Journal of Computer Applications
计算机应用
ISSN 1001-9081
CODEN JYIIDU
2019-01-28
http://www.joca.cn

DOI:10.11772/j.issn.1001-9081.2018102049

基于投票机制的拜占庭容错共识算法

王海勇[1]，郭凯璇[2*]，潘启青[2]

(1.南京邮电大学 计算机学院，江苏省 南京市 210003；
2.南京邮电大学 物联网学院，江苏省 南京市 210003)

(*通信作者电子邮箱 m15253171930@163.com)

摘 要: 针对现有的区块链中实用拜占庭容错（Practical Byzantine Fault Tolerance, PBFT）共识算法、动态授权的拜占庭容错（ DDBFT）共识算法、联盟拜占庭容错（ CBFT）共识算法普遍存在能耗高、效率低、扩展性差等问题，本文通过引入投票机制，提出了基于投票机制的拜占庭容错（Practical Byzantine Fault Tolerant based on Voting, VPBFT）共识算法。VPBFT 算法将网络中的节点划分为四类具有不同职责的节点，分别为投票节点、候选节点、生产节点、普通节点，节点之间的状态随时间变化可动态调整。通过性能仿真分析，本算法相比于 PBFT、 DDBFT、CBFT 等共识算法，在能耗、时延、容错、动态性等性能方面都有了明显提升。

关键词: 区块链；拜占庭容错；投票机制；共识算法；数据块

中图分类号:TP301.6 **文献标志码：A**

The Byzantine Fault Tolerance consensus algorithm based on voting mechanism

WANG Haiyong[1], GUO Kaixuan[2*], PAN Qiqing[2]

(1.College of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing Jiangsu 210003, China
2. College of Internet Of Things, Nanjing University of Posts and Telecommunications, Nanjing Jiangsu 210003, China)

Abstract: Focused on the problems of high energy consumption, low efficiency and poor scalability of the Practical Byzantine Fault Tolerant (PBFT) consensus algorithm, the Dynamic authorization of Byzantine Fault Tolerant (DDBFT) consensus algorithm and the Consortium Byzantine Fault Tolerant (CBFT) consensus algorithm existed in the blockchain, this paper proposes a Practical Byzantine Fault Tolerant consensus algorithm based on voting mechanism (VPBFT) by introducing voting mechanism. The VPBFT algorithm divides the nodes in the network into four types of nodes with different responsibilities, which are voting nodes, candidate nodes, production nodes, and ordinary nodes. The state between nodes could be dynamically adjusted at any time. Through performance simulation analysis, the proposed algorithm has significantly improved performance in terms of energy consumption, delay, fault tolerance and dynamics compared with consensus algorithms such as PBFT, DDBFT and CBFT.

Keywords: blockchain; Byzantine fault tolerance; voting mechanism; consensus algorithm; data block

0 引言

自 2008 年由名为中本聪（Satoshi Nakamoto）的人（或团队）提出“一种完全通过点对点技术实现的电子现金货币”（即比特币）起[1]，区块链技术正一步一步的得到重视。区块链具有去中心化、分布式、点对点等特点[2]，随着区块链技术的发展，各种共识算法也层出不穷，比如工作量证明（Proof of Work, POW）算法[1]、实用拜占庭容错（Practical Byzantine Fault Tolerance, PBFT）算法[3]等。

拜占庭将军问题（Byzantine failures）是区块链中共识算法会考虑到的基本问题[4]，由莱斯利.兰伯特（Leslie Lamport）提出。这是一个描述分布式系统一致性的协议问题，拜占庭的将军们必须全体一致的决定是否同时对敌军发起攻击，但在将军中存在叛徒，叛徒会发出虚假信息来影响其他将军们的决定，将军们如何在存有叛徒的前提下达成一致的决定，

收稿日期: 2018-10-10; **修回日期:** 2018-12-19; **录用日期:** 2018-12-27。
基金项目: 江苏省教育信息化研究资助重点课题（20172105）、江苏省现代教育技术研究 2017 年度智慧校园专项课题（2017-R-59518）、南京邮电大学教学改革重点项目（JG06717JX66）、南京邮电大学校园信息化创新项目（NYXX217002、NYXX217004）、赛尔网络下一代互联网技术创新项目（NGII20180620）
作者简介: 王海勇(1979—)，男，江苏南京人，副研究员，博士，主要研究方向：计算机网络与安全、信息网络应用技术；郭凯璇(1991—)，女，山东枣庄人，硕士研究生，主要研究方向：区块链、共识算法、物联网；潘启青(1994—)，女，江苏南京人，硕士研究生，主要研究方向：区块链、物联网。

并最终获得胜利正是该问题所要解决的。拜占庭容错问题，在计算机领域可以表述为：如何在存有恶意节点的系统网络中确保系统运行的良好以及信息数据的完整、可靠和一致性，从而做出正确的决策。

在目前现有的共识算法中，Paxos 算法、Raft 算法和 PBFT 算法是较为经典的分布式一致性算法[5]。但是，Paxos 算法[6]和 Raft 算法[7]均是面向数据而不是面向交易的，并未考虑到拜占庭问题，即没有考虑系统中存在恶意节点的情况，一旦系统内的恶意节点发送虚假消息，那么整个系统将会存储虚假错误的信息。为解决拜占庭问题，PBFT 算法被提出[3, 8]，通过大多数诚实节点来忽略掉恶意节点的消息，该算法能够容忍不超过*(n-1)/3* 个节点失效（其中，*n* 为节点总数）。但是 PBFT 算法[7]采用的是 C/S 架构[9]，不能适应 P2P 网络，无法动态感知节点数目的变化。

随着区块链技术的进一步发展，一些新的共识算法也层出不穷，其证明方式趋向于多样化和混合化。动态授权拜占庭(Dynamic authorization of Byzantine Fault Tolerant, DDBFT)算法[10]，将 DPOS 算法应用于 PBFT 算法，使得 PBFT 算法具有动态性的特点，同时也能够提高吞吐量、降低时延，但是由于网络带宽有限，需要确认的区块较大且超出一个节点的处理能力，就会造成阻塞，降低吞吐量。联盟拜占庭容错（Consortium Byzantine Fault Tolerant, CBFT）算法[5]，以 PBFT 算法为基础，通过区块缓存、区块同步与签名、节点变更实现，具有更高的吞吐量和较低的时延，但是其交易处理的效率和达成共识的效率等需要进一步提升。

在对比分析了已有的一些共识算法后，本文提出了一种改进的 PBFT 算法，即基于投票机制的拜占庭容错（Practical Byzantine Fault Tolerant based on Voting, VPBFT）共识算法，将投票证明（Proof of Vote, POV）与 PBFT 结合，具有安全可靠、高效、低延迟和动态性等特点。

全文共分为以下几个部分：第二部分介绍已有算法；第三部分提出改进算法——VPBFT 算法；第四部分对 VPBFT 算法的性能分析；第五部分进行总结。

1 已有算法

针对解决分布式系统一致性问题的算法，已经被提出了许多。从 Paxos 算法到 Raft 算法，再到 PBFT 算法，以及对 PBFT 算法改进后形成的基于动态授权的拜占庭容错（Dynamic authorization of Byzantine Fault Tolerant, DDBFT）共识算法和联盟拜占庭容错（Consortium Byzantine Fault Tolerant, CBFT）共识算法，都在针对解决分布式系统的一致性问题。

1.1 已有算法

1990 年，莱斯利.兰伯特（Leslie Lamport）提出了 Paxos 算法[6]，该算法是基于消息传递的，旨在解决在分布式系统内如何就某一个内容达成一致的问题[11]，在分布式系统内的所有节点的初始状态一致，在执行了相同的操作后，所有节点就能够得到一致的结果。Paxos 算法具有高度的容错性，但较为难懂且难以实现，于是出现了它的简化版——Raft 算法[12]。但是，Paxos 和 Raft 算法是面向数据而不是面向交易的，没有考虑系统中存在恶意节点的情况，一旦系统内的恶意节点发送虚假消息，那么整个系统将会存储虚假错误的信息。

除此之外，由卡斯特罗（Miguel Castro）和利斯科夫（Barbara Liskov）在 1999 年提出的 PBFT 算法[14]，也是专门针对解决拜占庭将军问题的算法，该算法旨在解决如何在整个网络中存在恶意节点的情况下保证最终决策的一致性、正确性的问题。在 PBFT 算法中，所有节点被分为客户节点、主节点和备份节点 3 种类型，其中，主节点和备份节点被称为副本节点。算法流程分为 3 个阶段：预准备阶段，准备阶段，确认阶段。具体过程如图 1：

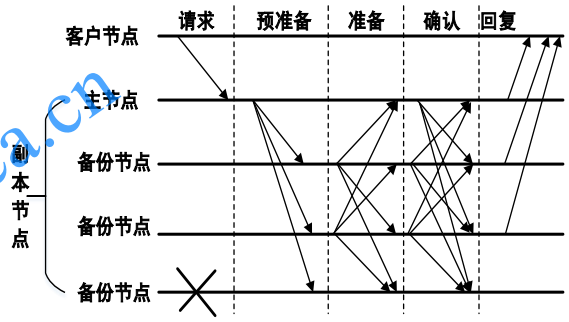

图 1 PBFT 算法流程

Fig. 1 The process of PBFT algorithm

当客户节点收到至少 *n+1* 个副本节点的结果是相同的情况下，才认可结果有效。PBFT 算法 针对分布式系统，而且系统中的指令顺序执行，是基于 C/S 架构的[8, 10]。算法的整个过程分为三阶段，具有三次信息的广播，这对网络带宽造成了一定的浪费。另外，在 PBFT 算法中，整个网络的节点数目是固定，一旦发生变动时系统无法感知，不具备扩展性。

除了上述经典的共识算法外，还有一些新提出的针对 PBFT 算法进行改进的算法：DDBFT 算法、CBFT 算法。DDBFT 算法，主要针对 PBFT 算法缺乏动态性的不足，将 DPOS 算法应用于 PBFT 算法，使得 PBFT 算法具有动态性的特点，同时也能够提高吞吐量、降低时延，但是由于网络带宽有限，需要确认的区块较大且超出一个节点的处理能力，就会造成阻塞，降低吞吐量。CBFT 算法，是以 PBFT 算法为基础，通过区块缓存、区块同步与签名、节点变更实现，具有更高的吞吐量和较低的时延，但是其交易处理的效率和达成共识的效率等需要进一步提升，且在共识流程、区块同步和节点管理方面仍存在问题。

由此可见，每种算法都具有其各自的优势及不足。其中，PBFT 算法扩展性较差，不能够适应动态变化的网络系统。DDBFT 和 CBFT 算法虽然在能耗、吞吐量、扩展性等方面有所改进，但仍然存在效率低、能耗高等不同的问题。由此可见，共识算法仍有一定的改进空间。

2 基于投票机制的拜占庭容错共识算法

通过对已有算法的分析，尤其是分布式系统的共识算法：PBFT 算法、DDBFT 算法、CBFT 算法等，本文针对这些共识算法的不足之处，提出了基于投票机制的拜占庭容错（Practical Byzantine Fault Tolerant based on Voting, VPBFT）共识算法，即 VPBFT 算法。在本算法中，将 POV 机制应用于传统的 PBFT 算法，将网络中的节点划分为四类，不同类别的节点具有不同的职责，不同类别的节点之间具有一定的数量关系。

2.1 POV 机制

POV 机制将整个联盟网络中的节点分为四类：投票者、管理者、候选人、普通用户[15]。其中，投票者具有推荐、投票管理者的权利，能够对产生的交易进行验证和转发，也能够对产生的区块进行验证；管理者只能来自于候选人，被随机的指定生成区块，有一定的任命周期，周期结束后重新被投票；候选人，由经过注册并获得多于 1 名投票者推荐的普通用户组成，也可以是投票者自荐组成；而普通用户则可以随时加入和退出。网络中所有节点都能够发生、转发并验证交易数据，数据有效才发送给投票者和管理者，并由管理者将数据放入交易池。而管理者被任命在其任命周期里生产块，且需要至少 *1+Nc/2* 个投票者的同意才能生产相应的数据块，其中 *Nc* 为投票者节点的数目。

POV 机制中的普通接节点能够随时加入网络，具有一定的扩展性，而且网络中的节点具有不同的身份和职责，在一定程度上避免了中心化。VPBFT 算法将 POV 机制引入 PBFT 算法，能够利用 POV 机制动态性的特点弥补 PBFT 算法的不足。

2.2 VPBFT 算法的网络模型

在 VPBFT 算法中，将整个网络中的节点分为四类：投票节点，生产节点，候选节点，普通节点。其网络模型如图 2 所示：

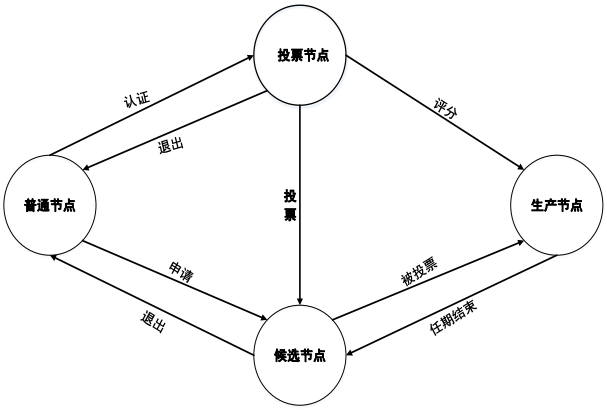

图 2 VPBFT 算法的网络模型

Fig. 2 The network model of VPBFT algorithm

其中，普通节点经过实名认证成为投票节点，投票节点负责对候选节点投票和对生产节点评分，其数目可设为 *Nv*，编号为{*0, 1, ... , Nv-1*}；生产节点由投票节点从候选节点中选出，负责生产块，数目为 *Np*，编号为{*0, 1, ... , Np-1*}；候选节点由普通节点申请成为，数目为 *Nc*，普通节点的数目为 *No*。那么，整个网络中的节点总数可以用公式 1 表示：

$$Nall = Nv + Np + Nc + No \tag{1}$$

另外，从公式 1 可以看出不同类别节点之间的数量关系，整个网络中，每种类型节点的数目是可变的，节点总数也是可变的。

网络中的所有节点都能够发生交易，且能够转发验证交易数据，有效的交易数据由生产节点放入交易池。交易池中的交易数据由被投票者节点选出来的生产节点打包，并发送给投票节点验证，验证后由生产节点生成数据块。

2.3 VPBFT 算法的算法流程

VPBFT 算法的算法流程可以分为两个阶段：准备阶段和确认阶段。其过程如下：

1）网络中所有节点都能够发生交易，并产生交易数据，交易池中存放着产生的大量有效的交易数据。

2）编号为 *i*（*i=R*）的生产节点从交易池取出一些交易数据进行打包，将生产数据块的请求以及所要生产的数据块广播发送给投票节点。这一阶段为准备阶段。其中 *R* 为随机数，包含在上一个生产节点生成的数据块中，若生产者将要生产的数据块是创世块，则 *R* 为 0。

3）投票节点收到请求后对收到的数据块进行验证，验证数据块没有被恶意篡改后，进行签名和加盖时间戳，广播回复确认消息及该数据块，此阶段为确认阶段。

4）生产节点在收到至少 *1+Nv/2* 个投票节点的确认消息后，生产该数据块。若在一定的时间内该生产节点没有生成

数据块，则由编号为 *R+1* 的生产节点继续生成数据块，重复过程 2。

整个过程可以简化为图 3：

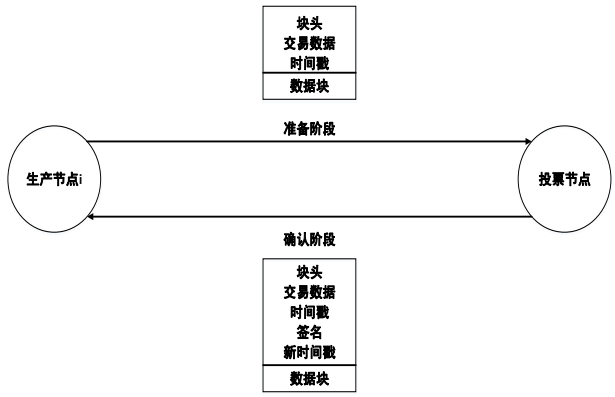

图 3 VPBFT 算法流程

Fig. 3 The process of VPBFT algorithm

在上述过程中，生产节点需要在其任命周期 *Tp* 内生成 *Bp* 个数据块。其中，前 *Bp-1* 个块为普通数据块，包含交易数据、时间戳、投票节点验证后的签名及加盖的时间戳等信息，最后一个为特殊数据块，不包含交易数据，包含投票节点给出的票数信息，用以确定下一轮生产数据块的生产节点。一轮的周期为 *T*，每一轮生产者节点的数目为 *Np*，每个生产节点的任命周期为 *Tp*，则 $T = Np \times Tp$。投票节点是否对生产节点进行投票使其进入下一轮，依据的是它们在本轮的表现：若候选节点成功被投票成为生产者节点，则加 1 分，在一轮之内，生产节点表现诚实并在其任期内成功生产出有效的数据块，则加 1 分，否则减 1 分。一轮结束后，获得 2 分的生产节点将优先被考虑进入下一轮。

2.4 随机数 *R* 的确定

每一轮中，第一个生产数据块的生产节点由随机数 *R* 确定，$0 \le R \le Np$。若生产的数据块为创世块或 *R*>*Np* 时，则 *R* 从 0 开始，逐次加 1；当数据块为非创世块时，*R* 由上一个生成的数据块确定。其确定方法如下：

假设生产节点收到来自投票节点的 *K* 个签名，也即得到 *K* 个投票，设为：

$Signature[i] (0 \le i \le K, Nv/2<K \le Nv-1)$，*Timestramp* 为时间戳，则 *Rsource* 可以由公式 2 得到[15]：

$$Rsource = (\bigoplus_{i=0}^{K-1} Signature[i]) \oplus TimeStramp \quad (2)$$

然后对 *Rsource* 进行哈希计算，取最后 32 位，化为整数后，利用公式 3 求得 *R*[15]：

$$\begin{aligned} H &= Hash(Rsource) \\ R &= StrToInt(SubStrEnd32(H)) \bmod Np \end{aligned} \quad (3)$$

2.5 *K* 的取值

在确定随机数 *R* 的过程中，投票数 *K* 是一个重要参数，那么，在网络中，如何获得 *K* 的值呢？假设在每一轮中每个投票节点投出 *K* 票，不考虑评分的影响，投票时随机的，且分别投给 *Nc* 个候选节点中的 *K* 个节点，则每个获选节点获得一票的概率相同，设为 *P1*，由公式 4 所得：

$$P1 = K / Nc \quad (4)$$

则每个候选节点获得 *X* 票的概率可设为 *P2*，由公式 5 得到：

$$\begin{aligned} P2 &= C_{N_v}^{X} \times (\frac{K}{N_c})^{X} \times (1-\frac{K}{N_c})^{N_v-X} \\ &= \frac{N_v!}{i!(N_v-i)!} \times (\frac{K}{N_c})^{X} \times (1-\frac{K}{N_c})^{N_v-X} \end{aligned} \quad (5)$$

为了使得投票结果更有效，生产节点获得的投票数应超过 *Nv/2*，因此可设 *P3* 为获得多于 *Nv/2* 个投票的概率，即为成功被选为生产节点的概率，如公式 6 所示：

$$P3 = \sum_{i=\frac{N_v}{2}+1}^{N_v} \frac{N_v!}{i!(N_v-i)!} \times (\frac{K}{N_c})^{i} \times (1-\frac{K}{N_c})^{N_v-i} \quad (6)$$

另外，*Np* 个生产节点是从 *Nc* 个候选节点中选出，那么，候选节点成功被选为生产节点的概率为 *P4*，即：

$$P4 = Np / Nc \quad (7)$$

当 *P3*=*P4* 时，即可求得 *K* 的值。当取 *Nv*=20，*Np*=50，*Nc*=200 时，可以得到图 4，从图中可以看出，当 *P3*=*P4* 时，*K* 值取值为 64。

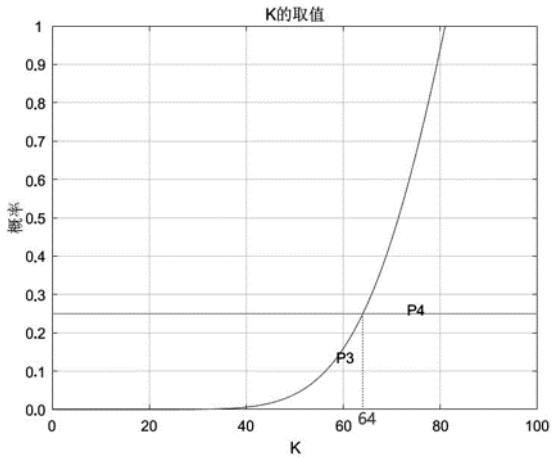

图 4 K 的取值

Fig. 4 The value of K.

2.6 小结

本文提出了一种基于投票机制的拜占庭容错共识算法，即 VPBFT 算法。在该算法中，充分应用了 POV 机制，将网络中的节点划分为四类具有不同职责的节点，并赋予一定的数量关系，根据上文对随机数 R 以及最佳投票数 K 计算的描述，当节点数目发生动态变化时，系统可自行根据行营的公式计算相应的参数，无需重新启动系统，确保了算法的动态性和可扩展性。另外，在本算法中，节点的投票权和生产权是分开的，能够确保算法的独立性。

3 性能分析

本文提出的 VPBFT 算法，是在 PBFT 算法的基础上引入 POV 机制，具有一定的动态性，同时在功耗、时延、动态性等方面也得到了进一步的改善。考虑到本算法是针对 PBFT 算法进行改进后得到的，因此本节内容，将在配置为 I5-8250U 处理器、8GB 内存、256GHz SSD 固态硬盘的 Windows10 系统下，通过 Matlab2017a 对 VPBFT 算法、PBFT 算法以及 DDBFT 算法、CBFT 算法等针对 PBFT 算法进行改进后的算法进行数学计算仿真。

3.1 低功耗

在整个网络中，每一种算法都需要进行数据传输，其所需要使用的网络带宽可用公式 8 表示：

$$Bandwith = N \times (N-1) \times Blocksize \quad (8)$$

其中，*Bandwith* 为所需要使用的网络带宽，*N* 为网络中的节点总数，*Blocksize* 为传输数据的大小，在区块链应用中，一个区块的大小约为 990KB。由公式 8 可以看出，在 *Blocksize* 大小一定时，随着 *N* 的增加，所需要使用的网络带宽随之增加，如图 5 的 *Bandwith*。

3.1.1 与 PBFT 算法相比

在前文中已知，PBFT 算法的整个过程分为预准备、准备和确认三个阶段，具有三次信息数据的广播传输；而 VPBFT 算法在具有准备和确认两个阶段，具有两次信息数据的广播传输。因此假设两种算法中的节点数目一致，则两种算法每次广播信息数据消耗的带宽一样，均为 *Bandwith*。则在整体上，VPBFT 算法则消耗带宽为 2 倍的 *Bandwith*，即图 5 中的 *Bandwith1*，PBFT 算法消耗的带宽为 3 倍的 *Bandwith*，如图 5 中的 *Bandwith2*。

3.1.2 与 DDBFT 算法相比

DDBFT 算法是将 DPOS 机制应用于 PBFT 算法，使得 PBFT 算法具有动态性。该算法整个共识过程为共识提案和共识确认两个阶段。共识提案阶段由主节点先广播交易数据，经过一定的时间后再广播共识提案；共识确认阶段由其他节点在对收到的交易数据进行验证后向主节点回复确认消息，若验证失败则广播发送配置变更消息。除此之外，在公式过程之前，网络中的代表节点需要广播告知其余节点自己的身份。因此，在 DDBFT 算法中，具有四次信息数据的广播传输。在同一网络环境中，假设 DDBFT 算法与 VPBFT 算法中的节点数目一定，则两种算法每次广播传输的信息数据消耗的带宽一样，均为 Bandwith。那么，在整体上，DDBFT 算法消耗的带宽为 4 倍的 Bandwith，如图 5 中的 Bandwith3。

3.1.3 与 CBFT 算法相比

CBFT 算法是以 PBFT 算法为基础，通过区块缓存、区块同步与签名、节点变更实现等三个阶段来实现的。该算法仍具有 PBFT 算法流程的三阶段，只是当备份节点向所有副本节点广播发送准备消息时，其他副本节点会率先形成确认消息，在收到准备消息后进行验证。若验证可靠则直接发送已形成的确认消息，否则更改确认消息后再发送。因此，CBFT 算法对网络带宽的消耗同 PBFT 算法，为 3 倍的 Bandwith，如图 5 中的 Bandwith4。

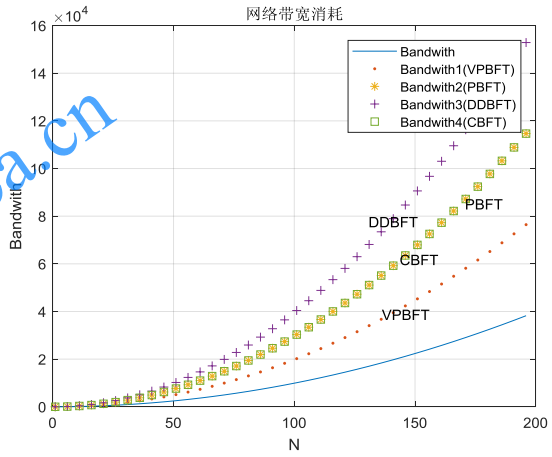

图 5 VPBFT 算法与其他算法所需网络带宽对比

Fig. 5 Comparison of network bandwidth required by VPBFT algorithm and other algorithms

3.1.4 小结

通过前 3 节以及图 5 可知，在同一个网络环境中，以节点数目相同且传输的信息数据大小相同为前提，VPBFT 算法需要的的总网络带宽明显低于 PBFT、DDBFT 以及 CBFT 算法，VPBFT 算法具更低的能耗。

3.2 可靠性

为了能够获得投票节点的投票和认可，生产节点在赢得投票后，在其任命周期内必须诚实的工作，生产出有效的数据块，完成自己的任务。在 VPBFT 算法中，生产节点会越来越可靠。如果生产节点在任命期内没有生成有效的数据块，且有如恶意篡改数据等不诚实行为，或者其生产的数据块不

被投票节点认可，那么它的分数将会下降，在下一轮中它被投票的可能性将会降低甚至可能得不到投票。没有获得投票的生产节点将失去生产数据块的机会，同时也就失去了获得工资的机会。由 VPBFT 算法的过程可知，候选节点成功被投票成为生产者节点时可获得 1 分，若在一轮之内，表现诚实并在任期内成功生产出有效的数据块，再加 1 分，否则减 1 分。一轮之后，可靠的节点获得 2 分，恶意节点获得 0 分。因此，恶意节点将难以获得投票，而可靠的生产节点更有可能被投票，从而使得整个系统更加的可靠。生产节点的可靠性是可以通过投票数 *K*、生产节点获得工资 *W* 来调整控制生产节点的可靠性。

首先是投票数 *K*。假设参数 *a* 的大小为候选节点因获得评分高低而被投票的可能性的大小，则候选节点被成功投票为生产节点的概率可用公式 9 表示：

$$P=\sum_{i=\frac{N_v}{2}+1}^{N_v}\frac{N_v!}{i!(N_v-i)!}\times(\frac{K}{N_c}+a)^i\times(1-\frac{K}{N_c}-a)^{N_v-i} \quad (9)$$

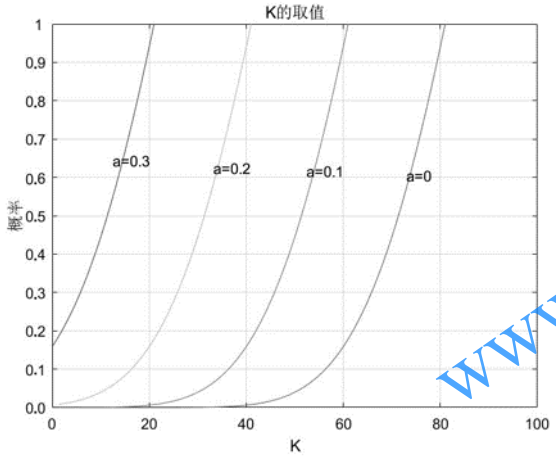

图 6 考虑评分时候选节点成功被投票为生产节点的概率

Fig. 6 Probability of candidate nodes being successfully voted as production nodes when considering scoring

分别取 *a* 为 0，0.1，0.2，0.3，根据公式 9 可画出图 6。从图中可已看出，当 *K* 值一定时，可靠节点因获得较高的评分而更有可能被投票成为生产节点。

其次是生产节点获得的总工资 *W*。由公式 9 可知，候选节点被投票为生产节点的概率随着投票数 *K* 和评分高 *a* 的增加而增加，假设每一次成功生产有效的数据块可获得工资数为 *w*，则生产节点累计获得的总工资 *W* 可以表示为：$W=P\times w$，可见随着概率 *P* 的增加所得工资数 *W* 也增加，表现可靠的生产节点将能够获得较高的工资。

投票数 *K*、生产节点获得工资 *W* 能够有效的调整控制生产节点的可靠性，越可靠越有可能获得投票，越可靠获得的工资越多，高获票概率、高工资将激励生产节点更加可靠的工作。

3.3 动态性

PBFT 算法是基于状态机复制原理的，采用 C/S 的请求响应模式，是静态网络拓扑结构的算法，无法动态地感知节点加入或离开网络，尤其是节点数目增加时，更是无法感知，甚至需要重新启动系统，重新开始计算、传输信息数据。一旦节点数目发生变化，且未重新启动系统，仍按照之前的节点数目进行运算，将使得新加入的节点资源的浪费或为不存在节点占用一定的系统资源。VPBFT 算法在一定程度上解决了动态性的问题。

VPBFT 算法，将整个网络系统中的节点划分为四类并加以量化，其中投票节点能够对生产节点进行评分以及对候选节点进行投票。被投票选中的候选节点成为生产节点并在投票节点的监督下进行数据块的生产。根据公式 1 可知，一旦节点发生变化，相应的节点参数 *Nv*，*Nc*，*Np*，No 也将发生变化，根据公式 6 和公式 7 即可求得相应的 *K* 值和 *R* 值，从而确定投票数和第一个生产数据块的生产节点，相应地，也能够调整最大容忍恶意节点的数目。

由此可见，相对于 PBFT 算法，VPBFT 算法能够动态地感知节点加入或离开网络，当节点数目增加时，不需要重新启动系统。不会出现新加入的节点资源的浪费或为不存在节点占用一定的系统资源的情况。

3.4 容错性

在 VPBFT 算法中能够容忍的失效节点数 f1 不超过 1+Nv/2，最多为 Nv/2，其中 Nv 为网络中投票节点的数目。在 PBFT 算法中，所有节点被分为三种类型：客户节点、主节点和备份节点。其中，主节点和备份节点被称为副本节点，且副本节点总数为 *Nt*，编号为{*0, 1, ... , Nt-1*}。PBFT 算法中最多能够容忍的恶意节点数为 f2=(Nt-1)/3。假设 *Nv=Nt*，也就是在两种算法中对数据具有验证权的节点数目相同的前提下，通过公式 10 可以得到 f2<f1 即：

$$f2-f1=(Nt-1)/3-Nv/2=-Nv/6-1/3 \quad (10)$$

其中，*Nv=Nt* 且均大于 0，故 *f1-f2<0* 恒成立，即 *f1<f2* 恒成立，也就是说，在 *Nv=Nt* 的前提下， VPBFT 算法的容错性比 PBFT 算法的高，能够容忍的恶意节点数目更大。

在 DDBFT 算法中，其容错性为 f3=(Nf-1)/3，其中 Nf 为具有授权权利的节点，假设 Nf=Nv，则由公式 11 可知 f3<f1，即 VPBFT 算法的容错性高于 DDBFT 算法。

$$f3-f1=(Nf-1)/3-Nv/2=-Nv/6-1/3 \quad (11)$$

在 CBFT 算法中，容错性为 f4=(N-1)/3，其中 N 为节点总数，则由公式 12 可得 f4 与 f1 的大小关系跟 N 和 Nv 有关。当 N 与 Nv 满足 N<1+3Nv/2 时，f1>f4，即 VPBFT 的容错性高于 CBFT。

$$f4 - f1 = (N - 1) / 3 - Nv/2 = (3Nv-2N)/6+1/3 \quad (12)$$

3.5 低时延

在计算机网络中，时延包括发送时延、处理时延、传输时延。同一个网络环境中，PBFT 算法与 VPBFT 算法对于信息数据的发送时延是一样的，且每次对数据的处理时延也是一样的。但是，PBFT 算法是三阶段三广播，对数据进行三次处理时延，而 VPBFT 算法是二阶段二广播，只有两次处理时延。因此，VPBFT 算法的总处理时延低于 PBFT 算法。同时，VPBFT 算法有效地提高了信息数据的传输速率，缩短了传输时间。因此，VPBFT 算法相比于 PBFT 算法有效地降低了传输信息数据的时延，提高了效率。

3.6 安全性

假设在 VPBFT 算法中非法数据块能够被验证通过。那么，由于生产节点必须获取至少 *1+Nv/2* 的投票节点的确认消息才能确定生产该数据块，所以在有效的投票节点数量多于 *1+Nv/2* 的情况下，有效投票节点不会认可非法数据块。因此，非法数据块得到的确认消息最多为 *Nv-(1+Nv/2)=Nv/2-1*，不能够被验证通过。这与假设相矛盾，因此假设不成立，非法数据块不能够被验证通过，VPBFT 算法具有一定的安全性。

4 结语

本文首先介绍了拜占庭问题以及一些共识算法，如 POW、Paxos、Raft 和 PBFT 算法以及对 PBFT 进行改进的算法：DDBFT、CBFT 等。通过分析对比已有算法，发现各有不足，其中 PBFT 算法的三阶段三广播，浪费了一定的网络带宽，且无法感知网络中节点数目的变动，不具备动态性；DDBFT 算法和 CBFT 算法，虽然在一定程度上具备了动态性，但其容错性、能耗方面还存在缺陷。针对这些不足，尤其是 PBFT 算法的不足之处，本文提出了 VPBFT 算法。该算法以 PBFT 算法为基础，引入投票机制，将网络中的节点划分为四类，不同身份的节点具有不同的职责，在一定程度上弱化了中心制。该算法中，各类节点之间具有一定的数量关系，当网络中节点数目发生变动时，能够根据数量关系进行相关参数的调整，无需重新启动系统，具有一定的动态性。通过对比可知，VPBFT 算法与 PBFT 算法相比，具有更低的能耗和时延、更高的容错性，以及一定的动态性和可靠性；与 DDBFT 算法相比，具有更低的能耗和时延、更高的容错性；与 CBFT 算法相比，具有更低的能耗和时延。VPBFT 算法还存在一些问题，如在容错性上不能够保证能够高于 CBFT 算法、节点处理数据能力有限等，需要更深入的研究。

参考文献

[1] Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.
[2] 长铗, 韩锋. 区块链：从数字货币到信用社会[M]. 北京：中信出版社, 2016：54-63. (CHANG J, HAN F. Blockchain: from digital currency to credit society[M]. Beijing: China CITIC Press, 2016: 54-63.).
[3] Bracha G, Toueg S. Asynchronous Consensus and Broadcast Protocols[J]. Journal of the Acm. 1985,32(4):824-840.
[4] Lamport, byzantine, Problem B G, et al. Seminal research document related to the field of Byzantine fault tolerance[J]. ACM Transactions on Programming Languages and Systems. Association for Computing Machinery. 1982.
[5] 李剑锋. 基于拜占庭容错机制的区块链共识算法研究与应用[D]. 郑州大学, 2018. (Li J F. Research and Application of Blockchain Consensus Algorithm Based on Byzantine Fault Tolerance Mechanism[D].2018.)
[6] Lamport. The Part-Time Parliament[J]. Acm Computing Surveys. 1998,16(2) :133-169.
[7] Ongaro D, Ousterhout J. In search of an understandable consensus algorithm[C]. Proceedings of the ATC14, USENIX Annual Technical Conference, 2014, 305-319.
[8] Reiter M K. A secure group membership protocol[C]. IEEE Journals & Magazines. 1996,22(1):31-42.
[9] Nagaraju GP, Aliya S, Zafar SF, Basha R, Diaz R, El-Rayes BF. A survey of peer-to-peer content distribution technologies[J]. Acm Computing Surveys. 2004,36(4):335-371.
[10] 刘肖飞. 基于动态授权的拜占庭容错共识算法的区块链性能改进研究[D]. 浙江大学, 2017. (LIU X F. Research on blockchain performance improvement based on Byzantine Fault Tolerance consensus algorithm based on dynamic authorization[D]. Zhejiang University,2017) [11] 分布式系统 Paxos 算法[EB/OL]. [2018-12-18]. https://www.jdon.com/artichect/paxos.html. (Distributed system Paxos algorithm [EB/OL].[2018-12-18]. https://www.jdon.com/artichect/paxos.html.).
[12] Ongaro D, Ousterhout J. In search of an understandable consensus algorithm (Extended Version) [EB/OL]. [2018-12-18]. https://raft.github.io/raft.pdf.
[13] Dwork C, Naor M. Pricing via Processing, Or, Combating Junk Mail, Advances in Cryptology[C]. International Crytology Conference, 1993.
[14] Castro M, Liskov B. Practical Byzantine Fault Tolerance[C]. Proceeding of the third symposium on operating systems design and implementation. New Orleans: USENIX association, 1999,173－186.
[15] Li K, Li H, Hou H, et al. Proof of Vote: A High-Performance Consensus Protocol Based on Vote Mechanism & Consortium Blockchain[C]. IEEE, International Conference on High PERFORMANCE Computing and Communications; IEEE, International Conference on Smart City; IEEE, International Conference on Data Science and Systems. IEEE, 2017:466-73.

This work is partially supported by Jiangsu Province Education Information Research funded topic (20172105、2017-R-59518), Nanjing University of Posts and Telecommunications Teaching Reform Project (JG06717JX66) and Nanjing University of Posts and Telecommunications Campus Information Innovation Project (NYXX217002、NYXX217004), CERNET Innovation Project (NGII20180620). The authors thank the sponsors for their support and the reviewers for helpful comments.

WANG Haiyong, born in 1979, Ph. D., Associate researcher. His research interests include computer network and security, information network application technology.
GUO Kaixuan, born in 1991, M. S. candidate. Her research interests include blockchain, consensus algorithm, internet of things.
PAN Qiqing, born in 1994, M. S. candidate. Her research interests include blockchain, internet of things.