# Survey of Blockchain consensus algorithm

- Fu X, Wang H M, Shi P C, et al. A survey of Blockchain consensus algorithms: mechanism, design, and applications. Sci China Inf Sci, 2021, 64(2): 121101, https://doi.org/10.1007/s11432-019-2790-1
- LIU Y Z, LIU J W, ZHANG Z Y, XU T G, YU H. Overview on Blockchain consensus mechanisms[J]. Journal of Cryptologic Research, 2019, 6(4): 395–432.

汇报人：0xDktb

2021.11.13

# 目录

# PART 01 Requirements & Process model

# Requirements & Process model

receiver cannot know when the message will arrive. The majority of the current consensus algorithms are based on either the complete synchronous communication model or the weak synchronous communication model, where a timeout mechanism is set for the transmission of messages. Here, we assume that Blockchain consensus algorithms employ the weak synchronization communication model; that is, a message may be delayed but will eventually reach the receiver within a specific time limit, beyond which, the sending node will be considered as failed. Under this assumption, Blockchain consensus algorithms must guarantee the consistency and liveness properties.

一致性和活性保证

**Consistency.** If a transaction is valid on an honest node, it is also valid on other honest nodes. Consistency ensures that the double spending attack [18] will never succeed in the Blockchain system if honest nodes are in the majority.

**Liveness.** All valid transactions sent by honest nodes must be eventually confirmed, which ensures the availability of the system.

在吞吐量/可扩展性/资源开销/...上对共识算法进行优化

quirements. On the premise of guaranteeing consistency and liveness, other aspects of Blockchain can be designed according to application requirements, such as improving throughput, increasing scalability, and reducing resource costs. Consensus algorithms have been studied for a long time in the field of tra-

# Requirements & Process model

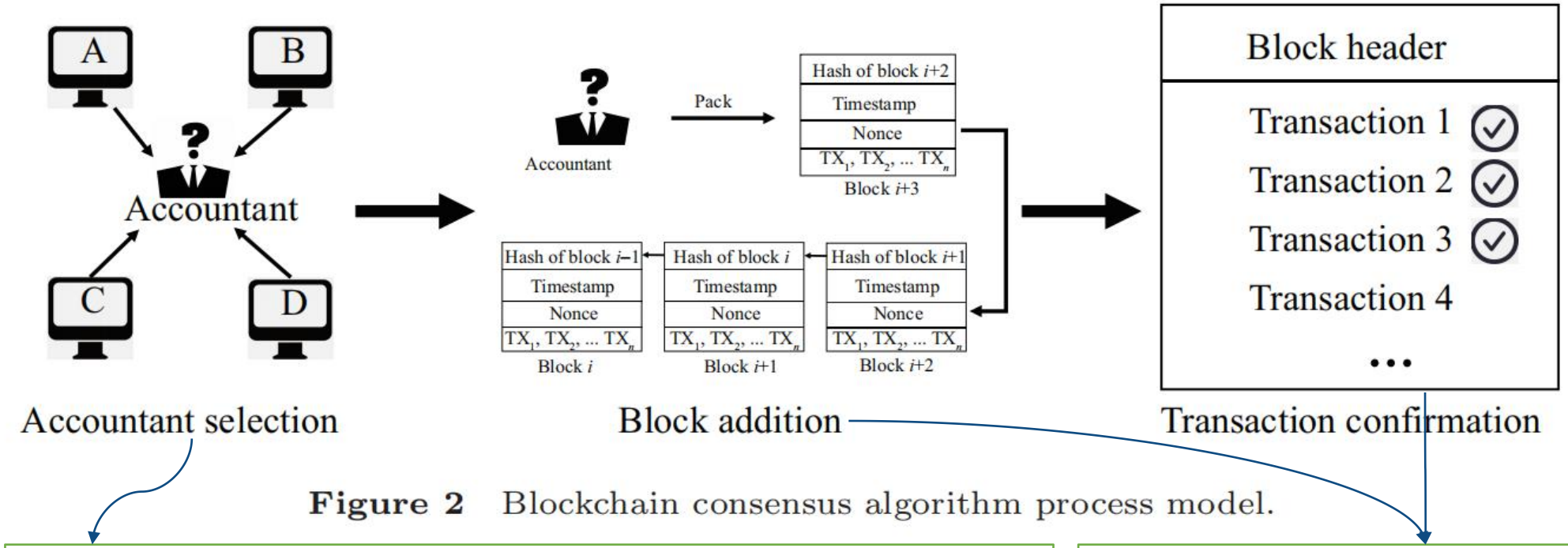



**Figure 2** Blockchain consensus algorithm process model.

## 打包区块权

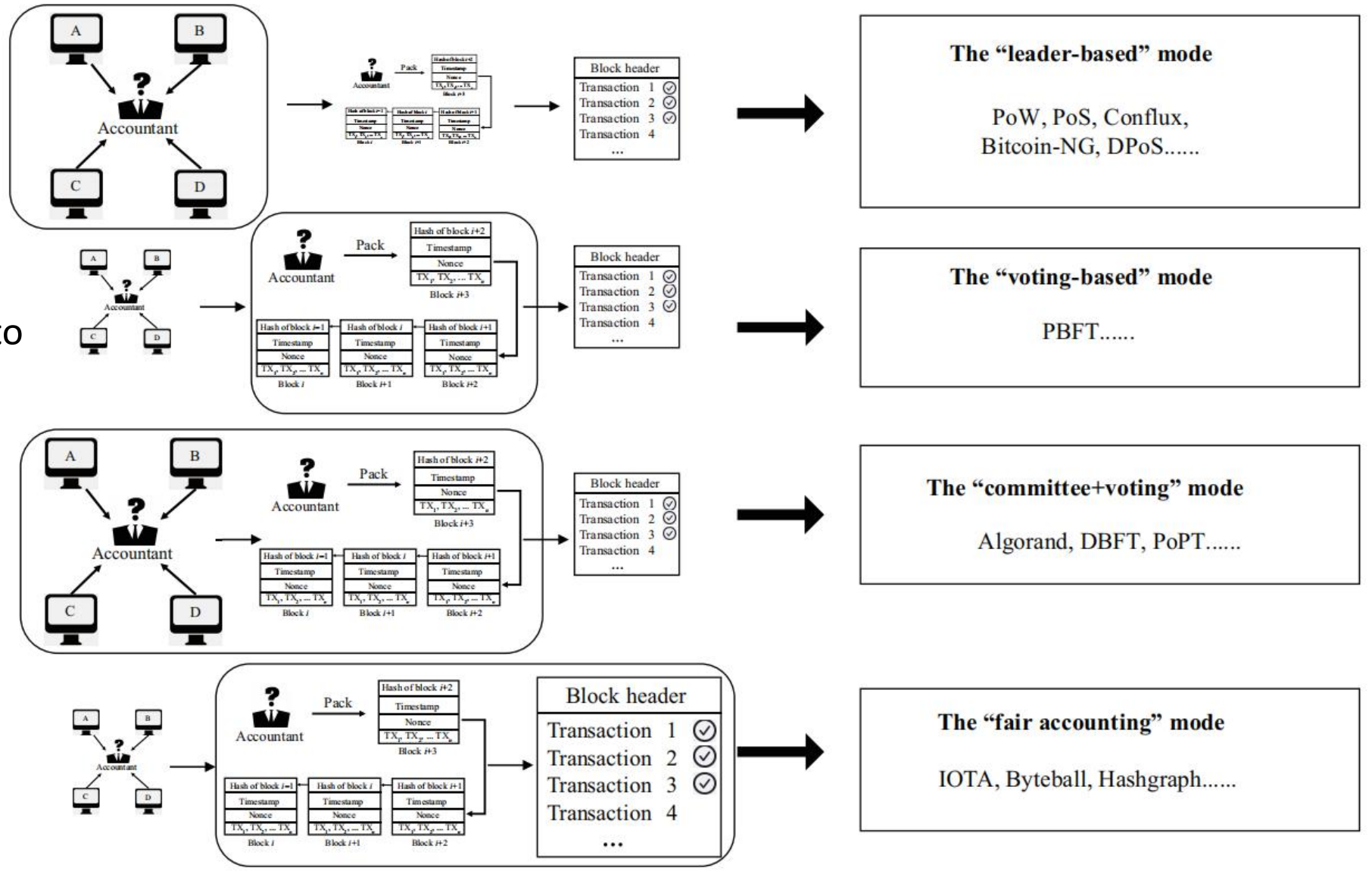
- Randomly choose using PoW/PoS/verifiable random functions(VRF)
- in a predetermined order (e.g. PBFT choose leader)
- Every node is an accountant (always in DAG)

- Validation of accountant and block header (prevHash/Merkle root/...)
- Require votes from nodes (e.g. PBFT)

If without real-time votes from the majority of nodes, confirmation after n blocks is required.
(**链分叉**→延后区块确认)

# Requirements & Process model

Consensus algorithms can be broken down into ***four modes*** as shown.



# PART 02 Classification

# "Leader-based" mode

nodes determines the attribution of the accounting rights of PoW. In the Bitcoin system, which is the first notable application of PoW-based Blockchain, each participating node solves a complex but easily verified mathematical problem based on their respective computing powers in the accountant selection phase. This mathematical problem can be expressed as finding a suitable random number (Nonce) for the block header according to the current difficulty value so that the SHA256 [33] value of the block header is less than or equal to the target value. This process is also known as "mining" [34]. The node solving the problem is first selected as the accountant and may receive an accounting reward. The design of the block addition phase is relatively simple: after verifying that a block was created by the accountant, the block is immediately added to the current Blockchain, if it passes the verification. The transaction confirmation phase follows the longest chain principle; that is, when a fork occurs, the longer chain is considered to be the legal one. If a block has six or more successor blocks, the transactions it contains

**PoW**

All nodes participate in the competition for the accounting right, which leads to a waste of computing power.

Solutions:

① For accountant selection: the mining process in Bitcoin-NG does not include packing transactions（即竞争打包权与真正打包以不同速率工作，出块者负责打包一段时期（←不需要挖矿）内的所有区块）

② 将非最长链上的算力也纳入有效算力（GHOST：最长链 → 最重子树；思路推广至接受深度≥2的孤块 → DAG → e.g. Conflux）

# ”Leader-based” mode

power has a higher probability of obtaining the accounting right. When we talk about the PoS algorithm, we do not refer to a specific algorithm, but the consensus algorithms that rely on the stake to compete for accounting rights. There are several implementations of PoS. In the simplest one, the node with the highest stake directly obtains the accounting rights in the accountant selection phase, or, based on PoW, the mining difficulty changes according to the proportion and time of the token each node holds, thus speeding up the search for Nonce. PoS primarily uses the proof of stake to replace the proof of hash-based procedure in PoW. Thus, the accounting rights are assigned to the nodes with the highest stake rather than the highest computing power. Although this method reduces the waste of computing power, it may have the risk of monopoly, which leads to the centralization trend of the system while allowing malicious attackers to have a clear target to attack, undermining the security of the system. To reduce the monopoly risk of PoS, researchers proposed the DPoS consensus algorithm [36]. In DPoS, the owner of the stake grants the rights to other nodes to vote for accountant candidates, with the top-ranked candidates taking turns in having the accounting rights. The next two phases are the same as in PoS. The PoS and DPoS

基于随机数：Snow White PoS

# ”Voting-based” mode

## PBFT(Pratical Byzantine Fault Tolerance)

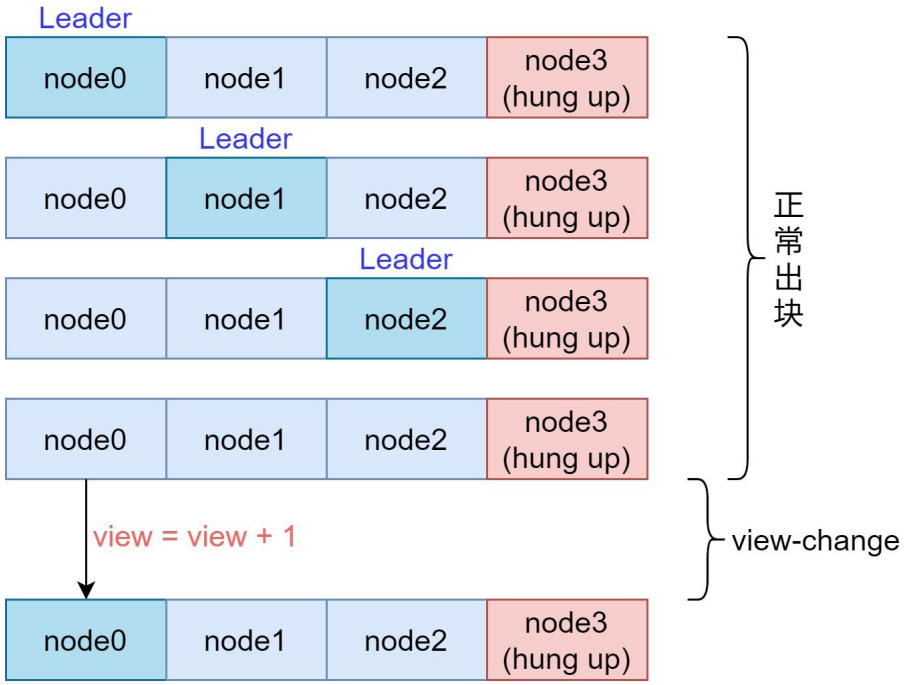
### 节点类型

- Leader/Primary: 负责将交易打包成区块和区块共识，每轮共识过程中有且仅有一个leader，为了防止leader伪造区块，每轮PBFT共识后，均会切换leader
- Replica：副本节点，负责区块共识
- Observer：观察者节点，负责从共识节点获取最新区块，执行并验证执行结果后，将区块上链

其中Leader和Replica共称为共识节点

### Accountant Selection

*leader_idx = (view + block_num) % node_num*



# ”Voting-based” mode

PBFT(Pratical Byzantine Fault Tolerance)

**共识消息**

- PrepareReqPacket: 包含区块的请求包，由leader产生并向所有Replica节点广播，Replica收到Prepare包后，验证PrepareReq签名、执行区块并缓存区块执行结果；
- SignReqPacket: 带有区块执行结果的签名请求，由收到Prepare包并执行完区块的共识节点产生，SignReq请求带有执行后区块的hash以及该hash的签名，分别记为SignReq.block_hash和SignReq.sig；
- CommitReqPacket: 用于确认区块执行结果的提交请求，由收集满(2f+1)个block_hash相同且来自不同节点SignReq请求的节点产生；
- ViewChangeReqPacket: 视图切换请求，当leader无法提供正常服务(如网络连接不正常、服务器宕机等)时，其他共识节点会主动触发视图切换；

# ”Voting-based” mode

## PBFT(Pratical Byzantine Fault Tolerance)

### 共识消息

- PrepareReqPacket: 包含区块的
  PrepareReq签名、执行区块并
- SignReqPacket: 带有区块执行
  有执行后区块的hash以及该has
- CommitReqPacket: 用于确认区
  SignReq请求的节点产生；
- ViewChangeReqPacket: 视图
  识节点会主动触发视图切换；

这四类消息包包含的字段大致相同，所有消息包共有的字段如下：

| 字段名 | 字段含义 |
| --- | --- |
| 字段名 | 字段含义 |
| idx | 当前节点索引 |
| packetType | 请求包类型(包括PrepareReqPacket/SignReqPacket/CommitReqPacket/ViewChangeReqPacket) |
| height | 当前正在处理的区块高度(一般是本地区块高度加一) |
| blockHash | 当前正在处理的区块哈希 |
| view | 当前节点所处的视图 |
| sig | 当前节点对blockHash的签名 |

PrepareReqPacket类型消息包包含了正在处理的区块信息:

| 消息包类型 | 字段名 | 字段含义 |
| --- | --- | --- |
| PrepareReqPacket | block | 所有共识节点正在共识的区块数据 |

# "Voting-based" mode

## PBFT(Pratical Byzantine Fault Tolerance)

be added to the Blockchain. In the first pre-prepare stage, the accountant sends a block to other nodes for them to verify it (we call the accountant as the primary and the other nodes as the replicas). In the second prepare stage, each replica sends the verification result to all other nodes, and all nodes re-confirm the block according to the message sent by each replica (if there are more than $2f + 1$ "YES" votes, the block is valid). In the third commit stage, each node sends the verification result of the prepare stage to all other nodes again, and each node makes a final confirmation of the block according to the received messages.

Q: 执行后BlockHash change，
猜测类似Ethereum维护动态Merkle Trees?

- Pre-prepare: leader从TxPool取交易进行打包新块，并将该新块编码到PrepareReqPacket广播给所有Replica;
- Prepare: 判断PrevHash/...（若Tx.num=0，即空块，触发view-change），合法后执行区块并缓存执行后的区块，产生SignReqPacket并广播;
- Commit:
  判断收到SignReqPacket的hash是否与Prepare阶段缓存的相同（合法性），若不同还需判断是否未来块;
  缓存合法的SignReqPacket；若收集满2f+1则广播Commit包;

  - **若收集满签名包，备份prepare阶段缓存的Prepare包落盘**：为了防止Commit阶段区块未提交到数据库之前超过 2*f+1 个节点宕机，这些节点启动后重新出块，导致区块链分叉(剩余的节点最新区块与这些节点最高区块不同)，还需要备份prepare阶段缓存的Prepare包到数据库，节点重启后，优先处理备份的Prepare包。

  Commit包hash合法性判断；缓存合法的Commit包；
  同样，若收集满2f+1则将缓存的区块数据写入数据库（落盘）；

# ”Voting-based” mode

PBFT(Pratical Byzantine Fault Tolerance)

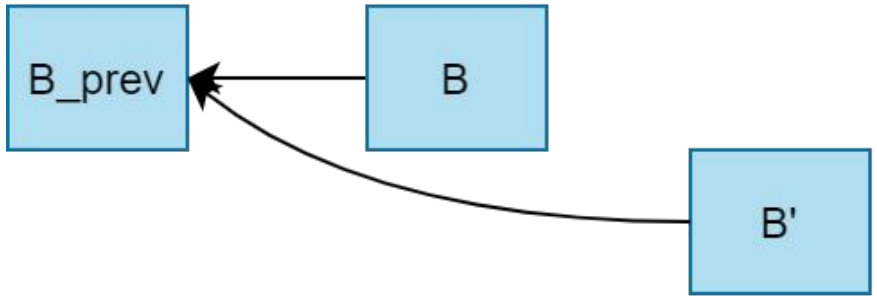
※ Why is the COMMIT phase in PBFT necessary?

假设Prepare phase后即可确认

诚实节点a收到2f+1个关于区块B的SignReqPacket，到达“Prepared”状态，于是a将区块B上链

但此时发生了***大规模的网络异常*** → 没有其他诚实节点收到2f+1个区块B的SignPacket

因此timeout触发view-change，新一轮公式中其他诚实节点确认了不同的区块B'，导致一致性失效




**增加COMMIT phase：**

则在view-change后，由于诚实节点a收不到2f+1个关于区块B的CommitPacket，但收到2f+1个关于区块B’的SignReqPacket时，于是将抛弃对区块B的预备，而切换到B’上，保证了一致性和活性

*陷入相同问题*

**考虑以下场景：**

但如果诚实节点a收到2f+1个关于区块B的CommitPacket，于是a将区块B上链

而在此时又发生了大规模的网络异常 → 其他诚实节点未达到”committed-local”就因timeout发起了view-change

再一次造成了一致性的失效

# ”Voting-based” mode

PBFT(Pratical Byzantine Fault Tolerance)

PBFT中采用view-change时同时向其他节点广播自己状态的方法，即发送自己最新状态的一条PrepareReqPacket和对应的2f+1条SignReqPacket

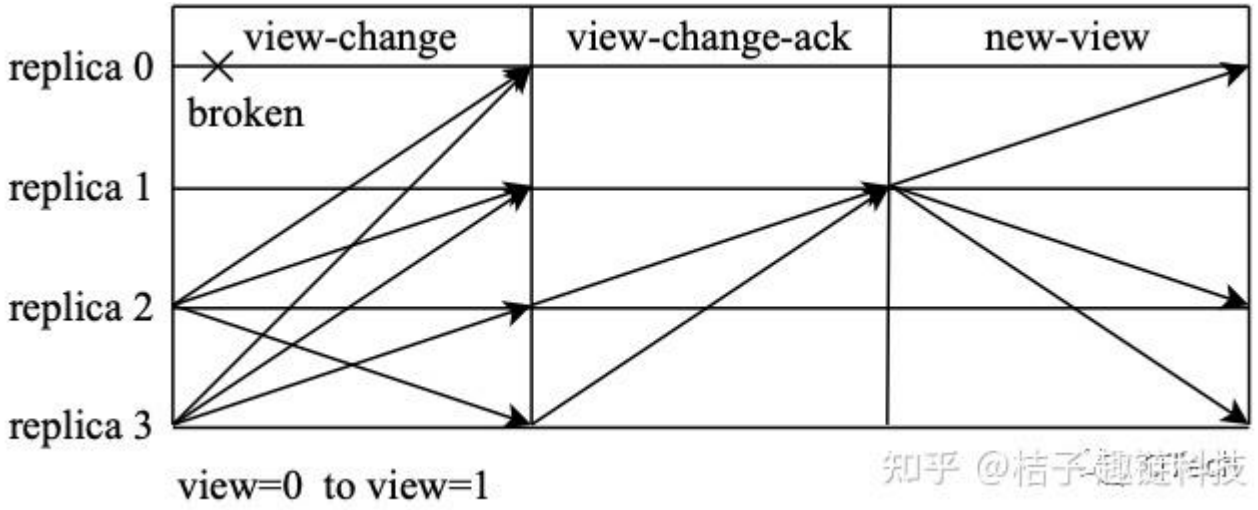
新的leader就根据收到信息中的状态决定是否要替换掉先前的区块B

**考虑以下场景：**

但如果诚实节点a收到2f+1个关于区块B的CommitPacket，于是a将区块B上链

而在此时又发生了大规模的网络异常 → 其他诚实节点未达到”committed-local”就因timeout发起了view-change

再一次造成了一致性的失效




一般情况下为O(n^2)

而因为view-change时

消息大小为O(n) →

切换视图时的复杂度为O(n^3)

# ”Voting-based” mode

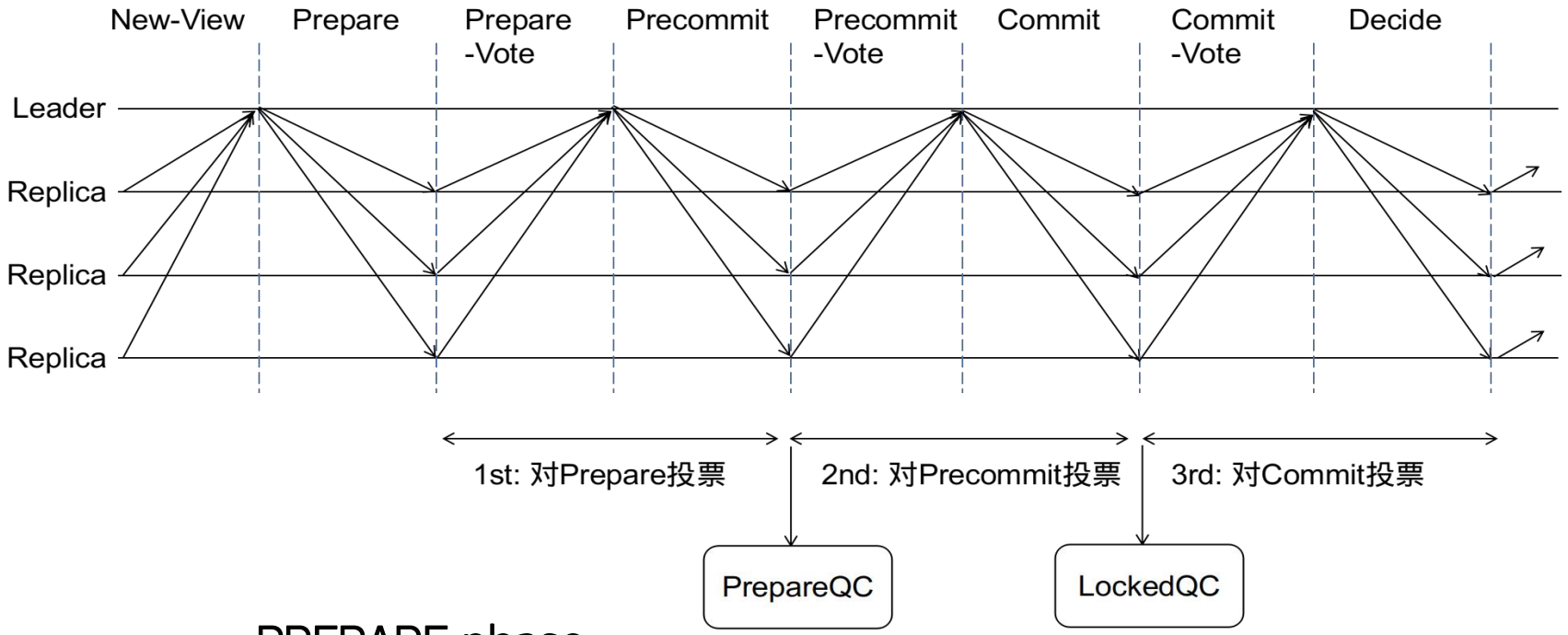
consensus algorithms in its version 0.6 and 1.0, respectively. To improve scalability, VMware and IBM proposed Hotstuff [42] and MirBFT [43], respectively. Hotstuff is a leader-based Byzantine fault-tolerant replication protocol for the partially synchronous model. Messages are sent only between the leader and replicas, and the leader collects votes from replicas. The time complexity of Hotstuff is reduced to $O(n)$.

HotStuff

- 门限签名
- Quorum Certificate, QC: leader收到至少quorum个节点的带签名vote后，聚合成的完整签名，表示对该次提案达成一次共识；
- View: 一个视图至多达成一次共识；
- State Machine Replication, SMR: 系统的每个节点都有着相同的SMR；
- prepareQC: 对于某个prepare信息，leader收集齐quorum个节点签名后聚合成的完整签名（第一轮投票达成的证据）；
- lockedQC: 对于某个precommit信息，leader收集齐quorum个节点签名后聚合成的完整签名（第二轮投票达成的证据）；

# ”Voting-based” mode

HotStuff




PREPARE phase:

leader收集quorum个replica发送的NEW-WAVE包（包含发送节点上最高的prepareQC）

leader广播prepare包（其中包含新区块和highQC，highQC = max(prepareQCs)）

replica验证(1)prepare包中的新区块是本机lockedQC的子孙节点;(2)highQC.view > lockedQC.view（未来区块，保证活性）中的一个条件满足即可，生成prepare-vote包并附带签名发送给leader；（区别于PBFT的广播）

# ”Voting-based” mode

HotStuff

PRE-COMMIT phase:

leader收到quorum个prepare-vote包时，通过聚合签名得到prepareQC
缓存在本地并将其加入precommit包广播给replica

replica收到precommit包并验签成功后，对其签名加入precommit-vote包发送给leader

COMMIT phase:

leader收到quorum个prepare-commit包时，通过聚合签名得到precommitQC
缓存在本地并将其加入commit包广播给replica

replica收到commit包并验签成功后，将lockedQC更新为precommitQC，对其签名生成commit-vote包发送给leader

分割线后的部分为HotStuff提出的*三阶段共识*

DECIDE phase:

leader聚合得到commitQC，并加入decide包广播给replica;
replica收到commitQC后，执行已确定分支上的交易，并令view_num自增1，开启新一轮共识

# ”Voting-based” mode

HotStuff

⚠注意，HotStuff中一笔交易从开始到提交进行了三轮共识，第三轮共识的加入，克服了经典两阶段范式的共识算法扩展的瓶颈。PBFT中为了保证系统在遇到恶意节点时能继续工作（活性），需要进行视图切换，而新视图中为了确定上一个视图中的区块是否达成共识，需要在view-change消息中附带自己收集到的quorum个prepare消息和相对应的一个pre-prepare消息作为证明，然后每个节点广播view-change消息请求视图切换，此时广播的消息复杂度o(n^2)，消息的量级为o(n)，因此视图切换的复杂度为o(n^3)。安全性和活性让PBFT需要o(n^3)的通信复杂度，对于网络的负载极大，限制了其向大规模网络的扩展。而HotStuff中如果我们对某一区块达成了两轮共识，在更换主节点时便能确定，主节点只需要基于最新的两轮共识节点产生新节点是安全的。换句话说，只需要根据**区块自身的状态**（经过几轮共识）就可以确定是否在新的视图中基于该区块打包块新区块，降低了在视图切换时候的通信复杂度。

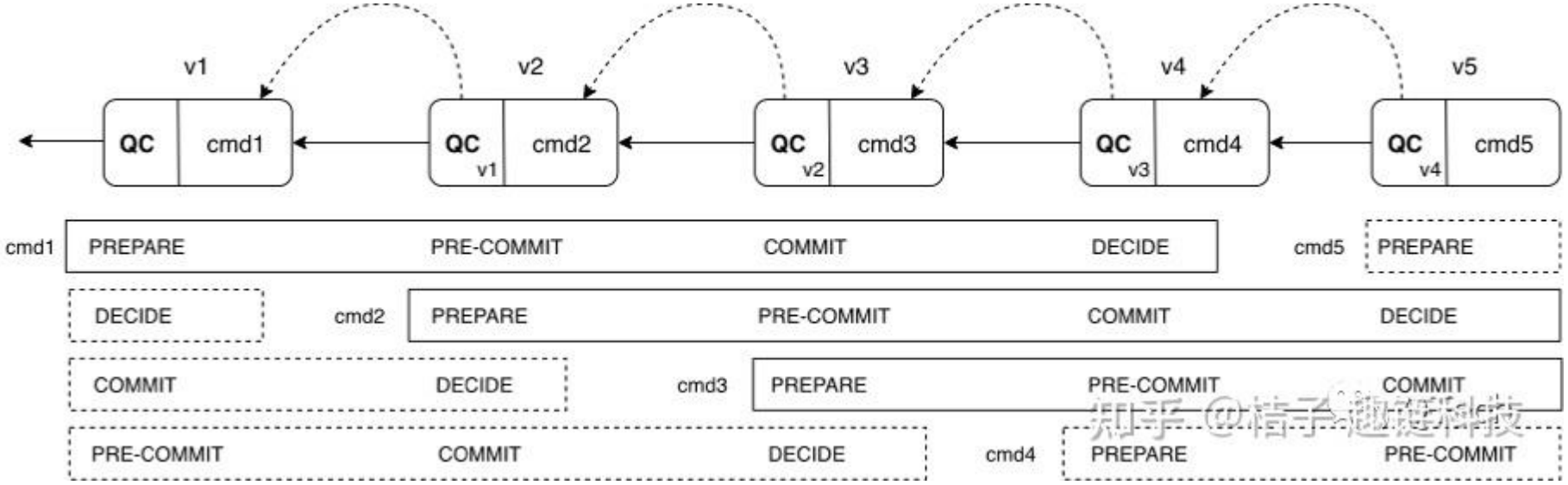
*why需要三阶段共识*

因此HotStuff在设计上节点轮换时（一般timeout触发），replica广播自己的状态并申请更换leader；
leader只需判断 **是否有quorum个节点处于“经过两轮或更高共识”的状态**

if num < quorum then
　不可能有一个诚实节点进入确认交易并执行阶段，因此新leader出的块在区块B_prev处分叉；
else
　不可能有一个诚实节点确认了一条不包括区块B的链，因此新leader出的块在区块B的子孙节点中；

# ”Voting-based” mode

## Chained HotStuff




Chained HotStuff中每个阶段都会切换视图，可以看到：
PREPARE phase的prepare-vote包发给下一视图的leader，
即下一视图的leader在处理本视图PREPARE phase的同时，也在处理上一视图的PRE-COMMIT phase
以此类推。

# ”Committee+Voting” mode

Focus on the accountant selection and block addition phases

DBFT

algorand. In DBFT, there are two types of nodes: the accounting node and ordinary nodes. The ordinary nodes vote to choose the accounting nodes based on the proportion of the stake they hold. The committee consists only of accounting nodes. In the accountant selection phase of DBFT, just like in PBFT, the accountant who generates the block is selected from the accounting nodes through polling. In the block addition phase, the committee runs the PBFT consensus. The PoPT consensus algorithm ranks

**Difference: Construct Committee**

once the block is added to the Blockchain in the above three algorithms. The core idea of consensus algorithms of the committee + voting mode is to narrow the range of nodes participating in the consensus process, thereby reducing the communication complexity. Moreover, there is no sacrifice of the system's

# ”Fair accountant” mode

The majority of the Blockchains with the consensus algorithms of the “fair accounting” mode use DAG-based data structures. In the accountant selection phase, each node is an accountant, which means that a node is only responsible for packing the transactions it generates. Therefore, every node has a fair chance to be an accountant. In the block addition phase, there are always some block-linking rules for

## Challenges for DAG:

- **定序问题**(e.g. Conflux → Reference Edge indicate which block generated first)
- **重复交易问题**(responsible for own transactions, instead of TxPool, e.g. IOTA)

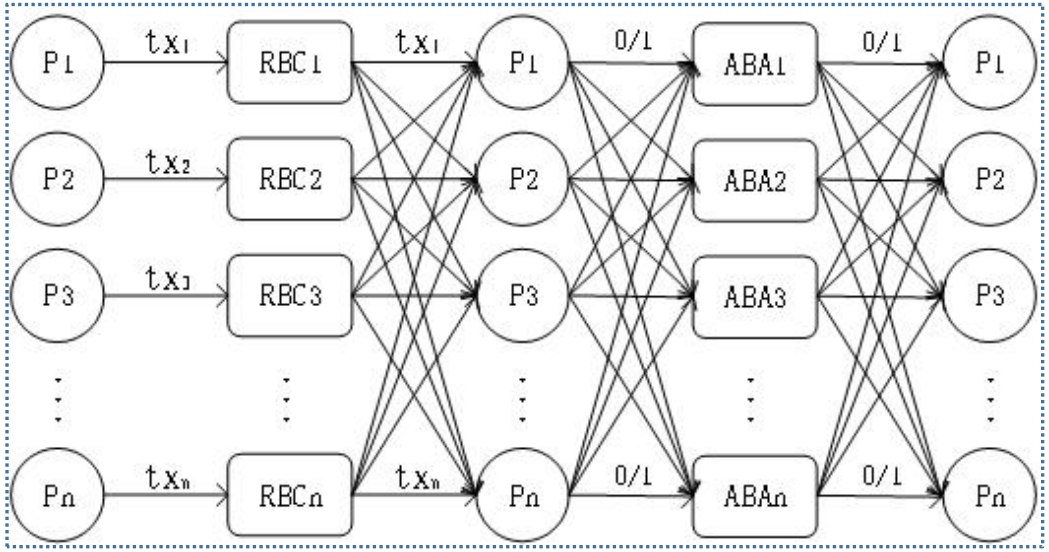
# ”Fair accountant” mode

HB-BFT(Honey Badger BFT) ⟸ *The first **asynchronous** BFT*

1. 每个节点从本地交易池（buf）中随机选取$\lfloor B/N \rfloor$个交易，作为$proposed$。并使用门限加密模块Setup后的共享公钥$PK$对$proposed$进行加密$\rightarrow x$；
2. 每个节点都并行执行$\{RBC_i\}_N, \{ABA_i\}_N$，组成ACS模块。将$x$作为各自ACS模块的输入（实际上对于节点$\mathcal{P}_j$，即作为本地$RBC_j$的输入，见Figure_4），得到系统共识“YES”后的*enc_proposed*集合；
3. 对Step 2输出的*enc_proposed*集合进行遍历处理：广播自己的秘密份额$e_j$，并通过f+1份share message解密得到明文交易集；
4. 排序交易集，写入区块（但官方论文中的e.g. lexicographically有点迷惑哈...区块交易可以这么定序的嘛❓）
5. 从本地交易池中remove已确认交易；

RBC协议（可靠广播）
- 基于Merkle Tree Root验证
- 基于Reed-Solomon纠删码

ABA协议（异步二进制协定）
- 引入基于门限加密的随机源



# ”Fair accountant” mode

## asynchronous BFT:

- Hashgraph
- Dumbo BFT (✔ better than Honey Badger BFT)

# PART 03 Comparasion

# Comparasion of consensus algorithms

To sum up, the scalability of the leader-based mode is high, while there is a fairness problem, and usually, the efficiency is low. In the voting-based mode, transactions can be confirmed once they are added to the Blockchain, and the confirmation delay is low. However, there is a network synchronization is needed, communication overheads are considerable, and scalability is low. The committee + voting mode combines the advantages of the former two. However, when the number of committee members is large, there is a problem of large communication overhead. The fair accounting mode (mostly based on DAG data structures) has low resource consumption, its scalability and efficiency are high. However, there is a centralization risk, and the storage cost is always higher than that of the chain-structure Blockchains. It is mostly used in consortium Blockchains. A brief description of the three-phase process

# Thx

汇报人：0xDktb

2021.11.13