计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2020,29(9):255−259 [doi: 10.15888/j.cnki.csa.007592]

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

基于信用分级的 PBFT 共识算法改进方案①

丁庭琛, 陈世平

(上海理工大学 光电信息与计算机工程学院, 上海 200093)
通讯作者: 丁庭琛, E-mail: 741907920@qq.com

摘　要: 针对现有实用拜占庭容错算法 (PBFT) 在联盟链应用场景下存在扩展性差, 通信开销大, 效率低等问题, 提出了一种基于信用分级的拜占庭容错共识算法, 即 CLBFT (Credit- Layered Byzantine Fault Tolerance). 在 PBFT 基础上, 制定节点信用积分规则. 提出一种基于信用等级划分的机制, 把节点划分成 4 类, 增强可信节点的主动性, 减少异常节点的参与, 达到系统良好运行的目的. 实验结果表明, 在长期运行状态下, CLBFT 明显减少了通信开销, 提高了系统效率.

关键词: 联盟链; 共识机制; 实用拜占庭容错算法; CLBFT

引用格式: 丁庭琛,陈世平.基于信用分级的 PBFT 共识算法改进方案.计算机系统应用,2020,29(9):255–259. http://www.c-s-a.org.cn/1003-3254/7592.html

Improved PBFT Consensus Mechanism Based on Credit-Layered Mechanism

DING Ting-Chen, CHEN Shi-Ping

(School of Optical Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China)

Abstract: Since the existing Practical Byzantine Fault Tolerance (PBFT) consensus algorithm applied to the consortium Blockchain has the problems of poor scalability, high communication overhead, and low efficiency, the Credit-Layered Byzantine Fault Tolerance (CLBFT) consensus algorithm was proposed. Based on the PBFT, a node credit score rule was formulated. A mechanism based on credit-layered was proposed, which divides the nodes into four categories, to enhance the initiative of trusted nodes and reduce the participation of abnormal nodes in order to achieve the purpose of good system operation. The experimental results show that under long-term operation, CLBFT can reduce communication overhead and improve system efficiency.

Key words: consortium Blockchain; consensus mechanism; PBFT; CLBFT

区块链技术最早出现在比特币中[1], 作为已知的分布式账本, 在过去几年中引起各界广泛的研究. Blockchain 是一种点对点分布式系统, 具有高安全性和分散存储, 高容错和加密性等特性. 为了解决现有中心化机构效率低、成本高、数字资源垄断等问题, 区块链整合密码学、计算机和通信等领域等技术, 所用技术有非对称加密、时间戳、共识机制和点对点通信[2], 实现中心化分布式系统. 区块链技术被认为是引起人类社会颠覆性变革的关键技术之一[3].

公有链和联盟链是区块链的两种主要形式[4]. 比特币是区块链最早的应用, 也是公共区块链最著名的例子. 比特币作为最早的数字货币, 仅使用单一的去中心化技术, 存在很大功能局限性. 随着智能合约[5] 的发展, 公有链有了更加智能的应用平台. 例如以太坊 (etherum) 平台就是一个可编程的区块链平台[6], 在系统资产自动化管理方面有显著进步. 公有区块链是完

① 基金项目: 国家自然科学基金 (61472256, 61170277); 上海理工大学科技发展基金 (16KJFZ035, 2017KJFZ033)
Foundation item: National Natural Science Foundation of China (61472256, 61170277); Science and Technology Development Fund of University of Shanghai for Science and Technology (16KJFZ035, 2017KJFZ033)
收稿时间: 2020-01-20; 修改时间: 2020-02-25; 采用时间: 2020-03-24; csa 在线出版时间: 2020-09-04

全去中心化系统, 没有管理和监督, 任何人都可以参与公有链并且访问数据. 因此, 公有链存在一定的弊端, 隐私和安全性难以得到保障, 可靠性差[7]. 为了更好地保护用户的隐私和监督数据, 提出了联盟链概念, 其中, Hyperledger 是经典的联盟链系统[8]. 联盟链在企业级组织内部广泛发展[9]. 在联盟区块链中, 只有特定允许的节点才能访问网络. 因此, 联盟区块链可以很好地支持企业级应用程序, 并在公共和政府服务中被广泛采用.

共识机制是区块链的核心, 区块链中有 PoW 和 PoS 算法[10], Raft 算法[11], PBFT 算法[12] 等共识机制. 一般情况下存在拜占庭问题都由 PBFT 算法解决. 拜占庭式故障模型[13] 假设副本可以通过任意行动而被恶意攻击[14], 该模型适用于区块链系统. 针对模型中的共识问题, 许多理论成果相继出现, 不过较早的协议都是从理论上解决问题, 无法在实际场景中应用. PBFT 是第一个为实际应用开发的协议, 解决了分布式文件系统可容忍拜占庭式故障问题. 不过 PBFT 无法在实际场景中广泛应用, 它存在扩展性差, 通信开销大, 效率低等问题[15].

针对联盟链的应用场景, 本文提出了一种基于 PBFT 改进的一致性算法, 称为 CLBFT (Credit- Layered Byzantine Fault Tolerance). 受 PoW 共识机制的启发, 基于信用奖励和惩罚节点, 基于节点信用值的基础上给节点分级, 旨在长期维持系统良好运行. 通过这种方案, 可以大大提高参与者的积极性, 减少恶意节点参与带来的影响, 提高系统的安全性和效率.

1 PBFT

Castro 和 Liskov 在 1999 年提出的实用的拜占庭容错协议 (PBFT) 被认为是解决拜占庭将军问题的最经典协议. PBFT 是为解决分布式系统中存在拜占庭故障节点从而达成信息一致性的问题. 为保障系统正常运转, 当系统中无效或者恶意点数为 f 时, 要求总节点 n 不小于 $3f+1$. PBFT 算法主要包括一致性协议, 视图切换协议和检查点协议 3 个部分.

1.1 一致性协议

一致性协议是 PBFT 算法的核心, 主要作用是保证区块链系统中信息的正确和相同. 在 PBFT 算法中存在客户端 (client), 主节点 (primary), 从节点 (replica) 3 种角色. 客户端 c 主要作用是向主节点发送请求 <REQUEST, o, t, c>, o 为请求的具体操作, t 为时间戳. 主节点通过视图编号以及节点数集合来确定, 主节点公式如下: $p = v \bmod |R|$, 其中 v 是视图编号, $|R|$是节点个数, p 是主节点编号. 主节点和从节点作为副本节点参与算法的主要 3 个通信阶段: 预准备阶段 (pre-prepare)、准备阶段 (prepare)、确认阶段 (commit).

1) 预准备阶段: 主节点接受到客户端 c 的请求, 丢弃错误请求, 对正确的请求进行排序, 并分配编号 n. 然后主节点向系统中的其他从节点广播一条<<PRE-PERPARE, v, n, d>, m>消息.

2) 准备阶段: 节点对收到的预准备消息进行判断, 如果同意预准备消息, 则进入准备阶段, 然后向其他节点 (包括主节点) 广播<PREPARE, v, n, d, i>消息.

3) 确认阶段: 当节点收到 $2f+1$ 个通过验证的消息时准备阶段结束进入确认阶段, 节点向包括主节点在内的其他节点发送<COMMIT, v, n, $D(m)$, i>消息.

图 1 是一次一致性协议过程. “Client”是客户端, “Primary”是主节点, “Replica1”、“Replica2”、“Replica3”是 3 个从节点. 即使“Replica3”是故障节点, 系统任可以通过一致性协议.

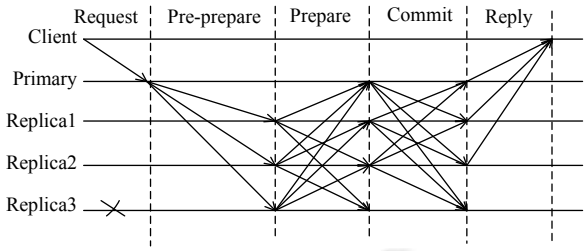

图 1 一致性协议交互过程

1.2 视图切换协议

视图切换协议是针对主节点发生故障时, 维持系统正确运行的协议. 1 个视图中有 1 个主节点, 当主节点发生故障时, 视图需要改变, 视图编号加 1 即 $v+1$, 更换新的主节点. 主节点发生故障时, 从节点触发视图切换协议, 系统设置两个触发条件: 从最近完成的一个区块的时间戳 T 开始, 在限定时间 T_1 内没有收到新主节点的 Pre-prepare 广播, 或是在限定时间 T_2 内没有生成新的区块, 其中 $T_1<T_2$. 上述两个触发条件满足其中一个就会触发视图切换协议. 为了保证系统的正确性和一致性, 视图切换也要进行节点间的交互通信. View-Change 的工作过程如下:

1) 视图切换协议开启后, 从节点进入视图 $v+1$, 并向所有节点广播 View-Change 消息.

2) 副本节点在收到 $2f+1$ 条 (包括自身)View-Change 消息后, 向视图 $v+1$ 中的主节点发送 View-Change-Ack 消息, 新主节点收到 View-Change-Ack 消息后进入

New-View 阶段.

3) 新的主节点选择检查点作为“New-View”请求的起始状态, 然后根据本地块链接数据执行一致性协议.

1.3 检查点协议

在共识过程中, 节点会生成大量的日志, 系统长期运行下就会存储大量信息. 检查点协议的作用就是减小节点信息存储规模, 释放经过共识认证的日志消息, 降低系统内存开销. 某些节点由于自身故障或网络问题没有和其他节点同步, 影响系统的运行, 因此需要 Checkpoint 协议周期性工作, 它在确认节点的一致性后清除经过验证的证书.

1.4 PBFT 存在的问题

虽然 PBFT 算法对区块链的共识性能改善很大, 但是任然存在通信开销大, 扩展性差, 效率低等问题. 首先 PBFT 算法是一种部分同步模式共识算法, 为了保证非故障节点以相同顺序执行客户端的请求, 需要三阶广播通信实现异步模式下的安全性, 造成大量的通信开销. 其次, PBFT 算法需要点对点通信进行拜占庭容错共识. 在 N 个节点的网络中, PBFT 通信的时间复杂度是$\mathrm{O}\left(N^2\right)$, 经过三阶段共识之后消息传输次数为 $2N(N-1)$. 由于通信的复杂性, 当节点数超过一定量时, PBFT 协议的性能会急剧下降. 最后是 PBFT 算法主节点选取随意, 选到恶意节点的概率偏高, 影响系统效率.

2 基于分级的高效共识机制

PBFT 算法中选取主节点随意. 它是根据编号的顺序依次得到主节点, 并且选出的主节点没有任何检验, 无法保证系统的安全性. 用这种方法选举出来的主节点存在恶意节点的可能性很大, 从识破恶意节点到通过视图切换协议更换主节点, 造成大量网络通信开销, 降低了系统效率. 本文采用信用评估和节点分级协议对 PBFT 算法加以改进, 提出了 CLBFT 共识机制. CLBFT 主要改进的地方有:

1) 制定信用积分规则, 评估节点信用状态.

2) 依据信用积分对节点进行分级, 选择积极节点参与一致性协议, 提高节点动态性.

3) 在可信节点层选择主节点, 大大减少恶意节点对系统运行的破坏, 减少通信开销, 提高系统效率.

2.1 信用分级协议

定义 C_i 代表节点的信用积分, 节点的信用级别划分为 4 个级别: A、B、C、D. 刚加入的节点的信用值为 C_{normal}, 根据节点在系统中的行为增加信用积分或减少信用积分. 节点在系统中参与一次有效区块的生成则节点信用积分加 1; 节点在系统中未生成有效区块则信用积分减 5. 信用值在 $C_{\text{normal}}\le C_i<C_{\text{good}}$ 区间的节点信用级别为 B, 初入节点也在此列; 当节点的信用值大于 C_{good} 时, 即 $C_i>C_{\text{good}}$, 节点信用级别为 A; 信用值为 $C_{\text{bad}}\le C_i<C_{\text{normal}}$ 时, 节点信用级别为 C; 当节点的信用值低于 C_{bad} 时, 即 $C_i<C_{\text{bad}}$ 节点的信用级别为 D. 节点信用分级转换如图 2 所示.

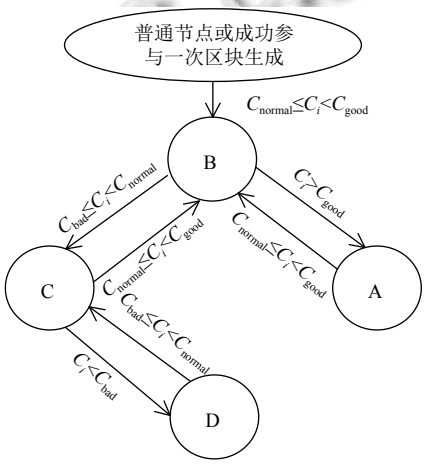

图 2 节点信用状态分级图

依据信用分级协议将节点划分为四类, 每类节点有不同的权限. A 类节点信用级别最高, 优先担任主节点. 其次是 B 类节点, 当 A 类节点被选择完毕或不存在 A 类节点, 可以从 B 类节点中选择主节点. C 类节点由于信用级别偏低不适合担任主节点, 但任然可以作为从节点参与区块共识. D 类节点信用级别太低, 不能参与共识. 权限分类不仅能够大大提高节点的积极性, 而且有效预防恶意节点成为主节点. 有效地减少恶意节点参与共识带来的通信损失和减少 View-Change 变更次数, 提高系统效率, 如表 1 区分节点权限.

表 1 节点权限分类

信用级别	优先担任主节点	担任主节点	担任从节点
A	√	√	√
B	x	√	√
C	x	x	√
D	x	x	x

2.2 CLBFT 共识过程

CLBFT 算法的过程如图 3 所示. 首先, 根据信用分级协议对节点进行分级. 信用分级协议的周期为

zCT, 其中 *CT* 是检查点协议的周期, *z* 为系数. 依据系统中节点数量调整合适的系数 *z* 按照周期间隔 *zCT* 更新节点信用积分, 节点依据信用积分分成不同级别. 然后, 不同级别的节点有不同的权限, 有相应权限的节点参与选出主节点. 节点根据自身的行为获得相应的信用积分, 当节点积分满足 A 类信用级别时, 进入集合 *A*. 集合 *A* 中的节点获得视图编号, 参与主节点选择, 保证主节点选取的最优性. 接下来, 主节点选出后参与一致性协议, 并监督一致性协议以判断主节点是否触发 View-Change 中设置的两个超时触发条件. 若超时, 触发视图切换协议, 更换主节点, 视图 *v*+1, 否则生成新区块写入区块链. 最后执行改进的检查点协议 (Checkpoint) 释放经过共识认证的日志消息, 降低系统内存开销.

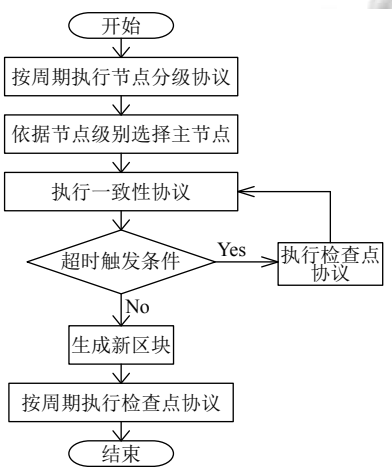

图 3 CLBFT 流程图

3 性能分析与实验

PBFT 算法的主节点选取较为随意, 是根据编号的顺序依次得到主节点. 用这种方法选举出来的主节点存在恶意节点的可能性很大, 视图切换协议较多会造成大量网络通信开销, 降低了系统效率. 本文提出的 CLBFT 算法是对 PBFT 算法的改进. 制定信用积分规则, 评估节点信用状态. 依据信用积分对节点进行分级, 积极节点参与一致性协议, 消极节点权限受限, 提高节点动态性. 在可信节点层选择主节点, 减少恶意节点对系统运行的破坏, 减少通信开销, 提高系统效率.

吞吐量一般指单位时间内系统处理的事务数, 吞吐量的高低显示了系统承受负载, 处理事务或者请求交易的能力. 在区块链领域中, 一般用每秒交易数 *TPS* (Transaction PerSecond) 来表示, 即:

$$TPS = transcations/\Delta t \tag{1}$$

其中, *transcations* 为出块时间内系统处理交易数, Δt 为出块时间.

系统硬件配置为: Inter Core i5-7300 CPU, 8GB 内存. Linux 系统是 Ubuntu16.04, 根据 Hyperledger Fabric V1.1 的环境建立仿真平台. 采用多节点运行分布式共识过程.

3.1 算法分析

假设系统的节点总数为 *n*, 系统发生视图切换协议的概率为 *p* (*p* 是平均发生视图切换协议次数占总共识次数的占比), *w* 用于统计通信次数.

PBFT 一致性协议经过三阶段广播, 通信的时间复杂度是 $O(N^2)$, 通信次数为 $2n(n-1)$. 视图切换协议的通信次数为 $n(n-1)$. 所以 PBFT 在 *p* 概率下发生视图切换协议后的总通信次数可以计算 $2n(n-1)+pn(n-1)$, 即:

$$w = (p+2)n(n-1) \tag{2}$$

CLBFT 使用信用分级协议有效地预防错误节点成为主节点, 并降低错误节点参与共识的概率. CLBFT 算法发生视图切换的概率为 p_1, $p_1<p$. 信用分级协议周期性触发, 系统通过一个周期中各个节点所得信用积分给节点分不同等级, 然后通过副本节点广播给其他节点, 所用通信次数为: $(n-1)$. 所以 CLPBFT算法在 p_1 概率下发生视图切换协议总通信次数为: $2n(n-1)+p_1n(n-1)+(n-1)$, 所以:

$$w = (p_1+2)n(n-1)+(n-1) \tag{3}$$

CLBFT 算法应用节点信用分级协议, 恶意节点参与区块共识几率下降, 视图切换概率大大降低, 所以 $p_1<p$. 系数对复杂度为 $O(N^2)$ 通信开销影响很大.

3.2 吞吐量测试

图 4 所示, 纵坐标为系统吞吐量, 横坐标为系统运行时间. 系统设置 100 个节点, 错误节点随机变化但不超过 33 个, 满足总节点 *n* 不少于 $3f+1$, *f* 为恶意节点, 比较 PBFT 和 CLBFT 的吞吐量随时间变化. 在容错范围内, PBFT 的效率在整个模拟过程中是稳定的. 随着系统长期运行, 基于信用分级的 CLBFT 算法有效地提高系统的吐吞量. 由于恶意节点参与共识概率大大降低, 主节点错误率下降, 视图切换协议的概率随之下降, 系统稳定性和效率得到提升.

3.3 通信开销验证

在图 5 中, 纵坐标为区块平均生成一次的通信次

数, 横坐标为系统中的节点数, 节点数分别取 20,40, 60,80,100 比较 PBFT 和 CLBFT 随着节点增多通信变化. 随着系统内节点的增加, CLBFT 通信开销比 PBFT 越来越小, 如图 5 所示.

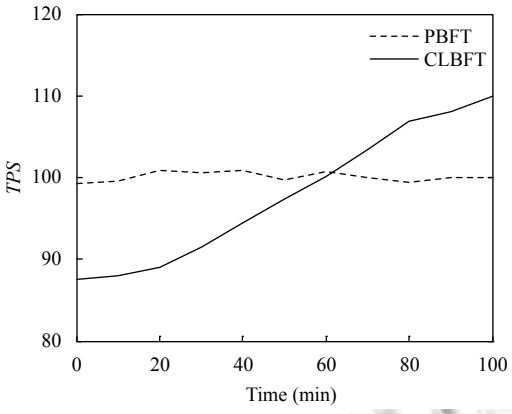

图 4 PBFT 和 CLBFT 的 TPS 随时间比较

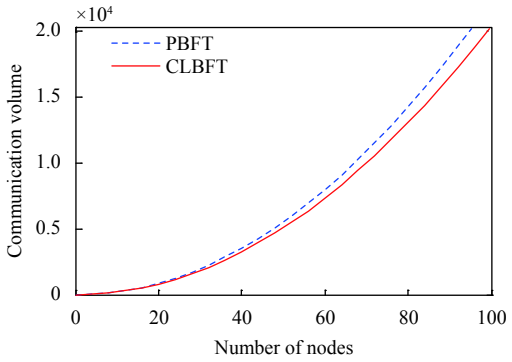

图 5 PBFT 和 CLBFT 随着节点增多通信变化

4 结论

近年来, 区块链在多个领域日益流行. 作为区块链的两种主要形式, 公有链和联盟链在不同应用领域研究各自核心共识机制. 针对联盟应用场景, 现有实用拜占庭容错算法 (PBFT) 存在视图变更频繁, 系统通信开销过大, 大量节点加入后系统效率低下等问题. 本文提出了一种基于 PBFT 改进的 CLBFT 算法, 设置节点基于信用分级的方法, 依据节点在系统中的表现赋予节点不同的权限. 降低恶意节点参与共识的概率, 从而有效避免频繁的视图变更带来的通信资源浪费. 仿真结果表明, 在系统长期运行下, CLBFT 降低了系统通信开销, 提高了系统效率.

参考文献

1 Nakamoto S. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/en/bitcoin-paper. 2008.
2 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.
3 张宇. 区块链视域下的高校人力资源培训体系研究. 当代经济, 2018, (22): 144–148. [doi: 10.3969/j.issn.1007-9378.2018.22.060]
4 王宇昊. 面向供应链管理的区块链共识机制研究 [硕士学位论文]. 厦门: 华侨大学, 2019.
5 Liang XP, Shetty S, Tosh D, *et al*. ProvChain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID). Madrid, Spain. 2017. 468–477.
6 黄秋波, 安庆文, 苏厚勤. 一种改进 PBFT 算法作为以太坊共识机制的研究与实现. 计算机应用与软件, 2017, 34(10): 288–293, 297. [doi: 10.3969/j.issn.1000-386x.2017.10.051]
7 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.
8 邵奇峰, 金澈清, 张召, 等. 区块链技术: 架构及进展. 计算机学报, 2018, 41(5): 969–988. [doi: 10.11897/SP.J.1016.2018.00969]
9 Hull R. Blockchain: Distributed event-based processing in a data-centric world: Extended abstract. Proceedings of the 11th ACM International Conference on Distributed and Event-Based Systems. Barcelona, Spain. 2017. 2–4.
10 张亮, 刘百祥, 张如意, 等. 区块链技术综述. 计算机工程, 2019, 45(5): 1–12.
11 Ongaro D, Ousterhout J. In search of an understandable consensus algorithm. Proceedings of the 2014 USENIX Annual Technical Conference. Philadelphia, PA, USA. 2014. 305–319.
12 Castro M, Liskov B. Practical byzantine fault tolerance and proactive recovery. ACM Transactions on Computer Systems, 2002, 20(4): 398–461. [doi: 10.1145/571637.571640]
13 Lamport L, Shostak R, Pease M. The byzantine generals problem. ACM Transactions on Programming Languages and Systems, 1982, 4(3): 382–401. [doi: 10.1145/357172.357176]
14 Thai QT, Yim JC, Yoo TW, *et al*. Hierarchical Byzantine fault-tolerance protocol for permissioned blockchain systems. The Journal of Supercomputing, 2019, 75(11): 7337–7365. [doi: 10.1007/s11227-019-02939-x]
15 Wang YH, Cai SB, Lin CL, *et al*. Study of blockchains's consensus mechanism based on credit. IEEE Access, 2019, 7: 10224–10231. [doi: 10.1109/ACCESS.2019.2891065]