适用于 UTXO 模型定长存储的轻节点研究

刘蕊嘉,李　敏
(武汉科技大学 计算机科学与技术学院,湖北 武汉 430065)

摘　要:当前基于 UTXO 的区块链要求节点保留整个不断增长的 UTXO 集以验证交易,这对存储资源受限的设备构成了挑战,使网络趋向集中化。无状态区块链技术能通过累加器将庞大的 UTXO 集合组合成一个短且恒定大小的承诺,但交易处理中动态删除集合元素所需的计算开销巨大,且部分轻节点不支持提交交易功能。该文提出了一种适用于 UTXO 模型的定长存储的轻节点,该轻节点在恒定存储容量的条件下既能验证交易的合法性,也能提交交易。该方案使用了仅支持元素添加操作的两个数据结构替代 UTXO 集合:STXO 集合和 TXO 集合。有效的 UTXO 属于 TXO 集合但不在 STXO 集合内。通过 RSA 累加器和 MMR 分别构建新的 STXO 承诺和 TXO 承诺。轻节点只需保存最新的块承诺并与全节点交互,在降低存储开销的同时实现交易的合法性验证和提交。评估结果表明:轻节点能定长存储;轻节点仅需较低的计算负载即可实现交易验证;全节点在承诺更新过程展现了优化的性能。

关键词:UTXO;RSA 累加器;STXO 承诺;TXO 承诺;无状态区块链
中图分类号:TP311.13　　**文献标识码**:A　　**文章编号**:1673-629X(2025)05-0060-07
doi:10.20165/j.cnki.ISSN1673-629X.2024.0400

Research on Lightweight Nodes with Fixed-length Storage Based on UTXO Model

LIU Rui-jia, LI Min
(School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, China)

Abstract: The current blockchain based on the UTXO model requires nodes to retain the entire and ever-growing UTXO set for transaction verification, posing challenges to devices with limited storage resources and leading to network centralization. Stateless blockchain technology can combine the massive UTXO set into a short and fixed-size commitment through accumulators, but the computational overhead required for dynamically deleting set elements during transaction processing is immense, and some lightweight nodes do not support transaction submission. We propose a lightweight node with fixed-length storage suitable for the UTXO model, which can both verify the validity of transactions and submit transactions under constant storage capacity. This scheme utilizes two data structures that support only element addition operations to replace the UTXO set: the STXO set and the TXO set. Valid UTXOs belong to the TXO set but not the STXO set. New STXO and TXO commitments are constructed using RSA accumulators and MMR, respectively. Lightweight nodes only need to store the latest block commitments and interact with full nodes, achieving transaction validity verification and submission while reducing storage overhead. Evaluation results show that lightweight nodes can achieve fixed-length storage; lightweight nodes can verify transactions with low computational load; full nodes exhibit optimized performance during commitment updating processes.

Key words: UTXO; RSA accumulator; STXO commitment; TXO commitment; Stateless blockchain

0　引　言

区块链[1-2]凭借其去中心化、数据防篡改及高度可追溯的核心优势,有效应对了传统中心化模式下存在的数据可信度低、系统维护挑战重以及信息孤岛化等难题[3]。区块链核心特性之一是分布式账本机制[4],其中最具代表性的两种记账模式为:以比特币为代表的未花费交易输出(Unspent Transaction Output, UTXO)模型和以以太坊为代表的普通账户模型。其中,比特币交易使用一个或多个 UTXO 作为输入,并将输入的总余额重新分配到新的 UTXO 作为交易输

收稿日期:2024-10-07　　**修回日期**:2025-02-11
基金项目:国家自然科学基金(61902285)
作者简介:刘蕊嘉(2000-),女,硕士研究生,通信作者,研究方向为区块链安全。

出,以此实现货币价值转移。由于 UTXO 只能用作一次交易输入,因此区块链中的节点需维护 UTXO 集合并检查其是否在过去的交易中使用,以防止“双花”问题[5-6]。

在区块链网络中,全节点需存储所有交易历史和整个 UTXO 集以独立验证新交易,然而由于区块链的仅追加特性,随着交易数量的激增,其状态数据的累积存储成本对资源受限的设备构成重压。截至 2024 年 1 月,比特币节点维护的未花费交易集合已接近一亿条记录,占用空间突破 6 GB,并维持显著增长态势。由于存储和带宽需求过高,多数用户转向轻客户端(如 SPV 节点)进行操作,这些轻节点在交易验证方面依赖于一定数量的全节点,从而面临如远程攻击等潜在安全风险。此外,随着存储负担的进一步增加,只有少数超级节点可以运行交易验证功能,造成网络集中化,削弱了区块链的去中心化特性。

为减轻节点存储负担,研究者们提出了一系列轻节点方案,包括基于 UTXO 分片的轻节点[7-9]、基于区块压缩的轻节点[10-12]和基于累加器的轻节点[13-23]。然而,这些方法仍然存在一些挑战,包括轻节点存储仍在增加、证明消耗带宽较大以及由于删除累加器成员导致承诺更新复杂度高的问题。

基于 UTXO 分片的轻节点方案将 UTXO 按片段存储在不同的节点上,将原本依赖于全节点的数据转移至若干节点共同维护的分布式哈希表(Distributed Hash Table,DHT)中。尽管轻节点的存储需求得到了优化,但随着分片数量的线性增长,无法从根本上解决问题。基于区块压缩的轻节点方案通过共识算法和压缩协议将多个原始区块合并成一个新区块,但此技术对于需要存储非数字型数据的区块体,存在显著的局限性。

基于密码学累加器的无状态交易验证为解决区块链数据爆炸问题提供了创新策略,该机制利用累加器为交易输出生成数字承诺,使轻节点无需存储完整区块链数据,仅需验证定长承诺即可实现安全交易验证。累加器类型涵盖默克尔树和 RSA 累加器等。基于默克尔树累加器的方案在处理证明生成和验证等操作时,每个交易均需要消耗对数级的带宽开销。基于单 RSA 累加器的解决方案在处理元素删除时计算复杂度较高,文献[20]进一步提出双 RSA 累加器方案,分别为交易的输出(Transaction Output,TXO)集合和已用交易输出集合生成承诺,提高了承诺更新效率,但大量输入时证明生成速度减慢。文献[23]则采用 MMR(Merkle Mountain Range,MMR)为交易输出集合生成承诺,轻节点仅需存储 MMR 子根以验证合法性。由于子根数与集合大小的对数成正比,轻节点存储上限依旧增长。

针对上述问题,该文提出一种适用于 UTXO 模型的定长存储轻节点方案,旨在实现两大核心功能:一是验证交易合法性以支持区块生成,二是允许用户向区块链网络提交新交易。该方案使用仅支持添加的交易输出集合和已用交易输出(Spent Transaction Outputs,STXO)集合替代 UTXO 集合。为避免验证者存储所有块头导致的资源耗费,方案引入基于默克尔山脉变体的 TXO 承诺机制,通过仅存储最新区块的 STXO 承诺和 TXO 承诺,以快速验证特定输入的有效性。实验证明,该轻节点架构在定长存储条件下能独立验证交易,且性能开销在合理范围内,适用于存储受限的设备或环境。总体而言,基于上述问题能够实现以下改进:

(1)定长存储:轻节点存储开销较低,无需存储完整区块数据或保留所有区块头副本,仅维护最新区块承诺即可保持同步和验证的能力。

(2)去中心化:轻节点能提交并独立验证交易,无需依赖第三方节点,保持了去中心化特性。

(3)高效率:本方案避免元素删除操作,相比其他无状态区块链系统,具有较快的交易验证和承诺更新速度。

1　模型设计

1.1　轻节点框架

区块链是通过加密算法将数据块安全链接的分布式数据库。相较于有状态区块链,该文采用的无状态区块链仅需通过轻量级的加密状态摘要作为验证基础,对 UTXO 数据集进行存储。方案架构如图 1 所示,虚线与实线分别描绘了交易提交与验证的流程。轻节点作为资源受限的设备,仅保留包含默克尔根、最新 STXO 承诺和 TXO 承诺等关键信息的区块头;全节点负责维护区块链完整性并生成新区块,包括承诺的构建与更新。具体到轻节点参与交易验证的流程,核心步骤如下:

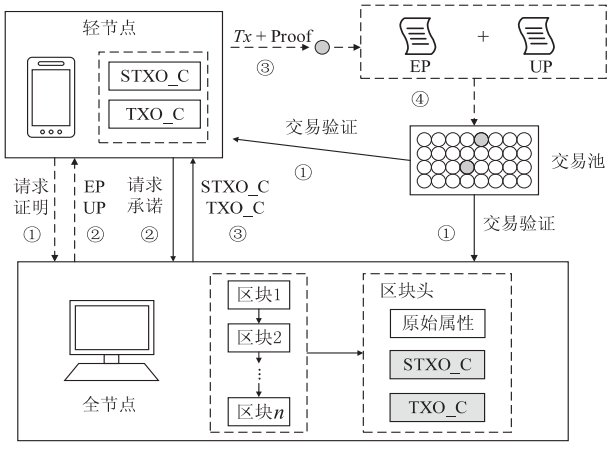

图 1　基于 UTXO 定长存储的节点框架

(1)当轻节点欲将交易提议至交易池时,系统首先激活验证机制,即向全节点发送请求,查询该交易输入(即引入的 UTXO)的合法性证明;

(2)全节点响应请求并返回两种关键性证明:存在证明(Existence Proof,EP)和未花费证明(Unspent Proof,UP)。随后轻节点进一步请求获取最新的 STXO 承诺和 TXO 承诺;

(3)轻节点利用接收到的证明和承诺,验证交易引用的 UTXO 有效且未被双重花费;

(4)验证通过后,轻节点将该交易暂存于本地交易池中。当全节点打包新区块并更新区块链时,调整 UTXO 集合及相关承诺,确保状态一致性。

1.2 区块设计

区块包含区块头和区块体。区块头存储整个区块的元数据,包括前一区块哈希、默克尔根和难度值等;区块体则维护全部的历史记录。假设区块链架构中共有 n 个区块,待验证交易输入为 tx 。该方案在区块头原有的属性基础上添加了两个新属性:STXO_C(STXO 承诺)和 TXO_C(TXO 承诺),如图 2 所示。合法的交易输入需满足 tx_outputs = { output | output ∈ TXO Set, output ∉ STXO Set },即交易引用的输出集合(tx_outputs)既能满足存在性(output ∈ TXO Set),同时也满足未花费性(output ∉ STXO Set)。

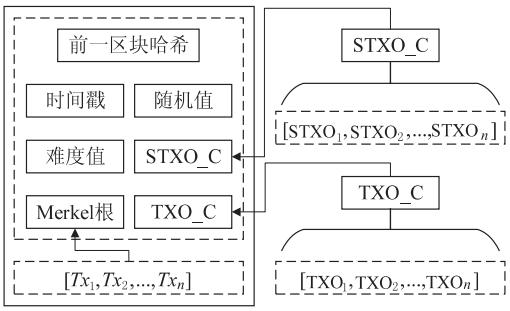

图 2 新区块头

(1)TXO 承诺。TXO 集合是由所有曾经生成的交易输出 TXO_i $(0 \leqslant i \leqslant n)$ 所构成的集合,所以有效的 tx 应来自于 TXO Set。存在性证明可通过提供输入交易的默克尔路径实现,这要求轻节点能快速访问所有区块头。该方案构建了一个由默克尔根组成的默克尔山脉,其根为 TXO 承诺并存储于区块头 TXO_C 中。轻节点验证基于 TXO 承诺的 EP 即可确认输入由区块链生成。

(2)STXO 承诺。STXO 集合是由所有使用过的交易输入 $STXO_i$ $(0 \leqslant i \leqslant n)$ 构成的集合,所以有效的交易输入不属于该集合。全节点通过 RSA 累加器为该集合内的所有元素生成一个 STXO 承诺。具体而言,每个区块头包含一个累加器,用于表示当前区块链内所有交易输入的 STXO 集合,存储在 STXO_C 中。轻节点提交输入时,全节点提供基于 STXO 承诺的未花费证明,轻节点验证 UP 与交易输入的数学关系,确认输入是否已消费。

1.3 安全模型

在该方案中,攻击者为已知区块头的恶意全节点 Adv,其可能发起两种攻击形式:(1)为不存在于区块链上的交易输入伪造有效的存在证明;(2)为存在于区块链上的交易提供有效的未使用证明。

2 方案构造

2.1 区块生成

该阶段由全节点执行,通过区块链的初始化参数配置,生成每个区块的 STXO 承诺和 TXO 承诺。

(1)参数初始化由特殊的全节点(Special Full Node,SFN)执行,生成创世区块(Genesis 块)和常见参数。系统生成 512 位随机数 num,使用 MillerRabin 算法找到素数 num,令 p = num, q 的生成方式与 p 相同。计算 RSA 累加器模数 $N = pq$,并随机选取 $g \in_R \mathbb{Z}_N^*$ 作为公开参数。参数 g 和 N 公开至区块链网络, p 和 q 则被丢弃。

(2)TXO 承诺生成。该方案采用称为 MMR 的默克尔树变体为所有区块头内的默克尔根构建承诺。构建过程包括 MMR 构造和子根默克尔化两部分。MMR 将累积的默克尔根散列成山脉状结构,动态构建尽可能大的完全二叉树,二叉树根为 MMR 峰值,叶子节点则存储根哈希值。当新区块产生时,其默克尔根加入 MMR。子根默克尔化则递归哈希所有子默克尔树根,从而构建 MMR 的根哈希值。每层元素为奇数时,视最后一个节点为该层子根,哈希拼接所有子根得 MMR 最终根,即 TXO 承诺。

(3)STXO 承诺生成。该方案使用 RSA 累加器优化区块链中 STXO 集合的管理。鉴于 RSA 累加器要求输入必须为素数,因此针对每个新区块 i ,使用 Python 3.10 中 sympy 库提供的 HashToPrime 函数将交易输入哈希成唯一素数。设 A_i 为第 i 个区块的累加值,对于区块链中每个新区块 i ,全节点获取区块内所有的新增交易输入 $TxIn_i$ 哈希值,并使用 $STXO_i$ 集合将所有新增 $TxIn_i$ 素数封装。通过公式 1 计算集合元素的乘积结果 v ,全节点通过公式 2 计算每个后续区块的累加值,并保存在各自区块头 STXO_C 中。

$$v = \prod_{TxIn_i \in stxo_i} \text{HashToPrime}(TxIn_i) \tag{1}$$

$$A_i = A_{i-1}^{v} \bmod N \tag{2}$$

2.2 交易提交

本小节侧重于全节点生成交易 tx 有效性证明的过程,其中包括两部分:存在证明和未使用证明。

2.2.1　存在证明生成

存在证明的目的在于确保 tx 输入确实曾经生成。tx 的存在证明包含两部分:基于默克尔根的输入存在证明和基于 TXO 承诺的默克尔根存在证明。假设最新 TXO 承诺为 TXO_C_n,为了证明第 m 个区块中确实生成了一个输入,证明生成步骤为:(1)全节点提供默克尔路径,以证实生成该输入的交易存在于第 m 个区块。(2)基于默克尔路径,全节点进一步构建 MMR 路径,证明第 m 个区块位于区块链上。此 MMR 路径的构建始于特定叶子节点至其所属山峰的验证路径,随后按左至右顺序,将相邻山峰纳入证明,直至形成完整的 MMR 路径。

图 3 和图 4 分别展示了默克尔路径和 MMR 路径示例。在包含 7 个区块、每区块 8 个交易的区块链中,若 tx 的输入 TXO_1 位于首个区块且该区块的默克尔根为 TMR_1,则 tx 的默克尔路径为[t_2,t_3t_4,t_{5-8}],MMR 路径为[[ ],[h_2,h_3h_4],[h_7,h_5h_6]]。此交易的存在证明为 [t_2,t_3t_4,t_{5-8}] 和 [[ ], [h_2,h_3h_4], [h_7,h_5h_6]],最新区块的 TXO 承诺为 TXO_C_7。

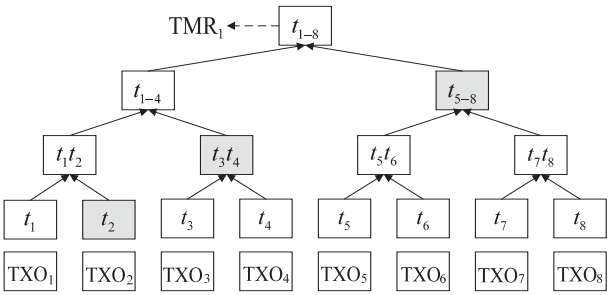

图 3　默克尔路径示例

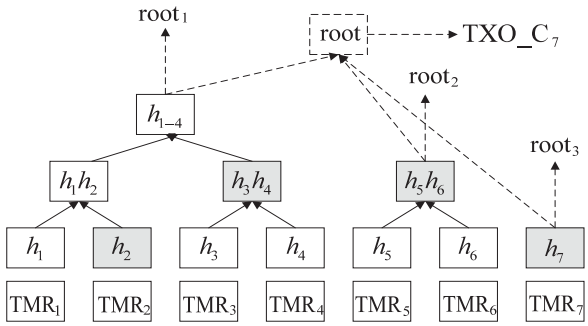

图 4　MMR 路径示例

2.2.2　未使用证明生成

未花费证明是与最新 STXO 承诺相对应的非成员证明。一旦交易在区块链中作为 TXO 被创建,就可以在后续的任何时间被花费。因此对于输入 tx,设从生成 tx 的区块 B_m 到最新区块 B_n 之间所有的 STXO 为 $STXO_{m:n}$,区块 B_m 的 STXO 承诺为 A_m,则非成员证明 $u_n(x)$ 的生成步骤如下:

(1)全节点将待验证交易 tx 转化成素数;

(2)计算 $STXO_{m:n}$ 中的所有 $stxo_i$ 的素数乘积 v。由于 tx 不是 v 的素数成员,因此使用扩展的欧拉定理找到 Bezout 系数 a 和 b,使得 $ax + bv = 1$;

(3)计算 $d = A_m^a \bmod N$,对应于 tx 的非成员证明 $u_n(x)$ 则为(d,b)。

2.3　交易验证

2.3.1　存在证明验证

本节侧重的是轻节点执行交易验证的过程,包含两个部分:存在证明验证和未使用证明验证。

轻节点利用待验证输入及其默克尔路径计算默克尔根,并结合从全节点获取的 MMR 路径计算出目标交易的 TXO 承诺。若此承诺与链上最新 TXO 承诺一致,则确认该输入存在于某交易输出中。待验证输入 tx 的 EP 验证过程如下:

(1)默克尔路径验证:轻节点遍历默克尔路径,对当前 tx 的素数 x 与路径中各元素进行哈希拼接;

(2)山峰峰值计算:通过迭代计算当前 x 和 MMR 特定山峰的默克尔证明,得出该山峰峰值 x;

(3)两侧峰值处理:将山峰左侧及右侧的峰值累加给 x,此时 x 即为验证计算得出的 TXO 承诺;

(4)承诺验证:轻节点将计算得出的 TXO 承诺与预期承诺 TXO_C 进行比较。若两者相等,则存在证明验证成功;若不相等,则此交易验证失败。

2.3.2　未使用证明验证

轻节点收到全节点发送的基于 tx 的非成员证明 $u_n(x) = (d,b)$ 后,验证 tx 的素数 x 是否使等式 3 成立。如果成立,则交易验证成功,轻节点确认该 tx 未曾使用;反之,交易验证失败。

$$A_m = d^x A_n^b \bmod N \tag{3}$$

3　安全性分析

轻节点能在定长存储限制下,根据交易输入存在证明与未花费证明验证输入合法性。安全性分析将证明恶意节点无法为非法输入提供有效证明。

定理 1　给定区块链 B 及相关参数(包括区块头承诺 TXO_C、STXO_C、初始化参数 N 和 g),攻击者 Adv 无法为不存在的输入构造有效存在证明,否则打破了哈希不可逆特性。

证明　当 Adv 欲伪造全新交易输入 txo_w 时,需提供指向区块链上某默克尔根的哈希认证路径,但基于哈希不可逆性,无法从哈希值反推出原始数据,因此构造此认证路径是不可能的。

定理 2　给定相同区块链 B 及相关参数,恶意全节点 Adv 无法为已经花费的输入提供有效的未使用证明,否则打破了贝祖定理。

证明　贝祖定理(Bezout theorem)陈述如下:对于任意两个非零整数 x 和 y,存在整数 a 和 b,使得它们

的最大公约数 d 可以表示为 $d = ax + by$。定理证明通常由扩展欧几里德算法完成。尽管 Adv 能相对容易地为存在于区块链上的输入 txo_w 生成存在证明,但无法为其生成有效的非成员证明。设 txo_w 位于区块 m,区块链总长度为 n,从 m 到 n 的所有的 STXO 集合表示为 $STXO_{m:n}$,累积值为 v,x = HashToPrime(txo_w)。由于 $x \in STXO_{m:n}$,所以 x 与 v 最大公约数不为 1。由贝祖定理可知无法生成整数 a 和 b,因此 Adv 无法生成有效的非成员证明。

4 实验与分析

4.1 实验环境与参数

该文在 Ubuntu 20.04.1 Linux 系统上使用 Python 3.8 进行仿真实验,配置为 Intel Core 2.60 GHz CPU 和 2.00 GB 内存。采用 hashlib 库的 sha256 哈希函数,RSA 累加器设置包括 1 024 位模数、128 位素数长度和 256 位默克尔根长度。

4.2 性能测试与分析

设当前区块链中一个交易消耗一个旧的 UTXO 并输出一个新的 UTXO,该文设置每个区块包含 2 000 笔交易,对每个测试实例进行了 10 次实验,并记录了每次实验运行时间的平均值。分析核心聚焦于时间开销、通信开销及存储开销三个维度,并将其性能与最先进的工作进行了比较。

(1)时间开销。

本节执行了对累加器更新性能和证明生成成本的时间复杂度分析测试。

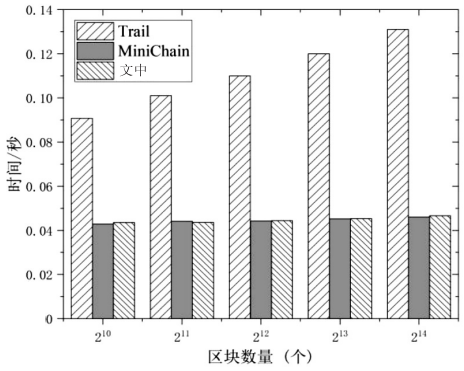

图 5 累加器更新的性能对比

图 5 展示了累加器更新性能的对比结果。将承诺更新的效率与 Trail[13] 和 MiniChain[23] 进行了比较。文中的承诺更新时间涉及 RSA 累加器和 MMR 的更新。RSA 累加器通过更紧凑的方式表示整个集合的状态,其更新不依赖于累积交易数量,更新效率可视为常量;而 MMR 更新时间则与区块数量相关。文中的承诺更新时间显著优于 Trail,原因在于 Trail 系统需移除已消耗的交易输出并标记新生成的 TXO 至树中,而文中采用仅添加元素的累加器,避免了删除操作,且 MMR 能够局部更新受影响的树部分,避免了 Trail 中更新开销翻倍的问题。此外,文中的承诺更新时间与 MiniChain 相近,时间复杂度均与区块链长度正相关。

图 6 展示了文中生成存在证明和未花费证明的时间开销。设待验证输入源自最新区块,未花费证明通过生成针对该区块内所有交易输出的非成员证明来计算,其时间复杂度与区块数量无关。本测试中未花费证明的平均生成时长为 0.108 秒,开销较低。存在证明通过构建待验证 UTXO 到 MMR 根节点的路径实现。随区块数量增加,路径构建的计算量会相应上升,但整体耗时仍维持在较低水平。

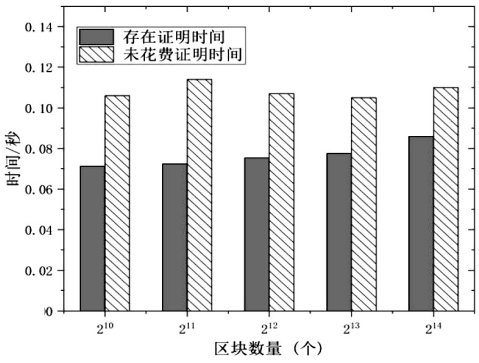

图 6 证明生成开销

(2)通信开销。

图 7 展示了不同方案下轻节点在验证交易时与全节点的通信开销。在文中环境中,轻节点与全节点之间的通信数据包括存在证明与未花费证明两部分,其中未使用证明大小为 256 字节,该部分通信开销可认为是固定的,存在证明大小由 MMR 构建高度与单个哈希值大小的乘积决定。而 Trail 无论是对已消耗的交易输出提供已花费证明,还是对未使用输入提供存在证明,皆使用 Merkle 树构造,证明长度与交易数量成正相关。在向全节点请求证明的场景中,证明的字节规模对传输效率和成本具有直接影响。当区块数量一定时,文中方案在通信开销上明显小于 Trail 和 MiniChain。

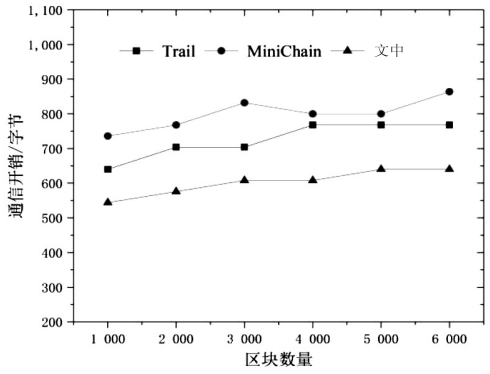

图 7 单个交易证明的通信开销对比

(3)存储开销。

图 8 展示了不同方案下轻节点存储开销的对比情况。分析表明,提出的存储方案开销恒定,与区块数量无关。具体而言,文中存储开销由 TXO 承诺和 STXO 承诺构成,总共占用 288 字节。在区块数量相同的前提下,由于 MMR 的构建高度普遍小于完全二叉树的高度,因此,文中方案轻节点的存储开销小于 Trail。另一方面,MiniChain 在利用 MMR 生成 TXO 承诺时,还需额外保留构建二叉树时产生的五至七个峰值,这直接导致生成的承诺体积显著超过文中方案。此外,考虑到区块链长度在实际应用中会发生变化,MiniChain 的存储开销可能随峰值的不断累积而呈指数级增加,进一步加剧了数据存储的膨胀问题。相比之下,文中轻节点的构建方法无需存储 MMR 的多个子根,结构更为紧凑,可保持为定长大小,从而为区块头节省了更多字节空间,有效减轻了轻节点的存储负担。

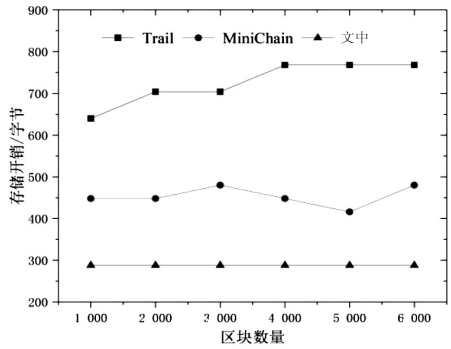

图 8　轻节点存储开销对比

4.3　与现有方案对比

表 1 给出了文中方案与现有方案的一些对比。文献[7]将 UTXO 分片并利用 DHT 进行分配存储。但其提出的安全轻节点模型仅支持交易验证,无法提交交易。文献[10-11]通过总结或压缩区块,形成大小远小于原始区块的新区块。但该方式仅适合区块体中保存金额、数字变化的场景,无法用于存储非数字型数据的区块体。

文献[13]提出将输入存储于 TXO 树结构中以管理交易使用状态。轻节点仅需存储自身 TXO 证明和最新区块树根,即可生成包含余额证明的交易。尽管 Trail 有效减轻存储负担并能独立验证交易,但其将所有输入构建于一棵树上,在累加器更新时的带宽开销呈对数级增长,构成显著负担。

文献[23]采用 RSA 累加器和 MMR 构造了一种新的 STXO 承诺 TXO 承诺方案,显著提升了累加器效率。轻节点仅存储最新区块的 STXO 承诺、TXO 承诺以及 MMR 子根,便能实现交易验证。由于 MMR 子根数量会发生变化,轻节点存储依旧增长,且仅实现了验证器的功能,不支持提交交易。

针对上述问题,该文提出了新的轻节点方案。当轻节点提交交易时,向邻近全节点请求输入的存在和未使用证明,据此提交包含证明的交易;在验证交易时,先更新本地的 STXO 和 TXO 承诺,根据输入证明验证交易的合法性。相较于其他轻节点方案,该方案不仅实现了定长存储,也兼备交易提交和验证功能,具备广泛的用户适用性。

表 1　UTXO 轻节点方案对比

方案	公有链/私有链	提交交易	独立验证交易	轻节点存储内容	定长存储
文献[7]	公有链	否	是	最新 6 个区块 UTXO 哈希列表和区块哈希列表	否
文献[10]	所有	是	是	所有总结区块	否
文献[11]	所有	是	是	压缩后所有的总结块	否
文献[13]	公有链	是	是	相关币的证明及最新承诺	否
文献[23]	公有链	否	是	最新 6 个区块头	是
文中	所有	是	是	最新 STXO 承诺和 TXO 承诺	是

5　结束语

针对现有 UTXO 轻节点存储开销不断增加的问题,提出了一种定长存储的轻节点方案。该方案无需部署额外的验证器,即可实现交易的提交与验证功能。并使用两个仅添加的数据结构取代原有 UTXO 集合,避免了删除成员元素造成的巨大存储开销,同时提高了全节点的承诺更新效率。引入的 STXO 承诺和 TXO 承诺将不断增长的状态组合为一个短小恒定的值,实现了轻节点的定长存储。因此,轻节点可以在低端设备上运行,保持了区块链的去中心化特性。实验和分析结果表明,该方案在性能开销和安全性方面都具备良好表现,尤其适用于低端设备,达到了预期设计目标。

参考文献:

[1] ISMAIL A,RANA A,MUSTAFA A,et al. Toward a model to enhance the applicability of blockchain in maritime shipping: a qualitative study from the middle east[J]. Measuring Business Excellence,2024,28(1):69-83.

[2] BUTERIN V,ILLUM J,NADLER M,et al. Blockchain pri-

vacy and regulatory compliance: towards a practical equilibrium[J]. Blockchain: Research and Applications, 2024, 5(1): 100176.

[3] 蔡晓晴,邓 尧,张 亮,等. 区块链原理及其核心技术[J]. 计算机学报,2021,44(1):84-131.

[4] MONRAT A A, SCHELÉN O, ANDERSSON K. A survey of blockchain from the perspectives of applications, challenges, and opportunities[J]. IEEE Access, 2019, 7: 117134-117151.

[5] CHEN Yujing, YANG Bin. Analysis on the evolution of shipping logistics service supply chain market structure under the application of blockchain technology[J]. Advanced Engineering Informatics, 2022, 53: 101714.

[6] 谢晴晴,董 凡. 轻量级区块链技术综述[J]. 软件学报,2023,34(1):33-49.

[7] FREY D, MAKKES M X, ROMAN P L, et al. Bringing secure bitcoin transactions to your smartphone[C]//Proceedings of the 15th international workshop on adaptive and reflective middleware. New York: ACM, 2016: 1-6.

[8] WANG Xiangyu, WANG Wenyong, ZENG Youlu, et al. A state sharding model on the blockchain[J]. Cluster Comput, 2022, 25(3): 1969-1979.

[9] 高应磊. 基于分片技术的区块链可扩展性研究[D]. 天津:天津大学,2019.

[10] PALAI A, VORA M, SHAH A. Empowering light nodes in blockchains with block summarization[C]//2018 9th IFIP international conference on new technologies, mobility and security (NTMS). Paris: IEEE, 2018: 1-5.

[11] NADIYA U, MUTIJARSA K, RIZQI C Y. Block summarization and compression in bitcoin blockchain[C]//2018 international symposium on electronics and smart devices (ISESD). Yogyakarta: IEEE, 2018: 1-4.

[12] FAN Xing, NIU Baoning, LIU Zhenliang. Scalable blockchain storage systems: research progress and models[J]. Computing, 2022, 104(6): 1-28.

[13] NAGAYAMA R, BANNO R, SHUDO K. Trail: a blockchain architecture for light nodes[C]//2020 IEEE symposium on computers and communications: IEEE symposium on computers and communications (ISCC 2020). Rennes: IEEE, 2020: 1-7.

[14] BHAYYAJI K D, GANDHARBA S. Data hiding and integrity verification based on quotient value differencing and Merkle tree[J]. Arabian Journal for Science and Engineering, 2022, 48(2): 1793-1805.

[15] LIU Chuyi. Throughput optimization for blockchain system with dynamic sharding[J]. Electronics, 2023, 12(24): 4915.

[16] PÂRIS J F, SCHWARZ T. Merkle hash grids instead of Merkle trees[C]//2020 28th international symposium on modeling, analysis, and simulation of computer and telecommunication systems (MASCOTS). Nice: IEEE, 2020: 1-8.

[17] DAHLBERG R, PULLS T, PEETERS R. Efficient sparse merkle trees: caching strategies and secure (non-) membership proofs[C]//Secure IT systems: 21st nordic conference, NordSec 2016. Oulu: Springer, 2016: 199-215.

[18] REYZIN L, YAKOUBOV S. Efficient asynchronous accumulators for distributed PKI[C]//Security and cryptography for networks: 10th international conference, SCN 2016. Amalfi: Springer, 2016: 292-309.

[19] SOBIN C C, RAYCHOUDHURY V, SAHA S. An incentive-based scheme for mitigating node selfishness in smart opportunistic mobile networks[J]. Wireless Personal Communications, 2017, 96(3): 3533-3551.

[20] 杨晋生,王 浩,高 镇,等. 基于双 RSA 累加器的无状态交易验证方案[J]. 浙江大学学报:工学版,2023,57(1):178-189.

[21] REDDY S B, REDDY U K T. Compactchain: an efficient stateless chain for utxo-model blockchain[J]. Frontiers of Computer Science, 2024, 18(2): 182806.

[22] LI Xingzhi, HAN Bei, LI Guojie, et al. Dynamic topology awareness in active distribution networks using blockchain-based state estimations[J]. IEEE Transactions on Power Systems, 2021, 36(6): 5185-5197.

[23] CHEN Huan, WANG Yijie. Minichain: a lightweight protocol to combat the utxo growth in public blockchain[J]. J Parallel Distrib Comput, 2020, 143(prepublish): 67-76.