doi:10.11835/j.issn.1000.582X.2024.07.011

基于k-means的动态多组织PBFT算法

杨雨浓[a,b],唐凌翔[a],王　洪[b]
（重庆师范大学 a. 计算机与信息科学学院；b. 数字教育工程与应用研究中心，重庆 401331）

摘要：联盟区块链系统被广泛用于金融和物流等场景。现有应用于区块链系统的实用拜占庭算法（practical Byzantine fault tolerance，PBFT）存在可扩展性较低及通信成本较高等问题，阻碍了区块链系统在大规模场景中的应用。针对上述问题，提出了一种动态多组织实用拜占庭容错算法（k-means-practical Byzantine fault tolerance，k-PBFT）。通过改进k-means算法，根据节点的时延以及节点间通信距离将节点分为多个自治组织，各组织之间通过组织代表节点进行通信。当新节点加入时，根据其特点将其分配到最合理的组织。同时，引入信誉机制以辨别系统中的诚实节点与恶意节点，从而提高系统的安全性。此外，该算法还引入节点任期机制，使区块链中每个诚实节点都有机会充当组织代表节点或主节点。实验结果表明，与PBFT算法相比，k-PBFT算法通信复杂度降低了75%；当节点数为100时，相比于PBFT算法，时延降低了210 ms，吞吐量提高了100%。在高延迟环境下，相较于基于信誉分组的PBFT改进算法，当节点数为100时，时延降低了20%，吞吐量提高了17%。

关键词：区块链；拜占庭容错算法；k-means算法；信誉机制；节点任期机制
中图分类号：TP311　　**文献标志码：**A　　**文章编号：**1000-582X(2024)07-125-15

Dynamic multi-organizational PBFT algorithm based on k-means

YANG Yunong[a,b], *TANG Lingxiang*[a], *WANG Hong*[b]
(a. College of Computer and Information Science; b. Digital Education Engineering Center for Technology and Applied Research, Chongqing Normal University, Chongqing 401331, P. R. China)

Abstract: Federated blockchain systems find widely application in fields such as finance and logistics. However, the practical Byzantine fault tolerance (PBFT) algorithm, commonly applied to these systems, encounters challenges of low scalability and high communication cost, limiting its effectiveness in large-scale scenarios. To address these issues, the k-PBFT consensus algorithm, a dynamic multi-organizational practical Byzantine fault-tolerant algorithm, is proposed. The k-PBFT algorithm leverages an enhanced k-means algorithm to partition nodes into multiple autonomous organizations based on node time delays and communication distances. Each organization communicates through designated representative nodes. When a new node joins, it is assigned to the most suitable organization based on its characteristics. Additionally, the k-PBFT algorithm improves system security by introducing a reputation mechanism to distinguish honest nodes from malicious ones. It also

收稿日期：2023-07-09　　**网络出版日期：**2024-06-28
基金项目：重庆市教委科学技术研究项目（KJZD-K202300515）。
Supported by the Science and Technology Research Program of Chongqing Municipal Education Commission (KJZD-K202300515).
作者简介：杨雨浓（1978—），男，副研究员，博士，主要从事人工智能、区块链方向研究，(E-mail)yangyunong@cqnu.edu.cn。
通信作者：王洪，男，高级实验师，(E-mail)tonywanghong@126.com。

implements a node tenure mechanism, giving honest node in the blockchain the opportunity to serve as representative or master node within an organization. Experimental results show significant improvements over the PBFT algorithm: the k-PBFT algorithm reduces communication complexity by 75%, decreases latency by 210 ms, and increases throughput by 100% with 100 nodes. In high-latency environments, the k-PBFT algorithm reduces latency by 20% and boosts throughput by 17% compared to the improved PBFT algorithm based on reputation grouping with 100 nodes.

Keywords: blockchain; Byzantine fault-tolerant algorithm; k-means algorithm; reputation mechanism; node tenure mechanism

在分布式网络中,共识问题是一个经典的问题,在20世纪90年代,Shankar等[1]对共识问题中的拜占庭容错问题进行了研究。随着学者Nakamoto在2008年提出了名为Bitcoin的数字货币[2],引发了区块链技术及其核心技术——共识算法[3]的研究热潮。近年来,学者们对共识算法进行了深入研究,并将其应用于医疗保健[4]、数据安全[5]和能源[6]等领域。

共识算法是区块链技术的核心,它确保分布式网络中的节点维护相同的账本。在几种流行的共识算法中,PoW(proof of work)算法[7]和PoS(proof of stake)算法[8]应用于公有区块链,DPoS(delegated proof of stake)算法[9]应用于私有区块链,Raft(raft consensus algorithm)算法[10]和PBFT算法[11]应用于联盟区块链。在上述算法中,PoW、PoS和DPoS算法需要引入链上代币激励层,根据中国的法律法规,不支持使用代币搭建区块链服务平台。因此,国家重点鼓励发展许可加入型的区块链——联盟区块链。在联盟区块链中,PBFT算法是最受欢迎的共识算法,这是因为在分布式网络中,即使有三分之一的节点作恶,PBFT算法也能确保整个网络的账本是正确的,并且具有可以脱离代币机制运行、共识机制简单、易于维护等特点[12-15]。虽然PBFT算法可以保持分布式网络的高容错性,但随着网络节点数量的增加,网络的通信开销会变得十分庞大,节点间达成共识效率会很低,这使得大型区块链项目无法开展。因此,许多学者改进了PBFT算法以适用于大型区块链。Yu等[12]提出基于分组共识的改进PBFT算法,这种共识算法的最终一致性是由组内代表节点达成的,而组内的一致性则由每个组节点来维持。然而,这种策略忽略了节点之间的时延。Wang等[14]利用信誉机制改进PBFT算法和评估每个节点的行为,只有值得信任的少数节点才能参与网络共识。这种方法可以降低通信的复杂度,增加网络的规模。然而,这种方法有以下缺点:首先,少数高信誉节点负载会很高,可能有崩溃的风险;其次,即使是高信誉节点也有作恶的可能,如果网络共识仅仅由少数几个节点完成,则会增加节点作恶的风险;最后,这种做法违背了去中心化的初衷,降低了区块链的民主性。

基于前述背景与研究现状,笔者引入改进的k-means算法、信誉机制及任期机制,对传统的PBFT算法进行改进,并提出了k-PBFT算法;在不损害现有区块链网络的民主性和安全性的前提下,提高了PBFT算法的可扩展性,使其更适用于大规模联盟区块链网络。

1 相关工作

近年来,学者们主要研究如何提高PBFT算法的可扩展性和降低其通信复杂度。PBFT算法的通信复杂度较高,导致其可扩展性较低,通常只适用于小型网络[16]。为了提高PBFT算法的可扩展性,学者们提出了很多解决方案。Li等[17]提出采用分层技术的改进PBFT算法来提高网络的可扩展性,避免大型网络中由于节点数量过多而导致通信成本增加的问题。与分层PBFT算法不同,Yang等[18]提出一种多组共识的PBFT算法,它首先在组内进行共识,然后进行组间共识,同样提高了网络的可扩展性。然而,可扩展性的提高可能会带来安全风险。通过网络分片的方式来增强网络的可扩展性,当某一个分片的数据丢失,会使整个记录无法查询[19]。分组或多中心的PBFT算法可以降低通信成本,但以此算法作为共识基础的区块链网络的安全性取决于组织代表节点及主节点,当代表节点数量过少时,可能会出现代表节点联合作恶,影响网络安全[20]。

与本文所提算法较为相似的是文献[21]报道的算法,一种采用k-medios算法对参与区块链共识的节点进

行分类的方法,采用惰性系数P来减少k-medios算法计算的复杂度。然而,该研究没有控制每个簇内的成员数量,当簇内成员数量过低时,会影响区块链网络的安全;并且该研究没有引用信誉机制来惩罚网络中的作恶节点,当作恶节点成为代表节点时,会造成网络频繁进行视图切换而崩溃。而且,若少数几个节点一直充当代表节点,会增加该节点的压力,且会提高网络中心化程度,违背了区块链网络去中心化和民主性的初衷。在笔者所提算法中,首先改进了k-means算法,降低孤立点对聚类中心的影响,并控制簇内成员数量,防止成员数量太少对区块链网络造成影响,最后引入信誉机制来检测并惩罚网络中的作恶节点,并引入节点任期机制来提高网络的去中心化程度,保持区块链网络的民主性。

上述算法提高了PBFT算法的可扩展性,但可能忽略了PBFT算法的容错性。虽然PBFT算法只适用于小型网络,但它可以容忍网络中33%的节点为作恶节点。在实践中,高可扩展性算法适用于大型网络,但需要高容错性来保证算法能够在恶劣的环境下正常运行。Yang等[22]对PBFT算法的容错性进行了深入研究,为保证分组共识的容错能力,提出了节点决策广播模型与阈值计票模型。Aifandi等[23]提出了能够容忍拜占庭错误的基于物联网的共识算法。

笔者在多组织设计中考虑了算法的容错性,并对k-means聚类过程进行了改进,以控制簇内成员的数量,以期在保留算法的容错性的同时,提高算法的可扩展性。目前的分组共识策略牺牲了区块链网络的去中心化特性以减少通信开销。在整个网络中由少数几个组代表节点执行共识的情况下,区块链网络的去中心化程度降低,同时降低网络资源的利用率。

考虑到上述问题,笔者设计了一种基于改进k-means的PBFT算法,即k-PBFT。该算法主要从以下几个方面进行了优化。

1)利用改进的k-means算法,根据节点间的时延和距离将节点分为多个自治组织。与其他基于分组的PBFT算法不同,k-PBFT算法中每个自治组织节点间的时延更低。

2)提出信誉值机制,根据节点的历史行为动态调整节点的信誉值,将网络中的节点分为诚实节点与作恶节点。每组中信誉值最高的节点会当选为组代表节点,而组节点代表中信誉值最高的节点当选为主节点。以此提高网络的安全性,并对作恶节点做出惩罚。

3)为了改善网络中仅有几个组代表节点与主节点参与共识的问题,提出节点任期机制,设置节点的任期。当节点充当组代表节点或主节点任期到期后,会重新选举组代表节点和主节点。可以降低少数几个节点的负载,并提高网络的民主性。

2　传统PBFT算法

2.1　PBFT算法共识流程

PBFT算法被用于解决如何在有节点作恶的环境中实现数据一致性的问题。它可以容忍区块链网络中存在三分之一的作恶节点。PBFT算法的基本通信过程如图1所示。

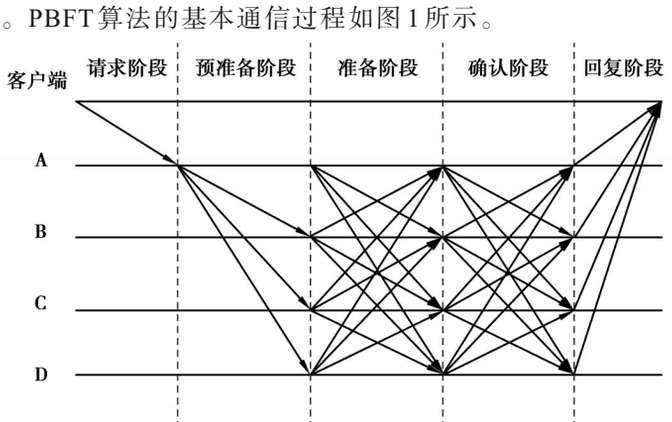

图 1　传统PBFT算法共识流程

Fig. 1　The traditional PBFT consensus process

PBFT算法的共识过程分为5阶段。

1)请求阶段,客户端向主节点发送请求消息。

2)预准备阶段,主节点收到了客户端发送的请求消息,并将预准备消息广播给其他节点。其他节点收到主节点发送的预准备消息后,验证该消息是否为主节点发送。

3)准备阶段,各个节点验证了主节点发送的预准备消息,向其他节点广播准备消息并收集其他节点发送过来的准备消息。当节点收到$2f+1$(其中f表示网络中作恶节点的数量)个正确的准备消息后进入下一阶段。

4)确认阶段,各个节点向其他节点发送确认消息,并收集其他节点发送过来的准备消息。当节点收到$2f+1$个正确的确认消息后,进入下一阶段。

5)回复阶段,各个节点向客户端发送回复消息。当客户端收到$f+1$个回复消息后,表示发送的请求消息已经通过了共识。

2.2 PBFT算法分析

PBFT算法通过准备阶段和提交阶段的广播消息验证,能够容忍网络中最多三分之一的节点为拜占庭节点,但也使得PBFT算法的通信开销变得十分庞大。因此,在联盟区块链中,PBFT算法是较好的选择,但该算法仍存在如下不足。

1)可扩展性较低。在PBFT算法的准备阶段和确认阶段需要向全网广播消息,当网络中的节点不断增加时,系统的通信开销快速增加,网络的通信时延不断上升,因此,现有的PBFT算法无法应用于大型网络。

2)主节点选取随意。PBFT算法的主节点通过随机选择产生,使得网络中的恶意节点可能成为主节点,导致频繁的视图更换,从而降低共识的效率,造成不必要的网络开销。并且PBFT算法缺少对恶意节点的惩罚措施,视图切换后,恶意节点仍可以参与接下来的共识,会持续地对网络安全造成影响。

3 k-PBFT算法的设计与实现

3.1 k-PBFT算法简介

k-PBFT算法以PBFT算法为基础,针对PBFT算法及其现有改进的算法存在的问题,引入改进的k-means算法以及信誉值机制和节点任期机制对算法进行优化,有效地解决了PBFT算法通信开销大,扩展性低,以及现有的基于信誉值分组的PBFT算法无法在节点时延较高的环境下保持良好共识效率的问题。

k-PBFT算法的基本流程为:首先,使用改进的k-means算法将节点聚类为多个共识组织。然后,节点的积分值被初始化。根据节点的行为,节点被划分为候选节点和正常节点。对于每个组织,具有最高信誉值的节点被选为该组织的代表节点。最后,从组织代表节点中选出主节点。当节点的回复信息数量达到预定水平时,节点会更新账本并完成共识。当确认信息的数量低于一定水平时,就会使用惩罚机制来处理不正确的节点。当主节点或超过三分之一的小组代表节点做恶时或当主节点或小组代表节点的任期结束时,视图切换机制被触发。k-PBFT算法的基本流程如图2所示。

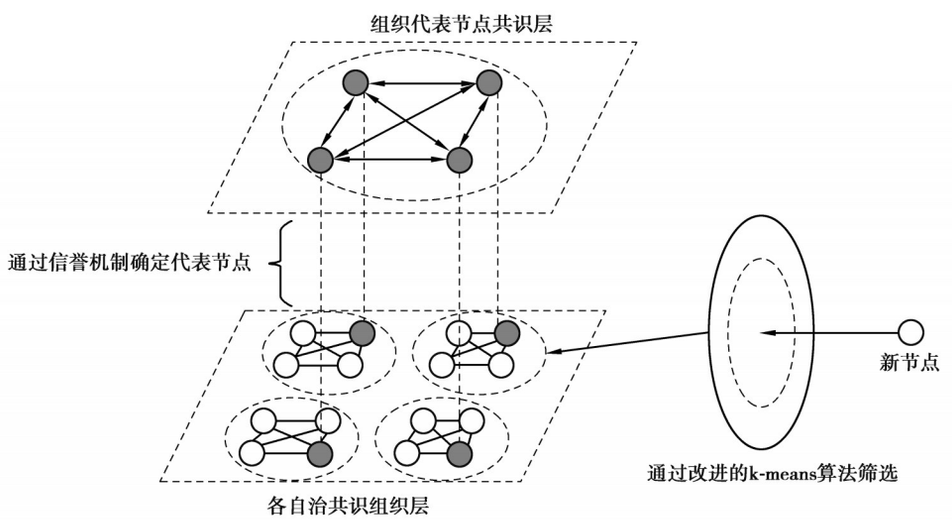

图2 k-PBFT算法基本流程

Fig.2 The basic process of k-PBFT algorithm

3.2　改进的k-means算法

3.2.1　k-means算法简介

k-means算法是一种聚类算法,其中means代表均值,k代表聚类的类别数。k-means算法通过预先设定的k值及每个类别的初始质心对相似的数据点进行划分,并通过划分后的均值迭代优化获得最优的聚类结果,k-means算法以欧氏距离作为相似度。

传统的k-means算法会因少量孤立点的加入而对聚类均值中心造成很大的影响,并且无法控制簇内成员数量,过少的成员数会影响区块链网络安全,为了使k-means算法更适用于区块链模型,本文对其进行了两方面的改进。

3.2.2　改进的k-means算法

1)减少孤立点对聚类均值中心的影响

为了减少孤立点对聚类均值中心的影响,引入初始的聚类中心之间的相互距离最大化改进方法。具体步骤如下。

步骤1:随机选取一个样本作为第一个平均值抽象点,即聚类中心。

步骤2:计算样本集X中每个样本x_i与当前已有的聚类中心的最短欧式距离$D(x_i)$,接着计算每个样本被选为下一个聚类中心的概率$P(x_i)=\frac{D(x)^2}{\sum_{x_i\in X}D(x)^2}$,$P(x_i)$越大,则该样本被选为聚类中心的概率越大。最后产生一个0~1的随机数r,计算每个样本的$P(x_i)$与r的差值,差值最小者为下一个聚类中心。

步骤3:重复步骤2,直到选出k个聚类中心。

使用随机数,可以减少孤立点对聚类中心的影响,当孤立点加入时,取与已有聚类中心距离最远的样本成为下一个聚类中心的方法会导致孤立点成为聚类中心,使用随机数会在几个样本中随机选择一个成为下一个聚类中心,减少孤立点的加入对簇内的聚类均值中心的影响。

2)控制簇内成员的数量

将k-means算法应用于区块链环境,希望利用k-means算法将节点分为不同的共识组织,并控制组织成员的数量,以提高每个组织的容错率。为了便于理解,给出以下定义。

定义1:整个网络中的节点数为n;每个组织中的节点数为m,$m=3f_1+1$,$f_1=1,2,3\cdots$;整个网络中的组织数量为$R=[(n-1)/m]$。

此外,根据$n\geqslant 3f+1$的结论(其中f代表可以容忍的拜占庭节点的最大数量),可知每个组织最多可以容忍$E=[(m-1)/3]$个拜占庭节点,整个网络最多可以容忍$w=[(R-1)/3]$个组织共识异常(其中R、E、w为整数,$R\geqslant 4$)。

因此,每个组织的节点数不能少于4,组织数不能少于4,这样才能保证区块链的安全性。基于以上问题,提出了改进k-means算法,具体流程见算法1。

算法1 改进的k-means算法

输入 设置数据集$D=\{x_1,x_2,\cdots,x_n\}$,分类数量为k,(k=int($\sqrt{n}$), int( )表示取整操作,分类集$C=\{C_1,C_2,\cdots,C_k\}$,最大迭代数为v。

输出 $C=\{C_1,C_2,\cdots,C_k\}$。

1. 随机选择一个样本成为第一个聚类中心;
2. 计算数据集中的样本与聚类中心的距离$D(x_i)$;
3. 计算每个样本被选为下一个聚类中心的概率$P(x_i)=\frac{D(x)^2}{\sum_{x_i\in X}D(x)^2}$,生成随机数$r$,选出聚类中心;
4. 重复3,直到选出k个聚类中心;
5. 计算x_i与聚类中心的欧氏距离;
6. 重新计算每个分类的聚类中心$\mu_k'=\frac{1}{|C_k|}\sum_{x\in|C_k|}x$;
7. 如果某个分类中样本数小于4;
8. 从分类$C_{\max}$中提取节点(样本数量最多的分类),并将其放入分类C_i中。

改进的k-means算法可以确保相互之间延迟较低的节点被分到同一个类中并且每个分类中的样本数不小于4,之后引入的信誉值机制以及节点任期机制可以整体上提高算法的容错性。

3.3 信誉值机制

为了增强算法的容错性以及使更稳定的诚实节点充当组织代表节点和主节点,设置了信誉值机制对网络中的节点进行分类和评估。基本流程如图3所示。

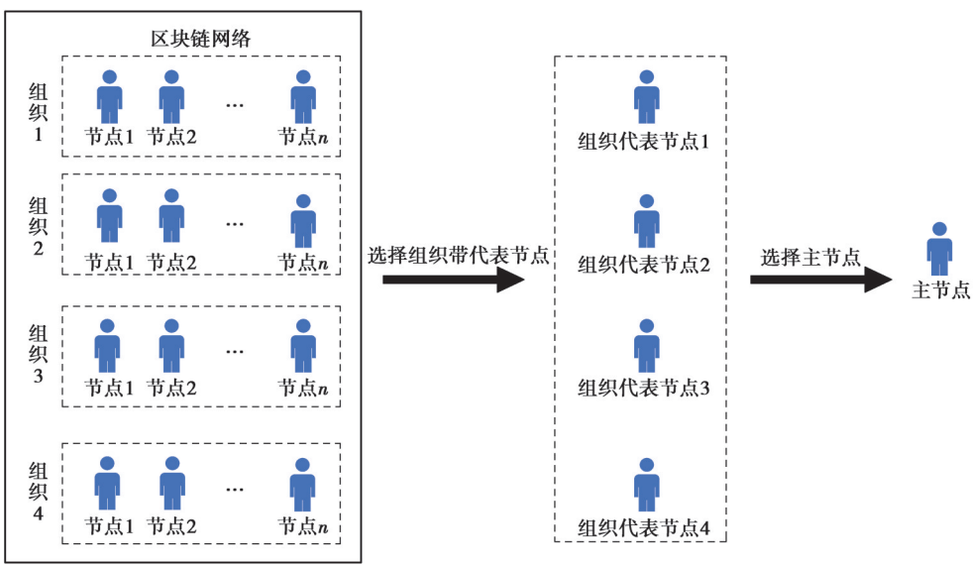

图3 信誉值机制流程

Fig. 3 The process of the credibility value mechanism

每个节点初始的信誉值为100。每个节点有4个状态:普通节点、候选节点、组织代表节点和主节点。信誉值低于100的节点为普通节点,大于等于100的节点为候选节点(未当选代表节点或主节点)。在每个视图切换时或组织代表节点、主节点任期结束时,候选节点可以参加选举,成为组织代表节点。主节点从组织代表节点中选出。节点的上述4种状态按照以下规则切换。

信誉值奖励规则和惩罚规则:在每次共识结束时,客户端首次收到f+1个回复信息,这些发送节点的信誉值增加10。如果一个节点发送错误消息或超时,积分值减少20。如果组织代表节点或主节点有错误或超时,积分值被重置为100,执行视图切换操作。

节点状态变化规则:在初始状态下,从每个组织中随机选择一个节点作为组织代表节点,然后从组织代表节点中随机选择一个节点作为主节点。设主节点和组织代表节点的任期为S。节点任期结束或主节点发生错误时,执行视图切换。当视图切换被触发时,每个组织中具有最高信誉值的节点被选为组织代表节点,然后从组织代表节点中选择具有最高信誉值的节点作为主节点。

3.4 k-PBFT算法共识流程

图4所示由一个客户端和16个节点构成的区块链网络。k-PBFT算法的共识过程有7个步骤:请求阶段、预准备阶段、组内准备阶段、组外准备阶段、确认阶段、确认-回复阶段和回复阶段。在各准备阶段,每4个共识节点构成1个子网络。为方便描述,将节点[1.1]、节点[2.1]、节点[3.1]、节点[4.1]作为各组织的代表节点,将节点[1.1]作为主节点。客户端向主节点发送请求消息后,整个网络将被触发,各节点在完成共识后将更新账本。

请求阶段:客户端向主节点发送request消息,其他节点收到消息后检查发送者是否主节点。如果是主节点,则进入预准备阶段,否则将消息转发给主节点。

预准备阶段:当主节点收到request消息时,它为该请求分配一个序列号n,广播一个pre-prepare(预准备)消息,并将该消息加入日志。该消息包含序列号n、节点号i、请求消息m和视图号v。

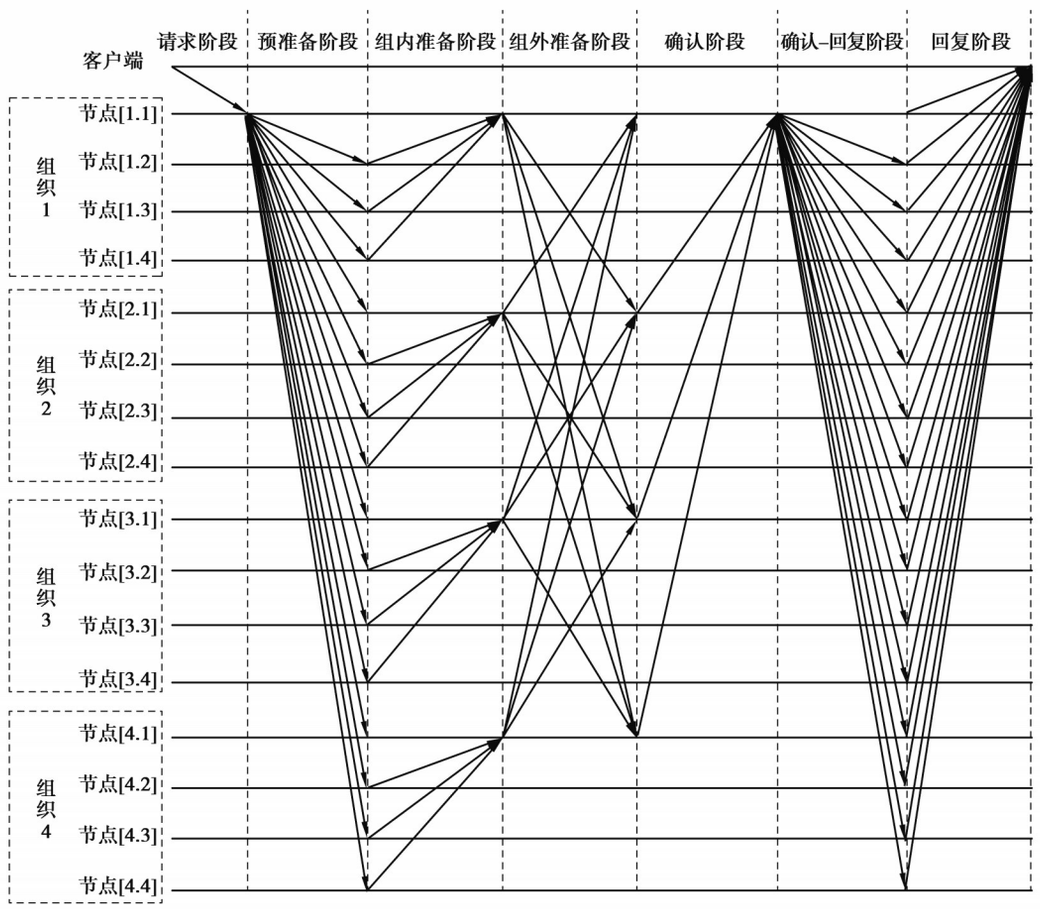

图 4　k-PBFT 算法共识流程

Fig. 4　The consensus process of k-PBFT algorithm

组内准备阶段:当网络中的节点收到 pre-prepare 消息后,检查该消息的发送者是否为主节点。如果验证通过,该消息被添加到日志中。然后,每个节点向自己的组织代表节点发送一条 in-prepare(组内准备)消息,并将该消息加入日志。该消息包含 request 消息的摘要 d、节点号 i 和序列号为 n 的视图号 v。

组外准备阶段:组织代表节点收到 in-prepare 消息后,检查消息的发送者是否同一个组织中的节点,以及摘要 d 是否正确。如果验证通过,该消息被添加到其日志中。然后,检查日志中是否已存在 $2/3\times m$ 个具有相同的 v、n、d 的 in-prepare。如果检查通过,则向其他组织代表节点发送一个 out-prepare(组外准备)消息,并将该消息加入其日志。该消息包括摘要 d、节点号 i 和视图号 v,序列号 n。

确认阶段:组织代表节点收到 out-prepare 消息后,检查该消息的发送者是否是组织代表节点,摘要 d 是否正确。如果验证通过,该消息被添加到其日志中。之后,检查在日志中是否存在 v、n 和 d 相同的 out-prepare 消息,直至该消息的数量达到 $2/3 \times R$ 个。如果检查通过,则向主节点发送一个 commit(确认)消息,并将该消息加入其日志。该消息包含摘要 d、节点号 i、视图号 v 和序列号 n。

确认-回复阶段:主节点收到 commit 消息后,检查消息的发送者是否是组织代表节点,摘要 d 是否正确。如果验证通过,该消息被添加到其日志中。然后检查日志中是否达到 $2/3 \times R$ 个 v、n、d 相同的 commit 消息。如果检查通过,则广播 commit-reply(确认-回复)消息,并将该消息加入其日志。该消息包括摘要 d、节点号 i、视图号 v 和序列号 n。

回复阶段:当网络中的节点收到 commit-reply 消息后,将检查该消息的发送者是否为主节点,如果验证通过,节点会向客户端发送一条 reply 消息,并将该消息添加到其日志中。该消息包括摘要 d、节点编号 i、返回结果 r_1 和视图编号 v。

客户端收到 reply(回复)消息时,将该消息添加到日志中,并检查日志,查看在 t 时间发送的 request 消息

是否已经收到了f+1个具有相同返回结果r_1的reply消息。如果检查通过,则执行该客户的请求,并将数据写到区块链上。

3.5 任期机制

任期机制通过允许系统在主节点失效时改变到下一个正确的视图状态来提供安全性。它还通过设置任期来确保区块链网络的民主性。视图变化由超时触发,可以防止节点无限期地等待执行请求。视图切换由任期触发。系统通过使用信誉值表确定新的主节点和组织代表节点。

每个节点在收到请求消息$request_i$时启动计时器$timer_i$,在$request_i$结束时关闭$timer_i$。当一个节点发生错误或任期结束时,$timer_i$超时。如果一个节点在当前视图v下超时,分2种情况分析。

1)若它是主节点或组织代表节点,节点的信誉值重置为100,并向其他节点发送视图切换view-change消息。该信息包括新的视图号$v+1$、$2f+1$个检查点消息checkpoint、序列号n、低水位h和$2f$个预准备消息pre-prepare。

2)若它是一个候选节点或普通节点,该节点的信誉值被减去20,并向其他节点发送一个view-change消息。

当一个节点收到一个view-change消息,将确定其视图号v是否小于当前视图,以及序列号n是否小于低水位h。如果验证通过,它将把该消息添加到日志中,并检查缓存中具有相同视图v和序列号n的view-change消息是否已经达到$2f+1$个。如果没有达到,那么系统可以在一些节点超时后继续运行。如果达到,系统将根据信誉值表重新选择组织代表节点和主节点。

每个节点在执行下一个操作之前都会确认其身份:

1)如果它是主节点,它将把视图v更新为$v+1$,把view-change消息中的低水位h更新为n,并向其他节点发送新视图new-view消息,其中包含新视图号$v+1$,新视图$v+1$的checkpoint消息集合,以及低水位h以上序列号n的pre-prepare消息集合;

2)如果它是一个候选节点或一个普通节点,它将把视图v更新为$v+1$,并把view-change消息中的低水位h更新为n;

3)当节点收到new-view消息时,它们会在新视图v+1中重新处理view-change消息中的pre-prepare消息集合。

基本流程如伪代码算法2所示。

算法2 任期机制

```
1.if TimeOutMsg,n<=lastRepNum //判断超时消息的序列号是否大于最后一条回复消息
2.    return
3. end if
4. isTimeOut = true
5.    reputation -= 20 //惩罚机制
6.    Map < > s = checkPoints.get(h) //收集日志中的检查点消息
7.    Message vm = new ViewChangeMsg( )
8.    addMessageToLog(vm)
9.    sendMsgToOthers(vm)
10.selectNewRNode( ) //选择新的组织代表节点以及主节点
```

3.6 节点加入以及退出

有新节点加入时,算法重新计算该节点与组织代表节点之间的特征关系,并分配最合适的共识组织。当一个节点需要退出时,如果成员数量太少,算法自动调整组织节点,避免因节点退出影响网络安全。具体过程如算法3所示。

算法3 节点加入及退出机制

```
1.if node_Join(node)
2.    k-means(org_Represent_Node.features , node.features) //分析该节点与组织节点间的特征关系
3.    distribution_Organization(node) //将节点分配到某组织中
4.    if node_Exit(node)
5.      if(member_check)//检查该组织成员数是否过少
6.        k-means()//重新建立组织
```

4　实验分析

4.1　实验设计

为了从多个方面测试算法,设计了几个实验来证明k-PBFT算法可以应用于大型区块链网络。

4.1.1　实验环境

为了方便测试k-PBFT算法在不同参数下的性能,省略了区块链网络中的其他步骤,只保留了共识过程。共识过程是:客户端向主节点发送request消息,然后网络中的节点执行k-PBFT算法的共识过程,最后,客户端收到$f+1$个reply消息。这个过程是通过队列来模拟的,编程语言为Java,运行环境是Java Development Kit 1.8。每个节点根据时间顺序来处理收到的消息。为了便于测试算法的延迟,队列中的时间由整数变量模拟,并从小到大排序。因此,节点根据时间顺序来接收消息。

4.1.2　容错测试

容错测试是实验设计中最重要的部分。为了证明算法可以应用于大型网络,设计了2个实验来测试该算法的容错性。实验中的拜占庭节点试图通过延迟非故障节点或它们之间的通信使网络崩溃。

4.1.3　通信成本测试

通过计算网络中传播的消息的总大小来确定算法的通信成本。实验设计中,消息的大小是一个整数常数,通信成本是一个变量。每当有消息被添加到队列中,通信成本就会增加。

4.1.4　时间延迟测试

实验中,时间延迟是指从客户端发送一个request消息到收到$f+1$个reply消息所需的时间。当共识从一个阶段进入下一个阶段时,时间延迟会增加,在等待足够数量的准备消息或确认消息时,时间延迟也会增加。

4.1.5　吞吐量测试

实验中,吞吐量是网络在每单位时间内,即1 s内所能处理的请求消息的数量。它是在时间戳timestamps内处理的request消息的数量。timestamps是客户端收到最后一个reply消息的时间戳。

4.2　容错性测试

分析k-PBFT算法的容错极限,证明它可以在不影响容错的情况下优化PBFT算法。在k-PBFT共识方法中,只有经过一轮组织内共识和两轮组织间共识,才能确定最终共识结果。组外准备阶段有R个共识组织。在这个阶段,可以容忍w组节点作恶。然而,其余的$(R-w)$组织节点也可能作恶。接下来将详细讨论该算法的容错限制。

只要网络中超过2/3的节点为诚实节点,PBFT算法便可保证正确的共识。可以容忍的拜占庭节点的最大数量是1/3。然而,k-PBFT算法的容错上限是一个区间而不是一个固定值。如果所有的组织代表节点都是拜占庭节点,那么在组外准备阶段就不可能达成共识。在这种情况下,共识组织的数量R等于最小容错数。此外,当$(R-w)$个组织达到最大容错数$E=(m-1)/3$时,仍可以保证$(R-w)$个组织节点之间的共识是正确的,并且,此时拜占庭组织不能超过w个。即$(R-w)$个组织具有正确的一致性并达到最大容错数$E=(m-1)/3$,且w个组织中的节点都为拜占庭节点。k-PBFT算法的容错区间为$[R,M]$,M代表最大容错节点数,由式(1)计算得到:

$$M=\left[\frac{R-1}{3}\right]m+\left[R-\frac{R-1}{3}\right]E。\tag{1}$$

根据公式(1),当网络中的拜占庭节点数超过w(可容忍的最大拜占庭组织数)时,存在网络中1/3以上的组织节点为拜占庭节点的情形,当网络中的拜占庭节点数超过网络中的组织数R时,存在所有组织代表节点为拜占庭节点的情形。

因此,对所有组织代表节点为拜占庭节点的情形进行了概率分析。假设每个节点都是相互独立的,可以得到如下公式。

$$\begin{cases}F_1=\dfrac{C_i^w}{C_n^w},\\ F_2=\dfrac{C_i^R}{C_n^R}。\end{cases}\tag{2}$$

式中:i表示网络中拜占庭节点的数量;F_1表示当i的数量超过w时的失败概率;F_2表示当i的数量超过R时的失败概率;C_n^w表示当拜占庭节点数超过w时,从n个共识节点中随机选择w个节点;C_i^w表示$R/3$个组织的代表节点均来自i个拜占庭节点;C_n^R表示当拜占庭节点数超过组织数时,从n个共识节点中随机选择R节点;C_i^R表示R个组织的所有代表节点来自i个拜占庭节点。

最后,当网络中存在i个拜占庭节点时,得到所有组织代表节点都是拜占庭节点的概率。根据公式(2),取n=26,m=5,R,i取0~25,w=2(w取整数),实验结果如图5所示。

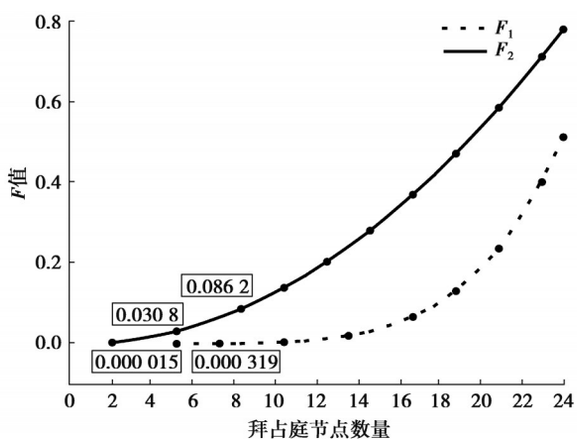

图5 组织代表节点联合作恶概率

Fig. 5 The probability of joint criminality by representative node

根据式(1),该环境下容错区间为[4,8]。从图5可以看出,当拜占庭节点数在上述容错区间[R,M]时,超过1/3的组织节点为拜占庭节点的概率非常低(F_1),在容错范围内所有组织节点为拜占庭节点的概率几乎为零(F_2)。

然而,当拜占庭节点的数量不断增加时,即使拜占庭节点的数量不超过M,共识也有可能失败。例如,在组外准备阶段,整个网络最多可以容忍w个组织的节点作恶,每个组织可以容忍E个节点作恶。但只要有($R-w$)个组织中的拜占庭节点大于或等于$E+1$,就会导致共识失败。因此,联合作恶的拜占庭节点的最小数量为($R-w$)($E+1$)。因此,可得到共同作恶的节点数与节点总数的比值:

$$J=\frac{(R-w)(E+1)}{n}。\tag{3}$$

实验分析了m = 10,20,25时J的变化。

当网络中的节点数为n,每个组织中的节点数为m时,J值反映($R-w$)组织中的节点共同作恶而导致共识失败的概率。结果如图6所示。

随着网络中节点数量的不断增加,无论m如何变化,J始终保持在1/3左右。因此,k-PBFT算法的容错能力达到了拜占庭容错算法的水平。

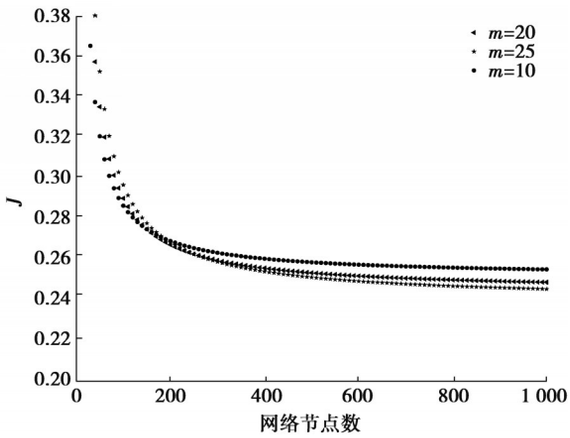

图6　诚实组织联合作恶概率

Fig. 6　The probability of joint criminality between honest organizations

4.3　通信成本测试

为了验证改进后的共识算法模型在通信次数上优于传统的PBFT共识算法,设计了通信复杂度对比分析和通信成本对比实验。

4.3.1　通信复杂度分析

1)计算PBFT算法的通信次数

假设现在有n个($n>3$)节点参与PBFT共识。在预准备阶段,共识网络中的通信次数为($n-1$)。在准备阶段,共识网络中的通信次数为$(n-1)\cdot(n-1)$。在确认阶段,共识网络中的通信次数为$n\cdot(n-1)$,则完成PBFT共识过程所需的通信总次数为

$$Z_1=2n\cdot(n-1)。\tag{4}$$

2)计算k-PBFT算法的通信次数

同样,假设现在有n个($n>3$)节点参与k-PBFT共识,设组织数为R。默认情况下,每个组织的共识节点数为n/R,组织代表节点数为R。

由图4可知,组织内的通信次数为$2n-R$。在组外准备阶段,通信次数为$R\cdot(R-1)$。在确认、确认-回复和回复阶段,通信次数共为$2(n-1)+R$。根据上述推导过程,通信总次数为

$$Z_2=2\cdot(2n-1)+R\cdot(R-1)。\tag{5}$$

$R=(n-1)/m$,则可得出k-PBFT算法的通信复杂度为$O\left([(n-1)/m]^2\right)$,小于PBFT算法的通信复杂度。表1为k-PBFT算法与常见的共识算法对比。

表1　k-PBFT算法与常见的共识算法对比

Table 1　The k-PBFT algorithm compares with the common consensus algorithm

算法	是否拜占庭容错	容错性	通信复杂度	可扩展性
PBFT	Yes	1/3	$O(n^2)$	Low
RAFT	No	1/2	$O(n)$	High
PoW	Yes	1/2	$O(n)$	High
PoS	Yes	1/2	$O(n\log_2 n)$	High
k-PBFT	Yes	1/3	$O\left([(n-1)/m]^2\right)$	High

k-PBFT算法具有高度的可扩展性和容错性,同时保持了去中心化的特性。虽然通信复杂度不是最优的,但k-PBFT算法在各个方面较为均衡。在区块链系统中,k-PBFT算法可以容忍拜占庭错误,同时可以扩展到大型区块链网络。

4.3.2 通信成本实验

为了进一步比较k-PBFT算法、基于分组的group-practical Byzantine algorithm(GPBFT)算法和PBFT算法的通信消耗，建立了一个实验来模拟共识过程。

在PBFT算法中，请求消息的大小为100 bytes，预准备消息的大小为104 bytes，准备消息的大小为36 bytes，确认消息的大小为36 bytes，回复消息的大小为16 bytes。

在k-PBFT算法中，请求消息的大小为100 bytes，预准备消息的大小为104 bytes，组内准备消息和组外准备消息的大小为36 bytes，确认消息和确认-回复消息的大小为36 bytes，回复消息的大小为16 bytes。GPBFT算法的配置同k-PBFT一致。

各节点间的基本时延为15 ms，节点间的时延扰动范围为3 ms，节点与客户端间的基本时延为30 ms，节点与客户端间的网络时延扰动范围为5 ms。

客户端发送50个请求消息。随机初始化单个节点之间以及节点与客户端之间的时延。通过消息队列模拟共识网络，比较k-PBFT算法和PBFT算法和GPBFT算法的流量。传输大小等于客户端发送50个请求消息后在网络中传播的消息大小之和。

客户端发送一个请求消息直至收到$f+1$个回复消息，在此期间网络中传播的消息总大小为S。那么本次实验流量traffic如下：

$$\text{traffic}=\sum_{n=0}^{50}S_n, \tag{6}$$

式中，$n\in\{1,2,...,50\}$。

根据式(6)进行实验，统计流量大小，结果如图7所示。

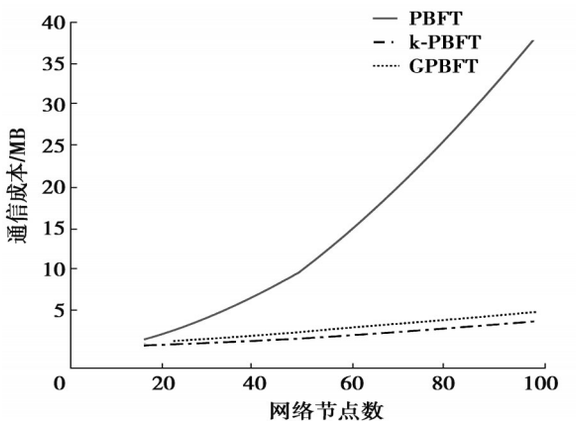

图7 通信成本实验

Fig. 7 Communication Cost Experiment

随着节点数量增加，PBFT算法、GPBFT算法和k-PBFT算法的通信成本呈上升趋势。在通信成本方面，k-PBFT算法比PBFT算法降低了很多。此外，k-PBFT算法采用基于k-means多组织模型，每个组织内部的通信成本较低，因此通信成本比简单分组低。当网络延迟较高时，k-PBFT算法的优势更加明显。

4.4 时延测试

时间延迟是指客户端从发送请求消息到接收到$f+1$回复消息所花费的时间，是评估共识算法的一个重要指标。当共识算法用于构建大型网络时，过高的时间延迟将对通信产生重大影响。本文中，客户端发送了50条请求消息。比较PBFT算法、GPBFT算法和k-PBFT算法随节点数增加而产生的时间延迟变化。参数与通信成本实验相同。结果如图8所示。

图8表明，随着节点数量的增加，PBFT算法、GPBFT算法和k-PBFT算法的时延都呈上升趋势。尽管节点数量不断增加，k-PBFT算法仍能保持最小的时延。此外，k-PBFT算法采用基于k-means多组织模型，使得各组织的时延低于GPBFT算法的分组方式。如果时延较大，自适应多组织策略的优势将更加明显。

因此,为了分析在高时延情况下简单分组策略与基于k-means多组织策略的区别,进行了如下实验。

选取节点间的基本时延为50 ms,节点间时延扰动范围为20 ms;节点与客户端间基本时延为60 ms,节点与客户端间时延扰动范围为30 ms。结果如图9所示。

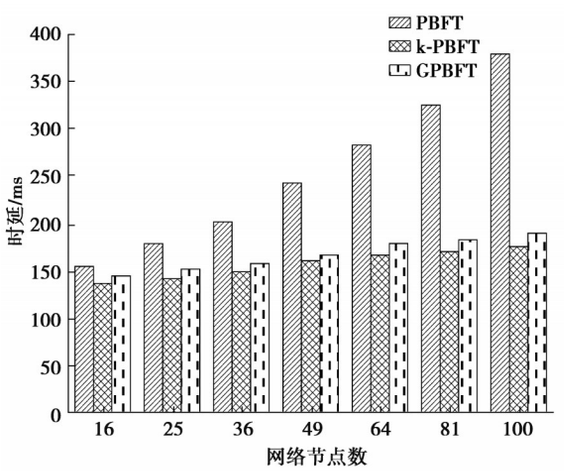

图 8　时延对比实验

Fig. 8　Delay comparison experiments

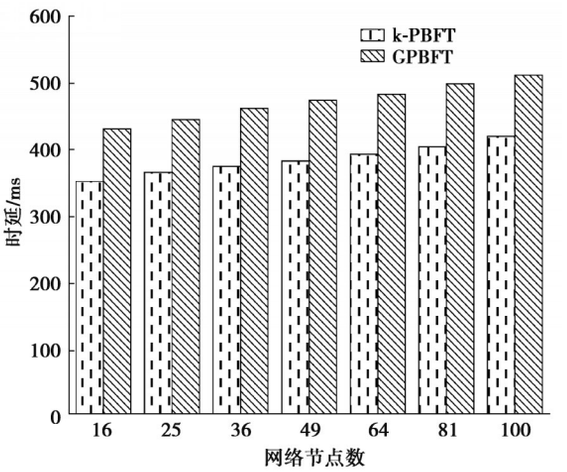

图 9　高时延对比实验

Fig. 9　Delay comparison experiments(high latency)

在高时延情况下,k-PBFT算法将两者之间通信时延较小的节点分组在同一个组织中。因此,与GPBFT算法的简单分组策略相比,k-PBFT算法能够保持较低的时延。

4.5　吞吐量测试

吞吐量(TPS)是指系统在单位时间内能够处理的事件数量,是共识算法的关键性能指标。以客户端发送的50条请求消息为基准,比较了PBFT、GPBFT和k-PBFT吞吐量随节点数增加而发生的变化。低时延情况下实验参数与通信成本实验相同,高时延情况下实验参数与高时延实验相同。结果如图10~11所示。

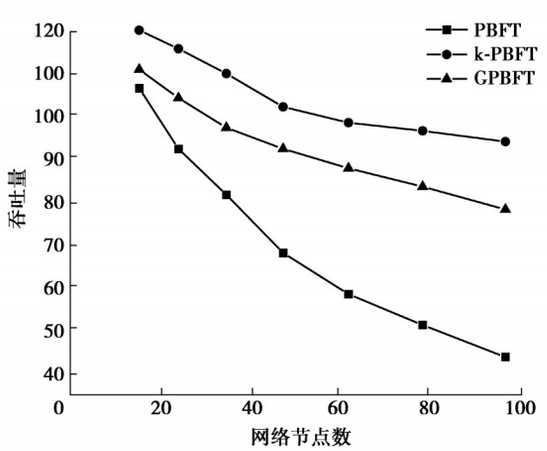

图 10　吞吐量对比图(低时延)

Fig. 10　Traffic comparison graph (low latency)

根据实验结果,随着节点数量不断增加,PBFT算法、GPBFT算法和k-PBFT算法的吞吐量呈下降趋势。由于k-PBFT算法采用k-means算法进行组织分配,各组织能够以较低的时延达成共识。因此,在图 10和图11中,k-PBFT算法的吞吐量下降趋势并不明显。

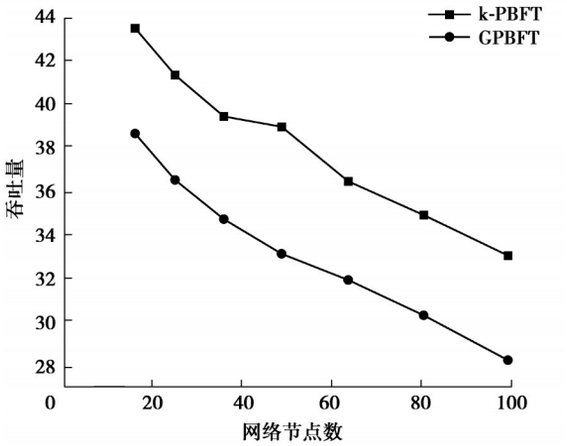

图11 吞吐量对比图(高时延)

Fig. 11 Traffic comparison graph (high latency)

5 结束语

为了解决拜占庭容错算法在实际应用中存在的通信成本高、可扩展性差、时延大等问题,提出了一种自适应的多组织PBFT算法。首先,对k-means算法进行了优化,利用k-measn算法将网络聚类为多个组织,降低了PBFT算法的通信成本和时间延迟。其次,引入信誉值机制,将节点分为诚实节点与作恶节点,提高了算法的容错性。最后,设计了节点任期机制,确保区块链网络的民主性,减轻了每个节点的压力。通过设计实验,从多个角度验证了算法的容错等性能。在k-mean算法中,根据节点之间的时延进行分类。在后续的优化中,可以加入多种节点特征,如地理位置、性能等。此外,在组织数控制中,采用了类似于顺序查找的方法。这将降低大型区块链网络的初始化速度,后期研究中,将尝试使用更高效的查找方法。

最后,提出了几种可以提高算法性能的优化方案。例如,可以在每个组中使用阈值签名技术,以减少节点之间的通信。此外,由于组织代表节点具有任期,因此无需担心节点的压力。另外,在组外准备阶段,可以允许主节点汇总和转发消息以完成共识,从而降低一个阶段的通信成本。最后,在确认-回复阶段,应建立检测主节点的检查机制。

本文所使用的测试数据是通过队列模拟通信网络获得的,但尚未在大型网络中进行测试。未来,需要在金融、数据安全和能源等更多领域展开测试以进一步验证算法的有效性和适用性。

参考文献

[1] Shankar N. Mechanical verification of a generalized protocol for Byzantine fault tolerant clock synchronization[M]//Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992: 217-236.

[2] Nakamoto S. Bitcoin: a peer-to-peer electronic cash system[J]. Bitcoin, 2008, 4(2): 15.

[3] Du M X, Ma X F, Zhang Z, et al. A review on consensus algorithm of blockchain[C]//2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, Canada. IEEE, 2017: 2567-2572.

[4] 薛腾飞, 傅群超, 王枞, 等. 基于区块链的医疗数据共享模型研究[J]. 自动化学报, 2017, 43(9): 1555-1562.
Xue T F, Fu Q C, Wang C, et al. A medical data sharing model via blockchain[J]. Acta Automatica Sinica, 2017, 43(9): 1555-1562.(in Chinese)

[5] Daneshgar F, Ameri Sianaki O, Guruwacharya P. Blockchain: a research framework for data security and privacy[M]// Advances in intelligent systems and computing. Cham: Springer International Publishing, 2019: 966-974.

[6] Wang Q, Su M. Integrating blockchain technology into the energy sector – from theory of blockchain to research and application of energy blockchain[J]. Computer Science Review, 2020, 37: 100275.

[7] Wang E K, Sun R P, Chen C M, et al. Proof of X-repute blockchain consensus protocol for IoT systems[J]. Computers & Security, 2020, 95: 101871.

[8] 吴梦宇, 朱国胜, 吴善超. 基于工作量证明和权益证明改进的区块链共识机制[J]. 计算机应用, 2020, 40(8): 2274-2278.

Wu M Y, Zhu G S, Wu S C. Improved consensus mechanism of blockchain based on proof-of-work and proof-of-stake[J]. Journal of Computer Applications, 2020, 40(8): 2274-2278.(in Chinese)

[9] Liu J, Xie M Y, Chen S Y, et al. An improved DPoS consensus mechanism in blockchain based on PLTS for the smart autonomous multi-robot system[J]. Information Sciences: an International Journal, 2021, 575(C): 528-541.

[10] Castro M, Liskov B. Practical byzantine fault tolerance[C]//1999 3th USENIX Conference on Operating Systems Design and Implementation (OsDI), Washington, D.C., United States. OsDI ,1999: 99(1999): 173-186.

[11] Huang D Y, Ma X L, Zhang S L. Performance analysis of the raft consensus algorithm for private blockchains[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2020, 50(1): 172-181.

[12] Yu G, Wu B, Niu X X. Improved blockchain consensus mechanism based on PBFT algorithm[C]//2020 2nd International Conference on Advances in Computer Technology, Information Science and Communications (CTISC), Suzhou, China. IEEE, 2020: 14-21.

[13] Li C L, Zhang J, Yang X M. Scalable blockchain storage mechanism based on two-layer structure and improveddistributed consensus[J]. The Journal of Supercomputing, 2022, 78(4): 4850-4881.

[14] Wang Z Y, Wang J, Liu Y N, et al. Improvement of PBFT consensus mechanism based on credibility[C]//2021 18th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China. IEEE, 2021: 93-96.

[15] Zhang L, Li Q W. Research on consensus efficiency based on practical Byzantine fault tolerance[C]//2018 10th International Conference on Modelling, Identification and Control (ICMIC), Guiyang, China. IEEE, 2018: 1-6.

[16] Jiang Y J, Lian Z. High performance and scalable Byzantine fault tolerance[C]//2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China. IEEE, 2019: 1195-1202.

[17] Li W Y, Feng C L, Zhang L, et al. A scalable multi-layer PBFT consensus for blockchain[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(5): 1146-1160.

[18] Yang J, Jia Z H, Su R G, et al. Improved fault-tolerant consensus based on the PBFT algorithm[J]. IEEE Access, 2022, 10: 30274-30283.

[19] Sadalage P J, Fowler M. NoSQL distilled: a brief guide to the emerging world of polyglot persistence[M]. Upper Saddle River, NJ: Addison-Wesley, 2013.

[20] Qushtom H, Mišić J, Chang X L, et al. A scalable two-tier PBFT consensus for blockchain-based IoT data recording[C]//ICC 2021 - IEEE International Conference on Communications. Montreal, QC, Canada. IEEE, 2021: 1-6.

[21] 陈子豪, 李强 . 基于 K-medoids 的改进 PBFT 共识机制 [J]. 计算机科学, 2019, 46(12): 101-107.
Chen Z H, Li Q. Improved PFFT consensus mechanism based on K-medoids [J]. Computer Science, 2019, 46(12): 101-107.

[22] Yang J, Jia Z, Su R, et al. Improved fault-tolerant consensus based on the PBFT algorithm[J]. IEEE Access, 2022, 10: 30274-30283.

[23] Alfandi O, Otoum S, Jararweh Y. Blockchain solution for IoT-based critical infrastructures: Byzantine fault tolerance[C]// NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium. ACM, 2020: 1-4.

（编辑 吕建斌）