计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2022,31(2):350−357 [doi: 10.15888/j.cnki.csa.008305]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于积分选择 PBFT 共识算法的果品质量溯源①

安　洋, 李　坤, 李军怀, 王怀军

(西安理工大学 计算机科学与工程学院, 西安 710048)
通信作者: 安　洋, E-mail: anyang@xaut.edu.cn

摘　要: 针对基于区块链的果品质量溯源系统中存在的共识算法吞吐量低、时延高、主节点随机选择等问题, 本文提出了一种基于积分选择的改进 PBFT (practical Byzantine fault tolerance) 共识算法. 该算法引入积分选择协议, 通过对一致性协议、视图转换协议以及垃圾回收机制的优化, 提高诚实主节点被选择的概率、减少节点间通讯开销, 从而提升共识算法执行效率. 同时, 在运行垃圾回收机制时, 给所有参与节点重新分配积分, 达到了动态更改节点数量的目的. 实验表明, 本文提出的方法在提升共识算法吞吐量和降低时延方面具有更好的性能.

关键词: 溯源; 区块链; 共识算法; PBFT; 积分选择

引用格式: 安洋,李坤,李军怀,王怀军.基于积分选择 PBFT 共识算法的果品质量溯源.计算机系统应用,2022,31(2):350–357. http://www.c-s-a.org.cn/1003-3254/8305.html

Consensus Algorithm Based on Integral Selection PBFT for Fruit Quality Traceability

AN Yang, LI Kun, LI Jun-Huai, WANG Huai-Jun

(School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China)

Abstract: To address the low throughput, high latency and random selection of master nodes in the Blockchain-based fruit quality traceability system, this study proposes an improved practical Byzantine fault tolerance (PBFT) consensus algorithm based on integral selection. The algorithm introduces the integral selection protocol and optimizes the consistency protocol, view change protocol and garbage collection mechanism to improve the probability of honest master nodes being selected and reduce the communication overhead between nodes, thus improving the efficiency of consensus algorithm execution. At the same time, when the garbage collection mechanism is operated, the integrals are reallocated to all participating nodes for the dynamic change in the node number. Experiments show that the method proposed in this study has better performance in improving the throughput and reducing the latency of consensus algorithms.

Key words: traceability; Blockchain; consensus algorithm; PBFT; integral selection

食品事关国运民生, 食品安全是国家安全的重要基础, 与公众健康、生活水平、经济发展乃至社会稳定息息相关[1]. 果品作为常见食品受众面很广, 其质量安全溯源受到广泛关注. 果品质量溯源是将果品从果园培育到销售的完整产业链中产生的所有数据进行管理以实现监管, 这需要整个供应链中的参与者共同实现[2]. 然而, 现有的主流果品质量溯源系统的溯源数据都是集中化控制管理的, 数据信息存储在中央数据库, 因此追溯数据极易被人刻意篡改且难以发现, 追溯数据的采集比较单一, 数据是否完整也无法被验证[3]. 集中式管理数据的溯源系统并不能保证查询数据的真实性, 不能完全满足果品溯源数据安全完整的需求.

区块链去中心化等特点满足果品质量溯源系统的

① 基金项目: 陕西省科技计划 (2018HJCG-05)
收稿时间: 2021-04-19; 修改时间: 2021-05-19; 采用时间: 2021-05-27; csa 在线出版时间: 2022-01-17

需求[4]. 在供应流程中产生的果园基地的环境数据、施肥、防虫、除害、加工、销售、物流等溯源数据都可以通过物联网设备或人工录入的方式进行信息上链, 一旦数据上链便不能修改. 区块链上的所有数据都需要信息背书, 这样可以有效减少人为错误, 供应链上的企业能够共同维护数据, 消费者只需通过溯源码进行链上数据查询即可获得果品溯源数据, 因此可以很大程度上解决消费者对数据的不信任问题.

在基于区块链的果品质量溯源系统中, 所有环节产生的数据都在一个去中心化、不可伪造和不可篡改的安全环境中进行流通, 共识机制是保证这些特征的区块链底层技术之一[5]. 由于果品质量溯源系统应用于供应链当中, 每天的交易和账户数据更新较为频繁, 系统需要良好的吞吐量以及低交易时延. 虽然区块链去中心化等特点适用于供应链, 但共识算法存在吞吐量低、时延高、节点数不能动态变化等问题, 影响了基于区块链的果品质量溯源系统的整体性能.

鉴于上述原因, 研究实现一种可以良好的对抗拜占庭将军问题[6] 且可以满足高吞吐量和低延时的区块链共识机制成为本文的研究重点.

1 相关研究

追溯体系的构建需要依托溯源技术. 目前, 国内外的追溯技术主要是通过二维码、RFID 技术、NFC 技术和生物 DNA 等技术手段对产品标记和记录, 实现信息存储, 通过互联网软件系统或手持设备等手段查询信息真伪. 传统溯源技术虽然发展比较成熟[7–9], 但其存在溯源数据采集单一、易被篡改、数据完整性无法验证等问题. 区块链技术的去中心化、可追溯性及不可篡改等特性能够很好的解决传统溯源技术存在的问题, 基于区块链的溯源方法成为国内外众多学者的研究热点[10–12].

区块链的核心技术包括共识机制、分布式存储技术、密码学和智能合约[13]. 其中, 共识机制主要解决分布式系统的一致性问题, 保证所有节点维护的数据副本的一致性. 共识算法已经有了非常丰富的实例, 从区块链应用衍生出来的共识算法有 PoW、PoS、DPoS 等; 从传统一致性算法衍生出来的共识算法有 Paxos、Raft、PBFT 等. 6 种共识机制具体对比结果如表 1 所示. 通过对区块链中几种常见的共识机制对比分析, 结合果品质量溯源系统的应用需求可得出: PoW 和 PoS 虽然拥有非常良好的拜占庭容错性但吞吐量与交易时延并不能满足该系统的性能要求, 且其资源消耗普通企业无法承担. Raft、Paxos 和 Kafka 虽然都拥有良好的吞吐量、交易时延和低消耗, 但均不具备拜占庭容错能力, 并不满足果品质量溯源系统对安全性的要求. PBFT 虽然在吞吐量、交易时延和消耗都不是最优, 但是其具备抗拜占庭能力, 基本满足果品质量溯源系统的需求, 因此本文选择 PBFT 共识算法作为基础, 结合果品质量溯源系统的应用场景展开研究工作.

表 1 6 种共识机制多指标对比

指标	PoW	PoS	Raft	Paxos	Kafka	PBFT
吞吐量	低	低	很高	很高	高	高
交易时延	分钟级	分钟级	秒级	秒级	秒级	秒级
资源消耗	很高	高	低	低	低	低
可扩展性	高	高	低	低	低	低
去中心化程度	去中心化	去中心化	半中心化	半中心化	半中心化	半中心化
容错率 (%)	50	50	50	50	99	33
拜占庭容错能力	是	是	否	否	否	是
可监管性	弱	弱	强	强	强	强
一致性	有分叉	有分叉	无分叉	无分叉	无分叉	无分叉
适用场景	公有链	公有链	私有链	私有链	联盟链	联盟链

2 基于积分选择的改进 PBFT 共识算法

针对共识机制低吞吐、高时延和主节点随机选择的问题, 文献 [14] 提出使用投票选举主节点的方式提高共识效率, 文献 [15] 通过简化一致性协议提高共识效率, 文献 [16] 提出分组概念将节点分为共识节点与记账节点, 但这些方法假设所有节点都是诚实节点并未考虑拜占庭问题, 且根据时间戳进行垃圾回收并不满足果品质量溯源系统需求. 本文引入积分机制[17], 对抗拜占庭节点, 对一致性协议步骤进行优化, 通过积分来执行垃圾回收机制, 以提高吞吐量、降低通信开销,

从而提高共识效率.

2.1 PBFT 共识算法

PBFT 共识算法的核心流程, 如图 1 所示, 算法的核心阶段分别是预准备阶段 (pre-prepare)、准备阶段 (prepare) 和提交阶段 (commit). 图中的 C 代表客户端, N_0, N_1, N_2, N_3代表节点的编号, N_3代表可能故障的节点或者是作恶节点, N_0是主节点. 整个过程如图 1, 其中, f代表故障节点数量.

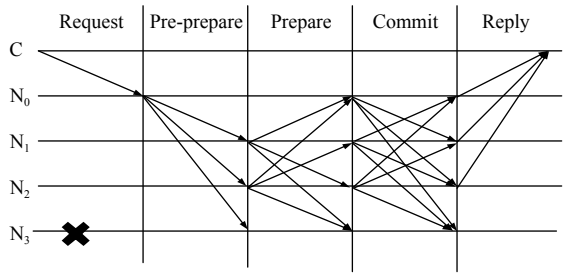


图 1 PBFT 共识算法流程

(1) 从所有参与共识的节点中随机选择一个节点作为主节点, 主节点的主要工作是负责接收客户端信息、广播信息以及生成新的区块;

(2) 预准备阶段: 主节点将从客户端接收到的交易请求进行校验, 校验通过后, 加上自己的签名通过对等网络广播至所有参与共识的从节点, 并且将该交易保存在日志文件中;

(3) 准备阶段: 所有从节点收到消息之后, 首先对消息进行校验, 包括主节点签名等, 校验通过后, 将该交易保存在日志文件并向全网广播一条准备消息;

(4) 确认阶段: 节点对收到的准备消息进行统计, 若收到 $2f$ 条通过校验且和自己信息一致的消息, 就广播一条确认消息;

(5) 回复阶段: 节点对收到的确认消息进行统计, 若收到 $2f$+1 条确认消息, 就将新区块更新到本地账本, 并向客户端发送消息;

(6) 客户端若收到 f+1 条相同消息, 共识结束.

通过对 PBFT 算法共识过程的分析可以得出, PBFT 共识算法存在以下问题: 1) 在执行完整的一致性协议时节点间需要进行大量的通信, 其时间复杂度为 $O(N^2)$; 2) 主节点是随机选择的, 增大了选择异常节点的概率, 从而导致视图转换协议调用次数增多; 3) 垃圾回收机制中需要确保至少 f+1 个节点已经执行了待回收的旧消息, 从而额外增加了节点间的通信开销.

2.2 改进的 PBFT 共识算法

针对传统 PBFT 共识算法存在的问题, 本文提出了一种基于积分选择的改进 PBFT 共识算法. 该算法通过积分选择, 从一致性协议、视图转换协议、垃圾回收机制几方面进行了优化, 提高了共识算法的效率. 系统中节点的积分是每个参与共识的节点在进行共识过程中根据共识行为进行相应加减. 在成功执行一次一致性协议之后, 对所有达成共识的节点 (除主节点), 将积分加 5; 对于未达成共识的节点, 将积分减 5; 主节点成功完成一次区块生成, 积分加 1. 积分选择协议如表 2 所示, 根据积分赋予节点不同角色.

表 2 积分选择协议

角色	选择依据	功能
主节点	$[3(n-1)/4, n-1]$	广播交易, 生成新块
共识节点	$[(n-1)/2, n-1]$	参与共识, 记账
记账节点	$[0, (n-1)/2)]$	记账

主节点的选取依据积分选择协议, 积分越高的节点其安全性越高且更稳定不易坏, 增大主节点选择的安全性, 从而降低了视图转化协议执行的概率, 提高共识效率. 通过不断的执行共识机制, 成功达成共识的节点将不断积累积分, 认为积分高的共识节点其安全性更高.

本文参考文献 [14–17] 对共识算法的改进思路, 进一步明确节点分组边界. 具体来说, 首先将所有节点的积分从小到大进行排序, 认为$[(n-1)/2, n-1]$范围内的节点可信度高, 作为共识节点, 共识节点至少为 4 个, 其余节点作为记账节点. 主节点从$[3(n-1)/4, n-1]$范围内随机选择, 算法整体流程如图 2 所示.

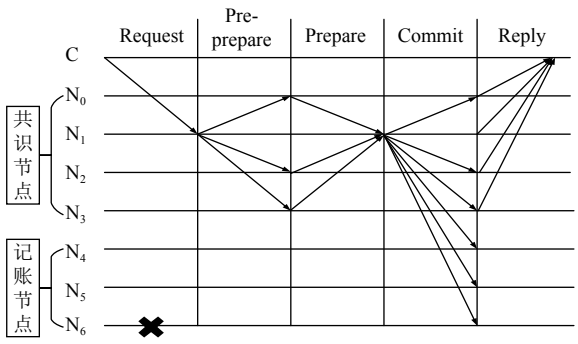


图 2 基于积分选择的改进 PBFT 共识算法流程

具体步骤如下:

(1) 首先将区块链所有节点根据积分选择协议的积分选择依据进行分组, 分为共识节点和记账节点;

(2) 客户端在共识节点中根据积分区间选择主节点N_1, 主节点的主要工作是负责接收客户端信息、广播信息以及生成新的区块;

(3) 主节点将从客户端接收到的交易请求进行校验, 校验通过之后, 加上自己的签名通过对等网络广播至所有参与共识的从节点, 并且将该交易保存在日志文件中;

(4) 从节点收到消息, 并对消息通过签名字段进行认证, 如果认可这条消息, 则将同样的消息加上签名发送给主节点, 并保留消息内容等待二次确认;

(5) 如果主节点收到的认可信息且消息内容没有更改的数量大于等于 $2f$, 则将认可信息打包再发给所有节点, 共识节点将收到的消息进行二次确认检查信息是否正确, 通过验证后进入 commit 状态, 将新区块更新到本地账本, 并向客户端发送消息, 记账节点接收消息更新本地账本;

(6) 如果客户端收到 f+1 条消息认为达成共识, 共识结束.

算法在执行过程中, 本文针对共识节点内拜占庭节点大于承载能力两种情况, 分别给出解决方法: 1) 若 3 阶段内任意阶段条件为满足, 认为共识节点中存在大于 f 个拜占庭节点, 则算法执行完整的一致性协议来解决拜占庭问题; 2) 若节点请求超时, 认为主节点为恶意节点, 则算法执行视图转换协议来解决主节点作恶问题.

每次执行一致性协议都会对节点积分产生影响, 当积分值积累到一定程度后执行垃圾回收机制更新并重置节点积分值. 优化的一致性协议通过简化信息交互过程, 降低了在共识过程中的通信开销, 并且由于引入积分机制使良好节点当选主节点的概率增高, 降低使用视图转换协议的概率, 提高了整体的效率.

2.3 视图转换协议和垃圾回收机制的优化

PBFT 共识算法中视图转换协议的主要目的是当判定主节点为错误节点时更换主节点, 具体过程如图 3 所示.

从节点需要在 view-change 阶段相互通讯来确定主节点为恶意节点, 然后在 view-change-ack 阶段随机选择新的节点成为主节点, 并舍弃未完成的交易. 通过积分机制, 在 view-change-ack 阶段根据当前节点积分选择新的主节点, 可以有效降低使用视图转换协议的概率, 达到提高共识效率的目的.

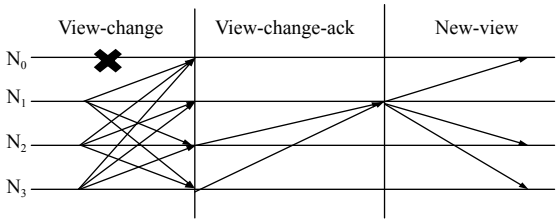


图 3 视图转换协议

PBFT 通过 3 阶段协议来对请求达成共识, 但各个阶段产生的消息如果不进行垃圾回收的话, 系统的存储空间将会不堪重负. 为此, PBFT 算法设计了垃圾回收机制来清除本地缓存. 根据前面的 3 阶段协议, 客户端收到某个请求的执行结果的时候, 表明该请求已经被至少 f+1 个节点提交过, 这个时候需要删除该消息. 垃圾回收机制是通过额外的通讯来提供证明, 证明节点状态正确. 如果每执行结束一次一致性协议都需要生成上述证明, 那么整个网络将会消耗大量资源.

在 PBFT 垃圾回收机制中, 执行方式是周期性执行, 目的是为了防止节点因为宕机、网络或自身故障等原因而产生节点信息不一致从而导致系统故障. 为了确保系统的正常运转以及安全, 节点在清除本地消息日志中旧消息时, 必须确保至少存在 f+1 个节点已经执行了这些旧消息, 因此需要进行节点间是否同步的确认通信, 这导致每次在执行垃圾回收机制时就会产生巨大的通信开销.

改进的 PBFT 算法在垃圾回收机制中实现了动态增加和退出节点的功能以及积分重新分配的功能. 当执行一致性协议时存在节点积分大于等于阈值的情况时, 完成一致性协议后运行垃圾回收机制, 所有参与共识的节点会清除本地日志中已执行过的交易请求, 达到降低网络通讯消耗的目的. 同时将系统中所有共识节点的信用全部清零, 并对所有参与共识节点的积分在一定范围内随机赋值, 提高了新加入网络节点成为主节点的可能, 从而达到了共识节点的动态增加和退出的目的.

3 实验结果与分析

本次实验是应用改进 PBFT 共识机制作为 Fabric 的自定义共识后端并通过 Caliper 对区块链框架中两种不同事务在不同情况下进行测试, 然后对测试结果进行分析讨论. 实验环境配置如表 3 所示.

分别进行提交请求和查询请求的测试, 观察在两

种请求下系统资源占用情况与区块链性能情况. 表 4、表 5 分别为在交易数量为 100 时提交请求和查询请求的内存和 CPU 占用率.

由表 4 和表 5 中的信息可以看出 peer1 节点的内存、CPU 等系统资源使用情况都是 0, 表明该节点在运行过程中并没有对系统提交的交易做出响应, 即该节点属于错误节点. 然而可以看出, 当系统发出提交事务请求的时候, 系统依旧完成了共识过程, 并没有因为某些失效节点或恶意节点造成区块链系统瘫痪. 因此基于积分选择的改进 PBFT 共识算法可以抵抗拜占庭错误, 安全性高.

表 3 实验环境配置

配置	详细情况
操作系统信息	Ubuntu 18.04
CPU	Intel® Core™ i7-8700
内存	2 GB
硬盘	20 GB

表 4 提交请求时资源占用率

Type	Name	Memory (max)(MB)	Memory (avg)(MB)	CPU (max)(%)	CPU (avg)(%)
Process	Node local-client.js(avg)	99.1	99.0	20.00	20.00
Docker	peer0.org1.example.com	63.9	63.9	543.42	543.42
Docker	peer1.org1.example.com	45.5	45.5	490.79	490.79
Docker	peer0.org2.example.com	34.8	34.8	566.69	566.69
Docker	peer1.org2.example.com	55.5	55.5	293.44	293.44
Docker	peer0.org3.example.com	69.1	69.1	472.39	472.39
Docker	peer1.org3.example.com	29.9	29.9	607.97	607.97
Docker	peer0.org4.example.com	39.0	39.0	355.83	355.83
Docker	peer1.org4.example.com	68.8	68.8	200.49	200.49
Docker	simplenetwork_ca_1	6.5	6.5	0.00	0.00
Docker	**simplenetwork_peer_1**	**0**	**0**	**0.00**	**0.00**
Docker	ca_peerOrg1	6.5	6.5	0.00	0.00
Docker	ca_peerOrg2	6.7	6.7	0.00	0.00
Docker	orderer.example.com	11.1	11.1	208.17	208.17

表 5 查询请求时资源占用率

Type	Name	Memory (max)(MB)	Memory (avg)(MB)	CPU (max)(%)	CPU (avg)(%)	Traffic In	Traffic Out
Process	Node local-client.js(avg)	120.7	120.3	5.86	4.82	—	—
Docker	peer0.org1.example	210.3	210.3	624.88	602.20	3.7 MB	4.7 MB
Docker	peer1.org1.example	248.7	228.7	718.74	712.43	3.7 MB	5.8 MB
Docker	peer0.org2.example	210.9	210.8	575.13	560.06	5.4 MB	6.5 MB
Docker	peer1.org2.example	210.7	210.6	764.83	793.44	5.3 MB	8.4 MB
Docker	peer0.org3.example	246.4	246.0	779.17	694.16	6.5 MB	9.3 MB
Docker	Peer1.org3.example	216.5	216.5	518.18	479.69	4.5 MB	6.3 MB
Docker	Peer0.org4.example	241.3	240.8	826.36	706.85	6.6 MB	9.4 MB
Docker	peer1.org4.example	200.8	200.4	556.13	507.66	4.5 MB	0 B
Docker	simplenetwork_ca_1	6.7	6.7	0.00	0.00	140 B	0 B
Docker	**simplenetwork_peer1**	**0**	**0**	**0.00**	**0.00**	**0 B**	—
Docker	ca_peerOrg1	109.8	109.8	0.96	0.42	70 B	0 B
Docker	ca_peerOrg2	6.7	6.7	0.00	0.00	70 B	0 B
Docker	orderer.example.com	6.3	6.3	0.00	0.00	177 B	0 B

表 6、表 7 表示在提交请求下不断增加交易数量以及查询请求下不断增加发送请求, 得出不同情况下系统处理返回数据的 TPS 以及请求时延情况 (实验数据为 10 次请求的平均值).

首先分析吞吐量情况, PBFT 共识机制与本文方法提交事务与查询事务吞吐量对比情况如图 4、图 5 所示.

结果表明, 在提交事务中随着交易数量的增加吞吐量逐渐增大并趋向饱和, 可以看出当交易数量达到一定程度之后吞吐量将维持在一个平稳的水平. 通过优化一致性协议、视图转换协议以及垃圾回收机制, 对比发送请求平均吞吐量从 604 TPS 提升到了 756 TPS 提升了 25% 而响应处理请求的平均吞吐量从 121 TPS 提升到了 137 TPS 提升了 13%. 在查询事务中设置交易数量为 5 000 在不同的请求发送速率下可得, 系统响

应平均吞吐量从 298 TPS 提升到了 350 TPS 提升了 17%.

接下来分析交易请求的时延情况, 传统 PBFT 共识机制与本文方法提交事务与查询事务的时延对比情况, 如图 6、图 7 所示.

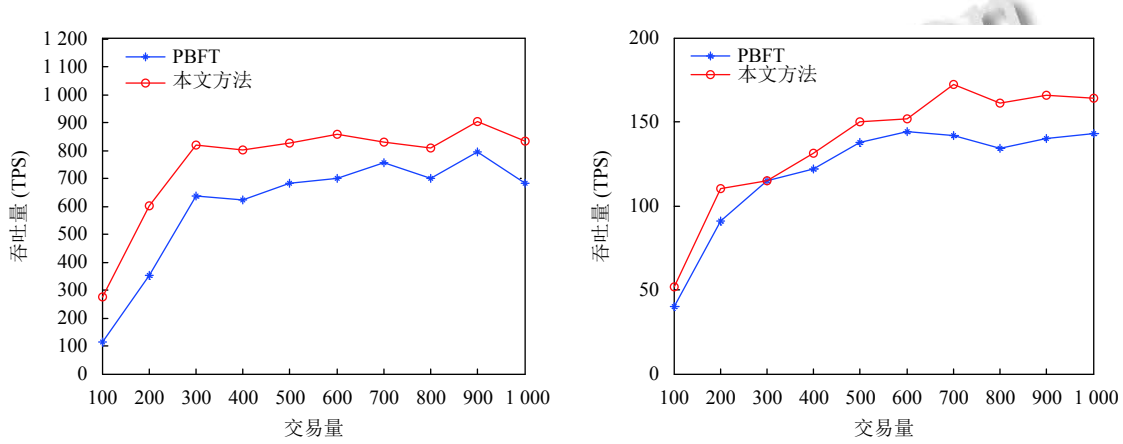
表 6 不同事务下 PBFT 的 TPS 以及时延

Test	Name	Succ	Send Rate (TPS)	Max Latency (s)	Min Latency (s)	Avg Latency (s)	Throughput (TPS)
1	submit	100	113	2.44	1.60	2.02	40
2	submit	200	351	2.13	1.60	1.89	91
3	submit	300	637	2.06	2.06	2.34	115
4	submit	400	622	3.25	1.84	2.50	122
5	submit	500	684	3.58	2.12	2.85	138
6	submit	600	699	4.14	2.67	3.39	144
7	submit	700	756	4.94	2.39	3.89	142
8	submit	800	699	5.98	2.90	4.56	134
9	submit	900	796	6.39	2.05	5.01	140
10	submit	1 000	684	6.71	2.43	5.18	143
11	query	5 000	100	0.07	0.00	0.01	100
12	query	5 000	200	0.29	0.01	0.02	200
13	query	5 000	300	6.67	0.01	2.58	284
14	query	5 000	398	9.65	0.01	4.96	334
15	query	5 000	483	12.68	0.01	4.92	354
16	query	5 000	573	12.73	0.01	7.13	354
17	query	5 000	646	13.92	0.46	7.97	345
18	query	5 000	763	14.49	1.18	9.16	343
19	query	5 000	839	14.96	1.04	10.36	324
20	query	5 000	957	14.56	4.10	10.72	343

结果表明, 在进行提交事务与查询事务时, 系统的最大时延、平均时延都随着交易数量的增加而增加, 提交事务的平均时延从 3.36 s 下降到 2.56 s, 查询事务的平均时延从 4.96 s 下降到 4.66 s, 有效的提升了果品质量溯源用户的系统使用体验.

表 7 不同事务下本文方法的 TPS 以及时延

Test	Name	Succ	Send Rate (TPS)	Max Latency (s)	Min Latency (s)	Avg Latency (s)	Throughput (TPS)
1	submit	100	277	2.21	1.60	1.81	52
2	submit	200	603	2.42	1.72	2.01	110
3	submit	300	820	2.45	2.06	2.13	115
4	submit	400	801	2.67	1.84	2.08	131
5	submit	500	826	2.86	2.12	2.28	150
6	submit	600	857	3.41	2.32	2.71	152
7	submit	700	831	3.94	2.51	3.11	172
8	submit	800	810	5.02	2.90	3.42	161
9	submit	900	904	5.17	2.21	4.01	166
10	submit	1 000	835	5.31	2.33	4.30	164
11	query	5 000	100	0.06	0.00	0.01	100
12	query	5 000	200	0.21	0.01	0.02	200
13	query	5 000	300	3.42	0.01	1.29	300
14	query	5 000	400	7.73	0.01	3.61	370
15	query	5 000	494	9.51	0.01	3.93	364
16	query	5 000	585	11.45	0.01	6.06	373
17	query	5 000	674	11.55	0.32	6.69	413
18	query	5 000	781	12.40	0.92	7.96	452
19	query	5 000	878	13.01	1.04	8.39	463
20	query	5 000	979	13.21	2.10	8.57	458

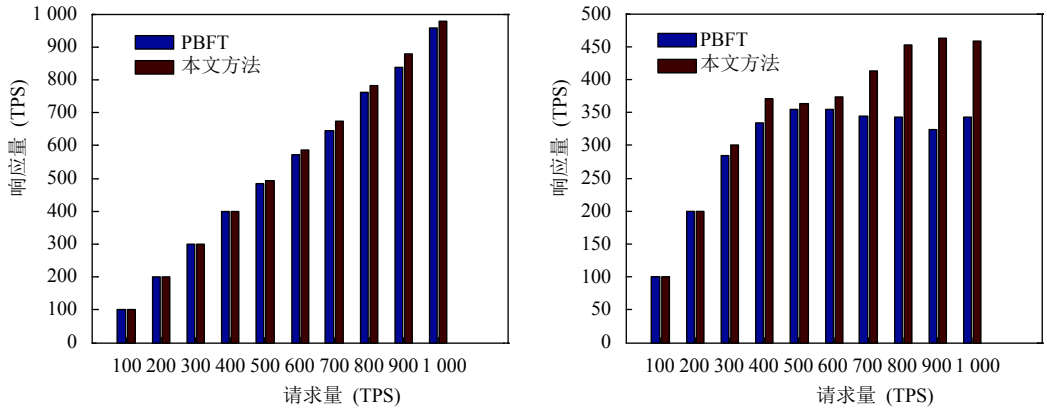


(a) 发送请求 (b) 处理请求

图 4 提交事务吞吐量对比

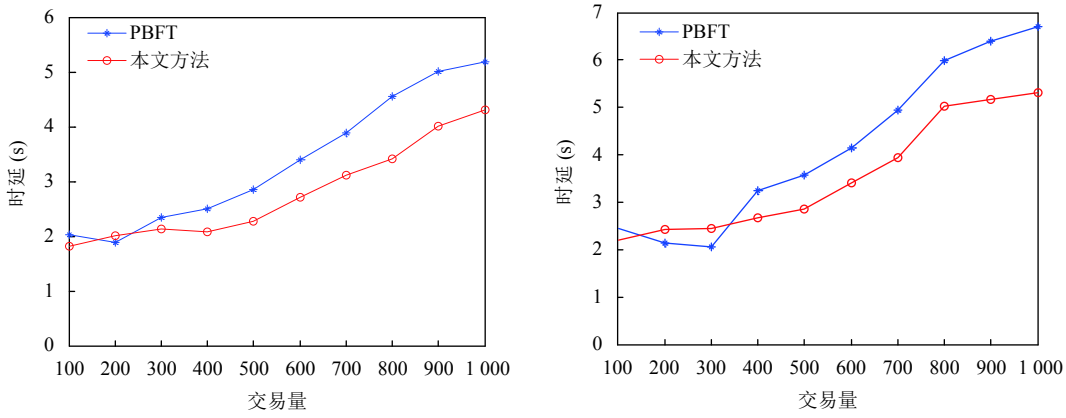
4 结论

针对基于区块链的果品质量溯源系统中存在的共识算法性能低下问题, 本文对 PBFT 算法进行分析并引入积分选择协议, 优化了一致性协议、视图转换协议以及垃圾回收机制. 在保证算法容错性的同时, 降低了共识过程中的传输消耗, 提高了吞吐量, 缩短了共识达成时间. 在运行垃圾回收机制时给所有节点重新分配积分, 达到了动态更改节点的目的. 最后, 通过实验证明了基于积分选择的改进 PBFT 共识机制的有效性.



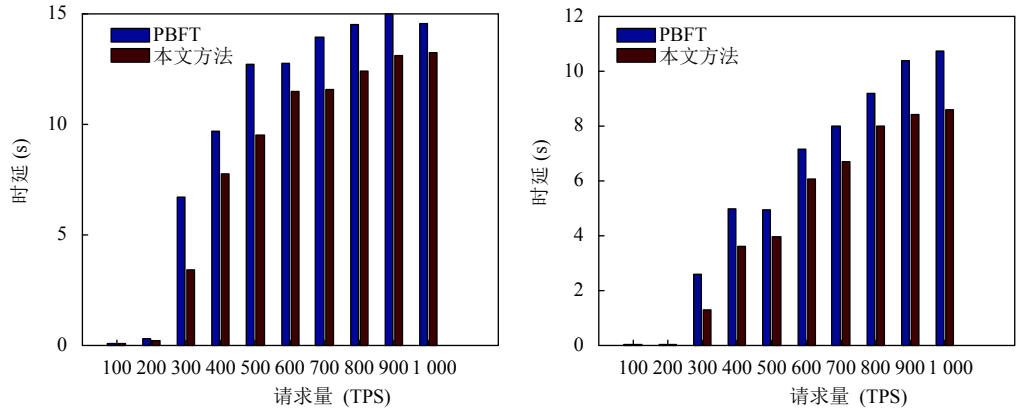
(a) 发送请求 (b) 处理请求

图 5 查询事务吞吐量对比



(a) 发送请求 (b) 处理请求

图 6 提交事务时延对比



(a) 发送请求 (b) 处理请求

图 7 查询事务时延对比

参考文献

1 Wang L, Hu Z, Liu SY, *et al*. Application of non-orthogonal multiple access for IoT in food traceability system. 2019 IEEE 4th Advanced Information Technology, Electronic and

Automation Control Conference (IAEAC). Chengdu: IEEE, 2019. 104–109.
2 杨晓, 秦一浪, 王进磊. 果品质量安全溯源管理系统设计. 现代农业科技, 2018, (16): 245–247. [doi: 10.3969/j.issn.1007-5739.2018.16.148]
3 何蕾, 马征, 王刚, 等. 基于区块链技术的茶叶质量溯源研究. 合作经济与科技, 2021, (5): 114–117.
4 王仕栋, 孙建明, 李昭, 等. 基于区块链技术的农产品质量溯源系统. 包装学报, 2020, 12(6): 80–85. [doi: 10.3969/j.issn.1674-7100.2020.06.011]
5 郑旭东, 杨现民, 岳婷燕. 教育政务数据开放平台的区块链技术架构与运行机制设计. 中国电化教育, 2021, (3): 71–78. [doi: 10.3969/j.issn.1006-9860.2021.03.011]
6 Driscoll K, Hall B, Paulitsch M, *et al*. The real Byzantine Generals. The 23rd Digital Avionics Systems Conference (IEEE Cat. No. 04CH37576). Salt Lake City: IEEE, 2004. 4–61.
7 王振辉. 基于二维码的食品溯源系统. 农业工程, 2017, 7(6): 29–32, 51. [doi: 10.3969/j.issn.2095-1795.2017.06.009]
8 Bibi F, Guillaume C, Gontard N, *et al*. A review: RFID technology having sensing aptitudes for food industry and their contribution to tracking and monitoring of food products. Trends in Food Science & Technology, 2017, 62: 91–103.
9 袁源, 郑嘉利, 石静, 等. 基于 Q-learning 的 RFID 多阅读器防碰撞算法. 计算机科学, 2019, 46(6): 124–127. [doi: 10.11896/j.issn.1002-137X.2019.06.018]
10 Salah K, Nizamuddin N, Jayaraman R, *et al*. Blockchain-based soybean traceability in agricultural supply chain. IEEE Access, 2019, 7: 73295–73305. [doi: 10.1109/ACCESS.2019.2918000]
11 Tsang YP, Choy KL, Wu CH, *et al*. Blockchain-driven IoT for food traceability with an integrated consensus mechanism. IEEE Access, 2019, 7: 129000–129017. [doi: 10.1109/ACCESS.2019.2940227]
12 工业和信息化部信息化和软件服务业司. 中国区块链技术和应用发展白皮书 (2016). http://www.ciotimes.com/recommend/119951.html. (2016-10-18).
13 黄冬艳, 李浪, 陈斌, 等. RBFT: 基于 Raft 集群的拜占庭容错共识机制. 通信学报, 2021, 42(3): 209–219.
14 杨绿林. 基于改进 PBFT 算法的区块链溯源系统设计与实现 [硕士学位论文]. 北京: 北京邮电大学, 2019.
15 徐治理, 封化民, 刘飚. 一种基于信用的改进 PBFT 高效共识机制. 计算机应用研究, 2019, 36(9): 2788–2791.
16 方维维, 王子岳, 宋慧丽, 等. 一种面向区块链的优化 PBFT 共识算法. 北京交通大学学报, 2019, 43(5): 58–64.
17 孙嘉豪, 孟翔斯, 张浩运, 等. 基于改进 PBFT 的区块链知识产权保护模型. 计算机工程, 2020, 46(12): 134–141.