https://crad.ict.ac.cn
63(6): 1613−1626, 2026

Climber: 一种线性视图切换且高鲁棒的拜占庭容错共识协议

李晓光　谢佳宏　李凌峰　杜岳峰　王俊陆
（辽宁大学信息学部　沈阳　110036）
（xgli@lnu.edu.cn）

Climber: A Linear View Change and Robust BFT Consensus Protocol

Li Xiaoguang, Xie Jiahong, Li Lingfeng, Du Yuefeng, and Wang Junlu
（*Faculty of Information*, *Liaoning University*, *Shenyang* 110036）

Abstract　An efficient and robust Byzantine fault tolerance (BFT) consensus protocol serves as the cornerstone of consortium blockchain applications. In existing proposal processes, most protocols require the primary node to transmit $O(n^2)$ signatures, which significantly impairs protocol efficiency. Some protocols achieve linear view change, where the primary node only needs to send $O(n)$ signatures to verify the validity of proposals, but at the cost of sacrificing optimistic responsiveness or introducing an extra round of message exchange. To address these inherent trade-offs, we propose Climber, a novel BFT consensus protocol that integrates linear view change, optimistic responsiveness, and two-phase commit into a unified framework. Specifically, during view change, the newly-elected primary node enumerates all unlocked commitments generated in the previous view to prove proposal validity without adding extra commit phases, ensuring the protocol's operational efficiency. Furthermore, the new primary node leverages aggregate signature technology to aggregate unlocked commitments from the same view into an unlocked certificate, reducing the communication complexity of view change to $O(n)$. Additionally, a blacklist mechanism for malicious nodes is introduced, enabling the primary node to verify signatures with a time complexity of $O(n)$. Simulation experiments are conducted under different network scales and malicious node ratios, and the results show that Climber not only maintains competitive throughput and latency performance but also exhibits superior robustness compared with Fast-HotStuff, making it more suitable for large-scale consortium blockchain scenarios.

Key words　BFT consensus protocol; consortium blockchains; linear view change; robustness; Fast-HotStuff

摘　要　高效鲁棒的拜占庭容错（Byzantine fault tolerance, BFT）共识协议是联盟链应用的基础。现有提案过程中，协议多需要主节点发送 $O(n^2)$ 数量的签名，严重影响了协议的效率。部分协议实现了线性视图切换，即主节点只需发送 $O(n)$ 数量的签名来证明提案的有效性，但放弃了积极乐观响应或者增加 1 轮消息交换。针对这些问题，提出了一种同时满足线性视图切换、积极乐观响应、2 阶段提交的 BFT 共识协议

收稿日期：2025-06-04；**修回日期**：2026-03-12
基金项目：辽宁省教育厅高校基本科研项目（JYTMS20230761）；教育部产学合作协同育人项目（241201160091308）；辽宁省科技厅科学技术计划项目（2025JH2/102800040）；辽宁省职工创新工作室项目；辽宁省公共舆情与网络安全信息系统重点实验室项目（d252453002）；国家重点研发计划项目（2023YFC3304900）；辽宁省科技计划联合计划项目（2025-MSLH-311）
This work was supported by the Liaoning Provincial Department of Education Basic Scientific Research Project for Higher Education Institutions (JYTMS20230761), the Ministry Education University-Industry Collaborative Education Program of China (241201160091308), the Liaoning Provincial Department of Science and Technology Science and Technology Plan Project (2025JH2/102800040), the Liaoning Provincial Employee Innovation Studio Project, the Liaoning Provincial Key Laboratory of Public Opinion and Network Security Information System Project (d252453002), the National Key Research and Development Program of China (2023YFC3304900), and the Liaoning Provincial Science and Technology Joint Program Project (2025-MSLH-311).
通信作者：王俊陆（wangjunlu@lnu.edu.cn）

Climber。首先,在发生视图切换时,新任主节点通过穷举未上锁承诺,在不增加提交阶段的同时,可以向副本节点证明提案的有效性;其次,新任主节点利用聚合签名技术,将相同视图的未上锁承诺聚合为未上锁证书,使视图切换时的通信复杂度降低至$O(n)$;最后,引入恶意节点黑名单机制,使主节点只需花费$O(n)$的时间代价验证签名。仿真实验结果表明,Climber相比Fast-HotStuff等先进BFT协议具有更好的鲁棒性。

关键词 BFT共识协议;联盟链;线性视图切换;鲁棒性;Fast-HotStuff

中图法分类号 TP393

DOI: 10.7544/issn1000-1239.202550378 **CSTR:** 32373.14.issn1000-1239.202550378

复制状态机(state machine replication, SMR)[1]是指在分布式系统(后文简称“系统”)中,在各个服务器节点(后文简称“节点”)收到的客户端请求的顺序有可能不一致的情况下,依然能确保各个节点响应客户端请求的顺序保持一致,即确保节点们的状态机保持一致。为了实现SMR, Lamport[2]设计了Paxos共识协议,Paxos从所有节点中选出1个主节点,由主节点对客户端的请求进行排序并由管道分发给副本节点,Paxos能够容纳节点故障宕机,但无法容忍恶意节点作恶,因此该种类型的协议,如Raft[3]等都被称作故障容错(crash fault tolerance, CFT)协议[4]。为了进一步解决CFT协议中拜占庭将军问题[5],诞生了拜占庭容错(Byzantine fault tolerance, BFT)协议[6]的构想,即能够在分布式系统中容纳一定数量的恶意节点的共识协议。

第1个BFT协议PBFT(practical Byzantine fault tolerance)[7]用2个提交阶段(PREPARE阶段和COMMIT阶段)以及视图切换机制确保了在恶意节点数量小于总节点数量1/3的情况下,无论主节点是否为恶意的,都能保证所有节点执行客户端请求的顺序一致。

自比特币诞生之后,区块链[8]尤其是联盟链的兴起,使得PBFT被大范围的应用[9],但其无法满足联盟链在大规模网络环境下仍要保持高吞吐量和低时延的需求,限制了联盟链的发展[10]。PBFT性能差的原因主要在于2点:一是在2个提交阶段所有节点都要广播消息,导致系统中所有节点的通信压力都很大;二是每次视图切换后,新任主节点在PRE-PREPARE阶段需要发送$O(n^3)$个签名(n为系统中节点总数量),并且每个副本节点都需要验证$O(n^2)$个签名。

在BFT协议的改进方面,SBFT(scalable Byzantine fault tolerance)[11]使用聚合签名[12]技术,将法定人数证书(quorum certificates, QC)中的签名数量从$O(n)$降为$O(1)$,进一步将新任主节点在PRE-PREPARE阶段需要发送的签名数量降到$O(n^2)$,但是副本节点验证签名的时间复杂度依然是$O(n^2)$。Tendermint[13]和Casper[14]放弃了大多数BFT协议追求的积极乐观响应[15],让新任领导者在使用聚合签名技术的情况下,只需发送$O(n)$个签名便可推动协议,即线性视图切换,这使得频繁进行视图切换成为可能,更加符合区块链去中心化的理念。

HotStuff[15]是第1个同时满足线性视图切换和积极乐观响应的BFT协议,它定义了PREPARE阶段的法定人数证书(QC)(记为PQC),并提出如果在视图切换时主节点不再向副本节点证明视图编号最大的PQC(记为$highPQC$)的有效性,则可以将PREPARE阶段主节点需要发送的签名数量和副本节点验证签名需要的时间都降为$O(n)$,但是应该增加1个提交阶段(PRE-COMMIT阶段)来保证协议的活性,即HotStuff有3个提交阶段。Fast-HotStuff[16]提出HotStuff的流水线模式发生恶意分叉攻击时会严重影响系统的性能,并且以仿真实验来说明有必要放弃线性视图切换来保证在分叉攻击下保持鲁棒,并指出虽然主节点在PREPARE阶段需要发送$O(n^2)$个签名向副本节点证明提案的有效性,但是副本节点们验证签名需要的时间可以降为$O(n)$。并且Fast-HotStuff声称在视图切换阶段可以将节点验证签名的时间降为$O(n)$,但没有考虑到主节点验证签名需要的时间无法降为$O(n)$。

综上所述,在PBFT, SBFT, Fast-HotStuff等协议中,无法实现线性视图切换,而能够实现线性视图切换的HotStuff等协议中主节点无法证明$highPQC$的有效性,导致需要增加提交阶段,降低了效率。

由此,本文基于超时副本节点建立未上锁承诺,新任主节点使用聚合签名技术将相同视图的未上锁承诺聚合为未上锁证书。此时,仅需通过$O(c+1)$数量的签名,即可向副本节点证明$highPQC$的有效性,使主节点视图切换时的签名发送量降至$O(n+cn)$,在不增加提交阶段的同时,实现$highPQC$有效性证明与线性视图切换。

此外,因为HotStuff等协议没有考虑到主节点验

证签名需要的时间，本文针对该问题引入恶意节点黑名单机制，主节点在 NEWVIEW 阶段收集消息时，实时剔除提交错误签名或无效承诺的恶意节点，最终将主节点总验证签名时间优化为$O(n+cn)$。图 1 为 Climber 结构示意图。

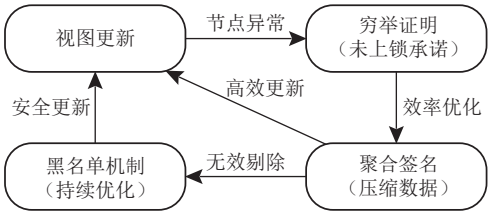


Fig. 1 Structure of Climber

图 1 Climber 结构

本文主要贡献包括 3 个方面：

1）提出了一种主节点能在视图切换时以$O(n+cn)$数量的签名代价向副本节点证明*highPQC*有效性的新方法来取代传统发送$O(n^2)$数量签名的方法，称之为穷举证明，并分析了c的大小取决于连续出现恶意主节点的数量，在恶意节点为系统节点总数 1/3 的极端情况下，c的数学期望为 1/2，且尾界上界呈指数衰减。

2）分析了在视图切换时主节点验证签名需要的时间不能降为$O(n)$的原因，并且引入恶意节点黑名单机制来完善 Fast-HotStuff，使主节点在视图切换时平均验证签名需要的时间降为$O(n+cn)$。

3）基于穷举证明和完善 Fast-HotStuff 的方法设计了一个同时满足线性视图切换、积极乐观响应、2 阶段提交 3 个特性的 BFT 协议 Climber，还分析了其安全性、活性以及签名复杂度。

1 相关工作

本节介绍 BFT 协议的发展过程，主要分析视图切换时主节点发送签名的通信复杂度以及节点验证签名的时间复杂度，其中通信成本主要为发送的签名数量和参与通信的节点数量，时间成本主要是验证签名所需的时间，各 BFT 协议对比如表 1 所示。

1.1 PBFT

PBFT[7] 为保证协议安全性，在其视图切换中规定副本节点i发送的 newview 消息中要包含节点i本地视图编号最大的 *PQC*（记为*lastPQC*），即最后一次上锁的证据，告知新任主节点*lastPQC*所在视图内至少有$n-f$个节点收到过该提案并审核通过。新任主节点在收到$n-f$条有效的 newview 消息后，将$n-f$个 *PQC* 打包形成 QC 集合并找出其中*highPQC*。为证明*highPQC*有效性主节点需将 QC 集合加入到提案消息中广播给所有的副本节点，QC 集合中的签名数量是$O(n^2)$，广播行为导致主节点需要发送$O(n^3)$个签名，同时每个副本节点验证 QC 集合所需要的时间为$O(n^2)$。PBFT 协议的流程图如图 2 所示。

Table 1 Comparison of Various BFT Protocols

表 1 各 BFT 协议对比

协议名称	聚合签名技术	积极乐观响应	线性视图切换	提交阶段数量	*highPQC* 有效性	主节点验证签名的时间复杂度	主节点发送签名的通信复杂度	副本节点验证签名的时间复杂度
PBFT	不使用	满足	不满足	2 个	证明	$O(n^2)$	$O(n^3)$	$O(n^2)$
SBFT	使用	满足	不满足	2 个	证明	$O(n^2)$	$O(n^2)$	$O(n^2)$
Tendermint/Casper	使用	不满足	满足	2 个	不证明	$O(n^2)$	$O(n)$	$O(n)$
HotStuff	使用	满足	满足	3 个	不证明	$O(n^2)$	$O(n)$	$O(n)$
HotStuff-2/HotStuff-1	使用	不满足	满足	2 个	不证明	$O(n^2)$	$O(n)$	$O(n)$
Moonshot	不使用	部分满足	不满足	2 个	证明	$O(n^2)$	$O(n^2)$	$O(n^2)$
Fast-HotStuff	使用	满足	不满足	2 个	证明	$O(n^2)$	$O(n^2)$	$O(n)$
Climber（本文）	使用	满足	满足	2 个	证明	$O(n+cn)$	$O(n+cn)$	$O(n+cn)$

1.2 SBFT

SBFT[11] 引入了收集器的概念，在 PREPARE 阶段和 COMMIT 阶段不再需要所有副本节点广播消息，而是由收集器收集后广播给所有副本节点，此种方式使得除收集器以外的副本节点在这 2 个阶段需要发送的签名数量从$O(n)$降为$O(1)$。

此外，SBFT 使用聚合签名技术，将单个 QC 的签名数量从$O(n)$降为$O(1)$，但是验证一个 QC 的时间复杂度依然是$O(n)$。因此在视图切换时，主节点需要发送的签名数量降为$O(n^2)$，但是副本节点验证签名的时间复杂度不变，依然为$O(n^2)$。SBFT 协议的流程图如图 3 所示。

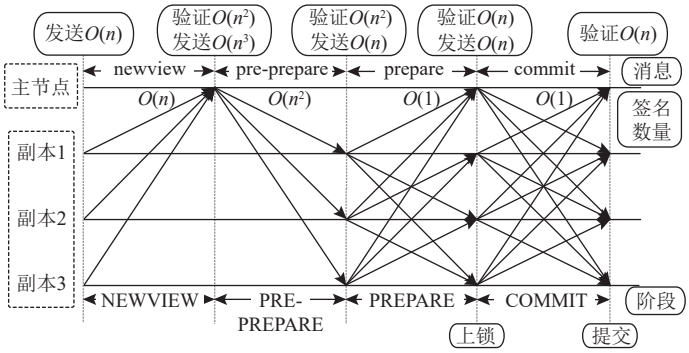


Fig. 2 Process of PBFT protocol

图 2 PBFT 协议流程

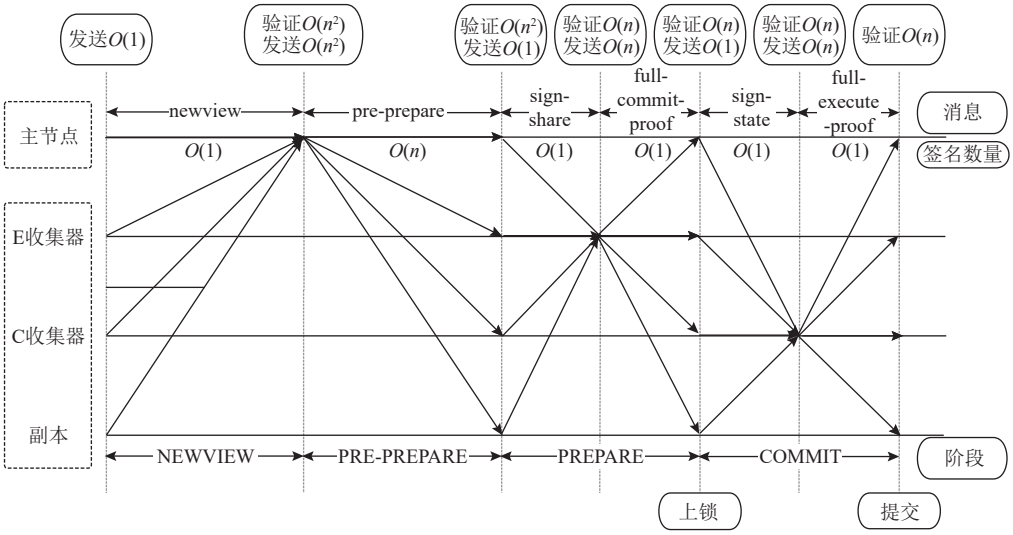


Fig. 3 Process of SBFT protocol

图 3 SBFT 协议流程

1.3 HotStuff

HotStuff[15] 将收集器的功能全部交给主节点，并且规定主节点不再发送 QC 集合证明 *highPQC* 的有效性，将主节点需要发送的签名数量和副本节点验证签名需要的时间都降为 $O(n)$，实现了线性视图切换。但是由于不再发送 QC 集合会导致部分诚实节点不敢解锁，HotStuff 必须增加 1 个提交阶段，通过设置 2 道锁来保证积极乐观响应。此外 HotStuff 还引入了流水线模式，HotStuff 协议的流程图如图 4 所示。

针对 HotStuff 协议，现有方法多致力于实现 2 阶段提交协议。HotStuff-2[17] 和 HotStuff-1[18] 使用双证书并要求新领导者等待最大网络延迟，以避免解锁问题，但代价是放弃乐观响应。Moonshot[19] 牺牲通信复杂度，换取了低延迟和吞吐量性能。Fast-HotStuff 放弃了线性切换，且在视图变更方面具有 2 次通信复杂度。

1.4 Fast-HotStuff

Fast-HotStuff[16] 论证了 HotStuff 流水线模式在面对分叉攻击时会对时延和吞吐量有很大影响。为了应对流水线模式的分叉攻击，Fast-HotStuff 放弃了线性视图切换，即主节点发送 QC 集合来证明 *highPQC* 有效以保证鲁棒。并且论证了副本节点们无需验证 QC 集合中包含的每个 *PQC*，只需要选出其中视图编号最大的 *highPQC* 而后验证即可。因此 Fast-HotStuff 得以改回 2 阶段提交，将视图切换时主节点需要发送的签名数量重新增加为和 SBFT 一样的 $O(n^2)$，副本节点们验证签名所需的时间依然为 $O(n)$。Fast-HotStuff 的流程图如图 5 所示。

但是 Fast-HotStuff 并没有考虑主节点是 QC 集合的收集者，其需要验证签名的数量和副本节点们不一致。如果主节点也只验证 *highPQC*，意味着主节点在收到一条 newview 消息时并不验证其中 *PQC* 的有效性，那么当主节点收到 $n-f$ 条 newview 消息后生成 QC 集合时，有可能出现其中 *highPQC* 是恶意节点伪造的情况，所以主节点验证签名所需的时间无法降为 $O(n)$，本文在 3.2 节中给出了完善 Fast-HotStuff，

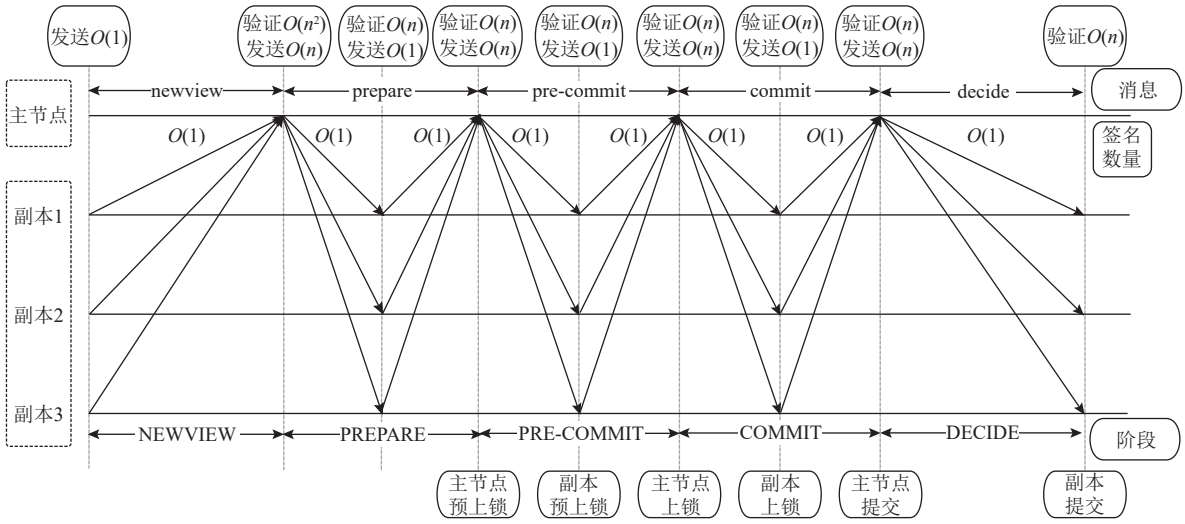


Fig. 4 Process of HotStuff protocol

图 4 HotStuff 协议流程

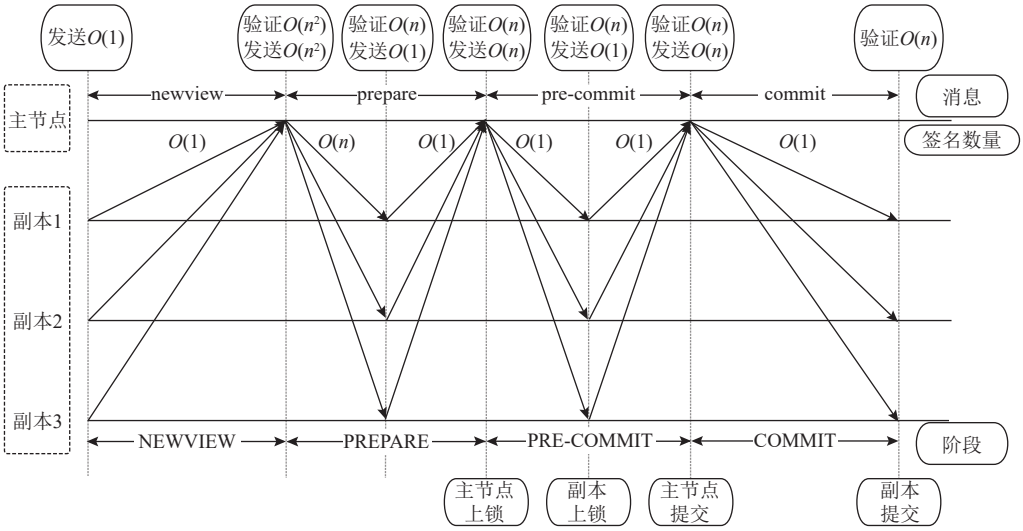


Fig. 5 Process of Fast-HotStuff protocol

图 5 Fast-HotStuff 协议流程

减少其视图切换时主节点验证签名所需时间的方案。

2 预备知识

2.1 提 案

在 BFT 共识协议中，提案（*block*）是一个客户端请求集合，这些请求还未被节点们响应，在区块链系统中通常被称作候选区块，由主节点打包生成并发送给副本节点们验证其有效性。提案中除了要包含请求集合，还要有高度（*height*）等信息。

2.2 线性视图切换

在 HotStuff、Fast-HotStuff 和本文 Climber 的系统模型中，所有节点轮流担任主节点，1 个节点担任 1 次主节点的这段时间称作 1 个视图，1 个视图内只对 1 个提案进行共识。视图由视图编号（*view*）标识，视图编号单调递增。所有节点使用 1 个确定性函数将视图编号转换为节点编号，换而言之，所有节点都知道每个视图的主节点。当前视图失败被迫更换主节点的行为称作视图切换，在视图切换时，如果主节点只需要发送 $O(n)$ 数量的签名，则协议实现了线性视图切换。

2.3 聚合签名

聚合签名技术[12]可以在不暴露个人私钥的前提下，让多方签署者以群体的形式生成数字签名。在通信协议领域中，聚合签名技术可以将 x 个不同节点对 x 条不同明文的 x 个签名聚合到一起，形成 1 条新

的签名。并且当明文相同的时候,可以有效降低消息内签名的空间复杂度,减少了消息的体积,进一步减小了网络带宽的压力。但是验证签名时消耗的时间依然随着签署人数的增加而呈线性增长。

2.4 法定人数证书

法定人数证书即 QC 是$n-f$个副本节点对相同数据或事件的签名集合,通常表示对于某些事件的证据,如果使用聚合签名技术,QC 的签名数量为$O(1)$。HotStuff 首次根据 QC 生成环节的不同定义了 PREPARE 阶段 QC 和 COMMIT 阶段 QC(记为 *CQC*)。*PQC* 是$n-f$个副本节点同意提案的签名集合,是诚实节点上锁即进入 PREPARE 状态的证据,表示在对应视图内可能存在诚实节点提交过区块。*CQC* 是$n-f$个副本节点声明已经上锁的签名集合,是诚实节点提交提案的证据。法定人数证书集合是指视图切换时新任主节点收集了$n-f$个 *PQC* 而后聚合形成的签名集合。

3 Climber 共识协议

3.1 数据结构

本节规定 Climber 协议中验证器和消息的格式。

1)验证器

Climber 中的验证器是对视图(*view*)与提案(*block*)以及验证器类型的签名或者法定人数证书的统称。某节点在视图v内未上锁承诺的验证器记作$NPSig(view, block)$,$n-f$个$NPSig$的聚合签名的验证器记作$NPQC(view,block)$,如果$NPSig$没有 *block* 字段,记作$NPSig(view)$或$NPQC(view)$。$NPSig$明文中包含的视图编号和提案信息分别用$NPSig.v$和$NPSig.b$表示。Climber 中所有验证器类型见表 2。

Table 2 Validator in Climber

表 2 Climber 中的验证器

验证器	含义
$NPSig$	某节点在视图v内没有上过锁的承诺
$NPQC$	$n-f$个$NPSig$的聚合签名
$Psig$	某节点在视图v内同意提交提案b的声明
PQC	$n-f$个$Psig$的聚合签名
$CSig$	某节点在视图v内已经对提案b上锁的声明
CQC	$n-f$个$CSig$的聚合签名

2)消息

Climber 协议中消息(m)具有固定的字段集,包括消息类型($m.type$)、视图编号($m.view$)、提案($m.block$)、消息内验证器集合($m.proofs$)和发送者对上述所有信息的签名($m.partialsig$),本文消息格式为

$$\langle m.type, m.view, m.block, m.proofs\rangle_{\text{消息发送者}}。$$

部分消息不需要包含提案字段。另外,后文中出现的消息有效只指$partialsig$字段正确,并不包含$proofs$字段。Climber 中各消息类型、$proofs$字段内容以及发送和验证$proofs$字段的代价见表 3。

Table 3 Messages in Climber

表 3 Climber 中的消息

消息	*proofs* 字段	发送通信代价	验证时间代价
newview	$CQC(curv-1)$	$O(1)$	$O(n)$
complain	$lastPQC$,c个$NPSig$	$O(c)$	$O(n+c)$
proposal	$CQC(curv-1)$	$O(1)$	$O(n)$
restore	$highPQC$,c个$NPQC$	$O(c)$	$O(cn)$
approve	$Psig(curv)$	$O(1)$	$O(1)$
prepare	$PQC(curv)$	$O(1)$	$O(n)$
precommit	$Csig(curv)$	$O(1)$	$O(1)$
commit	$CQC(curv)$	$O(1)$	$O(n)$

3.2 协议流程

本节给出 Climber 的协议流程,Climber 分为 4 个阶段:NEWVIEW, APPROVE, PREPARE, COMMIT。Climber 协议流程图见图 6。

主节点也需要执行副本节点的协议流程,每个阶段的功能以及消息描述如下:

1)NEWVIEW 阶段

在 Climber 中,不管副本节点i在当前视图($curv$)正常提交了提案还是超时,都会将 *curv* 的计数加 1,进入下个视图的 NEWVIEW 阶段,但 2 种情况执行的操作不同。

如果节点i在上个视图正常提交了提案,它将给新视图的主节点发送 newview 消息,告知其可以开启新视图,newview 消息格式如下:

$$\langle \text{“newview”}, curv, [CQC(curv-1, b_1)]\rangle_i。$$

newview 消息中的$proofs$字段只需要包含节点i在上个视图中提交提案的证据即可。如果节点i在上个视图中超时,它除了要找出 *lastPQC*,还要穷举出从视图$lastPQC.v+1$开始直到上个视图都没有进入过上锁状态的承诺(记为 *NPSig*),共a_i条(a_i=$curv-lastPQC.v-1$)。这些承诺$NPSig(lastPQC.v+1)$, $NPSig(lastPQC.v+2)$, …, $NPSig(curv-1)$记作$NPSigs$。而后节点i向新视图的主节点发送 complain 消息,其消息格式如下:

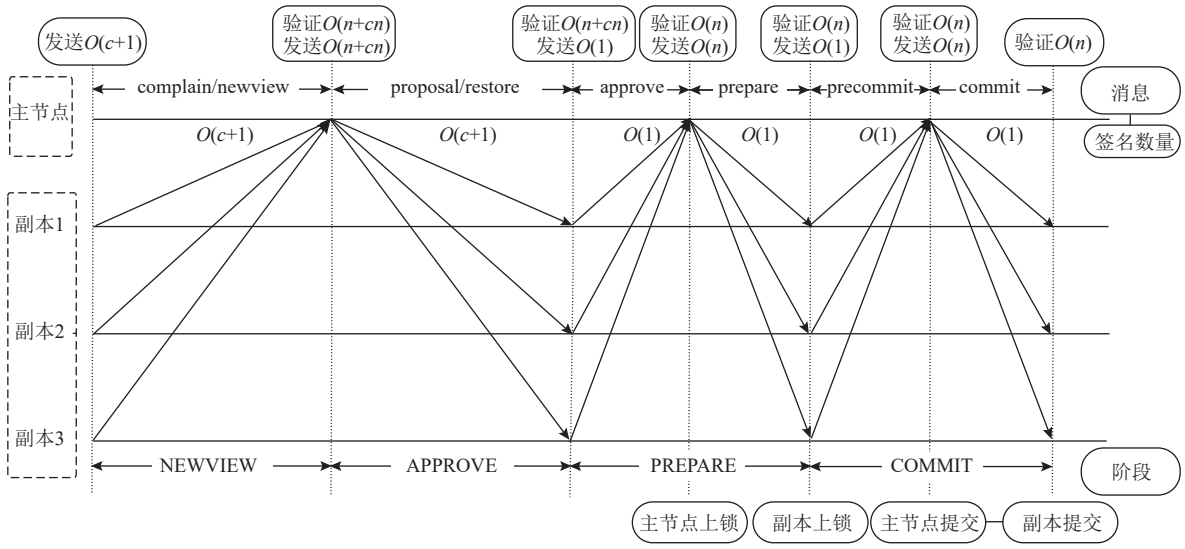


Fig. 6 Process of Climber protocol

图 6 Climber 协议流程

$\langle$“complain”,$curv$,[$lastPQC$,$NPSigs$]$\rangle_i$。

事实上，complain 消息中$partialsig$字段对$lastPQC$的部分签名，本质上是节点i做出的未上锁承诺，声明自身在$lastPQC.v$视图之后未对任何提案执行上锁操作。之所以要穷举出a_i条$NPSig$，是因为$n-f$条 complain 消息的$partialsig$字段的明文各不相同，无法进行有效的签名聚合，而$n-f$条明文相同的$NPSig$可以进行有效的签名聚合以降低签名数量。

新视图的主节点j在任意时刻，只要收到任意一个副本节点发来的，包含有效$CQC(curv-1,b_1)$且本身有效的 newview 消息，即可从本地的客户端请求池子里打包 1 个新的提案b，并广播 proposal 消息，其消息格式如下：

$\langle$“proposal”,$curv$,b_2,[$CQC(curv-1,b_1)$]$\rangle_j$。

否则，主节点j将持续等待，直到收到$n-f$条自身有效的 complain 消息，然后将这些 complain 消息按其中$lastPQC.v$从大到小排序再依次验证$lastPQC$。只要验证到一个有效的$lastPQC$则停止，并将该有效的$lastPQC$作为暂时的$highPQC$，并检查其他没有被验证$lastPQC$的 complain 消息中的$NPSig$集合是否有效且满足要求，将检查通过的$NPSig$集合取出备用。假设在上述过程中验证到了$e(e<f)$个错误的 complain 消息，意味着这e条消息都是来自恶意节点，将这些恶意节点记入黑名单，此视图内不再接收其消息，然后继续等待，这次每收到 1 个 complain 消息则先判断其中$lastPQC.v$和$highPQC.v$的大小，如果$lastPQC.v>highPQC.v$且$lastPQC$和$NPSig$集合有效，则切换$highPQC$，如果$lastPQC.v\leqslant highPQC.v$，则该 complain 消息只要有效且有满足要求的$NPSig$集合即可，直到节点j收集到$n-f$个满足要求的$NPSig$集合，即可将$curv$计数加 1 并开启新视图。

现在，节点j除了能找到$highPQC$，还有$cn-cf$条$NPSig(c=curv-highPQC.v-1)$，其中$NPSig(highPQC.v+1)$，$NPSig(highPQC.v+2)$，…，$NPSig(curv-1)$各有$n-f$条，节点$j$将它们分别聚合形成$NPQC(highPQC.v+1)$，$NPQC(highPQC.v+2)$，…，$NPQC(curv-1)$，记作$NPQCs$，而后广播 restore 消息，其格式如下：

$\langle$“restore”,$curv$,[$highPQC$,$NPQCs$]$\rangle_j$。

restore 消息中包含的c条$NPQC$可以告知所有副本节点，在视图$highPQC.v$之后直到当前视图，这段时间内任意视图中都至少有$n-f$个副本节点没有进入过上锁状态。换句话说任何副本节点在$highPQC.v$之后都不可能提交过区块，所以$highPQC$是真实有效的。

这种由进入视图切换的副本节点穷举出从$PQC.v+1$到$curv-1$期间没有进入过上锁状态的承诺，而后由新任主节点聚合即可向所有副本节点证明$highPQC$真实有效的证明方式本文称作穷举证明。由于每个$NPQC$的空间复杂度都是$O(1)$，所以单条 restore 消息的数量是$O(c+1)$，c的数学期望为 1/2(详见 3.6 节)，广播行为使新任主节点需要发送的签名数量为$O(n+cn)$，实现了线性视图切换。并且由于在 NEWVIEW 阶段新视图的主节点证明了$highPQC$的有效性，所以 Climber 并不需要像 HotStuff 一样来增加 1 个提交阶段以确保活性。

另外 Climber 中主节点j收集 complain 消息的过

程验证$lastPQC$的次数取决于恶意节点的数量,最小为1,最大为f。由于恶意节点在此环节中的作恶证据是容易被发现并且留存的,为避免每个视图在此环节都被恶意节点影响协议性能,可以在系统中加入根据作恶证据剔除恶意节点的机制,例如在区块链系统中加入相应的智能合约[20],让节点j能以客户端的身份发起剔除恶意节点请求。此种方式也可以用于完善Fast-HotStuff等协议。

2)APPROVE阶段

在APPROVE阶段副本节点i根据收到的消息类型不同进行区分。如果收到的是包含有效$CQC(curv-1,b_1)$的proposal消息,则节点i能确保在视图$curv-1$中已经有$n-f$个节点对b_1上锁,那么节点i可以直接提交b_1然后审核b_2,如果审核通过,令$b=b_2$。如果收到的是包含有效的$highPQC$以及$NPQC$集合的restore消息,此时节点i能够确定在视图$highPQC.v$之后,不存在任何诚实节点提交过区块,但是有可能有诚实节点提交过$highPQC.b$,因此即使节点i正处于上锁状态也可以解锁并令$b=highPQC.b$,而后给主节点发送approve消息,其格式如下:

$$\langle \text{“approve”}, curv, b, [PSig(curv,b)] \rangle_i 。$$

3)PREPARE阶段

主节点j在收到$n-f$条包含有效$Psig(curv,b)$的approve消息后,将$Psig(curv,b)$集合聚合生成$PQC(curv,b)$,并广播prepare消息,以告知所有副本节点已经有至少$n-f$个副本节点同意了当前提案,可以对提案b进行上锁然后进行下一阶段通信,prepare消息格式如下:

$$\langle \text{“prepare”}, curv, b, [PQC(curv,b)] \rangle_j 。$$

副本节点i在收到包含有效$PQC(curv,b)$的prepare消息后,对提案b进入上锁状态,除非后续提交了提案b或者在未来某个视图收到了正确的restore消息,否则节点i不会退出对b的上锁状态。然后向主节点发送precommit消息,其消息格式如下:

$$\langle \text{“precommit”}, curv, b, [CSig(curv,b)] \rangle_i 。$$

4)COMMIT阶段

与PREPARE阶段类似,主节点j在收到$n-f$条包含有效$CSig(curv,b)$的precommit消息后,将$CSig(curv,b)$集合聚合生成$CQC(curv,b)$,并广播commit消息,以告知所有副本节点已经至少有$n-f$个副本节点对提案b进入了上锁状态,即使后续触发视图切换,可以确保至少有1个对提案b进入上锁状态的诚实节点会进入视图切换。commit消息格式如下:

$$\langle \text{“commit”}, curv, b, [CQC(curv,b)] \rangle_j 。$$

副本节点i在收到包含有效$CQC(curv,b)$的commit消息后便可提交提案,即执行提案内的客户端命令而后将执行结果返回给客户端。

视图切换中副本节点和主节点算法分别见算法1和算法2。

算法1. 副本节点i的视图切换算法。

输入:当前视图编号$curv$,最后上锁证明$lastPQC$;

输出:签名集合$proofs$。

① $curv=curv+1$; /*切换视图*/
② $proofs=[]$; /*初始化验证器集合*/
③ $proofs.push(lastPQC)$; /*将$lastPQC$推入验证器集合*/
④ $k=lastPQC.v+1$;
⑤ while $k<curv$ do/*循环次数为$curv-lastPQC.v-1$*/
⑥ $proofs.push(NPSig(k))$;
⑦ $k=k+1$;
⑧ end while
⑨ return $proofs$。

算法2. 主节点j的视图切换算法。

输入:当前视图编号$curv$,收到的$n-f$条有效的complain消息集$complains$;

输出:签名集合$proofs$。

① 对收到的$n-f$条有效的complain消息集$complains$按$lastPQC.v$从大到小排序,初始化$highPQC$,定义$flag$=false; /*$flag$表示是否找到$highPQC$*/
② for m in $complains$ do
③ if $flag$=false and $m.proofs$有效 do
④ $highPQC=m.proofs.lastPQC$;
⑤ $flag$=true;
⑥ else
⑦ if $flag$=true and $m.proofs$中的$NPSig$集合有效 do/*不验证$lastPQC$,无需做任何事*/
⑧ else
⑨ $complains.delete(m)$; /*剔除*/
⑩ 将m的发送者计入黑名单;
⑪ end if
⑫ end if
⑬ end for
⑭ while 收到一条新的complain消息m do
⑮ if $m.proofs.lastPQC.v>highPQC.v$ and

```
        m.proofs 有效 do
⑯      highPQC = m.proofs.lastPQC;
⑰      complains.push(m);
⑱    else
⑲      if m.proofs中的NPSig集合有效 do
⑳        complains.push(m);
㉑      else
㉒        将m的发送者计入黑名单;
㉓      end if
㉔    end if
㉕    if complains中消息数量为n−f
㉖      curv = curv + 1;
㉗      proofs = [];
㉘      proofs.push(highPQC);
㉙      k = highPQC.v + 1;
㉚      while k < curv do
㉛        聚合n−f个NPSig(k)生成NPQC(k);
㉜        proofs.push(NPQC(k));
㉝      end while
㉞      作为客户端发起剔除恶意节点请求;
㉟      return proofs;
㊱    end if
㊲ end while
```

3.3 流水线模式

为了提高吞吐量，Climber也可以像HotStuff和Fast-HotStuff一样设计成流水线模式。具体的方式为当前视图的主节点在PREPARE阶段生成$PQC(curv, b)$之后，将其作为下一视图的$highPQC$发送给对应主节点，使下一视图可在当前视图结束前启动。Climber的流水线模式与Fast-HotStuff的区别只在于当视图失败需要视图切换时，Climber证明$highPQC$有效性的方法更加高效。

3.4 安全性分析

在本节和3.5节中，将分别讨论Climber是如何使用合理的上锁和解锁机制来保证协议的安全性和活性。

BFT共识协议的安全性本质是避免同一高度提案提交冲突，结合本文协议设计，安全性可形式化定义为：系统总节点数为$n=3f+1$（f为最大恶意节点数），若存在诚实节点$i\in H$（诚实节点集合记为H），在区块高度h提交提案b，即节点i验证通过对应提案的提交证书CQC并执行提案，则对任意与b不同的提案$b'\neq b$，若$b'.h=h$，则不存在任何诚实节点$j\in H$提交b'，且不存在有效提交证书$CQC(v',b')$（v'为任意视图编号）。

根据Climber协议设计，约定如下前提假设：恶意节点可伪造消息及延迟发送消息，但无法伪造诚实节点的数字签名，且无法突破$n=3f+1$的数量约束，法定人数证书需满足至少$n-f$个节点签名方可视为有效，且签名需通过节点公钥验证。诚实节点对提案的上锁行为具有排他性，若节点在视图v对提案b上锁，则在解锁前，拒绝对同高度其他提案执行上锁或签名操作，锁机制定义见表4。

Table 4 Definition of Lock Mechanism

表4 锁机制定义

符号	含义	约束规则
$Lock(i,v,b)$	诚实节点i在视图v对提案b上锁	① 触发前需验证视图v的$PQC(v,b)$有效 ② 本地无同高度已上锁未解锁提案
$UnLock(i,v,b)$	诚实节点i解除对提案b的锁定	① 触发需基于有效的$CQC(v,b)$或$NPQC(v)$ ② 解锁后清除本地同高度锁定记录

首先，提出矛盾假设，设定2个场景。在场景1及视图v_1中，提案b_1生成有效提交证书$CQC(v_1,b_1)$，构成该证书的诚实节点集合记为$A\subseteq H$，因CQC需$2f+1$个签名，且恶意节点小于f，故$|A|\geqslant f+1$。同理在场景2中，视图$v_2>v_1$中，提案b_2（$b_2\neq b_1$，$b_2.h=b_1.h$）生成有效提交证书$CQC(v_2,b_2)$，构成该证书的诚实节点集合记为$B\subseteq H$，满足$|B|\geqslant f+1$。

然后，基于场景1和场景2，可以推导出跨视图诚实节点的存在性。由于$|A\cup B|\leqslant|H|$，$|A|\geqslant f+1$，$|B|\geqslant f+1$，考虑到最坏情况恶意节点数为f，$|H|=2f+1$，可得$|A\cap B|\geqslant 1$，即存在至少1个跨视图诚实节$k\in A\cap B$。节点k既参与了$CQC(v_1,b_1)$的签名（对应$Lock(k,v_1,b_1)$触发），又参与了$CQC(v_2,b_2)$的签名（对应$Lock(k,v_2,b_2)$触发）。

最后，进行矛盾推导，根据上锁的排他性约束，若$v_2>v_1$且$b_2\neq b_1$，则$Lock(k,v_2,b_2)$触发前，节点k必须先执行$UnLock(k,v_1,b_1)$，否则违反同高度排他上锁。根据解锁的触发条件，$UnLock(k,v_1,b_1)$需满足以下二者之一：① 提案b_1已提交，即存在$CQC(v_1,b_1)$且节点k验证通过；②提案b_1确认无法提交，即节点k收到视图$curv>v_1$主节点广播的有效restore消息，且消息包含视图v_1的未上锁证书$NPQC(v_1)$。若满足条件①，则根据安全性定义，同高度的b_2无法生成有效$CQC(v_2,b_2)$，与初始假设矛盾。若满足条件②，则$NPQC(v_1)$需由$n-f$个节点的$NPSig(v_1)$聚合生成，设提供$NPSig(v_1)$的诚实节点集合为C，满足$|C|\geqslant 2f+1$。因为$|A\cup C|\leqslant 2f+1$，可得$|A\cap C|\geqslant 1$，即存在诚实节点$m\in A\cap C$。节点$m\in A$意味着m已经对提案上锁$Lock$

(m,v_1,b_1), 而$m \in C$意味着m生成了提案的未上锁签名$NPSig(v_1)$, 二者矛盾, 诚实节点无法同时执行$Lock(m, v_1,b_1)$与$NPSig(v_1)$的冲突行为, 产生矛盾。

综合上述推导可得, 存在同高度冲突提交证书的初始假设不成立, Climber 满足安全性。

3.5 活性分析

本节将讨论 Climber 是如何使用合理的解锁机制来保证协议的活性。解锁机制至关重要, 如果解锁机制不合理, 使得部分该解锁的上锁节点没有解锁, 会导致参与正常协议流程的诚实节点数量不够, 系统可能会卡死。BFT 协议的活性是指在恶意节点不超过系统总数 1/3 的情况下, 不管前面连续有多少个视图诚实节点没有提交提案, 且只要当前视图的主节点为诚实的且网络通信正常, 恶意节点怎么作恶, 诚实节点们都能在此视图内按照协议流程提交提案。

设定当前视图编号为$curv$, 上次提交提案的视图编号为v_3, 当前主节点收到的$n-f$个 complain 消息中的$highPQC.v$为v_4, 实际最后一次有诚实节点上锁的视图编号为v_5($v_3 < v_4 \leqslant v_5 \leqslant curv$)。

设任意诚实节点$i \in H$的最后一次上锁行为为$Lock(i,v_i,b_i)$, 其中$v_i < v_5$, $b_i.h = h_{curv}$, 最终需要基于 restore 消息证明 i 可执行$UnLock(i,v_i,b_i)$, 根据v_i与v_1的大小关系分 2 种情况推导:

第 1 种情况为$v_i \in [v_4,v_5]$, 即上锁视图在v_4之后, 此时 restore 消息的 *proofs* 字段必然包含$NPQC(v_i)$, 在视图v_i中对b_i上锁的节点最多为$2f$个(含所有恶意节点), 而提交提案需$2f+1$个节点的 *CSig* 签名, 故b_i在视图v_i中绝无可能生成有效$CQC(v_i,b_i)$, 节点 i验证$NPQC(v_i)$有效后, 可安全执行$UnLock(i,v_i,b_i)$, 并将本地待共识提案重置为b_4。

第 2 种情况为$v_i \in [v_3,v_4]$, $m \in C$, 即上锁视图在v_4之前, 根据假设可以算出v_4视图中至少$f+1$个诚实节点认可b_4的有效性, 假设b_i在视图v_i已提交, 根据锁的排他性约束, $f+1$诚实节点会因高度相同拒绝生成$PSig(v_4,b_4)$, 导致$highPQC(v_4,b_4)$无法满足$n-f$个签名, 这与设定中$highPQC$的有效性相矛盾, 故假设不成立, b_i在v_i未提交。节点 i验证$highPQC(v_4,b_4)$有效后, 可安全执行$UnLock(i,v_i,b_i)$, 并将本地待共识提案重置为 b_4。

所有诚实节点解锁后, 可正常参与当前视图$curv$的共识流程。

3.6 签名复杂度分析

设当前视图前连续未提交提案的视图数量为c, 系统总节点数$n=3f+1$, 最大恶意节点数量为f, 恶意节点集合M满足$|M| \leqslant f$, 且协议运行期间集合固定, 无动态加入/退出。根据 3.5 节活性分析, 视图v未提交提案的充要条件是该视图主节点$p_v \in M$。此时, 诚实主节点会按协议广播 proposal 或 restore 消息推进共识, 仅恶意主节点会拒绝广播, 导致视图超时未提交(3.2 节 NEWVIEW 阶段逻辑), 因此, c等价于当前视图前连续出现恶意主节点的视图数量。由于 Climber 采用确定性函数映射选择主节点, 设视图编号为v, 主节点 ID 为$p_v = v \bmod n$, 所有节点预先知晓该映射关系, 恶意节点无法篡改, 确保主节点轮换规则透明可验证。

1)视图间独立性证明

主节点 ID 按$p_v = v \bmod n$轮换, 周期为n, 相邻视图$v-1$与v的主节点满足$p_v = (p_{v-1}+1) \bmod n$, 因$M$固定, 任意视图$v$的主节点$p_v \in M$的概率$q$为

$$q = \frac{|M|}{n}。 \tag{1}$$

极端情况$|M| = f$, f足够大时

$$q = \lim_{f\to\infty} \frac{f}{3f+1} = \frac{1}{3}。 \tag{2}$$

对于任意视图v, 已知$p_{v-1} \in M$, p_v是否属M仅取决于节点$(p_{v-1}+1) \bmod n$是否在M中, 与p_{v-1}的恶意属性无关, 因此:

$$P(p_v \in M | p_{v-1} \in M) = \frac{|M|}{n} = P(p_v \in M), \tag{3}$$

即视图主节点恶意事件的发生在不同视图间相互独立, 满足伯努利实验条件。

2)c 的数学期望推导

由独立性可知, c服从几何分布, 视图主节点恶意的概率为q, 视图主节点诚实的概率为$p=1-q$, 其概率质量函数为

$$P(c=k) = q^k \cdot p, \tag{4}$$

其中$k \geqslant 0$, $k=0$表示当前视图前无连续恶意视图, $k=1$表示前一个视图主节点恶意, 以此类推, 由此可以进一步计算c的数学期望

$$E(c) = \sum_{k=1}^{\infty} k \cdot q^k \cdot p。 \tag{5}$$

当极端情况$q=\frac{1}{3}$, $p=\frac{2}{3}$时, 可得$E(c)=\frac{1}{2}$, 说明协议运行中c大概率取 0 或 1, 连续未提交视图数量极少。

3)开销可控性验证

为证明c取较大值的概率极低, 确保协议开销可控, 采用 Chernoff Bound 推导c的尾界, 即$P(c \geqslant k)$的上界。

对于几何分布$c \sim Geometric(p)$，其矩生成函数

$$M(t) = E[e^{tc}] = \frac{p}{1-qe^t}, \tag{6}$$

其中$t < -\ln q$，切尔诺夫界指出，对于任意$t > 0$，有

$$P(c \geqslant k) \leqslant e^{-tk} \cdot M(t) = e^{-tk} \cdot \frac{p}{1-qe^t}。 \tag{7}$$

通过求导最小化上界，解得最优

$$t^* = \ln\left(\frac{k}{q(1+k)}\right)。 \tag{8}$$

最后可以计算出c的尾界为

$$P(c \geqslant k) \leqslant 4\left(\frac{1}{3}\right)^k。 \tag{9}$$

由此计算可得，当$k=3$时，$P(c \geqslant 3) \leqslant 0.148$（实际精确值约为0.037）；当$k=5$时，$P(c \geqslant 5) \leqslant 0.016$（实际精确值约为0.004 1）；当$k=10$时，$P(c \geqslant 10) \leqslant 0.000\,046$（实际精确值约为0.000 017）。可见$c$的尾界呈指数衰减，虽然切尔诺夫界给出的上界比精确值宽松，但仍呈指数衰减，表明连续恶意主节点数量较大的概率极低，协议开销可控。

4）节点签名复杂度

对于主节点，由于其视图切换时需要发送n个restore消息，而单个restore消息的签名数量是$O(c+1)$，所以主节点在视图切换时发送签名的时间消耗为$O(n+cn)$。另外主节点在NEWVIEW阶段需要验证$NPSig$的数量是cn，因为有剔除恶意节点机制，其需要验证PQC的数量平均为$O(1)$，而验证1个QC的时间复杂度本身为$O(n)$，需要验证的签名复杂度为$O(n+cn)$。

对于普通节点，其在NEWVIEW阶段需要发送的签名数量为$O(c+1)$，在PROPOSAL阶段副本节点需要验证签名所需时间也为$O(n+cn)$。

4 仿真与实验分析

本文使用Golang语言编写Fast-HotStuff、完善后的Fast-HotStuff和Climber。实验在4台16核CPU、128 GB内存的服务器进行实现，服务器之间直接的网络通信延迟少于1 ms。为模拟广域网环境，实验设置随机50~300 ms跨区域延迟，将节点平均部署到每个服务器中，使用虚拟环境为每个节点分配1个容器，并设置1~10 ms随机网络抖动，0.01%~0.1%丢包，普通节点设置带宽上限为80 Mbps或40 Mbps，单个提案内包含的客户端请求数量选取500和5 000，在不同节点总数量（40, 70, 100, 130, 160, 190, 220, 250, 280, 310）和不同恶意节点占比（11%, 22%, 33%）下，对3种协议的流水线模式进行仿真实验，测试响应客户端请求时延和吞吐量。经过大量仿真实验，可知Climber对比Fast-HotStuff的优势大小取决于节点总数量、网络带宽、单个提案内包含的客户端请求数量、恶意节点占比4个方面。

实验在80 Mbps和40 Mbps网络带宽，提案内客户端请求数量分别为5 000条和500条的情况下，测量不同数量节点的时延和吞吐量，实验结果如图7所示。

由图7可知，网络带宽越小，节点数量越多，则单个提案内客户端请求数量越少（吞吐量），Climber的优化效果越好。在500条请求、40 Mbps网络、310节点的极端场景下，Climber优化幅度最大，相比Fast-HotStuff，时延降低约41.2%、吞吐量提升约63.3%，相比Fast-HotStuff（完善后）时延降低约19.4%、吞吐量提升约21.1%，优势明显。

在节点总数为310、请求数为500、网络带宽为40 Mbps时，不同恶意节点占比的实验结果如图8所示。恶意节点占比越大，3种协议的时延越高且吞吐量越低，但Climber具有更低的时延和更高的吞吐量，其抗分叉攻击、鲁棒性的优势突出。

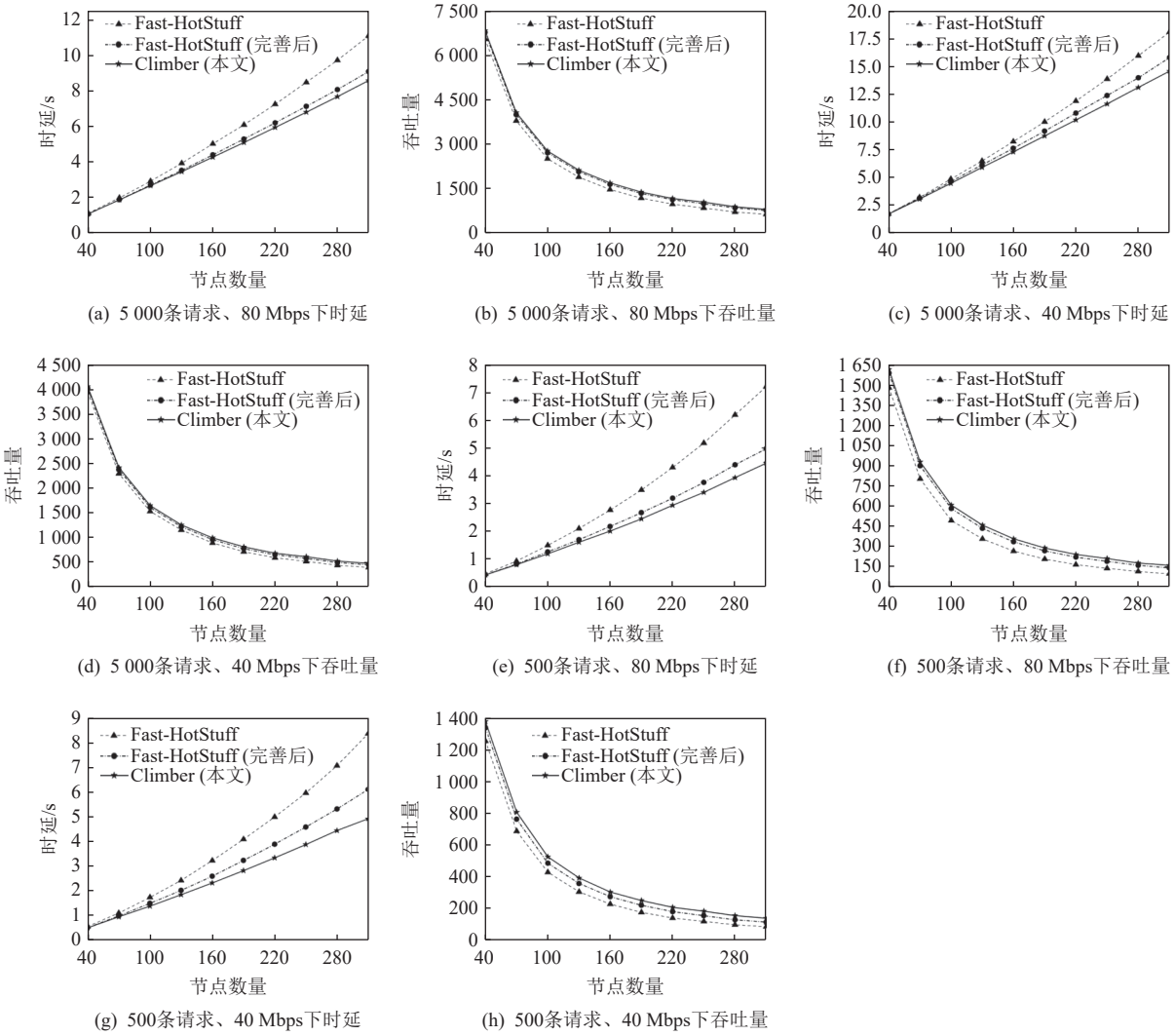
在图9实验中，对比了Climber和Fast-HotStuff中主节点在视图切换时发送的签名量。Climber是线性视图切换，其主节点发送的签名量很小，而Fast-HotStuffd的签名量随着节点数量增加呈二次增长。

Fast-HotStuff在不同提案内客户端请求数量下单个提案内签名占比如图10所示。同一节点规模，提案内客户端命令数越少，签名占比越高，随着节点数量的增多，不同命令数之间签名占比差距加大。签名数据在提案中占比过高会大幅增加带宽消耗和阻塞传输，所以Fast-HotStuff放弃了线性视图从而影响了其系统性能。

3种协议在最坏场景下（总节点数为310，其中33%为恶意节点；40 Mbps网络带宽；500客户端命令）的尾延迟累积分布函数（cumulative distribution function，CDF）如图11所示。随着3种协议的时延逐步增加，累计概率从0.9增加到1，Fast-HotStuff时延增加约23.4%，Fast-HotStuff（完善后）时延增加约19.5%，Climber时延增加约11.9%，Climber陡增现象不明显，体现了更优的极端场景稳定性。

5 结　论

随着区块链技术的兴起和应用，现有BFT共识



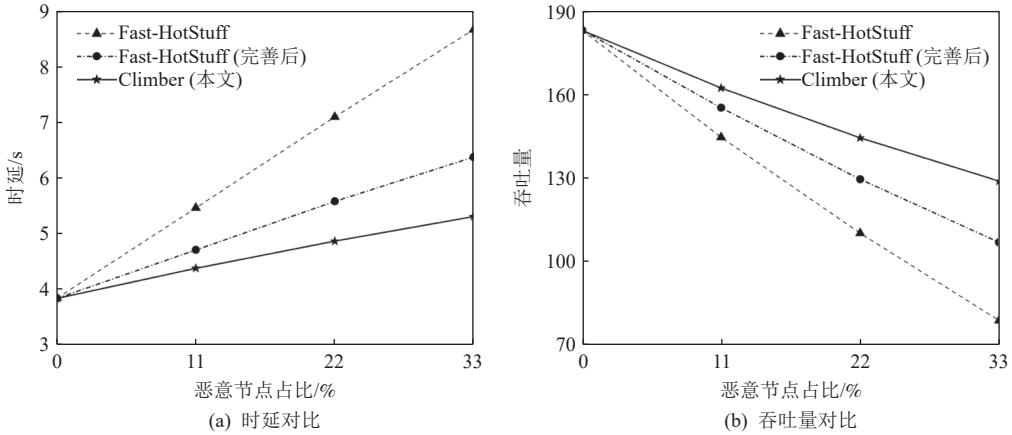
(a) 5 000条请求、80 Mbps下时延 (b) 5 000条请求、80 Mbps下吞吐量 (c) 5 000条请求、40 Mbps下时延

(d) 5 000条请求、40 Mbps下吞吐量 (e) 500条请求、80 Mbps下时延 (f) 500条请求、80 Mbps下吞吐量

(g) 500条请求、40 Mbps下时延 (h) 500条请求、40 Mbps下吞吐量

Fig. 7 Comparison of latency and throughput under different command quantities and bandwidth sizes

图 7 不同命令数量和带宽大小下时延和吞吐量对比



(a) 时延对比 (b) 吞吐量对比

Fig. 8 Comparison of latency and throughput under different proportion of malicious nodes

图 8 不同恶意节点占比下时延和吞吐量对比

协议难以满足大规模分布式网络高吞吐量、低时延、高鲁棒性需求。为此，本文提出一种线性视图切换且高鲁棒的 BFT 共识协议 Climber。该协议首先构建未上锁承诺并生成聚合证书，通过穷举证明，在不增加提交阶段的情况下，实现具有积极乐观响应的线性视图切换；在此基础上，引入恶意节点黑名单机制，

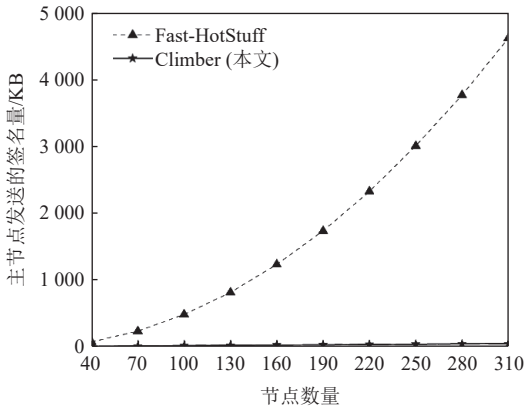


Fig. 9 Comparison of the number of signatures sent by the master node during view changes

图 9 视图切换时主节点发送的签名量对比

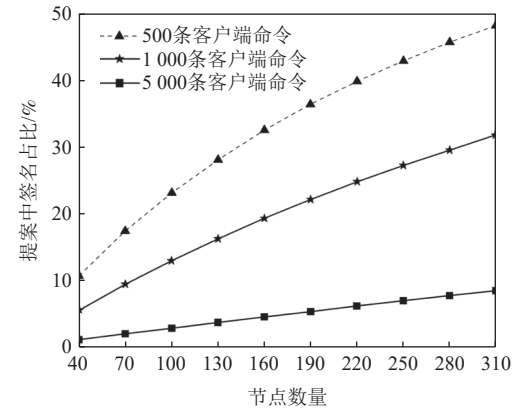


Fig. 10 The proportion of signatures in Fast-HotStuff proposal

图 10 Fast-HotStuff 提案中签名占比

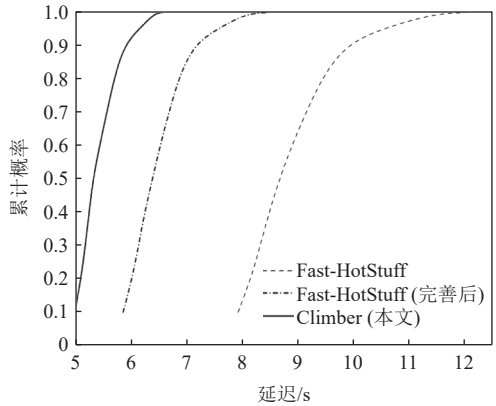


Fig. 11 Tail delay CDF

图 11 尾延迟 CDF

将主节点视图切换验证时间复杂性优化至线性级别，提升效率了的同时也增强了鲁棒性。实验结果表明，本文提出的 Climber 具有低时延、高吞吐量、高效性和抗分叉攻击的鲁棒性，有效突破了现有 BFT 协议的性能瓶颈，为高可用和抗攻击的分布式共识部署提供了新方案，助力 BFT 协议的进一步发展。

作者贡献声明：李晓光提出了新协议的设计思路和架构；谢佳宏、李凌峰负责论文撰写和修改；杜岳峰负责论文图片绘制和实验设计；王俊陆负责实验设计和论文修改。

参 考 文 献

[1] Davidson M. State machine replication and consensus with Byzantine adversaries[EB/OL]. 2023[2023-04-26]. https://csrc.nist.gov/Pubs/ir/8460/IPD

[2] Lamport L. The Part-time Parliament[M]//Concurrency: The Works of Leslie Lamport. New York: ACM, 2019: 277−317

[3] Ni Lin, Ye Qiukai, Yang Jun, et al. Research of raft algorithm improvement in blockchain[C]//Proc of the 16th Int Conf on Advanced Infocomm Technology (ICAIT). Piscataway, NJ: IEEE, 2024: 290−294

[4] Ge Ning, He Yukai, Zhai Shumao, et al. Formal verification of consensus protocols: Survey and perspective[J]. Journal of Software, 2023, 34(11): 4989−5007 (in Chinese)
(葛宁, 贺俞凯, 翟树茂, 等. 共识协议的形式化验证研究现状与展望[J]. 软件学报, 2023, 34(11): 4989−5007)

[5] Lamport L, Shostak R, Pease M. The Byzantine generals problem[M]//Concurrency: The Works of Leslie Lamport. New York: ACM, 2019: 203−226

[6] Zhang Gengrui, Pan Fei, Mao Yunhao, et al. Reaching consensus in the Byzantine empire: A comprehensive review of bft consensus algorithms[J]. ACM Computing Surveys, 2024, 56(5): 1−41

[7] Yuan Fujiang, Huang Xia, Zheng Long, et al. The evolution and optimization strategies of a PBFT consensus algorithm for consortium blockchains[J]. Information, 2025, 16(4): 268

[8] Shi Dong, Abbas K, Li Meixi, et al. Blockchain technology and application: An overview[J]. PeerJ Computer Science, 2023, 9: e1705

[9] Zhou Kai, Chen Fu, Lu Tianyuan, et al. Review of blockchain consensus algorithms[J]. Computer Science, 2025, 52(11): 255−269 (in Chinese)
(周凯, 陈福, 鲁添元, 等. 区块链共识算法综述[J]. 计算机科学, 2025, 52(11): 255−269)

[10] Wahab N H A, Zhang Dayong, Juniardi N F, et al. Advances in consortium chain scalability: A review of the practical Byzantine fault tolerance consensus algorithm[J]. International Journal of Advanced Computer Science and Applications, 2024, 15(7): 977−991

[11] Gueta G G, Abraham I, Grossman S, et al. SBFT: A scalable and decentralized trust infrastructure[C]//Proc of the 49th Annual IEEE/IFIP Int Conf on Dependable Systems and Networks (DSN). Piscataway, NJ: IEEE, 2019: 568−580

[12] Baldimtsi F, Chalkias K K, Garillot F, et al. Subset-optimized BLS multi-signature with key aggregation[C]//Proc of the Int Conf on Financial Cryptography and Data Security. Berlin: Springer, 2024: 188−205

[13] Amoussou-Guenou Y, Pozzo A D, Tucci-Piergiovanni S. Tendermint[M]//Blockchains: A Handbook on Fundamentals. Berlin: Springer, 2023: 195−221

[14] Buterin V, Griffith V. Casper the friendly finality gadget[J]. arXiv

preprint, arXiv: 1710.09437, 2017

[15] Yin Maofan, Malkhi D, Reiter M K, et al. HotStuff: BFT consensus with linearity and responsiveness[C]//Proc of the 2019 ACM Symp on Principles of Distributed Computing. New York: ACM, 2019: 347−356

[16] Jalalzai M M, Niu Jianyu, Feng Chen, et al. Fast-HotStuff: A fast and robust BFT protocol for blockchains[J]. IEEE Transactions on Dependable and Secure Computing, 2024, 21(4): 2478−2493

[17] Malkhi D, Nayak K. HotStuff-2: Optimal two-phase responsive BFT[J/OL]. Cryptology ePrint Archive, 2023[2023-04-17]. https://eprint.iacr.org/2023/397

[18] Kang Dakai, Gupta S, Malkhi D, et al. HotStuff-1: Linear consensus with one-phase speculation[J]. arXiv preprint, arXiv: 2408.04728, 2024

[19] Doidge I, Ramesh R, Shrestha N, et al. Moonshot: Optimizing block period and commit latency in chain-based rotating leader BFT[C]//Proc of the 54th Annual IEEE/IFIP Int Conf on Dependable Systems and Networks (DSN). Piscataway, NJ: IEEE, 2024: 470−482

[20] Wang Yongju, Du Xiujuan, Chen Haozhang. Overview of blockchain smart contract technology research[J]. Computer Simulation, 2023, 40(8): 1−4, 65 (in Chinese)
(汪永菊, 杜秀娟, 陈浩章. 区块链智能合约技术研究综述[J]. 计算机仿真, 2023, 40(8): 1−4, 65)



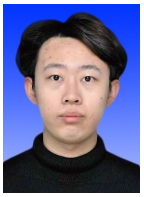
Li Xiaoguang, born in 1973. PhD, professor. Member of CCF. His main research interests include blockchain technology, machine learning, and graphdata analysis.

李晓光, 1973 年生。博士，教授。CCF 会员。主要研究方向为区块链技术、机器学习、图形数据分析。



Xie Jiahong, born in 2001. Master candidate. Student member of CCF. His main research interest includes blockchain technology.

谢佳宏, 2001 年生。硕士研究生。CCF 学生会员。主要研究方向为区块链技术。



Li Lingfeng, born in 2000. Master. His main research interest includes consensus technology.

李凌峰, 2000 年生。硕士。主要研究方向为共识技术。



Du Yuefeng, born in 1986. PhD, lecturer. Member of CCF. His main research interests include data quality management and data semantics analysis.

杜岳峰, 1986 年生。博士，讲师。CCF 会员。主要研究方向为数据质量管理、数据语义分析。



Wang Junlu, born in 1988. PhD, lecturer. Member of CCF. His main research interests include blockchain technology, big data processing technology, and streaming data processing technology.

王俊陆, 1988 年生。博士，讲师。CCF 会员。主要研究方向为区块链技术、大数据处理技术、流数据处理技术。