constructed, and a dynamic resource allocation algorithm was proposed to achieve fair resource distribution for third-party cloud servers, ensuring fairness and efficiency in resource utilization. Secondly, ciphertext conversion was implemented through proxy re-encryption technology to support multi-user homomorphic computation and meet fine-grained ciphertext sharing requirements. Finally, an authorized function-based homomorphic signature was employed to verify computation results and cloud server behaviors, with security proofs of the proposed scheme was provided under the standard model. Theoretical analysis and experimental results demonstrate that the proposed scheme exhibits re-encryption verifiability and enables multi-user homomorphic computation.

Keywords: proxy re-encryption, homomorphic encryption, game theory, provable security

---

收稿日期：2024–10–17；**修回日期：**2025–02–24
通信作者：丁红发，hfding@mail.gufe.edu.cn
基金项目：国家自然科学基金资助项目（No.62272124）；贵州省教育厅自然科学研究基金资助项目（No.[2023]065, No.[2023]064）；安徽财经大学校级科研基金资助项目（No.ACKYB24016）
Foundation Items: The National Natural Science Foundation of China (No.62272124), The Natural Science Research Program of D.o.E. of Guizhou (No.[2023]065, No.[2023]064), Anhui University of Finance & Economics University-level Research Project (No.ACKYB24016)

0　引言

随着人工智能技术的飞速发展，尤其是大规模预训练模型（LPM, large pretrained model）的广泛应用，云计算和数据共享的需求呈现指数级增长。LPM 凭借其强大的计算能力与任务泛化性，已成为数据驱动应用的核心引擎，推动用户将海量数据的存储与处理任务外包至第三方云服务。然而，云环境的信任缺失问题导致数据安全风险加剧，如何在不可信服务器上实现隐私保护与安全计算成为关键挑战。为满足数据共享与隐私保护需求，通常采取代理重加密（PRE, proxy re-encryption）技术[1]，该技术允许云服务器将用户 A 的密文直接转换为用户 B 的可解密密文，而不需要接触明文信息。然而，传统 PRE[2-5]技术存在显著缺陷，无法支持密文间的直接运算，导致海量数据在处理时需频繁解密，计算开销巨大。因此，如何在密文状态下实现高效、安全的联合计算，成为云环境隐私保护的核心问题。

全同态加密（FHE, fully homomorphic encryption）[6]因其对密文具有任意计算能力，被视为解决上述问题的理想工具。但是传统的 FHE[7-8]仅支持单用户密文计算，难以满足多用户场景下数据交互与可验证联合计算的需求。为此，研究者尝试将 FHE 与 PRE 结合，提出同态代理重加密（HPRE, homomorphic proxy re-encryption）方案。

Gentry 等[9]基于 GSW（Gentry-Sahai-Waters）框架首次实现了 HPRE[10]，通过将多用户密文统一转换为目标用户的合法密文，显著降低了通信开销。进一步，文献[11]利用 PRE 技术实现了 2 个用户密文的全同态运算。随后，Li 等[12]提出了首个多跳格基 HPRE 方案，通过短密文 FHE 技术[13]压缩密文维度，并利用分支程序（BP, branching program）优化重加密流程，实现了同态计算 BP 并输出多跳的重加密密文，但该方案存在密钥过长问题。针对先前的 HPRE 方案[14]只支持部分同态操作，Luo 等[15]设计了一种支持全同态操作的单跳 HPRE 方案的通用构造。该方案将密文和每个用户的索引相关联，形成索引与密文对，判断是否可以执行同态操作，利用密钥交换算法构建更细粒度的 HPRE 方案。为了实现更高的安全性和效率，李瑞琪等[16]利用密文扩张思想与重线性化技术提出了一种新的 NTRU 型多密钥 HPRE 方案，满足联邦学习中的分布式解密需求。

尽管上述研究推动了 HPRE 的发展，但其实际部署仍面临资源分配公平性与计算过程可验证性双重挑战。具体而言，云服务器作为半可信实体，需在隐私保护约束下实现计算资源与存储资源的公平调度，同时需提供可验证机制确保操作合法性。然而，现有方案多聚焦于功能实现，未充分考虑资源分配效率与验证机制的设计，导致理论成果与实际需求之间仍存在鸿沟。如何在确保隐私和安全共享的前提下公平地分配计算资源和存储资源，并验证云服务器操作的合法性是一个亟待解决的问题。

在云数据共享过程中，可验证性[17-18]和公平性[19]是构建可信系统的基石。可验证性确保数据的完整性和正确性，使各方能够确认数据在传输和处理过程中没有被篡改。这不仅提高了系统的透明度，还增强了用户对云服务的信任。同时，公平性保障多方参与者的权益均衡，避免资源垄断。通过公平资源分配，所有用户能在同样的条件下进行合法性操作，从而提高系统的整体效率和用户满意度。现有资源分配方法主要包括静态分配[20]、动态分配[21]和基于优先级[22]的分配等。但这些方法要么资源利用率低、缺乏弹性和复杂性高，要么无法公平分配资源。尽管基于博弈论的云计算资源分配方法[23]有助于实现资源利用率和分配公平性的提升，已部分解决传统分配策略的僵化问题，但现有的 HPRE 方案在可验证性和公平性方面仍存在不足。许多方案缺乏有效的机制验证云服务器操作的合法性，无法确保数据在重加密过程中的完整性和正确性。此外，资源分配的公平性也没有得到充分保障，导致部分用户无法获得足够的计算资源和存储资源。

针对上述挑战，围绕云环境中多用户共享和隐私安全等问题，确保云服务器计算过程的公平性与透明度，防止恶意行为的发生。本文提出了基于身份的公平可验证多用户同态代理重加密方案，旨在解决云环境中多用户密文计算、访问控制和共享等问题。本文主要的研究工作如下。

1) 设计了非合作博弈模型，采用拉格朗日乘数法求解纳什均衡点，提出动态资源分配算法，实现多用户场景下的按需资源调度。

2) 设计了代理重加密算法，将各参与方的密文转化为云服务器的密文，解决不同公钥加密下的

密文不能进行同态运算的问题。

3) 将授权函数同态签名作为黑盒嵌入同态代理重加密方案中，检验云服务器是否执行正确的函数计算，实现第三方的可验证性。

4) 安全性证明了本文方案在标准模型下是不可区分性选择明文攻击（IND-CPA, indistinguishable choice plaintext attack）安全。实验结果验证了本文方案能够支持多用户同态运算。

1　系统模型

针对云环境中多用户的密文计算等问题，构建了公平可验证同态代理重加密模型，允许多个数据提供者用自己的公钥加密数据来进行同态计算，同时也可将数据拥有者的数据分享给指定的数据接收者。该模型主要包括数据拥有者、云服务器、验证服务器和数据使用者。具体过程如图1所示。

1) 每个数据拥有者分别用自己的加密密钥对自己的明文数据进行加密，得到密文数据后上传给云服务器。

2) 云服务器根据所需实现的数据处理函数f，对接收到的密文数据进行同态运算，将运算后的密文计算结果C_f发送给验证服务器。

3) 验证服务器检验重加密密文，并输出验证结果。

4) 将密文计算结果C_f返回给所有参与计算的用户，实现多方计算。

5) 云服务器根据数据拥有者的要求将用户i的密文转化为用户j的密文，实现了密文共享。

在此模型下，利用博弈论建模和优化用户之间的行为、资源分配和系统性能，将公平性资源分配与多用户同态代理重加密有机融合，能够实现系统的公平性、安全性和高效性，为云计算等多用户环境中的加密资源管理提供切实可行的资源分配策略，确保在密文数据共享过程中，各方的权利和义务对等，从而避免资源分配不公平导致的系统性能下降问题。同时，为确保公平性，采用授权函数同态签名设计验证算法，允许用户验证密文的完整性与计算结果的可信性。本文用到的参数及其含义如表1所示。

表1　　参数及其含义

参数	含义
n	常数
$\boldsymbol{A}$（大写加粗字母）	矩阵$\boldsymbol{A}$
$\boldsymbol{a}$（小写加粗字母）	向量$\boldsymbol{a}$
$\boldsymbol{A}^{\mathrm{T}}$	矩阵$\boldsymbol{A}$的转置
$\boldsymbol{A}^{-1}$	矩阵$\boldsymbol{A}$的逆矩阵
χ	误差分布
$\mathbb{Z}_q^{n\times m}$	模q剩余类环上的$n\times m$维矩阵空间
$x\leftarrow D$	从D中随机均匀地选择样本x

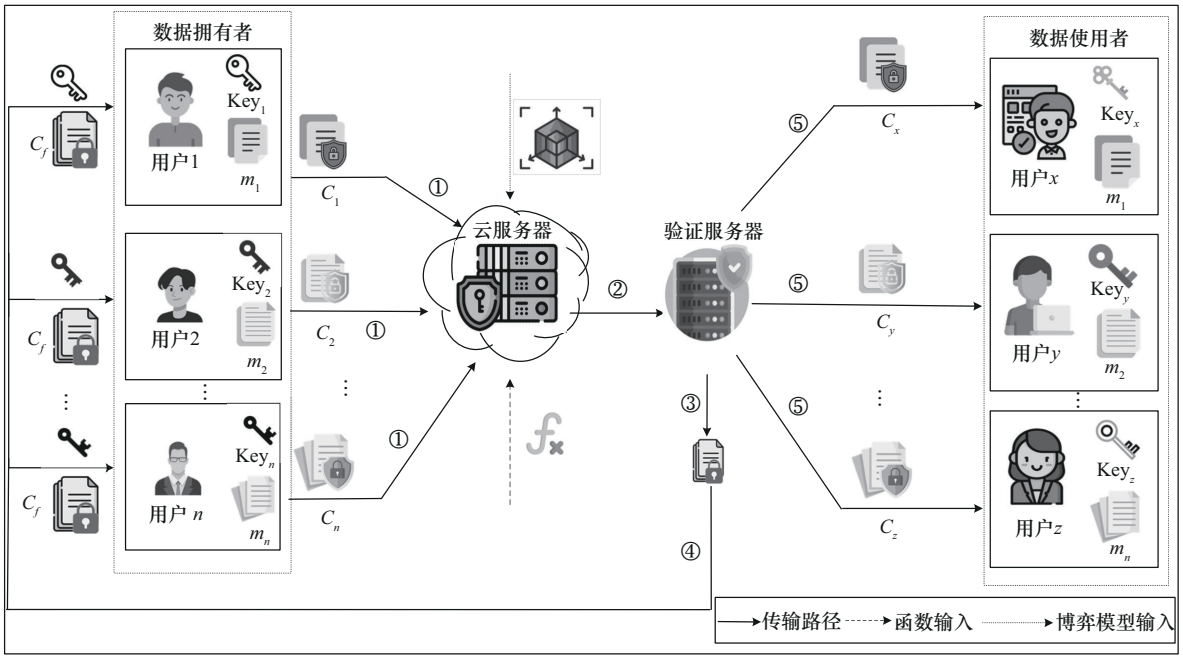

图1　公平可验证同态代理重加密模型

2　基于博弈论的公平性资源分配算法

在云计算环境中，多个用户通过代理重加密方案共享计算资源（如计算能力、存储、带宽等），不同用户可能会同时向云服务器提交计算请求，这些请求的性质和优先级可能不同，每个用户的资源需求可能是动态变化的，且云服务器是一个第三方，无法直接访问用户的加密数据。为了保证云服务器在为多个用户提供资源的同时能够公正地分配资源，必须建立一个公平的资源分配机制。公平性不仅意味着资源分配不应偏向某一用户，还要求系统能够在不同用户之间保持负载均衡，避免任何一方因资源分配不公平而受到损害。

本节探讨如何利用博弈论模型实现第三方云服务器的公平性资源分配。通过设计非合作博弈模型，采用拉格朗日乘数法求解纳什均衡点，提出了一种动态资源分配算法。该算法可以根据每个用户的请求特性和优先级合理分配计算资源，在保证用户资源分配公平性的同时，最大化系统整体效率，确保每个用户的计算任务能够及时完成，提升系统的响应能力和用户体验。

1) 模型设计

本文提出以下3个假设：① 用户的资源需求是已知的且可能动态变化；② 资源总量是有限的；③ 用户独立选择策略（申请的资源量），没有合作行为。用户的收益函数定义为用户实际获得的资源与所需资源的匹配度，如式(1)所示。

$$U_i = f\left(R_i, A_i\right) \tag{1}$$

其中，U_i表示第i个用户的收益，R_i表示第i个用户的资源需求，A_i表示第i个用户实际获得的资源。假设有N个用户和总量为C的资源，定义满意度函数为

$$U_i = \frac{A_i}{R_i} \tag{2}$$

约束条件为

$$\sum_{i=1}^{N} A_i \leqslant C \tag{3}$$

$$A_i \geqslant 0 \tag{4}$$

2) 纳什均衡

纳什均衡是指在所有用户选择最优策略的情况下，没有用户可以通过改变策略来提高收益。首先，构建拉格朗日函数为

$$L\left(A_1, A_2, \cdots, A_N, \lambda\right) = \sum_{i=1}^{N} U_i + \lambda\left(C - \sum_{i=1}^{N} A_i\right) \tag{5}$$

其中，λ为拉格朗日乘数。

其次，求解一阶条件为

$$\frac{\partial L}{\partial A_i} = \frac{1}{R_i} - \lambda = 0 \tag{6}$$

$$\frac{\partial L}{\partial \lambda} = C - \sum_{i=1}^{N} A_i = 0 \tag{7}$$

解得

$$A_i = \frac{CR_i}{\sum_{j=1}^{N} R_i} \tag{8}$$

3) 动态资源分配算法

设计基于比例公平的动态资源分配算法，考虑用户需求和资源总量，通过迭代调整达到均衡状态。

首先，初始化每个用户的初始资源分配量，表示为

$$A_i^{(0)} = \frac{C}{N} \tag{9}$$

其次，计算每个用户的资源缺口，表示为

$$G_i^{(k)} = R_i - A_i^{(k)} \tag{10}$$

最后，迭代调整资源分配量，优先满足资源缺口大的用户并更新资源分配量，表示为

$$A_i^{(k+1)} = A_i^{(k)} + \alpha G_i^{(k)} \tag{11}$$

其中，α表示调整步长。

4) 应用示例

假设有一个医疗研究机构，多个研究团队希望共享患者的医疗数据用于统计分析，而数据由不可信的云服务器进行存储和处理。在这种情况下，动态资源分配算法有以下优势。

① 根据每个研究团队的数据处理需求，动态分配云服务器的计算资源，确保数据分析任务能够及时完成。

② 优化代理重加密操作的执行顺序和资源的合理分配，有效提升数据处理效率，最大限度地提高系统的整体性能。

③ 管理和监控数据访问，确保数据的隐私和安全性不受损害。

3 多用户可验证同态代理重加密方案

3.1 形式化定义

一个多用户可验证同态代理重加密方案可以形式化定义为一个八元组（Setup, Extract, Encrypt, RKGen, ReEncrypt, Eval, Verify, Decrypt）。特别地，一个授权函数同态签名（AFHS, authorized function homomorphic signature）方案主要由以下5个时间多项式算法（AFHS.Setup, AFHS.Sign, AFHS.Verify, AFHS.KeyGen, AFHS.Eval）构成。其中，定义用户身份空间为I、消息空间为$\mathcal{M}$、数据集标签为$\tau\in\{0,1\}^*$以及电路函数族为$\mathcal{F}=\{f:\mathcal{M}^k\to\mathcal{M}\}$，且函数支持的最大数据集大小为$k$。

1) AFHS.Setup$(1^\lambda,k)$。输入安全参数1^λ及数据集大小k，输出密钥对(SK,PK)。数据拥有者保存主私钥为MSK，发布验证密钥为vk。

2) AFHS.KeyGen$(\mathrm{MSK},\mathrm{ID},f,\tau)$。① 输入主私钥MSK及用户身份ID$\in I$，输出用户身份ID对应的签名密钥$\mathrm{SK}_{\mathrm{ID}}$。② 输入函数$f\in\mathcal{F}$及数据集标签$\tau$，输出函数$f$和数据集标签$\tau$对应的签名密钥$\mathrm{SK}_{f,\tau}$。

3) AFHS.Sign$(\mathrm{SK}_{\mathrm{ID}},m_i,i,\tau)$。输入签名密钥$\mathrm{SK}_{\mathrm{ID}}$、数据集标签$\tau$和消息$m_i$，$i\in[1,k]$，输出消息$m_i$对应的签名$\sigma_i$，并记为$\boldsymbol{m}=(m_1,m_2,\cdots,m_k)$，$\boldsymbol{\sigma}=(\sigma_1,\sigma_2,\cdots,\sigma_k)$。

4) AFHS.Eval$(\mathrm{vk},\mathrm{ID},\mathrm{SK}_{f,\tau},\tau,f,(\boldsymbol{m},\boldsymbol{\sigma}))$。将验证密钥vk、用户身份ID、签名密钥$\mathrm{SK}_{f,\tau}$、数据集标签$\tau$、函数$f$、消息集$\boldsymbol{m}=(m_1,m_2,\cdots,m_k)$以及其对应签名集$\boldsymbol{\sigma}=(\sigma_1,\sigma_2,\cdots,\sigma_k)$作为输入，算法在概率多项式时间内输出签名向量$\boldsymbol{\sigma}'$以及消息向量$\boldsymbol{m}'=f(\boldsymbol{m})$。

5) AFHS.Verify$(\mathrm{vk},\mathrm{ID},\tau,(\boldsymbol{m}',\boldsymbol{\sigma}'),f)$。输入验证密钥vk、用户身份ID、数据集标签$\tau$、消息向量$\boldsymbol{m}'$及其对应签名向量$\boldsymbol{\sigma}'$和函数$f$，算法在概率多项式时间内输出$b\in\{0,1\}$。如果$b=1$，则验证成功；否则验证失败。

正确性。对于任意输入参数λ、k、$m\leftarrow\mathcal{M}$、$f\leftarrow\mathcal{F}$、AFHS.Setup$(1^\lambda,k)\to(\mathrm{MSK},\mathrm{vk})$、AFHS.KeyGen$(\mathrm{MSK},\mathrm{ID},f,\tau)\to\mathrm{SK}_{f,\tau}$和AFHS.Sign$(\mathrm{SK}_{\mathrm{ID}},m_i,i,\tau)\to\sigma_i$，均有式(12)成立，则说明该AFHS方案是正确的（此处省略AFHS）。

$$\Pr\left[\mathrm{Verify}\left(\mathrm{vk},\tau,f,\mathrm{Eval}\left(\mathrm{vk},\mathrm{SK}_{f,\tau},\tau,f,(\boldsymbol{m},\boldsymbol{\sigma})\right)=1\right)\right]=1 \tag{12}$$

下面以AFHS方案为基础，给出多用户可验证同态代理重加密方案的定义。

1) 初始化算法Setup$(1^\lambda,1^L)$。输入安全参数λ和方案能进行同态计算的最大乘法电路层数$L\in\mathbb{N}$，输出公共参数pp和主私钥msk。

2) 私钥提取算法Extract$(\mathrm{pp},\mathrm{msk},\mathrm{id})$。① 输入共参数pp及用户身份id，输出id的私钥$\mathrm{sk}_{\mathrm{id}}$。② 运行AFHS.KeyGen算法，输出用户身份的签名密钥$\mathrm{SK}_{\mathrm{id}}$及函数的签名密钥$\mathrm{SK}_{f,\tau}$。

3) 加密算法Encrypt$(\mathrm{pp},m,\mathrm{id})$。输入公共参数pp、待加密的消息$m$以及用户身份id，输出密文$\boldsymbol{c}_{\mathrm{id}}$。

4) 重加密密钥生成算法RKGen$(\mathrm{pp},\mathrm{id}_1,\mathrm{id}_2,\mathrm{sk}_{\mathrm{id}_1})$。① 输入公共参数pp、用户身份$\mathrm{id}_1$与$\mathrm{id}_2$及用户$\mathrm{id}_1$的私钥$\mathrm{sk}_{\mathrm{id}_1}$，输出从用户$\mathrm{id}_1$到用户$\mathrm{id}_2$的重加密密钥$\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}$。② 运行AFHS.Sign$(\mathrm{SK}_{\mathrm{id}},m_i,i,\tau)$算法，输出重加密密钥的签名$\sigma_i$。

5) 重加密算法ReEncrypt$(\mathrm{pp},\boldsymbol{c}_{\mathrm{id}_1},\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2})$。① 输入公共参数pp、重加密密钥$\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}$以及用户身份$\mathrm{id}_1$的密文$\boldsymbol{c}_{\mathrm{id}_1}$，输出用户身份$\mathrm{id}_2$的密文$\boldsymbol{c}_{\mathrm{id}_2}$。② 运行AFHS.Eval$(\mathrm{vk},\mathrm{ID},\mathrm{SK}_{f,\tau},\tau,f,(\boldsymbol{m},\boldsymbol{\sigma}))$算法，输出签名向量$\boldsymbol{\sigma}'$。

6) 密文计算算法Eval$(\mathrm{pp},\boldsymbol{c}_{\mathrm{id}_1},\boldsymbol{c}_{\mathrm{id}_2},f)$。输入公共参数pp、密文和$\boldsymbol{c}_{\mathrm{id}_1}$，$\boldsymbol{c}_{\mathrm{id}_2}$函数，输出密文的计算结果$\boldsymbol{c}'$。

7) 重加密验证算法Verify$(\mathrm{vk},\boldsymbol{c}_{\mathrm{id}_1},\boldsymbol{c}_{\mathrm{id}_2},f,\boldsymbol{\sigma}')$。输入验证密钥$\mathrm{vk}_{\mathrm{id}_1\to\mathrm{id}_2}$，如果$\boldsymbol{c}_{\mathrm{id}_2}$确实是原始密文$c_{\mathrm{id}_1}$经过转换后得到的，且是经过函数$f$计算得到的，那么算法输出$b=1$，否则输出$b=0$。

8) 解密算法Decrypt$(\mathrm{pp},\boldsymbol{c}_{\mathrm{id}},\mathrm{sk}_{\mathrm{id}})$。输入公共参数pp、密文$\boldsymbol{c}_{\mathrm{id}}$以及私钥$\mathrm{sk}_{\mathrm{id}}$，输出明文$m$或者终止$\perp$。

若一个多用户可验证同态代理重加密方案是正确的，则应该满足以下2个条件。

① 解密正确性。

$$\begin{aligned}&\mathrm{Decrypt}(\mathrm{pp},\mathrm{sk}_{\mathrm{id}_1},\boldsymbol{c}_{\mathrm{id}_1})=m\\&\mathrm{Decrypt}(\mathrm{pp},\mathrm{sk}_{\mathrm{id}_2},\mathrm{ReEncrypt}(\mathrm{pp},\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2},\boldsymbol{c}_{\mathrm{id}_1}))=m\\&\mathrm{Decrypt}(\mathrm{pp},\mathrm{sk}_i,\boldsymbol{c}')=f(m_1,\cdots,m_k)\end{aligned} \tag{13}$$

② 验证正确性。对于$\boldsymbol{c}_{\mathrm{id}_1}$和$\boldsymbol{c}_{\mathrm{id}_2}$，有

$$\Pr\left[\mathrm{Verify}\left(\mathrm{vk}_{\mathrm{id}_1\to\mathrm{id}_2},\boldsymbol{c}_{\mathrm{id}_1},\boldsymbol{c}_{\mathrm{id}_2},f\right)=1\right]=1 \tag{14}$$

3.2　安全模型

定义 1　基于身份的可验证同态代理重加密方案的IND-CPA安全性通过敌手$\mathcal{A}$与挑战者$\mathcal{C}$之间的交互游戏来定义，具体过程如下。

初始化。对于给定的安全参数λ及随机数$\varphi\in\{0,1\}$，运行$(\mathrm{pp},\mathrm{msk})\leftarrow\mathrm{Setup}(1^{\lambda},1^{L})$算法，挑战者保存主私钥msk，并将公共参数pp发送给敌手。

阶段1。敌手向挑战者发起用户私钥询问、重加密密钥询问、重加密询问和重加密验证询问。具体询问过程如下。

1) 私钥询问。敌手将用户身份id发送给挑战者，挑战者运行$\mathrm{Extract}(\mathrm{pp},\mathrm{msk},\mathrm{id})$算法生成用户身份对应的密钥$\mathrm{sk}_{\mathrm{id}}$，并将该密钥发送给敌手。

2) 重加密密钥询问。挑战者首先运行$\mathrm{Extract}(\mathrm{pp},\mathrm{msk},\mathrm{id})$算法，生成用户$\mathrm{id}_1$的私钥$\mathrm{sk}_{\mathrm{id}_1}$。其次运行$\mathrm{RKGen}\left(\mathrm{pp},\mathrm{id}_1,\mathrm{id}_2,\mathrm{sk}_{\mathrm{id}_1}\right)$算法，并将重加密密钥$\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}$返回给敌手。

3) 重加密询问。敌手将$\left(\mathrm{id}_1,\mathrm{id}_2,\boldsymbol{c}_{\mathrm{id}_1}\right)$发送给挑战者，挑战者运行$\mathrm{ReEncrypt}\left(\mathrm{pp},\boldsymbol{c}_{\mathrm{id}_1},\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}\right)$算法生成用户身份$\mathrm{id}_2$的重加密密文$\boldsymbol{c}_{\mathrm{id}_2}$。

4) 重加密验证询问。挑战者运行$\mathrm{Verify}(\mathrm{vk},\boldsymbol{c}_{\mathrm{id}_1},\boldsymbol{c}_{\mathrm{id}_2},f,\boldsymbol{\sigma}')$算法，根据返回结果判断重加密密文$\boldsymbol{c}_{\mathrm{id}_2}$是否合法有效。

挑战。敌手选择一个目标用户身份id^*和2个等长的消息$m_0,m_1\in\mathcal{M}$，$(m_0\neq m_1)$，并将(id^*,m_0,m_1)提交给挑战者。挑战者运行$\mathrm{Encrypt}(\mathrm{pp},m,\mathrm{id}^*)$算法，并将密文$\boldsymbol{c}^*_{\mathrm{id}^*}$返回给敌手，其中，$\varphi\in\{0,1\}$是随机选择的。

限制条件。挑战者身份id^*从来没有被$\mathrm{Extract}(\mathrm{pp},\mathrm{msk},\mathrm{id}^*)$询问过，且对于任意接收者的身份$\mathrm{id}^*$，也从未被$\mathrm{Extract}(\mathrm{pp},\mathrm{msk},\mathrm{id}')$和$\mathrm{RKGen}\left(\mathrm{pp},\mathrm{id}^*,\mathrm{id}',\mathrm{sk}_{\mathrm{id}^*}\right)$询问过。

阶段2。敌手继续按照阶段1的方式对用户身份$\mathrm{id}\neq\mathrm{id}^*$进行询问，挑战者的回复也与阶段1相同。但是敌手不能进行以下查询：1) $\mathrm{Extract}(\mathrm{pp},\mathrm{msk},\mathrm{id}^*)$；2) 对于任意接收者的身份$\mathrm{id}'$进行$\mathrm{Extract}(\mathrm{pp},\mathrm{msk},\mathrm{id}')$和$\mathrm{RKGen}\left(\mathrm{pp},\mathrm{id}^*,\mathrm{id}',\mathrm{sk}_{\mathrm{id}^*}\right)$询问；3)连续调用重加密密钥生成算法，形成初始身份id'和终止身份id''的查询链，并对任意身份id''进行$\mathrm{Extract}(\mathrm{pp},\mathrm{msk},\mathrm{id}'')$询问。

猜测阶段。敌手输出一个猜测$\varphi'\in\{0,1\}$，如果$\varphi'=\varphi$，那么敌手赢得游戏。其中，敌手的优势为

$$\mathrm{Adv}_{\mathcal{A}}^{\text{IND-CPA}}(\mathrm{id}^*)=\left|\Pr[\varphi'=\varphi]-\frac{1}{2}\right| \tag{15}$$

如果对于任意概率多项式时间攻击者$\mathcal{A}$来说，$\mathrm{Adv}_{\mathcal{A}}^{\text{IND-CPA}}(\mathrm{id}^*)$是可忽略的，则称该基于身份的可验证同态代理重加密方案是IND-CPA安全。

3.3　方案设计

为满足多用户进行密文运算的需求，首先，对不同用户上传的密文进行重加密，将其转化为云服务器的密文。其次，利用重加密机制使计算结果可选择地发送给指定且合法的非数据提供者，且其他用户即使获得了数据提供者的原始密文也无法对其进行解密。最后，将授权函数同态签名AFHS作为黑盒，验证计算结果的正确性与云服务器的计算行为。具体如下。

首先，执行算法1——系统建立算法，输入安全参数后生成主私钥，具体介绍如下。

算法 1　$\mathrm{Setup}(1^{\lambda},1^{L})$

输入　λ,L,k,m,n,q,χ

输出　$\mathrm{pp}=(n,q,\chi,m,\boldsymbol{A}_0,\boldsymbol{A}_1,\boldsymbol{A}_2,\boldsymbol{u})$，$\mathrm{msk}=\boldsymbol{T}_{\boldsymbol{A}_0}$

1) $\boldsymbol{A}_0\in\mathbb{Z}_q^{n\times m},\boldsymbol{T}_{\boldsymbol{A}_0}\in\mathbb{Z}_q^{m\times m}\leftarrow\mathrm{TrapGen}(q,n)$

2) $\boldsymbol{u}\leftarrow\mathbb{Z}_q^n$

3) $\boldsymbol{A}_1,\boldsymbol{A}_2\leftarrow\mathbb{Z}_q^{n\times m}$

输入用户身份以及算法1产生的公共参数和主私钥后，运行算法2——私钥提取算法，得到用户的公私钥对。

算法 2　$\mathrm{Extract}(\mathrm{pp},\mathrm{msk},\mathrm{id})$

输入　pp,msk,id

输出　$\mathrm{pk}_{\mathrm{id}}=\boldsymbol{B}_{\mathrm{id}},\mathrm{sk}_{\mathrm{id}}=\boldsymbol{t}$

1) $\boldsymbol{P}_{\mathrm{id}}=\left[\boldsymbol{A}_0\middle|\boldsymbol{A}_1+\boldsymbol{H}(\mathrm{id})\boldsymbol{A}_2\right]\in\mathbb{Z}_q^{n\times 2m}$

2) $\boldsymbol{s}\leftarrow\mathrm{SL}\left(\boldsymbol{A}_0,\boldsymbol{T}_{\boldsymbol{A}_0},\boldsymbol{A}_1+\boldsymbol{H}(\mathrm{id})\boldsymbol{A}_2,\boldsymbol{u},\sigma\right)$，其中，SL为SampleLeft算法

3) $\boldsymbol{P}_{\mathrm{id}}\boldsymbol{s}=\boldsymbol{u}$

4) $\boldsymbol{t}=\begin{pmatrix}1\\-\boldsymbol{s}^{\mathrm{T}}\end{pmatrix}\in\mathbb{Z}_q^{1+2m}$

5) $\boldsymbol{B}_{\mathrm{id}}=\left(\boldsymbol{u}|\boldsymbol{P}_{\mathrm{id}}\right)\in\mathbb{Z}_q^{n\times(1+2m)}$

6) $\boldsymbol{B}_{\mathrm{id}}\boldsymbol{t}=0$

使用公钥并执行算法3——加密算法，将用户原始明文数据变成密文数据。

算法3 $\mathrm{Encrypt}\left(\mathrm{pp},m,\mathrm{id}\right)$

输入 $\mathrm{pp},\mathrm{id},m\leftarrow\{0,1\}$

输出 $\boldsymbol{c}_{\mathrm{id}_1}$

1) $\boldsymbol{e}_i\leftarrow\chi^{(1+2m)\times 1}$

2) $\boldsymbol{r}\leftarrow\{-1,1\}^{n\times 1}$

3) $\boldsymbol{c}_{\mathrm{id}_1}=\boldsymbol{B}_{\mathrm{id}_1}^{\mathrm{T}}\boldsymbol{r}+\boldsymbol{e}_i+m\left(2^{\left(l_q-1\right)}\boldsymbol{x}_1\right)\left(\mathrm{mod}\,q\right)$，其中，$\boldsymbol{c}_{\mathrm{id}_1}\in\mathbb{Z}_q^{(1+2m)\times 1}$，$\boldsymbol{x}_1=(1,0,0,\cdots,0)$

注：这里仅使用工具矩阵$\boldsymbol{G}$的一部分，而不是整个工具矩阵，故将$\left(2^{\left(l_q-1\right)}\boldsymbol{x}_1\right)$记为$\tilde{\boldsymbol{G}}$。

进一步，数据拥有者使用自己的私钥，并根据云服务器和数据使用者的公钥通过算法4——重加密密钥生成算法，计算相应的重加密密钥并产生签名，为后续验证做准备。

算法4 $\mathrm{RKGen}\left(\mathrm{pp},\mathrm{id}_2,\mathrm{sk}_{\mathrm{id}_1}\right)$

输入 $\mathrm{sk}_{\mathrm{id}_1}=\boldsymbol{t}_i,\mathrm{pk}_{\mathrm{id}_o}=\boldsymbol{B}_{\mathrm{id}_o},\mathrm{pk}_{\mathrm{id}_2}=\boldsymbol{B}_{\mathrm{id}_2}$，其中，$\mathrm{pk}_{\mathrm{id}_o}$为云服务器的公钥

输出 $\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2},\sigma_i$

1) $\boldsymbol{Z}=\boldsymbol{r}^{\mathrm{T}}\boldsymbol{B}_{\mathrm{id}_o}\in\mathbb{Z}_q^{1\times(1+2m)}$

2) $\boldsymbol{X}\to\mathbb{Z}_q^{(1+2m)\times n}$

3) $\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_o}=\left(\mathrm{Power2}\left(\boldsymbol{t}_i\right)\boldsymbol{Z}\quad\boldsymbol{XG}\right)$

4) $\boldsymbol{e}_1,\boldsymbol{e}_2\leftarrow\chi^n$

5) $\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}=\left(\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{B}_{\mathrm{id}_2}\quad\boldsymbol{e}_2^{\mathrm{T}}-\mathrm{Power2}\left(\boldsymbol{t}_i^{\mathrm{T}}\right)\right)$

6) $\mathrm{SK}_{\mathrm{id}_1}\leftarrow\mathrm{AFHS.KeyGen}\left(\mathrm{MSK},f,\tau\right)$，其中，$\mathrm{SK}_{\mathrm{id}_1}$表示签名密钥

7) $\sigma_i\leftarrow\mathrm{AFHS.Sign}\left(\mathrm{SK}_{\mathrm{id}_1},x_i,i,\tau\right)$，其中，$x_i$表示$\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}$的每一行

基于重加密密钥，执行算法5——重加密算法，将原始密文分别转化为云服务器和数据使用者可解密的密文，之后执行签名的同态评估算法。

算法5 $\mathrm{ReEncrypt}\left(\mathrm{pp},\boldsymbol{c}_{\mathrm{id}_1},\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}\right)$

输入 $\mathrm{pp},\boldsymbol{c}_{\mathrm{id}_1},\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_o},\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}$

输出 $\boldsymbol{c}_{\mathrm{id}_2},\sigma_{\mathrm{id}_1\to\mathrm{id}_2}$

1) $\boldsymbol{c}_{\mathrm{id}_o}=\boldsymbol{G}^{-1}\left(\boldsymbol{c}_{\mathrm{id}_1}\right)\boldsymbol{X}^{-1}\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_o}$

2) $\boldsymbol{c}_{\mathrm{id}_2}=\mathrm{Bit}\left(\boldsymbol{c}_{\mathrm{id}_1}\right)\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}$

3) $\boldsymbol{\sigma}'\leftarrow\mathrm{AFHS.Eval}\left(\mathrm{pk},\mathrm{SK}_{f,\tau},\tau,f_{\boldsymbol{c}_{\mathrm{id}_1}},(\boldsymbol{m},\boldsymbol{\sigma})\right)$，其中，

$f_{\boldsymbol{c}_{\mathrm{id}_1}}\left(\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}\right)=\mathrm{Bit}\left(\boldsymbol{c}_{\mathrm{id}_1}\right)\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}$

然后，将密文与函数输入算法6——同态评估算法，实现密文同态计算。

算法6 $\mathrm{Eval}\left(\mathrm{pp},\boldsymbol{c}_{\mathrm{id}_1},\boldsymbol{c}_{\mathrm{id}_2},f\right)$

输入 $\mathrm{pp},\boldsymbol{c}_{\mathrm{id}_1},\boldsymbol{c}_{\mathrm{id}_2},f$

输出 $\boldsymbol{c}_{\mathrm{Add}},\boldsymbol{c}_{\mathrm{Mult}}$

1) $\boldsymbol{c}_{\mathrm{Add}}\triangleq\boldsymbol{c}_{\mathrm{id}_1}+\boldsymbol{c}_{\mathrm{id}_2}$

2) $\boldsymbol{c}_{\mathrm{Mult}}\triangleq\boldsymbol{c}_{\mathrm{id}_1}\tilde{\boldsymbol{G}}^{-1}\left(\boldsymbol{c}_{\mathrm{id}_2}\right)$

验证服务器根据授权同态函数同态签名运行算法7——验证算法，判断云服务器的重加密操作是否正确。

算法7 $\mathrm{Verify}\left(\mathrm{vk},\boldsymbol{c}_{\mathrm{id}_1},\boldsymbol{c}_{\mathrm{id}_2}f_{\boldsymbol{c}_{\mathrm{id}_1}},\sigma_{\mathrm{id}_1\to\mathrm{id}_2}\right)$

输入 $\mathrm{vk},\left(\boldsymbol{c}_{\mathrm{id}_1},\varnothing\right),\left(\boldsymbol{c}_{\mathrm{id}_2},\sigma_{\mathrm{id}_1\to\mathrm{id}_2}\right)$

输出 0/1

1) $0/1\leftarrow\mathrm{AFHS.Verify}\left(\mathrm{vk},\mathrm{id}_2,\tau,\boldsymbol{c}_{\mathrm{id}_2},f_{\boldsymbol{c}_{\mathrm{id}_1}}\right)$

最后，不同用户利用算法8——解密算法得到明文信息。

算法8 $\mathrm{Decrypt}\left(\mathrm{pp},\boldsymbol{c}_{\mathrm{id}},\mathrm{sk}_{\mathrm{id}}\right)$

输入 $\mathrm{pp},\boldsymbol{c}_{\mathrm{id}},\mathrm{sk}_{\mathrm{id}}$

输出 m'

1) $m'=\left[\left\langle\boldsymbol{c}_{\mathrm{id}_1},\mathrm{sk}_{\mathrm{id}_1}\right\rangle\right]_q$

2) $\boldsymbol{c}'_{\mathrm{id}_1}=\left[\left\langle\boldsymbol{c}_{\mathrm{id}_o},\mathrm{sk}_{\mathrm{id}_o}\right\rangle\right]_q$

3) $m'=\left[\left\langle\boldsymbol{c}_{\mathrm{id}_2},\mathrm{sk}_{\mathrm{id}_2}\right\rangle\right]_q$

通过上述8个算法，云服务器将经过同态计算后的结果发送给各参与方，各参与方通过自己的私钥解密获得最终的计算结果，实现了多方云计算，解决了由于不同的公钥加密产生的密文无法进行同

态计算的问题。

3.4　正确性分析

1) 对于用户身份 id_1 的密文，可用其私钥 $\mathrm{sk}_{\mathrm{id}_1}=\boldsymbol{t}_i$ 解密，有

$$\left[\left\langle \boldsymbol{c}_{\mathrm{id}_1},\mathrm{sk}_{\mathrm{id}_1}\right\rangle\right]_q=\boldsymbol{t}_i^{\mathrm{T}}m\left(2^{\left(l_q-1\right)}\boldsymbol{x}_1\right)+\mathrm{noise}_1 \tag{16}$$

2) 对于云服务器的重加密密文 $\boldsymbol{c}_{\mathrm{id}_o}$，云服务器使用其私钥 $\mathrm{sk}_{\mathrm{id}_o}=\boldsymbol{t}_o$ 解密，有

$$\boldsymbol{c}_{\mathrm{id}_1}{}'=\left[\left\langle \boldsymbol{c}_{\mathrm{id}_o},\mathrm{sk}_{\mathrm{id}_o}\right\rangle\right]_q=\left[\left\langle \boldsymbol{G}^{-1}\left(\boldsymbol{c}_{\mathrm{id}_1}\right)\boldsymbol{X}^{-1}\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_o},\mathrm{sk}_{\mathrm{id}_o}\right\rangle\right]_q=\boldsymbol{c}_{\mathrm{id}_1} \tag{17}$$

由式(17)可知，云服务器使用自己的私钥解密后得到的是各参与方的密文，有效地保证了各参与方的隐私。

3) 对于重加密密文 $\boldsymbol{c}_{\mathrm{id}_2}$，用户身份 id_2 使用其私钥 $\mathrm{sk}_{\mathrm{id}_2}=\boldsymbol{t}_j$ 解密，有

$$\begin{aligned}&\left[\left\langle \boldsymbol{c}_{\mathrm{id}_2},\mathrm{sk}_{\mathrm{id}_2}\right\rangle\right]_q=\left[\left\langle \mathrm{Bit}\left(\boldsymbol{c}_{\mathrm{id}_1}\right)\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2},\mathrm{sk}_{\mathrm{id}_2}\right\rangle\right]_q=\\&\mathrm{Bit}\left(\boldsymbol{c}_{\mathrm{id}_1}\right)\left(\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{B}_{\mathrm{id}_2}\boldsymbol{t}_j+\boldsymbol{e}_2-\mathrm{Power2}\left(\boldsymbol{t}_i^{\mathrm{T}}\right)\right)=\\&\boldsymbol{t}_i^{\mathrm{T}}m\left(2^{\left(l_q-1\right)}\boldsymbol{x}_1\right)+\mathrm{noise}_2\end{aligned} \tag{18}$$

注：上述误差应满足 $\left\|\mathrm{noise}_i\right\|<\dfrac{q}{4}$，方可正确解密。

3.5　同态性分析

本文方案满足同态性，即对于某函数集合 F，$\forall f\in F$，都有

$$\mathrm{Decrypt}\left(\mathrm{sk}_{\mathrm{id}},\boldsymbol{c}\right)=f\left(m_1,m_2,\cdots,m_k\right) \tag{19}$$

1) 针对同态加法，计算用户身份 id_1 与用户身份 id_2 的密文并输出，如式(20)所示。

$$\begin{aligned}&\boldsymbol{c}_{\mathrm{Add}}=\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\left(\boldsymbol{c}_{\mathrm{id}_1}+\boldsymbol{c}_{\mathrm{id}_2}\right)=\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\boldsymbol{c}_{\mathrm{id}_1}+\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\boldsymbol{c}_{\mathrm{id}_2}=\\&\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\left(\boldsymbol{B}_{\mathrm{id}_1}^{\mathrm{T}}\boldsymbol{r}+\boldsymbol{e}_i+m_i\left(2^{\left(l_q-1\right)}\boldsymbol{x}_1\right)\right)+\\&\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\left(\mathrm{Bit}\left(\boldsymbol{c}_{\mathrm{id}_1}\right)\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}\right)=\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\left(m_i\left(2^{\left(l_q-1\right)}\boldsymbol{x}_1\right)\right)+\\&\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}m_j\left(2^{\left(l_q-1\right)}\boldsymbol{x}_1\right)+\mathrm{noise}_3=\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\left(m_i+m_j\right)\left(2^{\left(l_q-1\right)}\boldsymbol{x}_1\right)+\\&\mathrm{noise}_3=\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\left(m_i+m_j\right)\tilde{\boldsymbol{G}}+\mathrm{noise}_3\end{aligned} \tag{20}$$

2) 针对同态乘法，计算用户身份 id_1 与用户身份 id_2 的密文并输出，如式(21)所示。

$$\begin{aligned}&\boldsymbol{c}_{\mathrm{Mult}}=\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\boldsymbol{c}_{\mathrm{Mult}}=\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\left(\boldsymbol{c}_{\mathrm{id}_1}\tilde{\boldsymbol{G}}^{-1}\left(\boldsymbol{c}_{\mathrm{id}_2}\right)\right)=\\&\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}\left(\left(\boldsymbol{B}_{\mathrm{id}_1}^{\mathrm{T}}\boldsymbol{r}+\boldsymbol{e}_i+m_i\tilde{\boldsymbol{G}}\right)\tilde{\boldsymbol{G}}^{-1}\left(\boldsymbol{c}_{\mathrm{id}_2}\right)\right)=\\&\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}m_im_j\left(2^{\left(l_q-1\right)}\boldsymbol{x}_1\right)+\mathrm{noise}_4=\\&\mathrm{sk}_{\mathrm{id}}^{\mathrm{T}}m_im_j\tilde{\boldsymbol{G}}+\mathrm{noise}_4\end{aligned} \tag{21}$$

注：在同态操作中，密文的尺寸（或维度）随着操作次数而增加，通过引入密文压缩技术，将维度膨胀控制在可接受的范围内。此外，密文中的噪声随操作次数的增加而累积，通过引入噪声刷新技术，在每次操作后对密文进行“重置”，可以消除累积的噪声，保证噪声水平在可控范围内。

3.6　安全性分析

首先，从以下两方面考虑方案的安全性：1) 密文及计算结果对于云服务器是保密的，通过代理重加密将密文转化为云服务器可解密的密文，云服务器解密后为各参与方的密文，但是由于云服务器并不知道各参与方的密钥，所以不能获得最终的计算结果；2) 参与方彼此之间的密文是保密的，由于用户之间的密钥是不相同的，所以各个参与方无法通过自己掌握的信息获取其他参与方的输入。

其次，通过定义 2 和定理 1 证明本文方案达到了 IND-CPA 安全。

定义 2　对于随机均匀选择的向量 $\boldsymbol{s}\in\mathbb{Z}_q^n$，通过 $\left(\boldsymbol{a}\in\mathbb{Z}_q^n,\boldsymbol{b}=\langle\boldsymbol{a},\boldsymbol{s}\rangle+\boldsymbol{e}\bmod q\right)$ 获得分布 $A_{S,\chi}\in\mathbb{Z}_q^n\times\mathbb{Z}_q$，其中，错误向量 $\boldsymbol{e}\in\mathbb{Z}_q$ 是根据错误概率分布 $\chi\leftarrow\mathbb{Z}_q$ 随机选择的，若能够以不可忽略的概率区分分布 $A_{S,\chi}$ 与均匀分布 $\mathbb{Z}_q^n\times\mathbb{Z}_q$ 上的实例，则称之为 LWE（判定性）问题。

定理 1　若参数设置合理，且 LWE 问题是困难的，那么方案在标准模型下是 IND-CPA 安全。

证明　通过以下 4 个不可区分的游戏证明对于一个多项式时间敌手 $\mathcal{A}$ 尝试攻破本文方案的优势是可忽略的。

游戏 0。这是一个敌手 $\mathcal{A}$ 与挑战者 $\mathcal{C}$ 之间标准的 IND-CPA 方案的游戏。设敌手要攻击的用户身份为 id^*，挑战者如实地运行 $\mathrm{Setup}(\cdot)$、$\mathrm{Extract}(\cdot)$、$\mathrm{Encrypt}(\cdot)$ 和 $\mathrm{RKGen}(\cdot)$ 算法，敌手得到相应的公共参数 pp、诚实用户的私钥 $\mathrm{sk}_{\mathrm{id}}$、重加密密钥 $\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}$ 及密文 $\boldsymbol{C}$。敌手的优势为 $\mathrm{Adv}[\mathcal{A}]\doteq\left|\Pr[\varphi'=\varphi]\right|$。

游戏1。除参数矩阵$\boldsymbol{A}_1$设置不同之外，其余参数的选取与游戏0完全相同。具体地，在游戏0中，矩阵$\boldsymbol{A}_1$是从$\mathbb{Z}_q^{n\times m}$中随机选取，但在该游戏中，随机选取一个均匀矩阵$\boldsymbol{R}^*\leftarrow\{-1,1\}^{m\times m}$，而矩阵$\boldsymbol{A}_1=\boldsymbol{A}_0\boldsymbol{R}^*-H(\mathrm{id}^*)\boldsymbol{A}_2$通过计算得到，其中$H$为身份编码函数。

游戏0和游戏1的不可区分性。游戏1只是用$\boldsymbol{A}_0\boldsymbol{R}^*$代替了随机产生的矩阵$\boldsymbol{A}_1$，根据引理1，$\boldsymbol{A}_0\boldsymbol{R}^*$也是均匀分布的，对于多项式时间内的敌手而言，游戏0和游戏1的概率不可区分。

引理1 假设$mn>(n+1)\mathrm{lb}\,q+\omega(\lg n)$，其中，$q$是一个素数，矩阵$\boldsymbol{A}$和矩阵$\boldsymbol{B}$取自均匀分布的$\mathbb{Z}_q^{n\times m}$，$\boldsymbol{R}^*$为一个$m\times m$的方阵，其值随机取自$\{-1,1\}^{m\times m}$，那么对于所有向量$\boldsymbol{v}$，$(\boldsymbol{A},\boldsymbol{A}\boldsymbol{R}^*,\boldsymbol{R}^{*\mathrm{T}}\boldsymbol{v})$的分布在统计上与分布$(\boldsymbol{A},\boldsymbol{B},\boldsymbol{R}^{*\mathrm{T}}\boldsymbol{v})$是不可区分的。

游戏2。不再通过$\mathrm{TrapGen}(q,n)$算法抽样矩阵$\boldsymbol{A}_0$，而是均匀随机地选择一个矩阵$\boldsymbol{A}_0\leftarrow\mathbb{Z}_q^{n\times m}$，利用$\mathrm{TrapGen}(q,n)$算法产生带有陷门基$\boldsymbol{T}_{\boldsymbol{A}_2}$的矩阵$\boldsymbol{A}_2$，剩余参数与游戏1保持一致。在游戏2中，敌手可以进行如下询问。

1) 私钥询问

① 如果$\mathrm{id}\neq\mathrm{id}^*$，挑战者利用陷门$\boldsymbol{T}_{\boldsymbol{A}_2}$按照算法9生成用户的私钥$\mathrm{sk}_{\mathrm{id}}$。

算法9 $\mathrm{SimExtract}(\mathrm{pp},\mathrm{msk},\mathrm{id})$

输入 pp,msk,id

输出 $\mathrm{sk}_{\mathrm{id}}$

1) $\boldsymbol{P}_{\mathrm{id}}=\left[\boldsymbol{A}_0\middle|\boldsymbol{A}_1+H(\mathrm{id})\boldsymbol{A}_2\right]=$

$\left[\boldsymbol{A}_0\middle|\boldsymbol{A}_0\boldsymbol{R}^*+\left[H(\mathrm{id})-H(\mathrm{id}^*)\right]\boldsymbol{A}_2\right]\in\mathbb{Z}_q^{n\times 2m}$

2) $\boldsymbol{T}_{\boldsymbol{A}_2}=(\boldsymbol{a}_1,\boldsymbol{a}_2,\cdots,\boldsymbol{a}_m)\in\mathbb{Z}_q^{m\times m}$，其中，$\boldsymbol{a}_i$为$\boldsymbol{T}_{\boldsymbol{A}_2}$的第$i$列

3) $\boldsymbol{d}_i=(-\boldsymbol{R}^*\boldsymbol{a}_i;\boldsymbol{a}_i)\in\mathbb{Z}_q^{2m}$

4) $\boldsymbol{P}_{\mathrm{id}}\boldsymbol{d}_i=0$

5) 对于$m+1\leqslant i\leqslant 2m$，设$\boldsymbol{r}_i$为单位矩阵$\boldsymbol{I}_{m\times n}$的第$i-mn$列，选取一个向量$\boldsymbol{w}_i$

6) $\left[H(\mathrm{id})-H(\mathrm{id}^*)\right]\boldsymbol{A}_2=\boldsymbol{A}_0\boldsymbol{w}_i$

7) $\boldsymbol{d}_i^*=(\boldsymbol{w}_i-\boldsymbol{R}^*\boldsymbol{r}_i|\boldsymbol{r}_i)\in\mathbb{Z}_q^{2m^*}$

8) $\boldsymbol{P}_{\mathrm{id}}\boldsymbol{d}_i^*=0$

9) $\boldsymbol{T}_{\boldsymbol{P}_{\mathrm{id}}}=(\boldsymbol{p}_1,\boldsymbol{p}_2,\cdots,\boldsymbol{p}_{2m})\in\mathbb{Z}_q^{2m\times 2m}$，其中，$\left\|\overline{\boldsymbol{T}_{\boldsymbol{P}_{id}}}\right\|\leqslant\left\|\boldsymbol{T}_{\boldsymbol{A}_2}\right\|\sqrt{m}\,\omega(\sqrt{\lg m})$，$\overline{\boldsymbol{T}_{\boldsymbol{P}_{\mathrm{id}}}}$中$2mn$个向量线性无关。

10) $\boldsymbol{s}\leftarrow\mathrm{SR}(\mathrm{id},\boldsymbol{A}_0,\boldsymbol{A}_1,\boldsymbol{A}_2,\boldsymbol{T}_{\boldsymbol{A}_2},\boldsymbol{u}_0,\sigma)$，其中，SR为SampleRight算法

11) $\boldsymbol{P}_{\mathrm{id}}\boldsymbol{s}=\boldsymbol{u}_0$

12) $\boldsymbol{t}=\begin{pmatrix}1\\-\boldsymbol{s}^{\mathrm{T}}\end{pmatrix}\in\mathbb{Z}_q^{1+2m}$

13) $\boldsymbol{B}_{\mathrm{id}}=(\boldsymbol{u}|\boldsymbol{P}_{\mathrm{id}})\in\mathbb{Z}_q^{n\times(1+2m)}$

14) $\boldsymbol{B}_{\mathrm{id}}\boldsymbol{t}=0$

15) $\mathrm{sk}_{\mathrm{id}}=\boldsymbol{t}$

② 如果$\mathrm{id}=\mathrm{id}^*$，则输出一个随机比特，并终止游戏。

2) 重加密密钥询问

当敌手发起该预言机询问时，考虑如下2种情况。① 当$\mathrm{id}\neq\mathrm{id}^*$，首先运行算法9生成用户私钥$\mathrm{sk}_{\mathrm{id}}$。其次，构造重加密密钥$\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}=\left(\boldsymbol{e}_1\boldsymbol{B}_j\boldsymbol{e}_2-\mathrm{Power2}(\mathrm{sk}_{\mathrm{id}_1})\right)$，其中，$\boldsymbol{e}_1,\boldsymbol{e}_2\leftarrow\chi^n$为噪声向量。② 如果$\mathrm{id}=\mathrm{id}^*$，则输出一个随机比特，并终止游戏。

3) 重加密询问

敌手将$(\mathrm{id}_1,\mathrm{id}_2,\boldsymbol{c}_{\mathrm{id}_1})$发送给挑战者，挑战者运行$\mathrm{ReEncrypt}(\mathrm{pp},\boldsymbol{c}_{\mathrm{id}_1},\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2})$算法生成用户身份$\mathrm{id}_2$的重加密密文$\boldsymbol{c}_{\mathrm{id}_2}=\mathrm{Bit}(\boldsymbol{c}_{\mathrm{id}_1})\mathrm{rk}_{\mathrm{id}_1\to\mathrm{id}_2}$。

4) 重加密验证询问

挑战者如实地运行$\mathrm{Verify}(\mathrm{vk},\boldsymbol{c}_{\mathrm{id}_1},\boldsymbol{c}_{\mathrm{id}_2},f_{\boldsymbol{c}_{\mathrm{id}_1}},\sigma)$算法，并将验证结果返回给敌手。

游戏1和游戏2的不可区分性。在敌手看来，私钥的生成过程不可见，且有

$$\begin{aligned}&\boldsymbol{P}_{\mathrm{id}}=\left[\boldsymbol{A}_0\middle|\boldsymbol{A}_1+H(\mathrm{id})\boldsymbol{A}_2\right]=\\&\boldsymbol{A}_0\middle|\boldsymbol{A}_0\boldsymbol{R}^*+\left[H(\mathrm{id})-H(\mathrm{id}^*)\right]\boldsymbol{A}_2=\\&\boldsymbol{A}_0\middle|\boldsymbol{A}_0\boldsymbol{R}^*-H(\mathrm{id}^*)\boldsymbol{A}_2+H(\mathrm{id})\boldsymbol{A}_2=\\&\boldsymbol{A}_0\middle|\boldsymbol{A}_1+H(\mathrm{id})\boldsymbol{A}_2\in\mathbb{Z}_q^{n\times 2m}\end{aligned}\tag{22}$$

因此，游戏2与游戏1的设置具有相同的形式，敌手所获得的私钥$\mathrm{sk}_{\mathrm{id}}$在统计上服从分布$D_{\mathbb{Z}^{1+2m},\sigma}$。对于重加密密钥，2个游戏中的重加密密钥都是通过$\mathrm{RKGen}(\mathrm{pp},\mathrm{id}_2,\mathrm{sk}_{\mathrm{id}_1})$算法生成的，因此，挑战者对于

重加密密钥的回答对敌手而言与游戏1是不可区分的。

游戏3。本游戏中的挑战密文与游戏2的不同之处在于不再使用$\mathrm{Encrypt}(\mathrm{pp},m,\mathrm{id})$算法得到，而是从密文空间中随机选取且相互独立的矩阵。

游戏2和游戏3的不可区分性。假设存在一个敌手，能够以不可忽略的概率区分游戏2和游戏3，那么就存在一个模拟者$\mathcal{S}$，能够解决判定性LWE困难假设。

假设存在一个LWE采样预言机$O(O_{\$}/O_S)$，其中，$O_{\$}$是真实的随机预言机，$O_S$是含有噪声的伪随机预言机。模拟者$\mathcal{S}$通过与敌手进行交互后回答这些采样是取自$O_{\$}$还是$O_S$。

实例化。模拟者$\mathcal{S}$向预言机O询问$m+1$次，并接收LWE样本$(\boldsymbol{u}_i,v_i)\in\mathbb{Z}_q^n\times\mathbb{Z}_q,0\leqslant i\leqslant m$。

攻击目标。敌手向模拟者宣布它要攻击的目标用户身份id^*。

初始化。令矩阵$\boldsymbol{B}_{\mathrm{id}}$的第$i$列为$\boldsymbol{u}_i$，则$\boldsymbol{B}_{\mathrm{id}}=(\boldsymbol{u}_1,\boldsymbol{u}_2,\cdots,\boldsymbol{u}_m)$，模拟者$\mathcal{S}$将公共参数$\mathrm{pp}=(n,q,\chi,m,\boldsymbol{A}_0,\boldsymbol{A}_1,\boldsymbol{A}_2,\boldsymbol{u})$发送给敌手$\mathcal{A}$。

阶段1。模拟者$\mathcal{S}$按照游戏2来回答来自敌手$\mathcal{A}$的各种询问。

挑战。敌手向模拟者发送一个消息位$b^*\in\{0,1\}$。模拟者设置$\boldsymbol{v}=[v_1,v_2,\cdots,v_m]$，计算挑战密文$\boldsymbol{c}^*_{\mathrm{id}^*}=\boldsymbol{v}+b^*\left\lfloor\frac{q}{2}\right\rfloor$，随机选取$\varphi\in\{0,1\}$。如果$\varphi=0$，将挑战密文$\boldsymbol{c}^*_{\mathrm{id}^*}$返回给敌手。如果$\varphi=1$，随机选择均匀分布的$\boldsymbol{c}^*_{\mathrm{id}^*}\in\mathbb{Z}_q^{(1+2m)\times 1}$作为挑战密文并返回给敌手$\mathcal{A}$。

阶段2。敌手$\mathcal{A}$继续发起询问，行为同阶段1。

猜测。敌手提交一个对φ的猜想$\varphi'\in\{0,1\}$。

如果$O=O_{\$}$，则挑战密文在$\mathbb{Z}_q^{(1+2m)\times 1}$中是均匀随机的，敌手$\mathcal{A}$猜测正确的$\varphi$的概率至多为$\frac{1}{2}$，模拟者$\mathcal{S}$输出1的概率至多为$\frac{1}{2}$。

如果$O=O_S$，有$\boldsymbol{v}=[v_1,v_2,\cdots,v_m]=(\boldsymbol{B}_{\mathrm{id}}^{\mathrm{T}}\boldsymbol{r}+\boldsymbol{e})\bmod q$，则挑战密文$\boldsymbol{C}^*$也是均匀随机的，这与采用$\mathrm{Encrypt}(\mathrm{pp},m,\mathrm{id})$算法得到的密文分布是相同的。故敌手猜测正确的$\varphi$的概率为$\frac{1+\varepsilon}{2}$，其中，$\varepsilon$为一个不可忽略的量。那么，模拟者$\mathcal{S}$输出1的概率也为$\frac{1+\varepsilon}{2}$。

综上可得，如果敌手$\mathcal{A}$能够以不可区分的优势区分游戏2和游戏3，那么模拟者$\mathcal{S}$至少能够以$\varepsilon'=\frac{1+\varepsilon}{2}-\frac{1}{2}=\frac{\varepsilon}{2}$的概率解决判定性LWE问题。由于判定性LWE问题是一个已知的公开困难问题，那么游戏2和游戏3是不可区分的。

通过以上4个等价游戏，本文方案在标准模型下满足IND-CPA安全。证毕。

3.7　性能分析

表2展示了相关类型方案与本文方案的性能对比结果，对比涉及是否支持多用户同态计算、云服务的重加密可验证性及公平性。从表2可以看出，文献[9]和文献[24]虽然具备全同态性质，但不支持多用户同态计算、可验证性和公平性。文献[10]、文献[11]、文献[12]、文献[25]和文献[26]虽然支持多用户同态计算，但无法实现云服务的可验证性和公平性。对于文献[17]而言，仅支持可验证性，可能无法适用于复杂的实际应用。尽管文献[19]实现了可验证和公平的多用户密文共享，却无法对密文进行计算。然而，本文方案不仅支持多用户的密文计算，而且提供了重加密的可验证性与云服务器的公平性方法，具有较大优势。

表2　　各方案性能对比

方案	多用户同态计算	同态性	可验证性	公平性
文献[9]	×	√	×	×
文献[10]	√	√	×	×
文献[11]	√	√	×	×
文献[12]	√	√	×	×
文献[17]	×	×	√	×
文献[19]	√	×	√	√
文献[24]	×	√	×	×
文献[25]	√	√	×	×
文献[26]	√	√	×	×
本文方案	√	√	√	√

为验证本文方案在存储效率上的优势，选取了与当前研究工作特性严格对齐的典型方案进行对比。具体而言，文献[10]、文献[11]、文献[12]和文献[26]均为基于格密码且支持多用户场景的同态代理重加密方案，表3展示了上述相关方案与本文方案中的密文和重加密密钥大小对比结果。其中，$N=(n+1)(\lfloor\mathrm{lb}\,q\rfloor+1)$，$M=(n+1)\lceil\mathrm{lb}\,q\rceil+m$。

表3　各方案密文和重加密密钥大小对比

方案	密文大小	重加密密钥大小
文献[10]	$O\left(n^2\mathrm{lb}\,q\right)$	$O\left(\left(n^3+n^2\right)\mathrm{lb}\,q\right)$
文献[11]	$O\left(N^2\mathrm{lb}\,q\right)$	$O\left(m(1+n)\mathrm{lb}\,q\right)$
文献[12]	$O\left((1+n)\mathrm{lb}\,q\right)$	$O\left(M(n+1)\mathrm{lb}\,q\right)$
文献[26]	$O\left((1+n)\mathrm{lb}\,q\right)$	$O\left((1+m)(1+n)\mathrm{lb}\,q\right)$
本文方案	$O\left((1+2m)\mathrm{lb}\,q\right)$	$O\left((1+2m+n)\mathrm{lb}\,q\right)$

由于本文方案使用了Mao等[21]提出的短密文技术，因此密文尺寸相较于文献[10]和文献[11]来说较小，略高于文献[12]和文献[26]。在重加密密钥大小方面，本文方案的重加密密钥尺寸最小，为$O\left((1+2m+n)\mathrm{lb}\,q\right)$。

综上所述，本文设计的基于身份的同态代理重加密方案不仅支持多用户密文计算、代理重加密可验证性和公平性，而且密文和重加密密钥尺寸较小，在云共享环境中具有较大优势。

4　实验分析

为测试本文方案的有效性，在一台操作系统为MacOS Monterey，处理器为2 GHz四核Intel Core i5，编译环境为Python 3.8的计算机上进行了实现。

4.1　公平性资源分配算法

实验分别测试5个和10个不同用户的资源函数分布及其满足纳什均衡时的资源分配情况，结果如图2~图5所示。图2和图3分别展示了5个不同用户的资源函数分布情况及其在满足纳什均衡条件下的资源分配结果。图4和图5分别展示了10个不同用户的资源函数分布及其对应的纳什均衡资源分配情况。

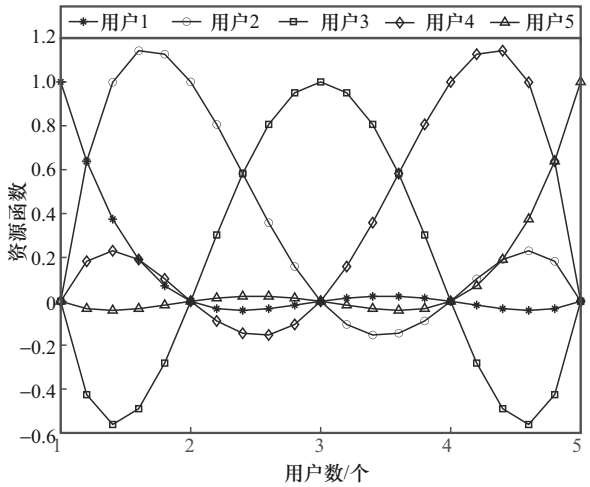

图2　5个不同用户的资源函数分布

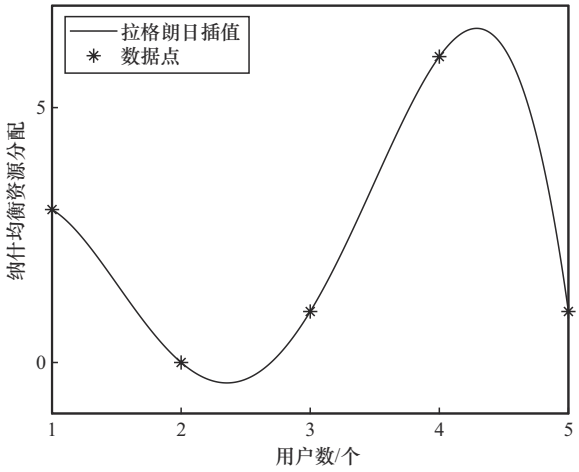

图3　5个不同用户满足纳什均衡的资源分配

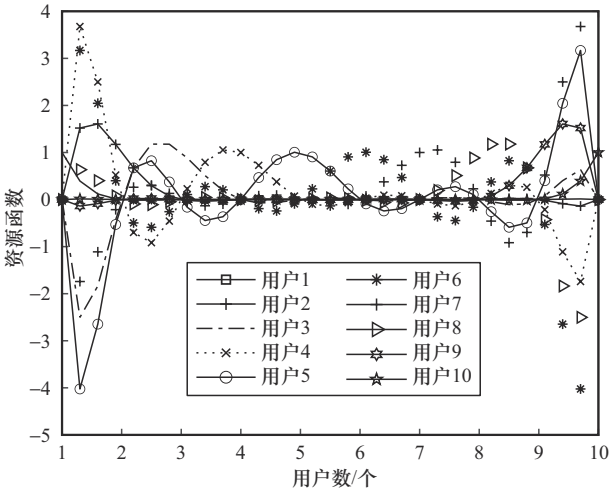

图4　10个不同用户的资源函数分布

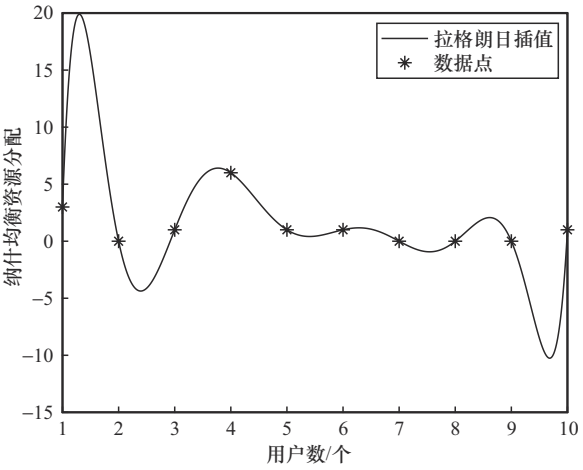

图5　10个不同用户满足纳什均衡的资源分配

通过对比分析这两组实验结果可以发现，资源需求差距较大的用户能够在分配过程中优先得到满足，并及时调整其资源分配量。此过程优化了系统

的资源利用率，实现了数据处理的最大化。同时，这种动态分配机制不仅提升了系统整体性能，还能有效管理和监控数据访问行为，从而更好地保障了数据的隐私性和安全性。实验结果表明，本文所提基于博弈论的公平动态资源分配算法在实际应用中与理论模型吻合，验证了其有效性与适用性。

4.2　多用户可验证同态代理重加密方案

首先，分别测试在 80 bit、128 bit、192 bit 和 256 bit 的安全级别下，本文方案与其他相关方案（文献[10]、文献[11]、文献[12]和文献[26]）的密文和重加密密钥大小，实验结果如图 6 和图 7 所示。

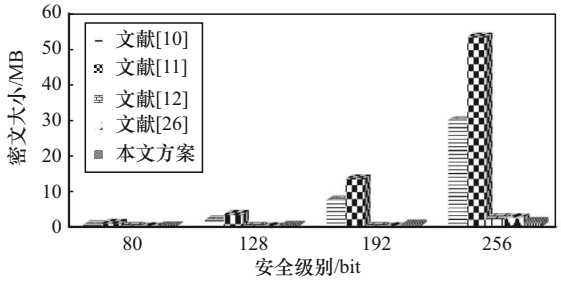

图6　密文大小随安全级别的变化

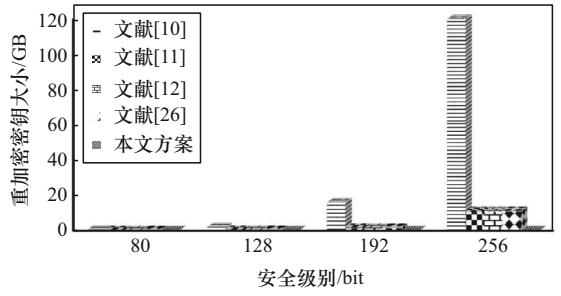

图7　重加密密钥大小随安全级别的变化

从图 6 可以看出，不同方案的密文大小随着安全级别的提高而增加。特别是文献[10]和文献[11]增长幅度较大，而文献[12]和本文方案增长速度较慢，在 256 bit 安全级别下，本文方案的密文大小为 1.38 MB。因此，相较于文献[10]和文献[11]，本文方案的密文尺寸较小，具有较大优势，与文献[12]和文献[26]中的密文大小接近，与理论分析一致。

从图 7 可以看出，文献[10]中的重加密密钥大小随着安全级别的提高而大幅增加，尤其是当达到 256 bit 安全级别时，重加密密钥需要占用 120 GB，严重影响了方案效率。文献[11]、文献[12]和文献[26]随着安全级别的增加而缓慢增加，而本文方案中的重加密密钥大小是 5 个方案中增长速度最慢，且占用空间最少，在 256 bit 安全级别下，仅占用 1.4 GB，与理论分析一致。

进一步地，当测试参数 n=12、14、16、18、20、22、24、26、28，$\lambda = n$ 时，执行本文方案的加解密算法时间和同态加法时间，如图 8~图 10 所示。

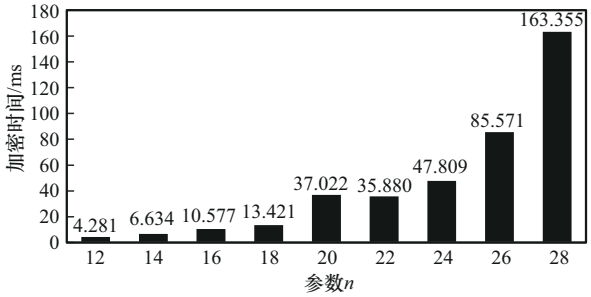

图8　加密时间随参数 n 的变化

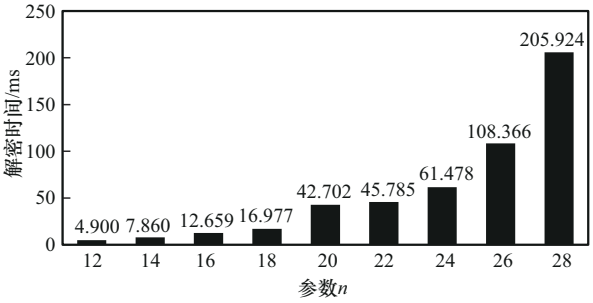

图9　解密时间随参数 n 的变化

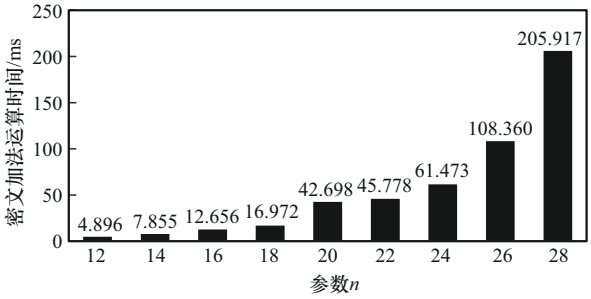

图10　密文加法运算时间随参数 n 的变化

从图 8~图 10 可以看出，本文方案的加密、解密和密文加法运算时间均随着参数 n 的不断增加而增加。特别地，当 $n = 20$ 时，各算法运行时间显著增加，当 $n = 28$ 时，解密时间和密文加法运算时间达到了 205 ms，这是由于方案的解密和同态加法运算的复杂度高于加密时的复杂度，因此需要花费更多的时间，但整体的运行结果是可以接受的，能够有效实现同态运算。

5 结束语

本文基于博弈论提出了一种公平性资源分配算法，并基于格设计了多用户可验证同态代理重加密方案，有效验证了云服务器的代理行为，解决了不同公钥加密的密文无法进行同态计算的问题，支持多方云计算和密文共享，并在标准模型下达到IND-CPA安全。理论分析和实验结果表明了本文方案的有效性。

下一步将深入探究公平性资源分配算法与同态代理重加密的有机融合，构建精确的公平性度量模型，实现更高效的基于环LWE问题的公平可验证多用户同态代理重加密方案。

参考文献：

[1] BLAZE M, BLEUMER G, STRAUSS M. Divertible protocols and atomic proxy cryptography[C]//International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 1998: 127-144.

[2] LUO F C, WANG H Y, SUSILO W, et al. Public trace-and-revoke proxy re-encryption for secure data sharing in clouds[J]. IEEE Transactions on Information Forensics and Security, 2024, 19: 2919-2934.

[3] PATIL R Y, PATIL Y H, BANNORE A, et al. Ensuring accountability in digital forensics with proxy re-encryption based chain of custody[J]. International Journal of Information Technology, 2024, 16(3): 1841-1853.

[4] ZHOU Y X, LIU S L, HAN S. Multi-hop fine-grained proxy re-encryption[C]//IACR International Conference on Public-Key Cryptography. Berlin: Springer, 2024: 161-192.

[5] ZHANG J, SU S S, ZHONG H, et al. Identity-based broadcast proxy re-encryption for flexible data sharing in VANETs[J]. IEEE Transactions on Information Forensics and Security, 2023, 18: 4830-4842.

[6] GENTRY C. Fully homomorphic encryption using ideal lattices[C]//Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing. New York: ACM Press, 2009: 169-178.

[7] GUPTA S, CAMMAROTA R, ŠIMUNIĆ T. MemFHE: end-to-end computing with fully homomorphic encryption in memory[J]. ACM Transactions on Embedded Computing Systems, 2024, 23(2): 1-23.

[8] WANG Z W, LI P N, HOU R, et al. HE-booster: an efficient polynomial arithmetic acceleration on GPUs for fully homomorphic encryption[J]. IEEE Transactions on Parallel and Distributed Systems, 2023, 34(4): 1067-1081.

[9] GENTRY C, SAHAI A, WATERS B. Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based[C]//Annual Cryptology Conference. Berlin: Springer, 2013: 75-92.

[10] MA C G, LI J Y, OUYANG W P. A homomorphic proxy re-encryption from lattices[C]//International Conference on Provable Security. Berlin: Springer, 2016: 353-372.

[11] 李陶深, 刘青, 黄汝维. 云环境中基于代理重加密的多用户全同态加密方案[J]. 清华大学学报(自然科学版), 2018, 58(2): 143-149.
LI T S, LIU Q, HUANG R W. Multi-user fully homomorphic encryption scheme based on proxy re-encryption for cloud computing[J]. Journal of Tsinghua University (Science and Technology), 2018, 58(2): 143-149.

[12] LI Z P, MA C G, WANG D. Achieving multi-hop PRE via branching program[J]. IEEE Transactions on Cloud Computing, 2020, 8(1): 45-58.

[13] BRAKERSKI Z, PERLMAN R. Lattice-based fully dynamic multi-key FHE with short ciphertexts[C]//Annual International Cryptology Conference. Berlin: Springer, 2016: 190-213.

[14] FAN J F, VERCAUTEREN F. Somewhat practical fully homomorphic encryption[J]. Cryptology ePrint Archive, 2012: 144-162.

[15] LUO F C, AL-KUWARI S, SUSILO W, et al. Chosen-ciphertext secure homomorphic proxy re-encryption[J]. IEEE Transactions on Cloud Computing, 2020, 10(4): 2398-2408.

[16] 李瑞琪, 贾春福, 王雅飞. 基于NTRU的多密钥同态代理重加密方案及其应用[J]. 通信学报, 2021, 42(3): 11-22.
LI R Q, JIA C F, WANG Y F. Multi-key homomorphic proxy re-encryption scheme based on NTRU and its application[J]. Journal on Communications, 2021, 42(3): 11-22.

[17] WU L Q, YANG X Y, ZHANG M Q, et al. IB-VPRE: adaptively secure identity-based proxy re-encryption scheme from LWE with re-encryption verifiability[J]. Journal of Ambient Intelligence and Humanized Computing, 2022, 13(1): 469-482.

[18] MANULIS M, NGUYEN J. Fully homomorphic encryption beyond IND-CCA1 security: integrity through verifiability[C]//International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2024: 63-93.

[19] GE C P, SUSILO W, BAEK J, et al. A verifiable and fair attribute-based proxy re-encryption scheme for data sharing in clouds[J]. IEEE Transactions on Dependable and Secure Computing, 2021, 19(5): 2907-2919.

[20] BUYYA R, YEO C S, VENUGOPAL S, et al. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility[J]. Future Generation Computer Systems, 2009, 25(6): 599-616.

[21] MAO M, HUMPHREY M. Auto-scaling to minimize cost and meet application deadlines in cloud workflows[C]//Proceedings of the 2011 International Conference for High Performance Computing, Networking, Storage and Analysis. Piscataway: IEEE Press, 2011: 1-12.

[22] LEE E A, SANGIOVANNI-VINCENTELLI A. A framework for comparing models of computation[J]. IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems, 1998, 17(12): 1217-1229.

[23] NAMAZI A, KHODABAKHSHI M. A novel game theoretic method on fair economic resource allocation with multiple Criteria[J]. Interna-

tional Journal of Management Science and Engineering Management, 2023, 18(3): 170-176.

[24] BRAKERSKI Z, GENTRY C, VAIKUNTANATHAN V. (Leveled) fully homomorphic encryption without bootstrap-ping[C]//Proceedings of the 3rd Innovations in Theoretical Com-puter Science Conference. New York: ACM Press, 2012: 309-325.

[25] SHEN X Y, LUO X, YUAN F, et al. Privacy-preserving multi-party deep learning based on homomorphic proxy re-encryption[J]. Journal of Systems Architecture, 2023, 144: 102983.

[26] ZHAO F X, WANG H X, WENG J. Constant-size unbounded multi-hop fully homomorphic proxy re-encryption from lattices[C]//European Symposium on Research in Computer Security. Berlin: Springer, 2024: 238-258.

[作者简介]

侯金秋（1994– ），女，安徽蚌埠人，博士，安徽财经大学讲师，主要研究方向为格密码、代理重加密、访问控制等。

彭长根（1963– ），男，贵州锦屏人，博士，贵州大学教授，主要研究方向为密码学、数据安全与隐私保护。

丁红发（1988– ），男，河南南阳人，博士，贵州财经大学副教授，主要研究方向为数据安全、密码算法与协议设计。

杨训（1988– ），男，贵州毕节人，博士，贵州工程应用技术学院讲师，主要研究方向为数据安全与隐私保护、算法博弈论、联邦学习等。