计算机研究与发展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202440477
61(10): 2526−2539, 2024

基于近似测量算法 Sketch 的高精度区块链异常检测机制

朱龙隆[1,2,5] 陈 翔[2] 陈浩东[1] 牛继堂[4] 刘雯靓[3] 林声睿[1] 张 栋[1,5] 吴春明[2,5]

[1](福州大学计算机与大数据学院 福州 350108)

[2](浙江大学计算机科学与技术学院 杭州 310013)

[3](浙江广播电视集团 杭州 310005)

[4](浙江大学工程师学院 杭州 310015)

[5](泉城实验室 济南 250103)

(kikionepin@gmail.com)

An Accurate Blockchain Anomaly Detection Mechanism Built on Approximate Sketch Algorithms

Zhu Longlong[1,2,5], Chen Xiang[2], Chen Haodong[1], Niu Jitang[4], Liu Wenjing[3], Lin Shengrui[1], Zhang Dong[1,5], and Wu Chunming[2,5]

[1](*College of Computer and Data Science*, *Fuzhou University*, *Fuzhou* 350108)

[2](*College of Computer Science and Technology*, *Zhejiang University*, *Hangzhou* 310013)

[3](*Zhejiang Media Group*, *Hangzhou* 310005)

[4](*Polytechnic Institute of Zhejiang University*, *Hangzhou* 310015)

[5](*Quan Cheng Laboratory*, *Jinan* 250103)

Abstract Blockchain suffers from network dynamics and management difficulties, making the anomalies such as DDoS attacks and account takeovers possible. Existing approaches that detect anomalies in blockchains extract features, such as historical transaction information and transaction frequencies, from blockchain accounts to identify anomalies. However, the increasing scale of blockchain data results in the challenge of high memory consumption and low detection accuracy in the feature extraction of existing approaches. To address this challenge, we propose a blockchain anomaly detection mechanism that achieves detection accuracy and reduces resource footprints. This mechanism embraces approximate sketching algorithms to transform the detection of blockchain anomalies into that of malicious accounts, including intra-block accounts and inter-block accounts. For intra-block accounts, i.e., the malicious accounts that occur inside a single block and the mechanism uses sketching algorithms to collectively filter out those accounts with high precision. For inter-block accounts, malicious accounts can be hardly detected by analyzing the information of a single block, it aggregates multi-block information to accurately detect those accounts. We evaluate our mechanism with real Ethereum block data comprising of 88 847 blocks. Our results indicate that compared with typical existing approaches, our mechanism improves the recall of detecting blockchain anomalies by up to 6.3 times and the *F*1 score by up to 4.4 times. Therefore, our proposed blockchain anomaly detection mechanism

收稿日期: 2024-06-05; **修回日期**: 2024-07-14

基金项目: 国家重点研发计划项目(2023YFB2904000, 2023YFB2904005); 泉城实验室重点项目(QCLZD202304); 山东省实验室项目(SYS202201); 国家自然科学基金项目(623B2090)

This work was supported by the National Key Research and Development Program of China (2023YFB2904000, 2023YFB2904005), the Key Project of Quan Cheng Laboratory (QCLZD202304), the Project of Provincial Laboratory of Shandong (SYS202201), and the National Natural Science Foundation of China (623B2090).

通信作者: 张栋(zhangdong@fzu.edu.cn)

can bring benefits to regulating blockchain transaction behaviors and maintain system security.

Key words blockchain; security; anomaly detection; approximate algorithms; Sketch

摘 要 区块链存在网络动态性强和其管理困难等问题，使得区块链普遍存在 DDoS 攻击和账户接管等异常现象. 现有区块链异常检测方法多从所有区块链账户中提取历史交易信息和交易频率等特征加以分析以甄别异常. 然而，随着区块链数据规模的不断扩大，现有方法在提取特征时面临内存消耗高、检测精度低的挑战. 为此，提出了一种高检测精度、低内存开销的区块链异常检测机制，该机制采用近似测量算法将区块链异常检测转化为异常交易账户检测，包括区块内异常账户和跨区块异常账户. 对于区块内异常账户，即仅存在于单个区块内的异常账户，使用 Sketch 算法进行账户识别，精度高. 而对于存在于多个区块且难以通过单个区块信息检测到的跨区块异常账户，则通过聚合和分析多区块信息进行账户的准确检测. 使用包含 88 847 个区块的真实区块链数据评估上述机制. 实验结果表明，与现有代表性方法对比，所提出的机制将区块链异常检测的召回率最高提升了 6.3 倍，*F*1 分数最高提升了 4.4 倍. 因此，提出的高精度区块链异常检测机制对于规范区块链交易行为、维护系统安全性具有意义.

关键词 区块链；安全；异常检测；近似算法；Sketch

中图法分类号 TP393

区块链是一种去中心化的新型分布式计算技术，其结构是由多个数据块（又名区块）组合成的链，每个区块都包含前一个区块的加密哈希、时间戳、事务列表和其他字段. 目前，区块链技术已被广泛应用于比特币、以太坊等加密货币的交易记录[1-2]，然而在网络层面临的安全问题日益严峻. 由于区块链网络流量动态性强、管理难度大，区块链易遭受由异常交易账户发起的网络层攻击[3-4]，如 DDoS 攻击，为避免区块链被攻击后导致的重大经济损失，区块链异常检测被普遍认为是一个亟待解决的前沿科学问题[5-7].

现有区块链网络层异常检测方法主要分为静态规则检测方法和采样检测方法两类[8-9]. 一方面，静态规则检测方法指的是通过提取静态的区块链交易特征，形成安全规则，从而检测异常交易账户的方法[10-11]，它们通过提取当前账户的交易历史信息、交易金额、交易频率和操作码等特征，对比当前账户与正常账户特征的统计信息，从而验证当前账户行为是否正常. 然而，如表 1 所示，随着区块链上交易数据规模的增加，静态规则检测方法的内存开销急剧增加，难以保证低内存开销. 另一方面，采样检测方法[12-15]从区块链交易数据中按照一定的采样率提取部分区块链账户特征以推断异常账户. 然而该方法依赖于采样，难以保证高检测精度.

Table 1 Comparison of Anomaly Detection Effect for Blockchain

表 1 区块链异常检测效果对比

算法类型	内存开销	准确度
静态规则检测方法	高 *O*(100 KB)~*O*(1 MB)	高（召回率为 100%、*F*1 分数为 1）
采样检测方法	低 *O*(10 KB)	低（召回率为 15%~80%、*F*1 分数为 0.22~0.64）
本文提出的检测机制	低 *O*(0.1 KB)	高（召回率为 95%、*F*1 分数为 0.97）

为解决上述现有方案存在的问题，本文提出了一种高检测精度、低内存开销的区块链异常检测机制. 本文的核心贡献主要包括 3 个方面：

1）通过分析常见区块链异常，包括 DDoS 攻击、欺诈、账户接管、非法活动等，从而将区块链异常检测问题转化为异常交易账户检测问题，这些异常账户存在高交易额、高交易频率、高交易对象数量的特征.

2）提出了一种基于近似测量算法 Sketch 的高精度区块链异常检测机制. Sketch 是一种用于信息统计的数据结构，以较小的内存开销保留海量数据中的关键信息，并支持快速的信息查询. 该机制包含 2 个检测阶段：在第 1 阶段，通过 Sketch 过滤得到高交易额、高交易频率、高交易对象数量的区块内异常账户. 在第 2 阶段，基于第 1 阶段的分析结果，以多个区块为一个检测窗口，聚合多区块交易信息，以准确检测跨区块异常账户（即存在多个区块且难以通过单个区块信息检测异常账户）.

3）基于包含 88 847 个区块的真实以太坊区块链数据评估了提出的区块链异常检测机制. 实验结果表明，与现有方法相比，本文提出的机制将区块链异常检测的召回率最高提升了 6.3 倍，*F*1 分数最高提升了 4.4 倍. 同时基于真实 DDoS 攻击数据集，验证了本文机制在 DDoS 攻击场景下的有效性，召回率达到 99%，达到理论最优精度.

1 相关工作

1.1 常见区块链异常类型

本节介绍常见区块链异常类型.

1)DDoS 攻击. DDoS 攻击是指运用大量计算机同时向受害者主机请求服务，占满受害者主机资源导致其拒绝为正常主机提供服务[16]，在区块链网络中，DDoS 攻击表现为交易金额和交易次数的突然上升[17].

2)欺诈. 欺诈者以诱人的远期收益为诱饵，来骗取投资者的投资，主要表现为庞氏骗局[18]. 在庞氏骗局中，欺诈者向初期投资者许诺短期内的投资能得到高回报，随后用新投资人的资金作为初期投资者的收益，以吸引更多的投资者. 在区块链中表现为一个账户拥有较多的资金流入，但只有很少的资金流出[19].

3)账户接管. 攻击者通过恶意软件或网络钓鱼的方式非法获取受害者设备或账户的权限，从而接管受害者账户以偷取资金[20]. 近年的市场分析报告指出 2024 年已有价值 1 693 万美元的加密货币被盗，数额是 2023 年的 2 倍. 在区块链中表现为一个非活跃账户短期内发生较大金额的资金流出.

4)非法活动. 由于区块链的匿名特性，数字货币也被用于洗钱等非法活动[21]. 在区块链中表现为一个活跃账户长期拥有高额的资金流入或流出[22].

1.2 区块链网络异常检测

本节介绍现有区块链异常检测方法，主要包含静态规则检测方法和采样检测方法.

静态规则检测方法主要通过提取以太坊钱包交易的特征进行异常检测. 现有研究表明，人为操作的以太坊钱包连续交易的时间差呈现幂次分布律[23]，因此 Zwang 等人[10] 利用以太坊钱包连续交易之间的时间差分布来识别周期性和不规则的异常；Chen 等人[11] 通过从交易中提取账户特征(如余额和支付率等)、操作码特征和交易频率等信息来识别庞氏骗局. 然而，随着攻击类型和手段的不断演变，静态规则检测方法中静态、单一的防御规则无法应对攻击变化[24]. 此外，随着区块链数据规模增大，内存开销也急剧上升.

采样检测方法从大量数据中按一定采样率提取所需特征，常被用于大流量数据的特征提取与分析[12-15]. 采样检测方法从区块链交易数据中按照一定的采样率提取数据子集，减少数据采集和分析成本. 通过分析数据子集中异常账户的特征，检测方法进而推断区块链交易中的异常交易账户. NetFlow[13] 改变采样率来权衡精度与内存. OpenSample[15] 将 sFlow[12] 的采样功能集成到控制器中以提升精度. 然而，采样检测方法依赖于采样率的设定，随着区块链上数据规模的增加，无法同时兼顾高检测精度和低内存开销.

2 基于 Sketch 的区块链异常检测

本节将详细介绍基于近似测量算法 Sketch 的高精度区块链异常检测机制. 首先定义本文的异常检测任务；其次提出基于 Sketch 的区块链异常检测机制以及关键算法；最后介绍检测机制的实现. 表 2 总结了本文主要符号的含义.

Table 2 Notion of Symbols

表 2 符号含义

符号	含义
T	高交易额、高交易频率异常账户检测中的阈值
C	区块链交易账户
$ValueC$	账户总交易额
$CountC$	账户总交易次数
$TotalValue$	总交易额
$TotalCount$	总交易次数
$DegC$	账户 C 的交易对象数量
θ	高交易对象数量异常账户检测中的阈值
$factor$	阈值θ的调整因子

2.1 异常检测任务的定义

本文分析区块链网络常见攻击，如 DDoS 攻击、欺诈、账户接管、非法活动等. 其核心特征为：在一段时间内，单个或多个账户进行了大量交易，拥有较高的资金流入或流出.

因此，本文专注于区块链交易中的异常账户的检测，其主要特征包括高交易额、高交易频率、高交易对象数量. 具体为：

1)高交易额或高交易频率的异常账户. 在一段时间内，考虑一个区块链网络内的总交易额或总交易次数. 如果某个区块链账户所涉及交易的总额或交易次数占区块总交易总额或总次数的主要部分，可认定为异常交易账户.

形式上，设定阈值T，将账户C的交易额、交易次数分别记为$ValueC$、$CountC$，总交易额记为$TotalValue$，总交易次数记为$TotalCount$. 异常交易账户判定条件为：

$$ValueC \geqslant T \times TotalValue, \tag{1}$$

$$CountC \geqslant T \times TotalCount. \tag{2}$$

2)高交易对象数量的异常账户. 在一段时间内，考虑一个区块链网络内各个账户的出度，即与其发生交易的不同账户的数量. 如果某个区块链账户的出度或入度超过设定的阈值，可认定为异常账户.

形式上，设定阈值为 θ，账户 C 的出度或入度记为 $DegC$. 异常账户判定条件为：

$$DegC \geqslant \theta. \tag{3}$$

2.2 基于 Sketch 的区块链异常检测框架

Sketch 作为经典的概率统计算法，支持对大规模数据的统计与查询，同时减少存储数据特征所需的空间[25]. Sketch 近似测量的原理是利用亚线性的摘要数据结构实现大规模数据的聚合、采样或压缩. 对于输入的数据流，Sketch 首先解析并获取数据包的信息，如源 IP 地址和目的 IP 地址；其次，利用哈希函数找到其在摘要数据结构中对应的计数器；再次，根据数据包值更新计数器统计值；最后，在一个测量周期结束时，遍历摘要数据结构获取最终统计值. 由于将属于相同流的数据映射到同一个计数器中，Sketch 大幅度降低了内存资源开销. 但由于流之间存在哈希碰撞，即多条流映射到同一个计数器中，导致 Sketch 得到的统计值存在一定误差. 以频率估计为例，假设在 1 个测量周期内，流 A 包含 1 个数据包，流 B 包含 9 个数据包，且流 A 与流 B 发生哈希碰撞. 当数据包传入时，首先，解析数据包获取键信息；其次，利用哈希函数定位计数器位置；再次，将计数器值累加 1；最后，查询计数器得到统计值为 10. 因此，对于占主导地位的大象流，Sketch 以较低的内存开销取得了较高的测量精度. Sketch 已被广泛用于实现大流检测[25]、熵估计[26] 等应用. 在区块链中，Sketch 主要应用于区块摘要[27]、支付验证[28]、唯一地址检查[29-30] 等，实现存储空间和查询精度之间的平衡. Rottenstreich[31] 提出使用 Sketch 生成多个连续区块的摘要. 相较于使用布隆过滤器的简易摘要，基于 Sketch 的区块摘要在维持较小存储开销的同时可提供更多交易信息.

Sketch 为解决已有区块链异常检测方法的高内存、低准确度挑战提供可行思路. HashPipe[32] 是一种广泛应用于识别网络数据流中大象流的 Sketch 算法，通过多级哈希表的流水线机制在有限内存中准确跟踪大流量数据. HyperLogLog[33] 是一种典型的用于估计大规模数据集合基数(即不同元素数量)的 Sketch 算法，通过概率算法和哈希函数在有限内存中提供高准确度的基数估计.

本文所提基于 Sketch 的区块链异常检测机制如图 1 所示，包括 2 个检测阶段：基于 HashPipe 与 HyperLogLog 的区块内检测和跨区块检测.

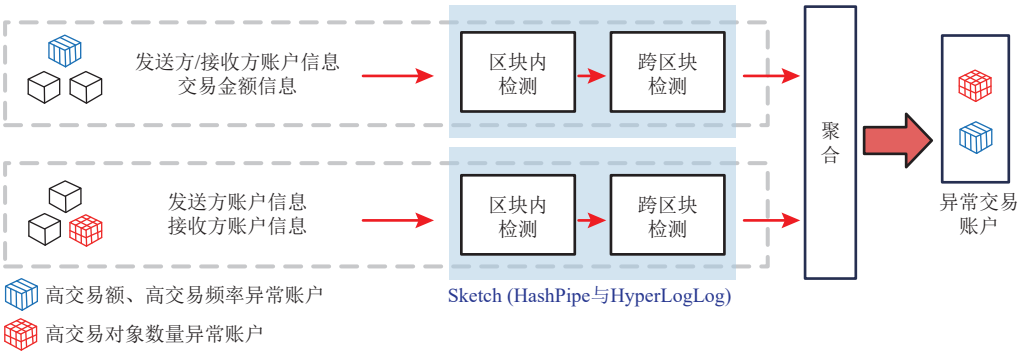


Fig. 1 Workflow of Sketch-based anomaly accounts detection in blockchain

图 1 基于 Sketch 的区块链异常账户检测工作流程

在第 1 个阶段，为检测高交易额或高交易频率的异常账户，在每个区块内，本文利用 HashPipe 近似维护 Top-K 交易额或交易频率的账户. 在每个检测周期开始时，创建包含若干计数器、大小固定的 HashPipe 数据结构. 针对每条新发起的交易，提取其发送方或接收方的账户信息、交易金额信息插入计数器. 在每个检测周期结束时，对比设定的阈值，判断每个计数器中的账户是否为区块内异常账户. 然而，难以检测存在于多个区块且难以通过单个区块信息检测到的异常账户. 对此，在第 2 个检测阶段，本文利用多区块信息. 为避免第 1 阶段的影响，可选择剔除已判定为异常的账户. 在一个检测周期内，同时考虑多个区块的交易信息检测跨区块异常账户. 最后，聚合区块内检测、多区块联合检测两阶段得到异常账户.

为检测高交易对象数的异常账户，在每个区块内，本文利用 HyperLogLog 近似维护 Top-K 交易账户数的账户. 对于每个检测周期，创建 K 个包含 m 个桶的 HyperLogLog 数据结构，维护 K 个账户的交易对象数量.

对于每条新发起的区块链交易, 提取其发送方、接收方信息, 插入 HyperLogLog. 检测周期结束时, 基于每个 HyperLogLog 内维护的交易对象数量判断其对应账户是否为异常. 同样地, 跨区块异常账户难以通过单个区块信息检测. 因此一方面, 本文在一个检测周期内处理多个区块的交易信息, 得到异常账户候选; 另一方面, 本文维护了一个全局的 HyperLogLog 结构, 聚合多个检测周期内得到的异常账户候选的信息. 最后一个检测周期结束后, 本文在全局 HyperLogLog 结构中得到异常账户.

2.3 检测指标

本文将区块链账户分类为异常账户与正常账户. 为衡量检测的性能, 本文与遍历完整区块数据集计算得到的准确异常账户分类结果比较, 基于对比结果计算真阳性 *TP*、假阳性 *FP*、真阴性 *TN*、假阴性 *FN*. 在此基础上, 计算精度(precision)、召回率(recall)、*F*1 分数为:

$$\text{精度} = \frac{TP}{TP+FP}\times 100\%, \tag{4}$$

$$\text{召回率} = \frac{TP}{TP+FN}\times 100\%, \tag{5}$$

$$F1\text{分数} = 2\times\frac{\text{精度}\times\text{召回率}}{\text{精度}+\text{召回率}}. \tag{6}$$

2.4 高交易额或高交易频率的异常账户检测

本文的目标是检测在一段时间内具备高交易额、高交易频率的账户. 遍历区块链记录, 所有账户交易记录的方法精度和召回率高. 然而, 面对大规模区块数据, 存储开销高. 对此, 本文通过一种基于多层流水线机制的 Sketch 应对这个问题, 即 HashPipe. 然而, 跨区块异常账户存在于多个区块且难以通过单个区块信息检测. 因此, 本文提出一种两阶段检测的算法. 第 1 阶段基于 HashPipe 检测区块内异常交易账户; 第 2 阶段同时考虑多个区块, 挖掘在单个区块内不占主导地位的跨区块异常账户. 聚合两阶段异常账户.

1)第 1 个检测阶段. HashPipe 的核心思想是构建流水线结构的多层 Sketch, 包括更新、查询 2 个关键操作. 如图 2 所示, HashPipe 由多个一维 Sketch 组成, 交易信息传入后可能经过多层 Sketch.

如算法 1 所示, 首先通过预设的哈希函数计算区块链账号 *Acc* 在当前层 Sketch 的桶位置 *L*. 如果该位置已被占用, 本文根据账号的交易额或者交易次数决定是否以新账号替换原区块链账号信息. 若替换, 则将原账号及其交易额或交易次数信息推送至下一层 Sketch, 重复上述操作. 若某个账号被踢出最后一层 Sketch, 则丢弃. 因此, 本文可以近似维护当前区块交易金额或交易频率最高的 *K* 个账户信息.

算法 1. HashPipe 更新算法.

输入: 交易信息(账户 *Acc*, 交易金额 *value*), HashPipe 数据结构 *h*, 多个哈希函数 *HF*;

输出: 更新后的 HashPipe 数据结构 *h*.

① for i←0 to d–1 do
② $L \leftarrow HF_i\ (Acc)$;
③ if $h[i][L].key == Acc$ then
④ $h[i][L].v$ += *value*; /*若统计频率则加 1*/
⑤ return;
⑥ end if
⑦ if $h[i][L].key == null$ then
⑧ $(h[i][L].key, h[i][L].v) \leftarrow (Acc, value)$;
⑨ return;
⑩ end if
⑪ if $h[i][L].key \neq null \&\& h[i][L].v < value$ then

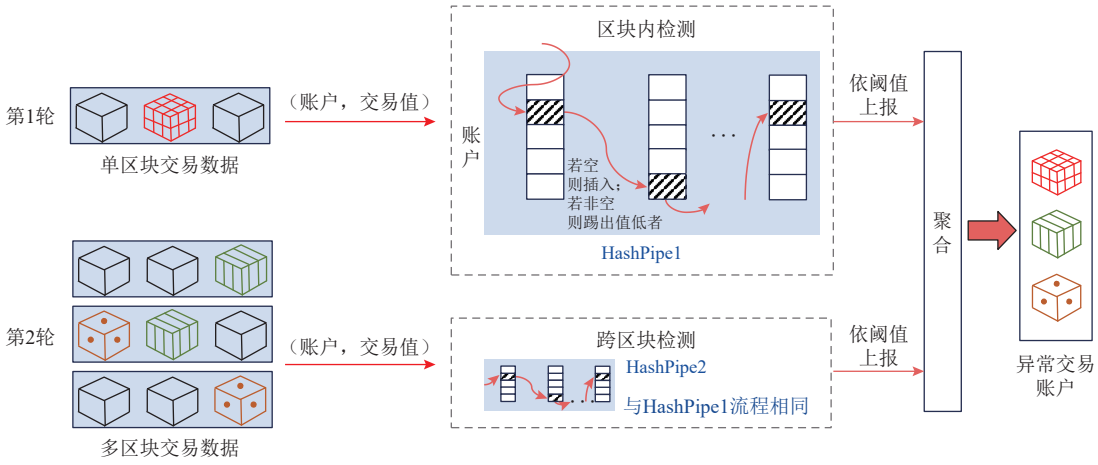


Fig. 2 HashPipe-based dual-stage anomaly accounts detection

图 2 基于 HashPipe 的异常账户两阶段检测

⑫ *swap* (*h*[*i*][*L*].*key*, *Acc*);
⑬ *swap* (*h*[*i*][*L*].*v*, *value*);
⑭ end if
⑮ end for
⑯ return *h*.

检测周期结束后，通过遍历 HashPipe 数据结构，本文得到统计值高于异常阈值 *T*×*TotalValue* 或 *T*×*TotalCount* 的异常账户. 具体过程如算法 2 所示.

算法 2. HashPipe 异常账户搜索算法.

输入：阈值 *T*, HashPipe 数据结构 *h*, 检测周期总交易额 *TotalValue*;

输出：异常账户集合 *List*.

① *List*←empty;
② *H*←*T*×*TotalValue*; /*以交易额为例，若统计交易次数则为 *T*×*TotalCount**/
③ for *i* ← 0 to *d*–1 do
④ for *key* in *h*[*i*] do
⑤ if *key.value* > *H* then
⑥ *List.append* (*key*);
⑦ end if
⑧ end for
⑨ end for
⑩ return *List*.

2）第 2 个检测阶段. 如算法 3 所示，以高交易额账户检测为例，本文首先联合 *n* 个区块的交易数据，然后构建一个大小为 *d*×*m*′的 HashPipe 数据结构用于交易额统计. 对于每一条交易数据（*Acc*, *value*），本文可以选择判断其账户 *Acc* 是否在第 1 轮检测中已被判定为异常. 若是，则不插入数据结构中，从而剔除单区块中高交易额账户的影响；否则，利用算法 1 将 *Acc* 插入 HashPipe. 检测周期结束后，本文利用算法 2 搜索异常账户. 最后，将 2 个阶段的检测结果聚合，得到最终的异常账户列表. 类似地，对于检测一段时间内前 *K* 个最高交易频率的账户，将算法 3 稍作修改即可，本文不再详细描述其过程.

算法 3. 多区块聚合的异常检测算法.

输入：阈值 *T*, 检测周期总交易额 *TotalValue*, *n* 个区块的交易数据 *Trans*[1],⋯,*Trans*[*n*], HashPipe 结构的层数 *d* 和每层的计数器数 *m*;

输出：异常账户集合 *List*.

① for *i*←1 to *n* do
② *TransTotal.append* (*Trans*[*i*]);
③ end for
④ *H*←*T*×*TotalValue*; /*以交易额为例，若统计交易次数则为 *T*×*TotalCount**/
⑤ *Construct* (*h*, *d*, *m*′); /*构建大小为 *d*×*m*′的 HashPipe 数据结构 *h**/
⑥ for (*Acc*, *value*) in *TransTotal* do
⑦ *Insert* (*Acc*, *value*, *h*);
⑧ end for
⑨ *List*←*Query* (*h*, *H*); /*遍历 *h*，搜索交易金额高于 *H* 的异常账户*/
⑩ return *List*.

对于高交易额或高交易频率的异常账户检测问题，第 1 个检测阶段，本文在 HashPipe 中维护 *k* 个计数器，其中 $k = d \times m$，*d* 和 *m* 分别表示 HashPipe 结构的层数和每层的计数器数. 每个计数器维护一个二元组（账户信息，账户交易总额或账户交易总次数）. 本文使用 5 个字节存储账户信息的前 40 位信息，4 个字节存储账户交易总额或总次数. 尽管未存储完整账户信息，但在一段时间内 2 个前 40 位信息完全一致的账户都是异常交易账户的概率可以忽略不计. 第 2 个检测阶段，类似地，本文对于 *n* 个区块维护一个大小为 $d \times m'$ 的 HashPipe 数据结构，其中，m' 表示多区块检测时构建的 HashPipe 结构每层的计数器数. 综上，对于单个区块而言，本文的内存开销为 $9 \times k \times (1 + m'/n)$ 字节. 特别地，在本文捕获的真实以太坊区块链数据集中，一个区块完整交易数据的平均大小约为 22 KB，当 k=10, m' 和 n=50 时，所提检测方法的内存仅为区块完整交易数据大小的 0.8%.

2.5 高交易对象数量的异常账户检测

本文的目标是检测在一段时间内发生交易的对象数量高的异常账户，即节点度高的账户. 由于出度、入度的估计方法一致，以出度高的异常账户检测为例.

本文利用 HyperLogLog 近似维护出度最高的 *k* 个账户信息及其出度. HyperLogLog 由 *m* 个计数器桶构成，核心思想是使用多个哈希函数将区块交易信息（统计出度，本文取发送方账户信息）映射到多个桶内，在计数器桶中记录首次出现 1 的位置，即连续前导 0 的数量. 利用 *m* 个计数器桶内的统计数据，使用调和平均值估计账户的出度. 考虑到每个区块内交易记录有限，异常账户在单个区块中可能不占主体地位. 因此，如图 3 所示，本文在 1 个检测周期内同时处理多个区块的交易信息. 此外，本文额外维护全局 HyperLogLog 结构，聚合多个检测周期得到的异常账户信息以实现更准确的高出度的异常账户检测，也便于在区块链网络中在线统计和离线检测.

具体如算法 4 所示，本文在每个检测周期内同时

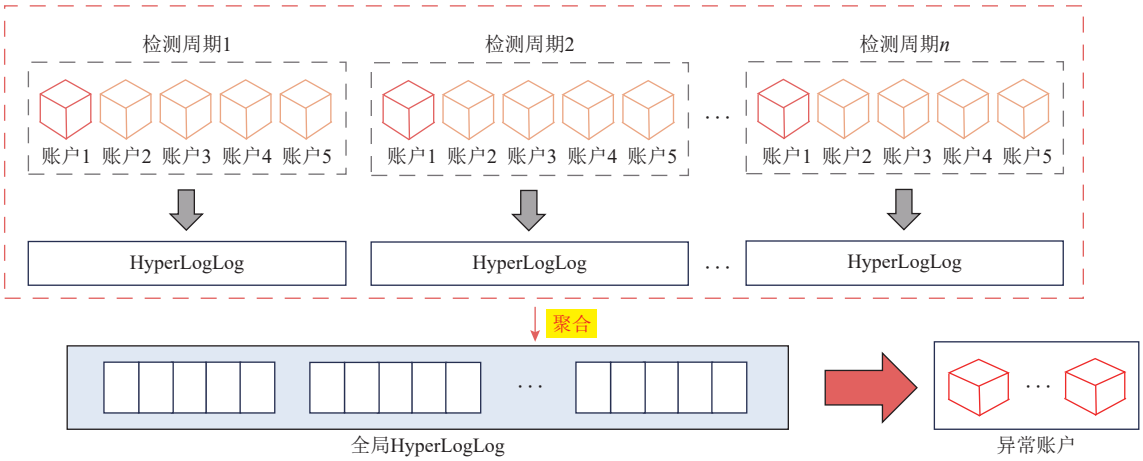


Fig. 3 HyperLogLog-based detection of anomaly accounts with high out-degree

图 3 基于 HyperLogLog 的高出度异常账户检测

考虑n个区块的交易信息，创建k个大小为m的 HyperLogLog 数据结构. 首先遍历所有交易账户信息，包含发送方 *Acc.from* 和接收方 *Acc.to*. 对于已与某个发送方 *Acc.from* 对应的 HyperLogLog 结构进行更新操作. 具体而言，对接收方 *Acc.to* 信息进行哈希操作，提取哈希值的前 b 位作为桶索引，提取剩余位计算前导 0 的数量，并与当前 *HLL*[i] 中维护的出度值比较，更新 *HLL*[i]. 对于未被保留的发送方 *Acc.from*，若创建的 k 个 HpyerLogLog 有未被使用者，则加入；否则，依据出度值大小进行k个 HyperLogLog 所维护发送方信息的替换. 每个检测周期结束后，计算各 HyperLogLog 中的调和平均值，估计出度，将其中高于阈值θ的发送方及其出度估计值合并至全局数据结构，更新全局异常账户集合 *List*. 最后，检测交易对象数量多的异常账户.

算法 4. 高交易对象数量的异常账户检测算法.

输入：阈值 θ, HyperLogLog 数量 k 及其的桶计数器数 m, n 个区块的交易数据 *Trans*[1],⋯,*Trans*[n];

输出：更新后的全局异常账户集合 *List*.

/*对于每个检测周期*/

① *Init* (*TransTotal*, *Trans*);

② *HLL* = *HLLConstruct* (k, m);

③ for *Acc* in *TransTotal* do /*遍历所有账户*/

④ for i = 1 to k do /*遍历 k 个数据结构*/

⑤ if *Acc.from* == *HLL*[i]. *key* then

⑥ *Update* (*HLL*[i], *Acc*); /*更新 *HLL**/

⑦ end if

⑧ if *HLL*[i] == *null* then

⑨ *Insert* (*HLL*[i], *Acc*); /*插入 *HLL**/

⑩ end if

⑪ if *HLL*[i] ≠ *null* && *Esti* (*HLL*[i])==1 then /*若出度为 1 替换为新账户相关信息*/

⑫ *swap* (*HLL*[i], *Acc*);

⑬ end if

⑭ end for

⑮ end for

⑯ *ListCurrenct*←*Query* (*HLL*, θ); /*遍历 k 个 *HLL*，计算调和平均，估计出度，返回出度估计值高于 θ 的异常账户*/

⑰ *Merge* (*List*, *ListCurrent*); /*合并到全局的 HyperLogLog，更新全局异常账户集合 *List**/

⑱ return *List*.

对于账户出度估计问题，本文在 HyperLogLog 中维护m个桶，即m个大小为 1 字节的计数器. 同样地，使用 5 个字节存储账户信息的前 40 位信息. 因此，对于每个需要估计出度的账户，本文的内存开销为 m+5字节，当k=14, m=5 时，仅占单个区块平均大小的 0.6%.

定理 1. 对于高交易对象数量的异常账户检测，以账户出度估计为例，本文在 HyperLogLog 中维护m个桶，每个需要估计出度账户的出度估计值的相对误差标准差约为 $\dfrac{1.04}{\sqrt{m}}$.

证明. 假定某个需要估计出度的账户，其不同交易对象的数量为 n，使用 m 个桶记录最大前导 0 的数量，分别为 $M[i]$. 根据泊松分布可知，估计值期望 $E=\alpha_m m^2\left(\sum_{i=1}^{m}2^{-M[i]}\right)^{-1}$，其中，校正常数 $\alpha_m=\left(m\int_0^{\infty}\times\left(\mathrm{lb}\left(\dfrac{2+t}{t}\right)\right)^m\mathrm{d}t\right)^{-1}$. 为分析$E$的方差，取$Z_i=2^{-M[i]}$，其期

望值$E[Z_i]=\frac{m}{2n}$，方差$V[Z_i]=E[Z_i^2]-(E[Z_i])^2$，展开计算得到$V[Z_i]\approx\frac{m}{4n}$. 由于$E$涉及$m$个桶，聚合$m$个独立同分布的$Z_i$的期望和方差，得$E\left[\sum_{i=1}^{m}Z_i\right]=\frac{m^2}{2n}$，$V\left[\sum_{i=1}^{m}Z_i\right]=\frac{m^2}{4n}$. 根据中心极限定理，标准差$\sigma_{\sum Z_i}=\sqrt{V\left[\sum_{i=1}^{m}Z_i\right]}=\frac{m}{2\sqrt{n}}$. 估计值$E$是基于$\sum_{i=1}^{m}Z_i$的反比值，即$E=\alpha_m\frac{m^2}{\sum_{i=1}^{m}Z_i}$，则$E$的标准差$\sigma_E\approx\alpha_m\frac{m^2\sigma_{Z_i}}{\left(E\left[\sum_{i=1}^{m}Z_i\right]\right)^2}=\alpha_m\frac{2n}{m}$. 因此，$E$的相对误差$\frac{\sigma_E}{E[E]}\approx\frac{4n^2}{m^3}$. 结合实验经验和理论计算，校正常数$\alpha_m$通常选为$\alpha_m=\frac{0.721\,3}{1+1.079/m}\approx 0.721\,3$. 所以，最终的相对误差标准差$\sigma\left(\frac{\hat{n}}{n}\right)\approx\frac{0.721\,3\sqrt{m}}{\sqrt{8}n}=\frac{1.04}{\sqrt{m}}$. 因此，账户出度估计值的相对误差标准差约为$\frac{1.04}{\sqrt{m}}$. 证毕.

2.6 检测机制实现

为实现所提异常账户检测机制，本文为发送方或接收方维护 2 个 HashPipe 结构用于统计交易金额和交易频率，维护多个 HyperLogLog 结构用于估计账户出度或入度信息.

1)统计阶段. 利用构建的大小固定的 HashPipe 数据结构存储多个区块中交易信息，包括发送方或接收方账户、交易金额. 利用多个大小固定的 HyperLogLog 数据结构进行账户出度或入度估计. 周期性发送统计数据.

2)处理阶段. 针对每个检测周期的统计数据，基于所提方法检测高交易金额、高交易频率、高交易对象数量的异常账户. 此外，可对比 2 个周期的检测结果以支持对交易金额、交易频率急剧变化账户的检测. 最后，由区块链网络管理员处理上报的异常账户.

当区块链数据不断增加时，本文执行以下步骤：首先，提取新交易记录中的部分信息. 以高交易金额异常账户检测为例，可提取新交易记录中的发送方信息、交易金额信息. 然后，利用更新操作将二元组(发送方，交易金额)记录到本文机制所构建的 HashPipe，HyperLogLog 数据结构中. 最后，在每个检测周期结束时，提取统计数据，清空 2 种数据结构已存储的信息以便于下一周期的检测，离线分析当前周期或结合以往多个周期的数据得到异常账户候选列表.

3 实验评估

3.1 实验设置

1)数据集捕获. 本文采集了真实以太坊区块链交易信息[34]. 数据集包含 88 847 个真实区块，每个区块平均包含约 98 条交易信息，大小约为 22 KB. 数据集总大小为 1908 MB，交易总次数为 8 705 166，包含 1 546 258 个不同的发送方以及 1 453 510 个不同接收方.

2)对比算法. 本文构造 2 类对比算法：针对异常账户的静态规则检测算法、采样检测方法.

① 静态规则检测算法. 由于现有静态规则检测方法往往针对特定攻击而非本文所提异常账户的检测，本文构造如下静态规则检测算法：使用交易账户信息、交易金额为特征，检测规则为本文所提出的判定为异常的条件. 现有静态规则检测算法[10-11]往往提取多个特征，内存开销显著高于所构造的静态规则检测算法.

② 采样检测方法. 本文采用 NetFlow 方法[13]对区块链数据进行采样，设置采样率γ为1/4 ~ 1/128.

3.2 数据集分析

以发送方为例，本文分析了区块链数据集中账户交易额、交易频率、出度的情况. 如表 3~5 所示，一方面，大多数账户的交易额低于 1 ETH，占账户总数的 75.768 9%；超过半数账户交易次数仅为 1，交易次数超过 100 的账户占比不超过 1%；超过 70% 的账户出度仅为 1，出度超过 10 的账户占比小于 1%. 另一方面，账户的平均交易额、交易频率、出度分别为 9.861 42 ETH、5.629 83 次、2.178 78 个. 综上所述，区块链中大多数账户的交易额、交易频率、交易对象数较低，易存在跨区块异常账户.

Table 3 Analysis of Account Transaction Volume

表 3 账户交易额分析

账户交易金额/ETH	账户数量	占数据集账户总数的比例/%
0	524 895	33.953 9
≤1	1 171 314	75.768 9
≤10	1 450 123	93.804 3
≤100	1 530 129	98.979 6
账户交易额平均值：9.861 42 ETH		

3.3 高交易额、高交易频率异常账户的检测性能

本文检测交易额高的区块内、跨区块异常账户，以及评估了仅执行第 1 个检测阶段时在不同阈值、内存开销条件下的召回率. 表 6 展示了平均召回率变

Table 4 Analysis of Account Transaction Frequencies

表 4 账户交易频率分析

账户交易次数	账户数量	占数据集账户总数的比例/%
1	1 020 785	66.016 5
≤2	1 269 947	82.130 3
≤10	1 504 157	97.277 2
≤100	1 542 453	99.753 9
账户交易次数：5.629 83		

Table 5 Analysis of Account Out-Degree

表 5 账户出度分析

账户出度	账户数量	占数据集账户总数的比例/%
1	1 223 157	79.139 4
≤3	1 470 610	95.149 9
≤10	1 535 291	99.334 8
≤20	1 541 798	99.755 8
≤50	1 544 064	99.902 4
≤100	1 544 681	99.942 4
≤1 000	1 545 441	99.991 5
账户出度个数平均值：2.178 78		

Table 6 Average Recall Rates of High Transaction Volume Anomaly Accounts Detection

表 6 高交易额异常账户检测的平均召回率 %

HashPipe 层数	阈值 T			
	5×10^{-4}	10^{-3}	5×10^{-3}	10^{-2}
2	20	22	29	35
4	35	38	50	59
6	48	52	67	77
8	57	61	77	87
10	65	70	86	93
12	72	76	91	97
14	77	82	95	100

化情况. 随着阈值 T 的增加以及被判定为异常的帐户数量降低, 在相同内存条件下, 召回率得到提升. 此外, 随着 HashPipe 内存开销的增加, 本文可以维护更多异常账户候选, 召回率增加. 在表 6 内所有配置下, 本文提出检测机制的检测平均精度均为 100%.

然而, 对于高交易频率的区块内异常账户检测, 召回率较低. 如表 7 所示, 在相同的内存和阈值条件下, $d=14$、$m=1$、阈值 T=10^{-2}, 高交易频率异常账户检测平均召回率为 60%, 远低于高交易额异常账户检测的平均召回率(100%). 对此, 注意到高交易频率问题中的异常账户往往是跨区块异常账户. 本文执行第 1 个、第 2 个检测阶段. 在第 2 个检测阶段中, 1 个检测周期包含 50 个区块的信息. 表 8 展示了 50 个区块下高交易频率异常账户检测的平均召回率, 由于跨区块异常账户占主体地位, 以 m'=50, d=14 且阈值 T=10^{-2} 为例, 平均召回率从 60% 提升至 100%. 随着内存大小的增加, 平均召回率也得到提升. 同样地, 表 8 中所有参数配置对应的检测平均精度均为 100%.

Table 7 Average Recall Rates of High Transaction Frequencies Anomaly Accounts Detection

表 7 高交易频率异常账户检测的平均召回率 %

HashPipe 层数	阈值 T			
	2×10^{-3}	7×10^{-3}	10^{-2}	2×10^{-2}
2	8	14	19	28
4	14	23	30	44
6	20	30	39	54
8	26	36	46	60
10	30	42	52	64
12	35	47	56	67
14	39	51	60	68

Table 8 Average Recall Rates of High Transaction Frequencies Anomaly Accounts Detection in 50 Blocks

表 8 50 个区块下高交易频率异常账户检测的平均召回率 %

HashPipe 层数	阈值 T			
	2×10^{-3}	7×10^{-3}	10^{-2}	2×10^{-2}
2	76	95	97	99
4	85	98	98	99
6	92	99	99	100
8	93	99	99	100
10	95	99	99	100
12	96	99	100	100
14	97	99	100	100

综上所述, 本文提出的检测机制对于高交易额异常账户检测任务, 在有限内存下实现较高召回率. 对于高交易频率异常账户检测, 单区块内检测的性能低, 本文检测跨区块异常账户, 相同阈值下召回率最高提升了 4.3 倍.

3.4 高交易对象数量异常账户的检测性能

对于账户交易对象数量(本文取出度), 使用基于 HyperLogLog 的算法. 对于每个区块, 维护最多 7 个 HyperLogLog 数据结构以存储出度最高的 7 个异常账户. 图 4 展示了检测周期不同区块数量和出度阈值θ下的平均精度和召回率. 本文取检测周期分别为

1, 5, 20 个区块，阈值θ分别为 50, 100, 500, 1000. 如图 4 所示，对于单个区块，当阈值θ为 50, 100 时，平均精度均为 100%. 然而其平均召回率较低，分别为 37% 和 46%. 当检测周期中包含 20 个区块的信息时，其平均召回率分别提升至 45% 和 52%. 本文同时统计了 50 个真实区块，而 HyperLogLog 的数量有限，当阈值θ较小时，异常账户数量多，导致平均召回率低. 而随着阈值θ增加，异常账户数量降低，平均召回率逐步提升. 本文提出的检测机制的平均召回率最高达 86%.

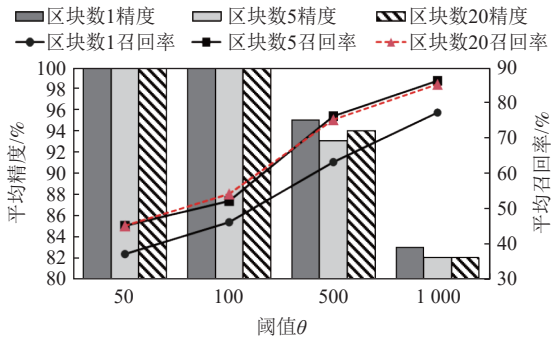


Fig. 4 Average precision and average recall rate of detection for high out-degree anomaly accounts

图 4 高出度异常账户检测的平均精度和平均召回率

本文改变异常账户判定标准，评估 HyperLogLog 的平均精度、平均召回率. 具体而言，固定检测周期为 5 个区块且阈值θ=500, 设定阈值调整因子 *factor*，即阈值 *factor*×θ，探究异常账户判定标准对检测性能的影响. 如图 5 所示，随着 *factor*的降低，平均召回率从 76% 提升至 83%. 与此同时，平均精度从 93% 下降至 87%. 在实际应用中，可以根据需求选择不同的 *factor*，权衡检测的召回率与精度.

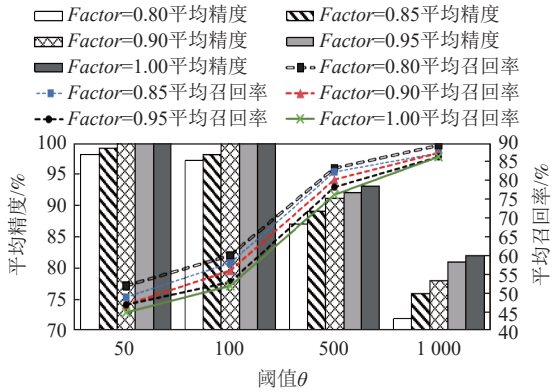


Fig. 5 Average precision and average recall rate of detection for high out-degree anomaly accounts introducing with *factor*

图 5 引入 *factor*的高出度异常账户检测平均精度和平均召回率

综上所述，本文提出的机制对于高交易对象数量异常账户，最高达到了 100% 的精度和 86% 的召回率；对跨区块异常账户，使用多区块信息提升召回率.

3.5 DDoS 数据集下异常账户的检测性能

本文从真实以太坊区块链完整数据集中提取 DDoS 特征的攻击数据集. 数据集包含 500 个区块，每个区块平均包含约 100 次交易. 数据集总大小为 10.73 MB, 包含 1 400 个不同发送方和 1 001 个不同接收方. 其中，400 个账户对同一个账户发起 30~80 次交易，构成对其的 DDoS 攻击.

对此，本文检测高交易频率的异常账户. 取检测阈值 T 为固定值 2×10^{-3}, 检测周期包含 50 个区块，m' 均取 50, 改变 d 的取值，评估 DDoS 账户检测的平均召回率、平均精度. 如图 6 所示，本文对于 DDoS 数据集的平均召回率最高达到 99%. 随着 d 值从 2 增加至 18, 高交易频率异常账户平均精度保持 100%. 而由于数据集内异常账户数量较多，随着内存降低，所能维持的异常账户候选数量降低，召回率随之下降，但精度仍保持在 100%. 综上所述，本文提出的机制可以在 DDoS 数据集下实现有效的异常账户检测，在有限内存条件下达到了 99% 的召回率和 100% 的精度.

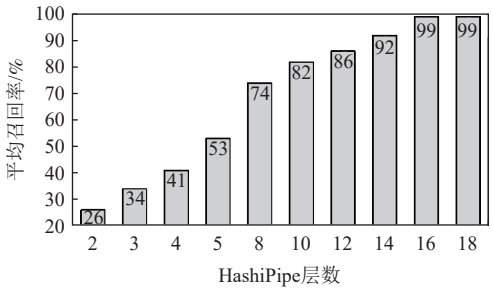


Fig. 6 Average recall rate of detection for anomaly accounts in DDoS dataset

图 6 DDoS 数据集下异常账户检测平均召回率

3.6 不同检测算法的检测性能分析

图 7 和图 8 分别展示了多种异常检测方法在高交易额异常账户检测下的平均召回率、*F*1 分数. 检测阈值 T 取 10^{-3}, 一个检测周期包含 50 个区块的信息. 本文所提检测机制的召回率、*F*1 分数趋近于基于静态规则的检测方法，高于采样检测方法. 召回率方面，本文检测方法召回率为 99.4%, 接近基于静态规则的检测方法，相较于采样检测方法的召回率(95.1%~96.3%)有所提升. *F*1 分数方面，本文检测方法接近基于静态规则的检测方法，相较于采样检测方法的 *F*1 分数(0.975~0.981)有所提升. 而本文提出的检测机制的内存开销远小于静态规则检测方法和采样检测方

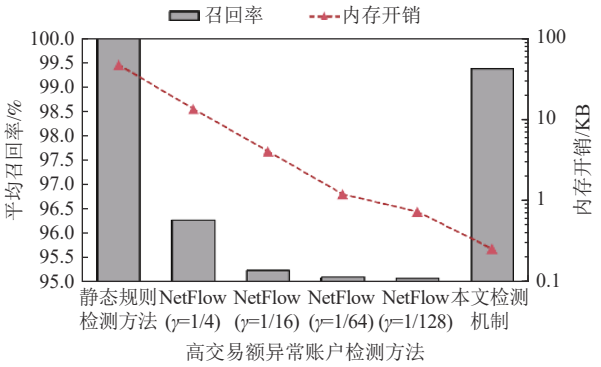


Fig. 7 Average recall rate and memory overhead of detection for high transaction volume anomaly accounts of different methods

图 7 不同方法的高交易额异常账户检测平均召回率和内存开销

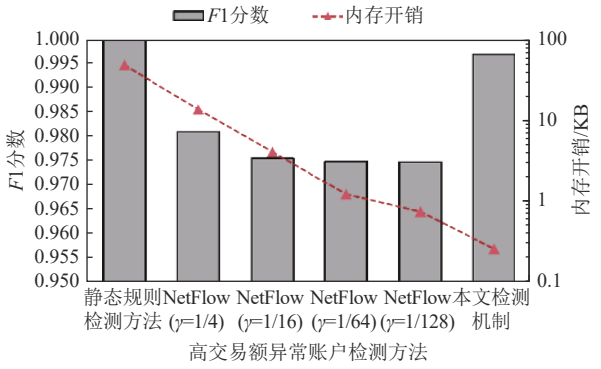


Fig. 8 Average $F1$ score and memory overhead of detection for high transaction volume anomaly accounts of different methods

图 8 不同方法的高交易额异常账户检测平均 $F1$ 分数和内存开销

法, 分别最高降低了 99.4% 和 98.1%.

图 9 和图 10 分别展示了多种异常检测方法在高交易频率异常账户检测下的平均召回率、$F1$ 分数. 检测阈值 T 取 10^{-3}, 1 个检测周期包含 50 个区块的信息. 如图 9 所示, 召回率方面, 本文所提检测机制的平均召回率为 95.1%, 接近基于静态规则的检测方法, 相较于采样检测方法的平均召回率, 最高提升了 6.3 倍. $F1$ 分数方面, 如图 10 所示, 本文所提检测机制的 $F1$ 分数为 0.975, 接近基于静态规则的检测方法, 相较于采样检测方法最高提升了 4.4 倍. 而本文提出的检测机制的内存开销远小于静态规则检测方法和采样检测方法, $F1$ 分数分别最高降低了 99.7 个百分点和 98.8 个百分点.

图 11 和图 12 分别展示了多种异常检测方法在高交易对象数量异常账户检测下的平均召回率和 $F1$ 分数. 当阈值 T 取 1 000 时, 本文提出的检测机制

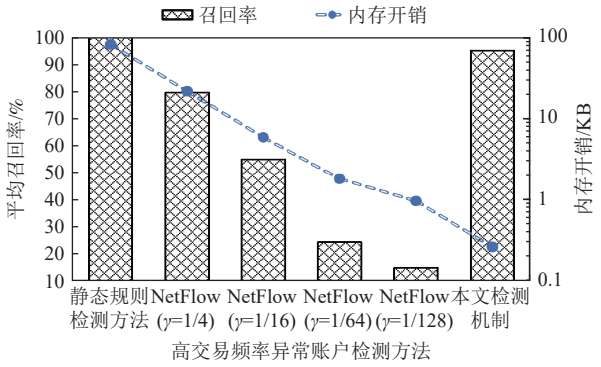


Fig. 9 Average rate recall and memory overhead of detection for high transaction frequencies anomaly accounts of different methods

图 9 不同方法的高交易频率异常账户检测平均召回率和内存开销

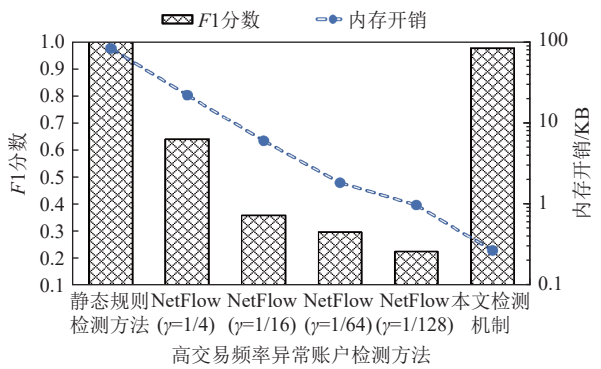


Fig. 10 Average $F1$ score and memory overhead of detection for high transaction frequencies anomaly accounts of different methods

图 10 不同方法高交易频率异常账户检测平均 $F1$ 分数和内存开销

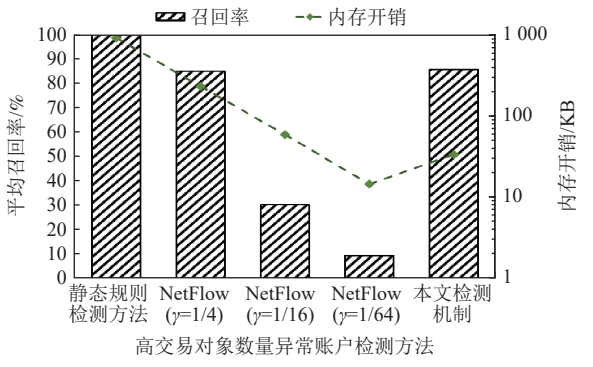


Fig. 11 Average recall rate and memory overhead of detection for high out-degree anomaly accounts of different methods

图 11 不同方法的高出度异常账户检测平均召回率和内存开销

的内存开销较低, 平均召回率和 $F1$ 分数接近静态规则检测方法. 召回率方面, 本文检测方法的内存开销

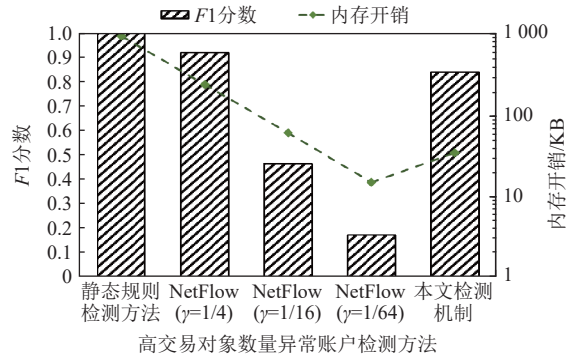


Fig. 12 Average $F1$ score and memory overhead of detection for high out-degree anomaly accounts of different methods

图 12 不同方法的高出度异常账户检测平均 $F1$ 分数和内存开销

较低，相比静态规则检测方法与采样检测方法（采样率 γ 取 1/4），每个检测周期所需的内存开销减少了 85.8% 和 96.4%，而本文达到了 86% 的平均召回率，接近于静态规则检测方法，优于采样检测方法在不同采样率下的平均召回率.

综上所述，相比与静态规则检测方法与采样检测方法，本文提出机制对区块链异常交易账户进行检测，使用较小的内存实现了高检测召回率和 $F1$ 分数.

4 结 论

本文提出了一种基于近似测量算法 Sketch 的高精度区块链异常检测机制. 该机制面向高交易额、高交易频率、高交易对象数量的异常账户检测，构建 2 个检测阶段：第 1 阶段，通过 Sketch 过滤得到高交易额、高交易频率、高交易对象数量的区块内异常账户. 第 2 阶段，基于第 1 阶段的分析结果，以多个区块为 1 个检测窗口，聚合多区块交易信息，以准确检测跨区块异常账户. 基于包含 88 847 个区块的真实以太坊区块链数据评估了提出的区块链异常检测机制. 实验结果表明，与现有方法相比，本文提出的机制将区块链异常检测的召回率最高提升了 6.3 倍，$F1$ 分数最高提升了 4.4 倍. 同时基于真实 DDoS 攻击数据集，验证了本文机制在 DDoS 攻击场景下的有效性，召回率达到 99%，达到理论最优精度.

未来的工作包括 2 个方面：1）关于检测任务类型的扩展，如何利用 Sketch 算法支持更多异常类型的检测. 2）如何在保证精度、召回率的前提下，将检测方法部署至硬件设备，嵌入真实区块链网络环境.

作者贡献声明：朱龙隆负责核心算法设计和论文撰写；陈翔负责系统架构并参与论文撰写；陈浩东负责实验方案设计；牛继堂负责核心算法实现；刘雯靓参与算法实现；林声睿指导本文设计；张栋指导本文设计并参与论文撰写；吴春明指导本文工作开展和论文撰写. 朱龙隆和陈翔对本文的贡献相同.

参 考 文 献

[1] Wood G. Ethereum: A secure decentralised generalised transaction ledger[J]. Ethereum Project Yellow Paper, 2014, 151(2014): 1−32

[2] Wright C S. Bitcoin: A peer-to-peer electronic cash system[J]. SSRN Electronic Journal, 2008, 3440802: 10−2139

[3] Han Xuan, Yuan Yong, Wang Feiyue. Security problems on blockchain: The state of the art and future trends[J]. Acta Automatica Sinica, 2019, 45(1): 206−225 (in Chinese)
(韩璇, 袁勇, 王飞跃. 区块链安全问题: 研究现状与展望[J]. 自动化学报, 2019, 45(1): 206−225)

[4] Jiang Peipei, Wang Qian, Chen Yanjiao, et al. Securing guarantee of the blockchain network: Attacks and countermeasures[J]. Journal on Communications, 2021, 42(1): 151−162 (in Chinese)
(江沛佩, 王骞, 陈艳姣, 等. 区块链网络安全保障: 攻击与防御[J]. 通信学报, 2021, 42(1): 151−162)

[5] Antonopoulos A M. Mastering Bitcoin: Unlocking Ddigital Cryptocurrencies[M]. Sebastopol: O'Reilly Media, 2017

[6] Ye Congcong, Li Guoqiang, Cai Hongming, et al. Security detection model of blockchain[J]. Journal of Software, 2018, 29(5): 1348−1359 (in Chinese)
(叶聪聪, 李国强, 蔡鸿明, 等. 区块链的安全检测模型[J]. 软件学报, 2018, 29(5): 1348−1359)

[7] Zeng Shiqin, Huo Ru, Huang Tao, et al. Survey of blockchain: Principle, progress and application[J]. Journal on Communications, 2020, 41(1): 134−151 (in Chinese)
(曾诗钦, 霍如, 黄韬, 等. 区块链技术研究综述: 原理、进展与应用[J]. 通信学报, 2020, 41(1): 134−151)

[8] Hassan M U, Rehmani M H, Chen Jinjun. Anomaly detection in blockchain networks: A comprehensive survey[J]. arXiv Preprint, arXiv: 2112.06089, 2021

[9] Kim J, Nakashima M, Fan W J, et al. Anomaly detection based on traffic monitoring for secure blockchain networking[C]//Proc of the 2021 IEEE Int Conf on Blockchain and Cryptocurrency (ICBC). Piscataway, NJ : IEEE , 2021: 1−9

[10] Zwang M, Somin S, Pentland A S, et al. Detecting bot activity in the Ethereum blockchain network[J]. arXiv preprint, arXiv: 1810.01591, 2018

[11] Chen Weili, Zheng Zibin, Ngai E, et al. Exploiting blockchain data to detect smart Ponzi schemes on ethereum[J]. IEEE Access, 2019, 7: 37575−37586

[12] Case J, Fedor M, Schoffstall M, et al. A simple network management protocol (SNMP)[EB/OL]. (1990-05-02)[2024-07-13]. https://www.ietf.org/rfc/rfc1157.txt

[13] Claise B. NetFlow[EB/OL]. (2004-10-02)[2024-07-13]. https://www.ietf.org/rfc/rfc3954.txt

[14] Phaal P, Panchen S, McKee N. sFlow[EB/OL]. (2001-09-02)[2024-07-13]. https://www.ietf.org/rfc/rfc3176.txt

[15] Suh J, Kwon T T, Dixon C, et al. OpenSample: A low-latency, sampling-based measurement platform for commodity SDN[C]//Proc of the 2014 Int Conf on Distributed Computing Systems. Piscataway, NJ: IEEE, 2014: 228−237

[16] Eliyan L F, Pietro R D. DoS and DDoS attacks in software defined networks: A survey of existing solutions and research challenges[J]. Future Generation Computer Systems, 2021, 122: 149−171

[17] Mirkin M, Ji Y, Pang J, et al. Bdos: Blockchain denial of service[C]//Proc of the 2020 ACM SIGSAC Conf on Computer and Communications Security. New York: ACM, 2020: 601−619

[18] Chen Weili, Zheng Zibin, Cui Jiahui, et al. Detecting Ponzi schemes on ethereum: Towards healthier blockchain technology[C]//Proc of the 2018 World Wide Web Conf. New York: ACM, 2018: 1409−1418

[19] Andryukhin A A. Phishing attacks and preventions in blockchain based projects[C]//Proc of the 2019 Int Conf on Engineering Technologies and Computer Science. Piscataway, NJ: IEEE, 2019: 15−19

[20] Chainalysis. Israeli government seizes cryptocurrency addresses associated with Hamas donation campaigns[EB/OL]. (2021-07-08)[2024-07-13]. https://blog.chainalysis.com/reports/israel-hamas-cryptocurrency-seizure-july-2021

[21] Voronov T, Raz D, Rottenstreich O. A framework for anomaly detection in blockchain networks with sketches[J]. IEEE/ACM Transactions on Networking, 2024, 1(32): 686−698

[22] Barabasi A L. The origin of bursts and heavy tails in human dynamics[J]. Nature, 2005, 435(7039): 207−211

[23] Zhou Xuanchen, Yang Wenzhong, Tian Xiaodan. Detecting phishing accounts on Ethereum based on transaction records and EGAT[J]. Electronics, 2023, 12(4): 993

[24] Voronov T, Raz D, Rottenstreich O. Scalable blockchain anomaly detection with sketches[C]//Proc of the 2021 IEEE Int Conf on Blockchain. Piscataway: IEEE, 2021: 1−10

[25] Tang Lu, Huang Qun, Lee P P C. MV-Sketch: A past and compact invertible sketch for heavy flow detection in network data streams[C]//Proc of the 2019 IEEE Conf on Computer Communications. Piscataway, NJ: IEEE, 2019: 2026−2034

[26] Lu Jie, Zhang Zheng, Chen Hongchang, et al. Filter-Sketch: A two-layer sketch for entropy estimation in the data plane[J]. IET Communications, 2022, 16(20): 2422−2430

[27] Rush T J. Enhanced adaptive Bloom filters for identifying transactions of interest in a blockchain[R/OL]. New York: QuickBlocks, 2017 [2020-06-05]. http://quickblocks.io

[28] Qin Kaihua, Hadass H, Gervais A, et al. Applying private information retrieval to lightweight bitcoin clients[C]//Proc of the 15th Crypto Valley Conf on Blockchain Technology (CVCBT). Piscataway, NJ: IEEE, 2019: 60−72

[29] Wang Tianyu, Zhu Wenbin, Ma Qun, et al. ABACUS: Address-partitioned Bloom filter on address checking for uniqueness in IoT blockchain[C]//Proc of the 39th Int Conf on Computer-Aided Design. Virtual Event, USA: ACM 2020: 1−7

[30] Shafeeq S, Zeadally S, Alam M, et al. Curbing address reuse in the IOTA distributed ledger: A cuckoo-filter-based approach[J]. IEEE Transactions on Engineering Management, 2019, 67(4): 1244−1255

[31] Rottenstreich O. Sketches for blockchains[C]//Proc of the 2021 Int Conf on Communication Systems & Networks. Piscataway, NJ: IEEE, 2021: 254−262

[32] Lin Yibing, Huang C C, Tsai S C. SDN soft computing application for detecting heavy hitters[J]. IEEE Transactions on Industrial Informatics, 2019, 15(10): 5690−5699

[33] Reviriego P, Martínez J, Rottenstreich O, et al. Remove minimum (RM): An error-tolerant scheme for cardinality estimate by HyperLogLog[J]. IEEE Transactions on Dependable and Secure Computing, 2020, 19(2): 966−977

[34] Etherscan. The Ethereum blockchain explorer[EB/OL]. (2024-06-02)[2024-07-13]. https://etherscan.io



Zhu Longlong, born in 2000. Master candidate. His main research interests include network measurement, programmable networks, and packet classification.

朱龙隆, 2000 年生. 硕士研究生. 主要研究方向为网络测量、可编程网络、包分类.



Chen Xiang, born in 1997. PhD candidate. His main research interests include programmable networks and network measurement.

陈　翔, 1997 年生. 博士研究生. 主要研究方向为可编程网络、网络测量.



Chen Haodong, born in 2000. Master candidate. His main research interests include programmable networks and datacenter congestion control.

陈浩东, 2000 年生. 硕士研究生. 主要研究方向为可编程网络、数据中心拥塞控制.



Niu Jitang, born in 2002. Master candidate. His main research interests include network measurement and programmable networks.

牛继堂, 2002 年生. 硕士研究生. 主要研究方向为网络测量、可编程网络.



Liu Wenjing, born in 1995. Bachelor. Her main research interests include convergence media network and network security.

刘雯靓, 1995 年生. 学士. 主要研究方向为融合媒体网络、网络安全.



Lin Shengrui, born in 1997. PhD candidate. His main research interests include programmable networks and in-network intelligence.

林声睿, 1997 年生. 博士研究生. 主要研究方向为可编程网络、网内智能.



Zhang Dong, born in 1981. PhD, professor, PhD supervisor. Member of CCF. His main research interests include software defined networking, network virtualization and Internet QoS.

张　栋, 1981 年生. 博士, 教授, 博士生导师. CCF 会员. 主要研究方向为软件定义网络、网络虚拟化、网络服务质量.



Wu Chunming, born in 1967. PhD, professor, PhD supervisor. Member of CCF. His main research interests include network architecture, Cyberspace Endogenous Safety and Security.

吴春明, 1967 年生. 博士, 教授, 博士生导师, CCF 会员. 主要研究方向为网络体系结构、网络空间内生安全.