1. 共识算法
   1. 概述

共识算法作为分布式计算中的关键问题之一，是指分布式系统环境下大部分节点就某一提案达成一致性意见的过程，来保证分布式系统数据的一致性、不可篡改和不可伪造性，其中提案可能是多个事件发生的顺序、交易数据等任何可以达成一致的信息。

在分布式计算中，如何将数据按照正确的顺序打包并提交至区块链上，保证不同节点数据的一致性，如何让多个节点对该提案达成共识已成为研究的重点。

理想情况下，如果分布式系统上的所有节点都能以无网络阻塞、不发生节点故障且诚信投票等理想情况稳定运行，则实现共识过程并不困难，但现实情况往往会出现网络阻塞、伪造信息等错误， 因此我们需要在容许部分错误的情况下实现共识。

一般地，把出现故障但不会伪造信息的情况称为非拜占庭错误或故障错误，把能伪造信息响应的错误称为拜占庭错误。根据解决的场景是否允许拜占庭错误情况，共识算法分为故障容错算法（CFT）和拜占庭容错算法（BFT）。对于非拜占庭错误的情况下，已经存在不少经典的算法，如Paxos 、Raft等，能够容忍不超过一半的故障节点，性能较好。而拜占庭容错算法，由于可能出现伪造信息人为破坏正常共识的情况，达成共识将更加困难，现在包括PBFT 等，能够容忍不超过1/3的拜占庭节点（一般在拜占庭容错算法下，拜占庭节点是指包括伪造信息响应的节点和出现故障的节点）。

然而，共识问题也存在理论界限，即使在网络通信可靠的情况下，分布式系统的共识问题也无通用解，此理论被称为FLP不可能理论：在网络可靠，但允许节点失效（即便只有一个）的最小化异步模型系统中，不存在一个可以解决一致性问题的确定性共识算法。此理论说明了为异步分布式系统设计面向任意场景的共识算法的不可能性，其中异步、同步以及弱同步的概念如下：

**同步:** 是指系统中的各个节点的时钟误差存在上限；并且消息传递必须在一定时间内完成，否则认为失败；同时各个节点完成处理消息的时间是一定的。因此同步系统中可以很容易的判断消息是否丢失，节点所发出的消息，在一个确定的时间内，肯定会到达目标节点；

**异步:** 是指系统中各个节点可能存在较大的时钟差异；同时消息传输时间是任意长的；各节点对消息进行处理的时间也可能是任意长的；节点所发出的消息，不能确定一定会到达目标节点；

**弱同步：**节点发出的消息，虽然会有延迟，但是最终会到达目标节点。

但是现实情况下，往往可以付出一些代价能够使共识算法在特定情况下达到可行。CAP理论阐述了分布式系统无法同时确保一致性（Consistency）、可用性（Availability）和分区容忍性（Partition），因此设计共识算法时往往需要弱化对某个特性的需求。

**一致性（Consistency）：**任何事务应该都是原子的，所有副本上的状态都是事务成功提交后的结果，并保持强一致；

**可用性（Availability）：**系统（非错误节点）能在有限时间内完成对操作请求的应答；

**分区容忍性（Partition）：**系统中的网络可能发生分区故障（成为多个子网，甚至出现节 点上线和下线），即节点之间的通信无法保障。而网络故障不应该影响到系统正常服务。

此外，科学的共识算法还应该满足以下四个特性：

**Agreement ：** 每个节点必须达成一致，这是实现一致性的必然要求

**Validity:** 如果所有正确节点都提出相同的value , 那么任何正确节点必定能够确定这value , 为了防止无论提交什么值有些节点一律不同意的情况，节点能验证块的正确性，并在其正确的时候表示赞同。

**Integrity：** 保证共识得到一致性不可逆，不可被修改，即节点不能对对一阶段进行两次投票。

**Termination:** 系统（包含错误节点）能在有限时间内完成对操作请求的应答，而不处于无限循环或等待的状况

* 1. 拜占庭容错算法

虽然故障容错算法在机器故障或者节点失效等因素的情况下达成一个分布式系统的共识，容错性能高，但由于分布式系统的现实情况错综复杂，往往也需要考虑到人为的恶意伪造和恶意篡改等行为，而拜占庭容错作为分布式计算领域的容错技术，能够容忍机器以任意方式出现故障，包括机器故意危害系统的方式。因此拜占庭共识算法能够更好地符合现实情况，而拜占庭共识算法最早是由拜占庭将军问题引发而来。

**拜占庭将军问题：**拜占庭罗马帝国国土辽阔，为了达到防御目的，每个军队相隔很远，将军与将军之间只能靠派遣差使来传递消息。在战争的时候，拜占庭军队内所有将军必须达成一致的共识，但由于将军之间可能存在叛徒或间谍并传播假的消息，干扰共识的达成，或者差使在路上可能被敌方截获，使得消息无法正常传递，这就是拜占庭将军问题。

拜占庭将军问题是对现实世界的模型化，由于硬件错误、网络拥塞或中断以及遭到恶意攻击等行为，计算机或网络出现不可预料的行为，而区块链网络环境符合拜占庭将军问题模型。虽然拜占庭容错算法用来解决拜占庭将军问题，但拜占庭共识算法不考虑通信兵被截获导致无法传递消息的情况，即消息传递的信道绝无问题，否则拜占庭问题无解。

根据Leslie Lamport 等人在论文《Reaching agreement in the presence of faults》的证明，对于拜占庭问题来说，假如节点总数为 N，故障节点数为 F，则当 N >= 3F + 1 时，问题才能有解。

计算过程如下：

假定有F 个叛变者和 L=N-F 个忠诚者，叛变者故意使坏，可以给出错误的结果，也可以不响应。某个时候 F个叛变者都不响应，则N-F个忠诚者取多数即能得到正确结果。但当F个叛变者都给出一个恶意的提案，并且L个忠诚者中由于网络延迟导致f个恶意节点的消息先被收到，剩下的 L - F 个忠诚者此时无法判断是否混入了叛变者，仍然要确保取多数能得到正确结果，因此，L - F > F，即 L > 2F 或 N - F > 2F，所以系统整体规模 N 要大于 3F，即N最少是3F+1。

例如，当N = 3，F = 1 时，若提案人不是叛变者，提案人发送一个提案出来，收到的叛变者可以宣称收到的是相反的命令。则对于第三个人（忠诚者）会收到两个相反的消息，无法判断谁是叛变者，则系统无法达到一致。若提案人是叛变者，发送两个相反的提案分别给另外两人，另外两人都收到两个相反的消息，无法判断究竟谁是叛变者，则系统无法达到一致。

更一般的，当提案人不是叛变者，提案人提出提案信息 1，则对于合作者来看，系统中会有 N - F 份确定的信息 1，和 F 份不确定的信息（可能为 0 或 1，假设叛变者会尽量干扰一致的达成），N − F > F，即 N > 2F 情况下才能达成一致。

当提案人是叛变者，会尽量发送相反的提案给 N - F 个合作者，从收到 1 的合作者看来，系统中会存在 (N - F)/2 个信息 1，以及 (N - F)/2 个信息 0；从收到 0 的合作者看来，系统中会存在 (N - F)/2 个信息 0，以及 (N - F)/2 个信息 1；另外存在 F − 1 个不确定的信息。合作者要想达成一致，必须进一步的对所获得的消息进行判定，询问其他人某个被怀疑对象的消息值，并通过取多数来作为被怀疑者的信息值。这个过程可以进一步递归下去。

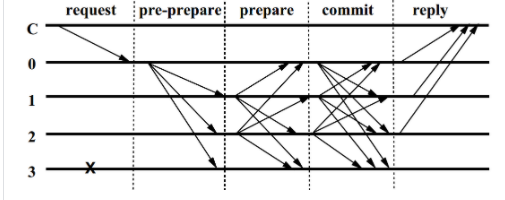
* 1. PBFT

PBFT作为一种比较典型的拜占庭容错算法，于1999年被发明却一直未被学术界使用，直到2008年中本聪将网络规模级别的分布式拜占庭容错算法设计到区块链方案中才使拜占庭容错得到推广。

PBFT 假设系统是弱同步的，节点通过网络连接，消息会被延迟，但是不会被无限延迟,。PBFT中参与共识过程的节点包括一个主节点和多个备份节点，当某个节点作为主节点时，称为系统的一个视图（view），当节点出现问题时，就进行试图更换，切换到下一个节点担任主节点，主节点的更替采用round-robin方式。

PBFT共识过程由三个阶段构成，分别为pre-prepare、prepare和commit。pre-prepare 阶段和prepare 阶段确保了在同一个视图下，正常节点对于消息m 达成了全局一致的顺序，在当前视图下，正常节点都会对消息m确认一个序号n。接下来的commit投票，如果commit投票失败，则在切换视图的过程中附带序号消息m的序号n，保证对于m的全局一致顺序，即视图切换后, 依然会对消息m，对同一序号n进行确认。

具体流程如下：



1. 客户端发送请求m 给主节点，主节点将请求m 广播给其他节点，同时给请求m 分配一个序号n 并广播给其他节点，产生pre-prepare 消息，进入共识流程。pre-prepare 阶段的消息格式<<PRE-PREPARE , v , n , d>p ,m> ，其中v 表示当前视图编号，n 表示给m 分配的序号，d 为m 的哈希，以及m 的原文。
2. 其他节点收到pre-prepare 消息后，验证该消息是否包含所在视图v 的提议节点的签名，且在当前视图没有收到序号为n 的其他消息，验证通过后则接收该消息，并广播prepare 消息进入prepare 阶段，同时将消息加入到本地的日志中
3. 节点收到prepare 消息时，会验证提议节点对消息的签名并检查是否是当前视图v 的消息，同时检查消息序号n 是否在当前的接收窗口内，验证通过则接受该消息，保存到本地log 中，当节点日志中存在m 的pre-prepare 消息和2f个来自其他节点的prepare 消息，则进入prepared 状态。至此，可以确保在视图不发生切换的情况下，对于消息m 有全局一致的顺序，即
4. 一个正常节点，不能对两个及以上的不同消息，达成相同序号n 的prepared 状态。
5. 两个正常节点，必须对相同的消息m ，达成相同序号n 的prepared 状态。

当节点达成prepared 状态后，节点向网络广播commit 消息。

1. 节点接收commit 消息后，验证提议节点对区块的签名并检查是否是当前视图v 的消息，同时检查消息序号n 是否在当前的接收窗口内。当节点达成prepared 状态，并且收到了2f+1个commit 消息，则该节点就可以执行该消息并响应客户端。

当节点检测到超时或主节点的恶意行为时，会发送视图更换消息，进入视图更换流程，节点发出视图更换消息时，将本地的prepared状态信息也打包到了消息中，传递给新的视图。当新视图所对应的主节点收到了2f 个有效的视图更新消息，它就会广播消息进行通知所有验证节点进行视图更换；

在新的视图中，验证节点是在上一轮视图中各个节点的prepared 状态基础上进行共识流程的。发生视图转换时，如果视图转换之前的消息m 被分配了序号n, 并且达到了prepared 状态，那么在视图切换之后，为了保证切换前后的Safety 特性，即在一次执行中只批准一个最终决议，该消息也将同样被分配序号n 。因为达到prepared 状态以后，就有可能存在某个节点收到超过 2/3的commit 消息，其他节点可能因为网络延迟而超时发起视图更换消息。要保证对于消息m 只允许一次提交，在视图转换之后，其他节点提交的消息应该是一样的序号。

* 1. Tendermint共识过程

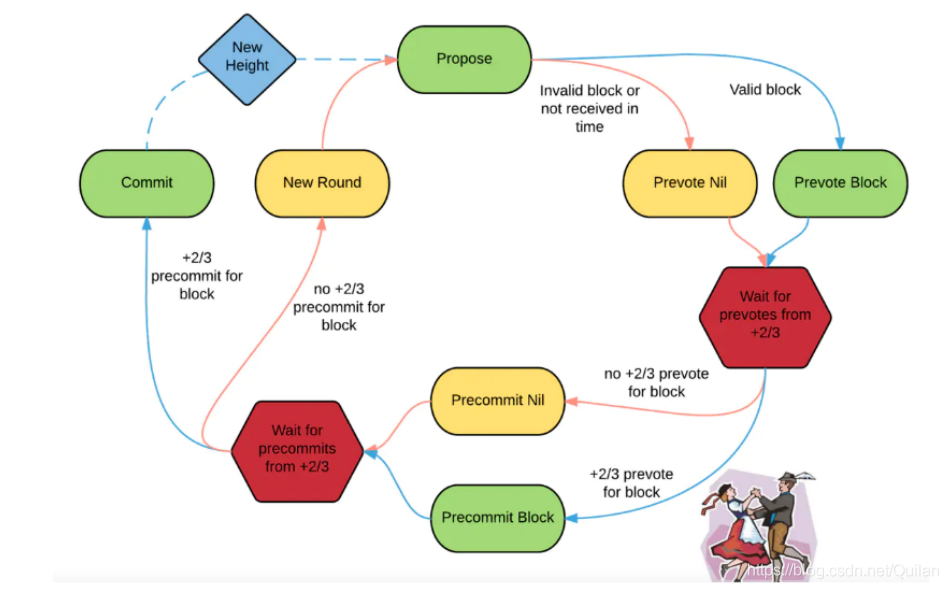
Tendermint是个能够在多机器上安全一致地复制应用的软件，是由Jae Kwon 在2014首次提出。其中，安全是指无论多达1/3的机器出现何种类型的故障，Tendermint 都可以正常工作。 一致是指每个正确运行的机器都可以获取完全相同的交易日志并计算相同的状态。

在许多系统中，验证者在共识协议中可能具有不同的投票“权重”。在tendermint 中，验证者在共识协议中可能具有不同的投票权重。因此，tendermint 并不关注验证者数目的三分之一或三分之二，而是关注总投票权的比例。此外，这个比例可能不是在各个验证者中均匀分布的

参与tendermint 共识过程的节点称为验证节点，将交易打包区块的节点称为提议节点，提议节点的选举采用weighted round-robin 算法，验证节点根据所占股权轮流担任提议节点。Tendermint 同样采用三阶段协议的BFT 共识算法，分别为预投票(prevote )、预提交 (precommit)、提交(commit)。但与PBFT三阶段共识不同的是，tendermint 引入了锁机制，每个节点只能对锁定的区块进行预提交和预投票。

Tendermint 的共识算法中高度(Height)指的是对成功提交某一区块的过程，如果对某一区块提交不成功，则高度不会增加。轮次(Round)指的是同一区块的共识过程可能需要进行多轮才能最终提交

具体流程如下：



1. Propose

当收到来自客户端的交易数据时，当前轮的提议节点将交易进行打包，并通过gossip协议向对等方广播提议，如果提议者被锁定在先前轮次中的某个区块上，则它提议被锁定的区块，并在提议中包含一个锁证明polc。其中提议区块包含高度和轮次等信息

如果验证节点被锁定在某个先前轮次的提议区块上，则它签名并广播锁定区块的预投票，进入prevote状态；如果没有锁定在某一区块，则对当前回合提议节点提议的区块进行签名并广播该提议区块的预投票，进入prevote状态；如果验证者不接受该提议或者超时后收到提议，则向网络中广播空区块nil，进入prevote状态。

1. Provote

当验证节点收到任何超过2/3的预投票后，会开启本地时钟并等待一段时间，如果验证者收到了对一个特定可接受块超过2/3的预投票，将它们打包成锁证明polc ，锁证明中包含的是超过2/3的预投票消息集合，其次验证者签名并广播该块的预提交，然后锁定在该块上并释放先前的锁，此过程被称为一个波尔卡，并进入预提交状态。如果该节点收到了超过2/3的对空区块的预投票，将它们打包成锁证明polc，同样对先前锁定的区块进行解锁，但不锁定在空区块上，然后对空块进行预提交，进入Precommit状态。否则，超时后某个节点未收到针对特定块（或无）的超过2/3的预选票，则它不会签名或锁定任何东西，对空区块的预提交进行广播，进入precommit状态。

1. Precommit

当验证节点收到任何超过2/3的预投票消息后将等待timeoutPrecommit时间；如果收到一个特定可接收块X的超过+2/3的预提交，则将该可接收块进行提交并进入新的高度；如果收到超过2/3空区块的预提交，进入新的一轮；否则，超时后验证节点未收到针对特定块的超过2/3的预提交，进入新的轮次。

参考文献

[1] The latest gossip on BFT consensus Ethan Buchman, Jae Kwon and Zarko Milosevic, September 24, 2018

[2] 区块链技术指南 邹均 张海宁 唐屹 李磊等

[3] 区块链共识机制之拜占庭算法 高丽芬 胡全贵 中国矿业大学(北京) 北京国网信通埃森哲信息技术有限公司

[4] 异步环境下的拜占庭共识算法研究 翁良 西安电子科技大学

[5] 基于区块链技术的共识算法研究 郭雨 吉林建筑科技学院