一种改进 PBFT 算法作为以太坊共识机制的研究与实现

黄秋波　安庆文　苏厚勤
（东华大学计算机科学与技术学院　上海 201620）

摘　要　针对以太坊中 PoW(Proof of Work)共识机制在联盟链场景下表现出的由于算力竞争造成的资源浪费和不可靠问题，提出了采用 PBFT(Practical Byzantine Fault Tolerance)算法作为以太坊共识机制，并结合以太坊结构对 PBFT 算法进行改进。改进 PBFT 算法中，检查点协议取消了定时检查清除证书的过程，节点同步过程采用向其他节点索要区块并校验的方式完成同步；视图切换协议在结合区块生成协议的基础上，采用超时机制进行视图切换。实验结果说明采用改进 PBFT 的以太坊适用于联盟链场景中，可以在很大程度上减少算力开销，并在一定程度上减少网络上的数据传输量。

关键词　以太坊　共识机制　PBFT　联盟链

中图分类号　TP3　　**文献标识码**　A　　DOI:10.3969/j.issn.1000-386x.2017.10.051

STUDY AND REALIZATION OF AN IMPROVED PBFT ALGORITHM AS AN ETHEREUM CONSENSUS MECHANISM

Huang Qiubo　An Qingwen　Su Houqin
(*School of Computer Science and Technology, Donghua University, Shanghai 201620, China*)

Abstract　Aiming at the resource waste and unreliability caused by computing power competition of PoW consensus mechanism in the scene of Consortium blockchain, PBFT algorithm is proposed as an ethereum consensus mechanism. And the PBFT algorithm is improved with the combination of the ethereum structure. In the improved PBFT, checkpoint mechanism cancelled the process of checking and eliminating certificate regularly and the synchro process of nodes was realized by the scheme of requesting blockchain from other nodes and verifying it. On the basis of blockchain production mechanism, view-change mechanism adopted timeout scheme to change view. Experimental results show that ethereum which is based on improved PBFT algorithm is better for the scene of consortium blockchain. The computing power is reduced a lot and the amount of data transmission is also reduced.

Keywords　Ethereum　Consensus mechanism　PBFT　Consortium blockchain

0　引　言

近些年来，互联网贸易越来越普及，给人们带来极大便利，然而这个过程需要一个可信任的第三方去保证交易的执行。2009 年中本聪[1]首次提出了区块链这一概念。区块链的产生实现了真正意义上的去中心化可信任交易系统。

区块链的本质是分布式系统，该系统具有去中心化与安全可信的特点。早期的区块链技术是借助虚拟电子货币实现去中心化的支付系统，在利用数字加密技术实现点对点的交易的基础之上，通过随机哈希将交易与时间戳等信息一并写入到一个基于工作证明 PoW 的无限延伸的链式结构中，并以激励机制鼓励全网共同维持区块链正确的延伸。以太坊[2]作为区块链技术的应用，具有一套可以执行图灵完备的脚本语言的虚拟机，并引入智能合约的概念，实现了去中心化应用 DApp(Decentralized Application)。以太坊的出现，使得区块链技术的应用不仅仅局限于电子货币的交易，基于区块链的去中心化应用也推动着区块链技术应用于各行各业。

随着区块链技术在商业化中的应用，区块链技术

收稿日期：2017-01-01。黄秋波，副教授，主研领域：车联网，大数据，分布式数据库。安庆文，硕士生。苏厚勤，教授级高工。

已经从公链形式向着联盟链方向发展。联盟链应用之一是解决金融领域中跨行交易清算相关问题,达成在清算过程中不依赖可信任第三方作为监理的目标。本文结合该目标以实现去中心任务发布平台作为背景展开研究。几家银行分别管理各自用户的账户信息,这些银行想要相互之间联合账户信息搭建跨企业用户的任务发布平台,并根据任务完成情况进行转账交易。采用传统的中心化系统运行该任务发布平台会造成银行间不信任的情况,因此需要依赖第三方进行监管,这种模式会造成大量开销。本文提出采用区块链技术进行解决,每个银行负责维护以太坊节点参与区块的生成与校验,实现账户间交易去中心化执行。但以太坊在应用于联盟链场景中面临着资源浪费和不可信的问题。本文针对这些问题,本文提出采用改进的 PBFT 共识算法进行解决,并加以实现。

1　相关工作

以太坊采用 PoW 共识机制,一方面会因为算力竞争造成大量的资源浪费,这是联盟链中不期望的;另一方面,联盟链中所有节点的总算力往往达不到公链中所有节点总算力的数量级,因此联盟链中一个节点很容易通过叠加 CPU 达到联盟链全网 50% 的算力并进行攻击,因此在联盟链中 PoW 共识机制并不可靠。

近些年人们对共识算法不断研究并取得一定进步。King 等[3] 首次提出了权益证明 Pos(Proof-of-Stake)的共识机制缓解了算力竞争造成的资源浪费,但仍然没有摆脱挖矿过程。Larimer[4] 提出了授权股权证明 DPos(Delegated-Proof-of-Stake)的共识机制,但该机制要依赖于代币,而在很多的商业应用中是没有代币的。Schwartz[5] 提出了 RPCA(Ripple Protocol Consensus Algorithm)共识机制,该算法结合拜占庭将军问题,摆脱了通过挖矿达成共识的限制,但 Ripple 对去中心化应用研究较少。Linux 基金会推出的超级账本项目[6] 采用了 PBFT[7-8] 算法达成全网共识。然而超级账本项目仍然处于开发阶段,系统可靠性并没有经过大规模检验。PBFT 算法首先由 Castro[7] 提出并用于解决异步分布式系统如何达成一致的问题。将 PBFT[7] 算法应用于区块链的共识机制,具有较高的可靠性。节点间通过协商达成一致,该算法可以保证当不多于三分之一的节点发生拜占庭错误时区块链仍能保证正常运转。

本文结合相关研究,提出基于改进 PBFT 算法的以太坊共识机制,并应用于区块的一致性校验中。采用改进 PBFT 共识算法,节点不需要通过工作证明来达成共识,从而解决了资源浪费的问题。与 PBFT 算法相比,在执行检查点协议和视图切换协议时,改进 PBFT 算法不需要进行数据传输,从而减少了传输消耗。

2　系统模型

本文以联盟链作为应用场景,基于以太坊设计了一个异步的分布式系统,系统中各节点通过网络进行交互。该系统类似于分布式数据库,并以区块的形式存储交易信息。系统中各节点间在相互不信任的前提下,利用分布式节点共识机制完成数据生成与存储的操作。系统中每个节点运行的程序相同,为改进后的以太坊。改进后的以太坊架构如图 1 所示。

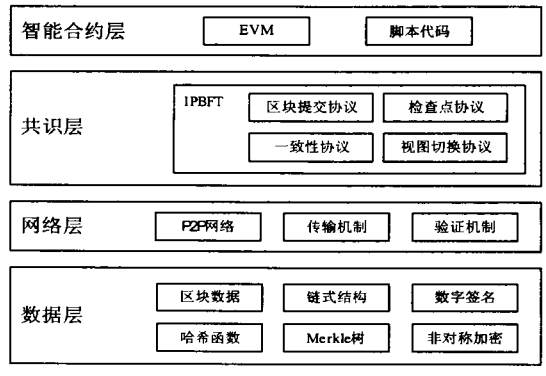

图 1　改进后以太坊系统架构

相比于以太坊架构,由于该系统模型应用于联盟链场景,节点不需要被激励也可以进行区块链的维护工作,因此取消了激励层。同时该架构保留了以太坊的智能合约层、网络层和数据层,提供以太坊中基本的服务。在共识层中,采用改进 PBFT,IPBFT(Improved Practical Byzantine Fault Tolerance)算法作为以太坊的共识机制。利用拜占庭容错算法使分布式系统中各节点对区块达成一致后,将区块添加到该节点维护的区块链中,实现数据的存储。在这个过程中,网络可能出现错误消息、延迟、重复消息等错误,但该系统仍然可以对区块的一致性达成共识。

3　构建以太坊共识机制

本文根据以太坊区块结构,设计了适用于以太坊的一种改进 PBFT 共识机制,解决了以太坊在联盟链场景下面临算力竞争造成的资源浪费和不可信的问题。其中对于 PBFT 算法中检查点协议和视图切换协议进行了如下改进:在检查点协议中,本文取消了证书删除节点间协商过程,系统同步区块过程采用节点索要的方式;在视图切换协议中,本文根据区块生成协

议,采用超时机制进行视图切换,在一定程度上减少网络通信量。

3.1 算法定义

在 PBFT 机制中有如下定义:

定义 1 Quorum 是由系统节点集合组成的,并且任意两个 Quorum 的交集不为空。假设系统节点集合为 U,$\partial = \{Q_1, Q_2, \cdots, Q_n\}$,且 $Q_1 \subseteq U$,满足式(1)则称 Q 为一个 Quorum。

$$\forall Q_i, Q_j \in \partial \text{ 且 } Q_i \cap Q_j \neq \phi \tag{1}$$

Quorum 有如下性质:

(1) 任意两个 Quorum 至少有一个共同的并且正确的节点。

(2) 一定存在着没有错误的 Quorum。

本文定义 $2f+1$ 个节点为一个 Quorum,其中 f 代表系统中最大可容忍错误节点个数,这样可以保证在一个 Quorum 至少有 $f+1$ 个节点没有发生错误。

定义 2 PBFT 机制中正确节点以一致性的形式观察群中节点的变化,把这样达成一致的群称为视图(View),视图进行编号,记作 v。假设一个视图共有 N 个节点,每一个节点的编号依次为 $\{0,1,2,\cdots,N-1\}$,其中主节点的编号记为 p(其余节点为备份节点称为 replica),满足:

$$p = v \bmod N \tag{2}$$

其中 v 是视图的编号。当群中主节点发生故障,则下一个编号的节点成为主节点,视图也随之发生切换,视图编号增加 1。

定义 3 证书:系统达成共识过程需要进行消息传输,该消息称作为证书。其中,证书信息中包含信息的编号,本文以区块号作为信息编号。

3.2 区块生成协议

在区块链中,交易以区块的形式提交并永久记录到区块链中。在本系统中,交易记录到区块链中的过程如下:

(1) 主节点对交易列表中的交易进行校验,并将交易列表中的交易写入到区块中,并向其他节点发送区块。

(2) 全网其他节点对该区块的合法性进行校验。

(3) 校验通过所有节点将该区块添加到区块链中,并从交易列表中清除该区块中包含的交易。该区块中的交易记录到区块链中并生效。

区块的校验包括对区块头信息的校验和区块体中交易信息的校验,区块头中包含上一区块的哈希值和当前区块的时间戳等信息。当交易到来时,交易列表不为空,此时由主节点将交易写入到区块中并广播该区块。当全网节点对该区块达成一致后,尝试将该区块添加到区块链中。整个过程是异步的,节点间通过区块号和区块记录的上一区块哈希值保证区块添加到区块链的有序性。当交易列表为空时,节点会监听区块链中最优区块的时间戳与系统时间间隔,当时间超过 t 会生成一个空的区块并添加到区块链中。这里考虑到在信息传输过程会产生网络延迟,假设区块从生成到达成共识并添加到区块链的最长时间为 Δt,其中 t 需要满足 $t > \Delta t$,这样可以保证在生成空区块时,之前的区块已经在全网达成一致。当区块链中添加空区块后,主节点停止生成区块,并等待交易到来时再重新触发生成区块。

3.3 一致性协议

该协议主要是保证区块在全网达成一致。由拜占庭算法可知,一个系统可以容忍不超过全部节点的三分之一发生拜占庭错误,因此这里假设在区块链系统中共有 $3f+1$ 个节点,一个 Quorum 至少包含 $2f+1$ 个节点。

在一个可以容忍拜占庭错误的系统中 f 的最小取值为 1,因此系统至少由 4 个节点组成。图 2 是由 4 个节点组成的系统进行一次正常请求的执行过程。

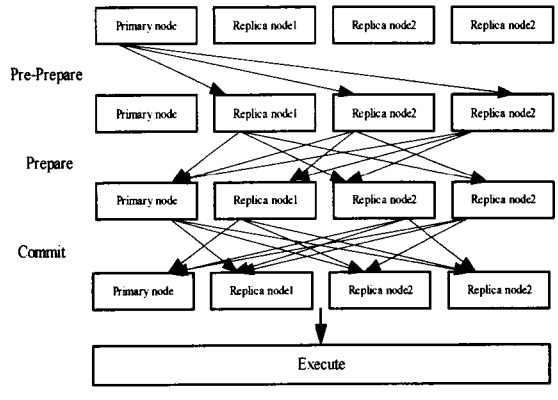

图 2 正常请求执行过程

由图 2 可知,一个信息达成共识并执行需要经过三阶段执行协商后达成一致执行,三阶段协商过程如下:

(1) 当主节点中满足生成区块条件时生成一个新区块时,主节点生成 Pre-Prepare 证书,将 Pre-Prepare 证书发送给其他节点后,主节点进入 Prepare 状态。

(2) replica 收到 Pre-Prepare 证书时就接收到了新生成区块的信息,同时该节点进入到 Prepare 状态。当该节点发现消息是来自于主节点时并且第一次接收时,会将 Prepare 证书发送给其他节点,并记录证书信息。当发现某一条证书通过 $2f$ 个节点同意的反馈,表明该区块信息通过了一个 Quorum 的同意,那么对于这条证书该节点进入 Commit 的状态,并向其他节点发送

Commit 消息。

(3) replica 会接收到来自其他节点的 Commit 的证书,当发现该信息得到了 $2f+1$ (包括自身)个节点同意,则认为该区块信息在系统中达成共识,并尝试将该区块添加到区块链中。

通过上述三阶段提交方式,使一个区块实现了全网节点达成一致。以图 2 为例考虑到当 replica 节点发生拜占庭错误时,由于另外两个 replica 是正确节点,因此仍可以满足 $2f+1$ 个节点通过验证,正确节点之间可以保证区块的一致性;当主节点发生拜占庭错误时,通过在 replica 节点中重新选出主节点生成区块并发送消息。接着将该区块添加到区块链,正确的区块会成功添加到区块链,并触发下一个区块的生成,这个过程是循环执行的。

执行过程主要对区块进行校验,当完成校验,并证明该区块是正确时,会将区块内包含的交易从该节点的交易列表中清除掉,并将该区块添加到区块链中。

3.4　检查点协议

该协议的主要目的是为了维护节点所存储信息的规模,解决证书信息的回收,从而降低节点的内存开销。PBFT 机制中定义检查点协议是通过节点间定时协商后再进行清除,这是为了防止个别节点不同步需要收集之前的证书。因此清除过期证书也是一个全网达成一致的过程,这势必又要进行 3 阶段提交过程,造成通信浪费。本文根据区块链中最优区块的时间戳进行证书清除。区块链是以链表的形式按照区块的生成时间相连而成,当一个区块添加到区块链中,则说明该区块时间戳之前的证书都已经被校验过,即该节点中这些证书相关的状态已经广播完毕,并可以进行清除,而证书的信息则以区块的形式永远保存在该节点中。因此对添加区块事件进行监听,每当有区块添加到区块链中,将该节点中该区块时间戳之前的证书进行清除。整个过程不需要节点间相互通信,同时也可以保证证书及时的清理。

以太坊中包含了节点间同步区块的解决方案,当一个节点发现自身维护的区块链与其可信任的节点维护的区块链的最优区块的区块号相差一定大小时,该节点会向其信任的节点(一般为开发者维护的节点)索要区块并将区块添加到区块链中。而在联盟链中并不存在完全可信的节点,因此该方案并不可行。本文提出当某节点区块链状态与其他节点不一致时,向该视图中的 $2f+1$ 个节点请求该区块链需要添加的区块的区块哈希,区块哈希是可以唯一标识区块的 256 位字节数组,当有不少于 $f+1$ 个节点返回的区块哈希一致,则认为该区块哈希对应的区块在全网达成共识。该节点首先会在 Pre-Prepare 证书中查找是否存在该区块哈希的证书,不存在会向其中一个节点索要该区块哈希对应的区块,并将该区块添加到区块链中,实现同步。该过程中数据传输量对于网络开销可以忽略不计。

3.5　视图切换协议

该协议是针对主节点发生故障,视图发生改变,需要更换主节点的协议。PBFT 机制详细地阐述了视图切换的过程,因为视图的切换不当会使得整个系统不一致,因此这个过程不可避免地也需要进行节点间的相互通信。在本系统中,采用对区块链最优区块监听的方式判断主节点是否发生故障,当满足添加区块的条件下,节点没有进行区块的添加则认为主节点发生故障,此时需要进行视图切换。具体算法如下:

算法 1　ViewChange( )

```
Input:该节点维护的区块链和交易列表
Output:是否进行视图切换
1:time←CurrentTime
2:Repeat
3:   if 交易列表不为空 then
4:      if BestBlock 的交易信息为空 then
5:         if CurrentTime-time 大于 t then
6:            ChangeView
7:      else
8:         time←BestBlock 的时间戳
9:         if CurrentTime-time 大于 t then
10:           ChangeView
11:     else
12:        if BestBlock 的交易信息为空 then
13:        if 下一个区块交易信息为空
14:           ViewChange
15:        else
16:           Thread. Sleep(t)
17:           time←CurrentTime
18:     else
19:        time←BestBlock 的时间戳
20:        if CurrentTime-time 大于 t then
21:           ChangeView
22:Until ViewChange
```

视图切换过程会将证书列表进行清除,并由新的主节点完成提交交易的操作,并继续维持系统的稳定。通过上述算法,当主节点发生错误时,系统会在时间内完成视图的切换。该算法需要更长的超时时间保证在主节点发生错误时,全网已经达成一致。这对于一般的分布式系统可能会造成服务中断等问题,而对于联

盟链环境下以太坊来说,服务并不会停止。其他正确节点仍然可以将交易存入交易列表中,并由其他节点各自维护的本地数据提供服务。整个视图切换过程根据区块链中最优区块时间戳采用超时触发,在区块链可容忍的延迟的范围,完成主节点的切换,不需要节点间相互通信,减少了通信消耗。

4 系统实现

本文采用了以太坊作为区块链框架,采用改进 PBFT 算法作为以太坊的共识机制,并将该系统与原有的以太坊开源框架进行比较。

本系统保留了以太坊的绝大优势,因此从功能方面实现了以太坊提供的功能。从资源浪费的角度,本文所采用的系统取消了挖矿的过程和激励机制,因此节点不需要进行大量的运算,并通过算力竞争来达到共识,解决了 PoW 共识算法所造成的资源浪费的问题。本文所采用的改进 PBFT 算法更加适用于联盟链的环境中,节点的可靠性较高,容错率要求虽然达不到 PoW 共识算法的 50%,但 33% 的容错率也足以满足场景需求。对于出块时间,由于 PoW 需要计算随机数,即使通过难度系数的调整也很难让出块时间稳定。一方面交易随着区块被记录到区块链的时间是随机的,因此交易被记录到区块链的时间也是随机的,另一方面,出块时间的不确定会造成区块链出现分支,不利于区块链的维护。而本文采用的共识算法,当有交易的时候会即时完成区块的一致性协商和区块校验,因此交易会第一时间记录到区块链中,同时由一个节点生成区块,不存在分支的情况,可以很好地维护区块链中的数据。

从 PBFT 算法角度看,本文针对以太坊改进的 PBFT 算法在检查点协议和视图切换的过程中,并没有采用各节点相互协商达成一致,而是利用区块链最优区块的时间戳在各节点不通信的情况下达成一致,减少了消息传输成本。

本文的模型适用于联盟链环境中,可以很好地支持以太坊联盟链的运行,而基于 PoW 共识机制的以太坊适用于节点多的公网之中。

5 实验验证

本文实验硬件采用 4 核 8 线程的 Intel(R) Core(TM) i7-4870HQ 处理器,16 GB 内存的硬件平台。虚拟机 8 台,1 GB 内存,CPU 共享主机的一个处理器核心,网络连接 100 Mbit/s LAN。

5.1 功能验证

本文采用 8 台虚拟机分别搭建了基于 PoW 共识机制和基于改进 PBFT 共识机制的以太坊联盟链,这里认为 8 台机器的算力是相同的,并使节点发生拜占庭错误进行实验,其实验结果如表 1 所示。从实验结果可以看到,本文设计的改进 PBFT 以太坊共识算法可以实现系统的去中心化运行,在容忍系统节点错误率的情况下,PoW 共识机制最多可以容忍 3 个节点发生任何形式的错误,而在改进 PBFT 共识机制中,当主节点发生错误时,可以完成视图切换以及主节点的重新选择,但是该系统最多可以容忍 2 个节点发生错误。该实验证明了该机制可以满足本文所提的要求,但在容错率中,不如 PoW 共识机制的容错率高。

表 1 联盟链节点一致性分析

名称	系统是否正常运行			
坏节点个数	1		2	3
节点类型	是主节点	非主节点		
PoW	是	是	是	是
改进 PBFT	是	是	是	否

5.2 算力开销验证

本文对 PoW 共识机制和改进 PBFT 共识机制的算力使用情况做了相关实验和分析。进行单节点测试,在相同的机器中分别运行基于 PoW 共识机制和改进 PBFT 共识机制的以太坊,其中基于 PoW 共识机制采用满线程运行。由于 PoW 共识机制中 Ethash 需要提前生成 1 GB 的内存空间,为了只对比稳定运行的 CPU 使用率,因此提前生成好内存哈希值。由于本文使用的 CPU 是 8 线程因此以太坊设置为 8 线程挖矿方式,CPU 使用率结果如图 3 所示。

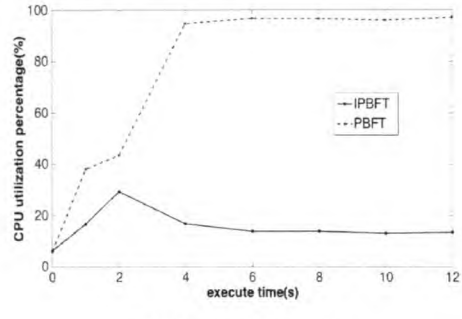

图 3 CPU 使用率

从结果可以看出基于 PoW 共识机制的 CPU 使用率接近 100%,而基于改进 PBFT(IPBFT)共识机制的 CPU 稳定使用率仅为 16% 左右,因此改进 PBFT 共识机制很大程度上减少了算力的使用,解决了 PoW 共识

机制因为算力竞争造成的资源浪费的问题。

5.3　数据传输量验证

本文分别对 PBFT 机制和改进 PBFT 机制进行验证分析。其中对于检查点协议,本文结合以太坊区块同步机制,取消了检查点中对证书清除的通信过程,因此数据传输量为零,而 PBFT 机制删除证书需要节点间达成一致,与以太坊区块同步机制重复,势必造成传输浪费。图 4 显示了当主节点发生错误的时候,一个完整的视图切换过程中的网络开销。图中横坐标代表系统内节点个数,纵坐标代表每次视图切换达成共识 Quorum 的数量,视图切换过程中,假设证书大小相同,因此通信开销和 Quorum 有关。从图中可以看出 PBFT 共识机制中,随着最大容错节点个数增多,开销增大,而改进 PBFT 共识机制中,该部分开销为零。由此也可以说明该改进 PBFT 共识机制在网络开销上较 PBFT 共识机制有一定的提高。

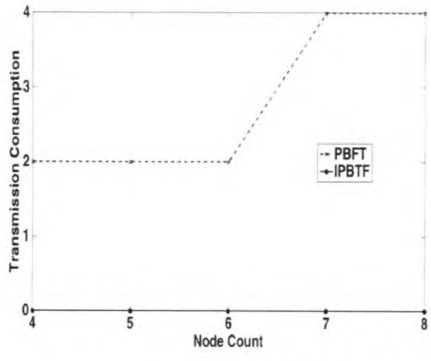

图 4　视图切换传输消耗

6　以太坊在联盟链中的应用

本节针对以太坊在联盟链场景下的应用进行描述。该应用通过联合企业各自用户的账户信息,形成公共账户管理平台,并实现了账户间交易的去中心化执行。该应用实现了任务发布系统中赏金在账户间流通的功能,这个过程类似于账户间进行交易。其中赏金的流动过程按照以太坊智能合约规则进行流通。图 5 是由四个联盟体组成的联盟链的系统框架。

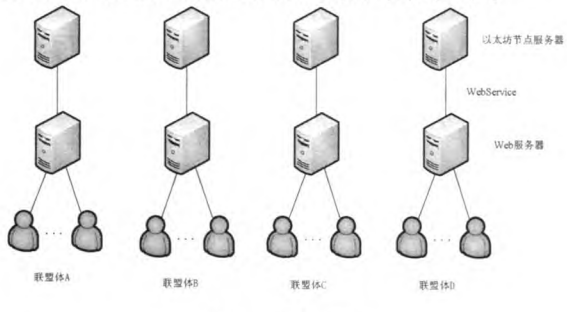

图 5　系统框架

每一个联盟体的用户与其 Web 服务器进行交互,以太坊节点服务器提供 WebService 接口供 Web 服务器调用。一个用户提交一笔交易会向 Web 服务器发出请求,Web 服务器相当于该联盟体管理的用户的代理服务器,因此交易提交到以太坊节点服务器都由 Web 服务器完成。以太坊节点服务器接收到交易将交易广播到其他节点,并按照改进 PBFT 算法对包含该交易的区块进行全网一致性验证,当该区块在全网达成共识后将该区块添加到区块链中,此时认为交易被执行并永久保存到区块链中。每一个以太坊节点地位相同,交易执行过程中,没有第三方进行中心化监管,因此实现了去中心化的功能。图 6 是以太坊区块链监控界面的截图信息,图中框内表示交易柱状图,从监控界面得知系统完成了去中心化执行交易的功能,并可以稳定运行。

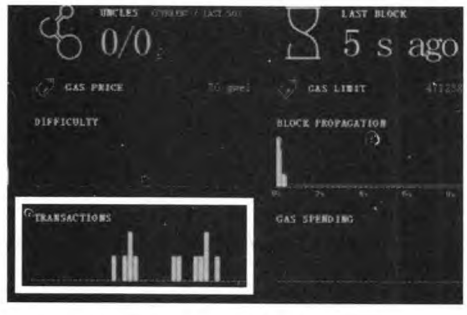

图 6　以太坊监控界面截图

7　结　语

针对以太坊区块链在联盟链的应用场景,基于工作证明共识机制存在的资源浪费和不可信问题,设计并实现了适用于联盟链环境下的以太坊共识机制。本文提出了采用 PBFT 算法作为以太坊共识机制,取消了基于工作证明共识机制中的挖矿过程。通过主节点生成区块,其他节点就该区块是否在全网达成一致进行协商的方式达成共识,解决了原系统中由于算力竞争造成的资源浪费和不可信问题。其中针对 PBFT 算法中检查点协议和视图切换协议进行改进,使该协议更适用于以太坊环境中,并在一定程度上减少了网络通信开销。最后采用该以太坊搭建联盟链,实现任务发布系统中赏金流动的去中心化执行。

参　考　文　献

[1] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system [J]. Consulted, 2009.

[2] Buterin V. A next-generation smart contract and decentralized application platform[J]. Etherum, 2014(1): 1-36.

(下转第 297 页)

差。然而,当迭代次数大于等于 3 时,其误比特与迫零算法趋于一致,这说明当迭代次数大于等于 3 时,就可以得到精确的值,说明了该算法收敛速度快,所需的计算量少。

考虑到基于雅克比预编码算法的并行性能,CPU 是 i3-3220 四核内存,GPU 是英伟达 Geforce GT620,共包含 32 个 CUDA 核,如图 4 所示。

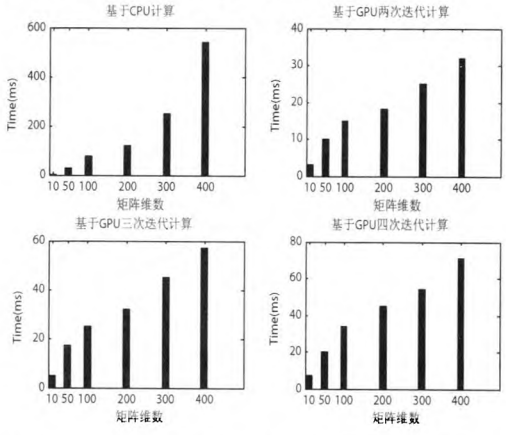

图 4　基于 CPU 串行与 GPU 并行计算时间比较

从图 4 可以看出,相同矩阵维数,基于 CPU 计算时间比 GPU 并行计算时间要长,为了定量知道并行程序的执行速度相对于串行程序的执行速度加快了多少,我们定义其加速比,表示的是同一个任务在单处理器系统和并行处理器系统中运行消耗的时间比率。如图 5 所示给出了雅克比算法次数为 2、3 和 4 时的加速比,仿真结果表明,随着矩阵维数增大,其加速比越大,说明时间缩短的比例越大。

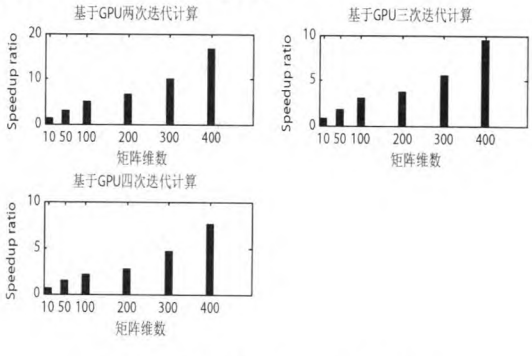

图 5　基于 GPU 并行计算加速比

4　结　语

本文针对无线通信系统预编码计算量大的缺点,提出了基于低复杂雅克比算法。该算法通过迭代,避免了矩阵求逆计算,与传统的迫零算法相比,计算量减少了一个数量级,仿真结果表明该算法收敛速度快。当迭代次数大于等于 3 时,就可以得到精确的值,所需的计算量少。然后,我们提出了快速雅克比预编码算法并行实现。仿真结果表明并行计算可以大大减少计算时间,当矩阵维数越大,其获得的加速比越大,说明了相对于串行计算时间而言,并行计算所需要的时间更短。

参　考　文　献

[1] Rusek F, Persson D, Lau B K, et al. Scaling Up MIMO: Opportunities and Challenges with Very Large Arrays[J]. IEEE Signal Processing Magazine, 2012, 30(1): 40 - 60.

[2] Gao X, Dai L, Zhang J, et al. Capacity-approaching linear precoding with low-complexity for large-scale MIMO systems [C]//2015 IEEE International Conference on Communications (ICC), 2015: 1577 - 1582.

[3] Costa M. Writing on dirty paper[J]. IEEE Transactions on Information Theory, 1983, 29(3): 439 - 441.

[4] Gao X, Dai L, Hu Y, et al. Matrix inversion-less signal detection using SOR method for uplink large-scale MIMO systems [C]//Global Communications Conference. IEEE, 2015: 3291 - 3295.

[5] Prabhu H, Rodrigues J, Edfors O, et al. Approximative matrix inverse computations for very-large MIMO and applications to linear pre-coding systems [C]//Wireless Communications and NETWORKING Conference. IEEE, 2013: 2710 - 2715.

[6] 陈国良. 并行计算[M]. 高等教育出版社, 2011.

[7] 卢凤顺, 宋君强, 银福康, 等. CPU/GPU 协同并行计算研究综述[J]. 计算机科学, 2011, 38(3): 5 - 9.

[8] Yu D, He S, Huang Y, et al. A fast parallel matrix inversion algorithm based on heterogeneous multicore architectures [C]//IEEE Global Conference on Signal and Information Processing, 2015: 903 - 907.

[9] Mahfoudhi R, Mahjoub Z, Nasri W. Parallel Communication-Avoiding Algorithm for Triangular Matrix Inversion on Homogeneous and Heterogeneous Platforms[J]. International Journal of Parallel Programming, 2015, 43(4): 631 - 655.

(上接第 293 页)

[3] King S, Nadal S. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake[OL]. 2012. http://ppcoin.org/static/ppcoin-paper.pdf.

[4] Larimer D. Delegated proof-of-stake white paper [OL]. 2014. http://www.bts.hk/dpos-baipishu.html.

[5] D Schwartz, N Youngs, A Britto. The Ripple protocol consensus algorithm[OL]. 2015. https://ripple.com/files/ripple_consensus_whitepaper.pdf.

[6] Cachin C. Architecture of the Hyperledger Blockchain Fabric [Z]. 2016.

[7] Castro M, Liskov B. Practical byzantine fault tolerance and proactive recovery[J]. Acm Transactions on Computer Systems, 2002, 20(4): 398 - 461.

[8] Platania M, Obenshain D, Tantillo T, et al. On Choosing Server-or Client-Side Solutions for BFT[J]. Acm Computing Surveys, 2016, 48(4): 1 - 30.