第3章

CHAPTER 3

区块链共识技术

区块链网络中的每个节点都拥有这个巨大账本，在缺乏第三方监管机构的情况下，当网络中的大部分人都拥有书写账本的权利时，区块链如何保持账本内容的一致性和内容不被恶意篡改呢？共识机制就是区块链中节点就区块信息达成全网一致共识的机制，保证最新的区块信息被准确添加至区块链、节点存储的区块链信息一致，并能够抵御恶意攻击。共识算法是区块链这个分布式账本能够实现的灵魂所在。

随着区块链技术的不断发展，不同的共识算法更新迭代，传统的如工作量证明和权益证明等算法已经通过实践检验了其有效性；而新的共识算法，不仅包括竞争类共识算法和选举类共识算法，还有基于有向无环图的共识算法，其出现也改变了区块的链式存储结构，并实现了无区块的概念。本章内容将对上述概念进行详细描述。

3.1 一致性与共识机制

何为一致性？具体而言，是指针对分布式系统中不同节点，给予一定操作，在约定协议的保障下，试图使得它们对处理结果达成“某种程度”的认同。在理想情况下，如果各服务节点严格遵循相同的处理协议，构成相同的处理状态，给定相同的初始状态及输入序列，则可以保证在处理过程中的每

个环节结果均相同。值得注意的是，一致性并不代表结果正确与否，而是系统对外呈现的状态是否一致，如果所有节点都达成失败状态也是一种一致性。对于区块链系统，想要达成一致性结果，必须满足：

（1）可终止性——结果在有限时间内能够完成。

（2）约同性——不同节点最终完成决策的结果是相同的。

（3）合法性——决策的结果必须是由某个节点提出的提案。

要实现绝对理想的严格一致性（strict consistency）的可能性不大，除非系统不发生任何故障，同时所有节点之间链接无须耗费时间，整个系统实质上等同于一台机器。从实际情况出发，越强的一致性往往会造成越弱的处理性能，以及越差的可拓展性。

进而，强一致性（Strong Consistency）开始被业界提出，主要包括顺序一致性、线性一致性，但依旧比较难实现，而且从实际需求出发，需求性不大，同时，强一致性的实现往往意味着高成本。因此，目前市场大部分系统的实现往往是通过所谓的最终一致性（Eventual Consistency），即总会存在一个时刻，让系统达到一致的状态。

随着区块链技术的不断发展，不同的共识机制不断涌现，现有的一部分共识机制和代表性应用介绍如表 3.1 所示。

表 3.1　共识机制举例

工作量证明-PoW	SHA256 算法	比特币（Bitcoin）、比特现金（Bitcoin Cash）
	Ethash 算法	以太坊（Ethereum）、以太经典（Ethereum Classic）
	Scrypt 算法	比特黄金（Bitcoin Gold）、莱特币（Litecoin）
	Equihash 算法	大零币（Zcash）、小零币（Zcoin）
	CryptoNote 算法	字节币（Bytecoin）、门罗币（Monero）
	X11 算法	达世币（Dash）、石油币（Petro）
权益证明-PoS	点点币（Peercoin）、黑币（Blackcoin）、量子链（Qtum）、以太坊第四阶段（Ethereum）	
委任权益证明-DPoS	柚子（EOS）、斯蒂姆币（Steem）、应用链（Lisk）	
随机权益证明-RPoS	Orabs、超脑链（Ultrain）	
有向无环图-DAG	埃欧塔（IOTA）	

续表

实用拜占庭容错算法-PBFT	超级账本(Hyperlegder)0.6 版、央行的数字货币
Pool 验证池	私有链
活跃证明-PoA	唯链(Vechain)、欧链(Oracles)
瑞波共识机制-RPCA	瑞波币(Ripple)
恒星共识协议-SCP	恒星币(Stellar)
容量证明-PoC	爆裂币(Burstcoin)
自定义共识机制及混合和共识机制	Hcash(红烧肉-Hshare)、授权拜占庭容错-dBFT(小蚁-NEO)、联邦拜占庭协议-FBA

3.2 拜占庭将军问题

拜占庭将军问题(Byzantine failures Problem)是由莱斯利·兰伯特(Leslie Lamport)和其他两人针对点对点通信于 1982 年提出的一个基本问题。问题描述为：在古代东罗马的首都，由于地域宽广，守卫边境的多个将军(系统中的多个节点)需要通过信使来传递消息，达成某些一致的决定。但由于将军中可能存在叛徒(系统中节点出错)，这些叛徒将努力向不同的将军发送不同的消息，试图干扰一致性的达成。拜占庭问题指的是在此情况下，如何让忠诚的将军们能达成行动的一致。

例如：10 个将军共同去攻打一座城堡，只有一半以上(也就是至少要 6 个)将军一起进攻，才可能攻破。但是，这中间有可能存在未知叛徒，造成真正进攻的军队数量小于或等于 5，致使进攻失败而遭受灭亡。那么如何相互通信，才能确保有 6 个将军同时发布进攻命令，从而使军队一致进攻而成功；或者确保小于 6 个将军的进攻命令，从而使军队一致不进攻避免被灭掉？也就是说，要么一半以上同意一起进攻而决定进攻，要么不到一半同意一起进攻而决定不进攻，但要避免达成进攻意见但命令却是不进攻，使那些进攻军队数小于或等于一半，造成进攻者的被灭。这种情况并不考虑进攻的命令是否准确有效，单纯就各位将军的命令在何种情况下能够确保一致。

拜占庭将军问题可以简化为：所有忠诚的将军能够相互间知晓对方的

真实意图，并最终做出一致行动。而形式化的要求就是一致性和正确性。兰伯特对拜占庭将军问题的研究结论是，如果叛徒的数量大于或等于 1/3，拜占庭问题不可解；如果叛徒个数小于将军总数的 1/3，在通信信道可靠的情况下，通过口头协议，可以构造满足一致性和正确性的解决方法，将军们能够做出正确决定。

口头协议指的是：将军们通过口头消息传递达到一致。隐藏条件是：每则消息都能够被正确传递；信息接收方确定信息的发送方；缺少的信息部分已知。如果同时将一个节点的信息传递给其他两个节点，这两个节点接收到消息后也分别传达给其他节点，这样每个节点都是信息的接受方和传递方……直到每个节点最后都收到所有节点发送的信息。在此过程中，若出现叛徒或虚假消息导致信息不匹配，所有节点按照少数服从多数的原则，行动便能够达成一致。缺点是：如果出现信息不一致的情况，因为对于信息的传送方未知，所以无法判断叛徒。

为了解决无法追溯根源的问题，还有一种方案：采用签名信息。将军们利用不可伪造的签名技术表达自己的意见，其他人可以验证签名的有效性，如果签名被本人之外的第三方篡改，则很容易被发现。但是这种方案需要解决如何实现真正可靠的签名体系。如果依赖第三方存储签名数据，那么这个网络本身就不再是前提中所假设的节点间互不信任的分布式结构，其次是签名造假的问题也无法避免，同时，异步协商带来的漫长的传输时间并不适合实际使用。

3.3 共识算法

3.3.1 工作量证明

作为一种分布式网络，区块链网络需要解决拜占庭将军问题，以达成工程上的相对一致。在比特币区块链中，通过工作量证明机制解决了如何在互不信任的分布式网络中确保各方利益的同时达成一致共识的难题。

工作量证明（Proof of Work，PoW）是一种对应于服务与资源滥用，或阻

断服务攻击的经济对策。一般要求用户进行一些用时适当的复杂运算，并且答案能被服务方快速验算，以耗用的时间、设备与能源作为担保成本，确保服务与资源是被真正的需求所使用。此概念最早由 Cynthia Dwork 和 Moni Naor 在 1993 年的学术论文提出，而“工作量证明”一词则是在 1999 年由 Markus Jakobsson 与 Ari Juels 共同发表。现在常用“工作量证明机制”指代应用于区块链技术中的一种主流共识机制。

工作量证明常用的技术原理是哈希函数。在比特币挖矿过程中使用的是 SHA256 哈希函数，无论输入值的大小是多少，SHA256 函数的输出的长度总是 256bit。该算法的规则是，节点通过解决密码学难题，也就是算得工作量证明解，来争夺唯一记账权。平均十分钟（具体时间受密码学问题的难度影响）会有一个节点记账成功，其他节点验证通过后复制这一记账结果。

矿工首先根据存储的交易池中的交易构造一个候选区块，计算区块头信息的哈希值，观察它是否小于当前的目标值。如果小于目标值，那么在没有其他节点广播信息的时候，矿工成功争夺记账权；如果哈希值不小于目标值，那么矿工会修改 Nonce 值，然后再试一次。具体目标值越小，找到小于该目标值的哈希值就会越难。以掷骰子举例，两个骰子的和小于 12 的概率远大于两个骰子的和小于 5 的。同时，无论哈希值前有多少位规定为 0，随着 0 位数的增加，难度都会越大。如果考虑的是 256bit 空间，每次哈希值前多一个 0，那么哈希查找的空间将缩减一半。

矿工成功挖矿代表得到了新区块的工作量证明解，并将迅速在网络中进行广播，其他节点在接受并验证后也会继续传播新区块，每个节点都会把它当作新区块添加到自身节点的区块链副本中。当挖矿节点收到并验证了这个新区块后，便会放弃之前对构建这个相同高度区块的计算，并立即开始计算区块链中的下一个区块。

在比特币网络中，实现所有节点的去中心化共识机制，不单需要工作量证明，还有其他 3 个独立的过程相互作用：每个全节点依据综合标准对每个交易进行独立验证；通过完成工作量证明算法的验算，挖矿节点将交易记录独立打包进新区块；每个节点独立地对新区块进行校验，并组装进区块链；每个节点对区块链进行独立选择，在工作量证明机制下选择累计工作量最

大的区块链。

其中，解决区块链的分叉问题遵循了累计工作量最大的链条为网络主链的原则。当有两名矿工几乎在同一时间算得新区块的工作量证明解，在分别对各自区块进行传播的过程中，就会出现两个不同版本的区块链。解决方法如下：总有一条最终会成为更长的链，所有节点会接收更长的链，网络就会重新达成共识，这就是最长合法链原则。

3.3.2 权益证明

工作量证明算法的优势明显，但为了维持其正常运转却需要大量的资源投入，尤其是电力资源和购置矿机的成本。根据 Digiconomist 调查，仅仅比特币矿工就要使用 54TW·h 的电力，这些电量足够支持美国五百万个家庭的用电，甚至整个新西兰或匈牙利的电力消耗，但是实际耗电不仅止于此。权益证明机制试图找到一个更为绿色环保的分布式共识机制。

2011 年，QuantumMechanic 在 Bitcointalk 论坛首次提出了权益证明，权益证明(Proof of Stake，PoS)是一类应用于公共区块链的共识算法，不同于工作量证明中新区块的挖掘完全取决于节点进行哈希碰撞的算力，在权益证明中，新区块的创建是通过随机、财富或币龄的各种组合来进行选择的，取决于节点在网络中的经济效益。它所蕴含的概念是：区块链应该由具有经济利益的人进行保障。通过选举，系统随机选择节点验证下一个区块，但要成为验证者，节点需要在网络中事先存入一定数量的货币作为权益，这类似于保证金机制。权益证明的运作方式是，当创造一个新区块时，节点需要创建一个 coinstake 交易，交易会按照一定比例将一些币发送给节点本身。根据节点拥有币的比例和时间，按照算法对难度目标进行调整，从而加快了节点找到符合难度目标随机数的速度，这极大地降低了系统达成共识所需要的时间。

权益证明并不单纯考虑账户余额，因为如果将账户余额定义为下一个有效区块的挖掘方式，那么单个最富有的节点将具有永久优势，这势必会导致网络的集中化。在目前的数字加密货币中，已经设计了多种不同的权益证明体系，如点点币(Peercoin)和黑币(Blackcoin)以及目前以太坊主链，采

用的都是不同的权益证明机制。

3.3.3 委托权益证明

委托权益证明(Delegated Proof of Stake,DPoS)共识算法由 Daniel Larimer 在 2014 年提出。例如,Bitshares、Steem、Ark 和 Lisk 都是使用委托权益证明共识算法的数字货币项目。委托权益证明区块链具有投票系统,利益相关者将他们的工作交付给第三方。换句话说,他们可以投票选出几个代表代替他们保护网络。代表们也被称为见证人,他们需要在产生和验证新区块的过程中达成共识。投票权与每个用户持有的币数量成正比。投票系统因项目而异,但总的来说,每位代表在投票时都会提出个人意见。通常,代表们会收集奖励并按比例分配给各自的投票者们。

因此,委托权益证明算法创造了一个直接取决于代表声誉的投票系统。如果选举的节点行为不当或不能有效工作,它将很快被驱逐并被另一个节点取代。

在性能方面,与工作量证明和权益证明相比,委托权益证明的区块链更具有可扩展性,每秒的事务处理(Transaction Per Second,TPS)更多。与权益证明相比,虽然委托权益证明在股份制的意义上是类似的,但委托权益证明提出了一种新颖的民主投票系统来选出区块生产者。委托权益证明的系统由选民维护,所以代表们的行为必须诚实且高效,否则便会被投票出局。此外,委托权益证明区块链在每秒事务处理方面比权益证明区块链更快。与工作量证明相比,不同于试图解决工作量证明问题的权益证明,委托权益证明旨在简化区块生成过程。因此,委托权益证明系统能够快速处理大量的链上交易。委托权益证明的使用方式与工作量证明、权益证明都不同。由于工作量证明仍然是公认的最安全的共识算法,所以大多数金融流动都发生于此。权益证明比工作量证明的工作效率更高,所以它具有更多的运用案例。委托权益证明限制了选举区块生产者的过程中股权的使用。与有着竞争体系的工作量证明系统不同,委托权益证明的实际区块生成是预定的,每个见证人都会轮流生产区块。因此,有人认为,委托权益证明应被视为一种权威证明系统。

3.3.4 Paxos 与 Raft 算法

1988 年，Brian M. Oki 和 Barbara H. Liskov 在论文 *Viewstamped Replication: A New Primary Copy Method to Support Highly-Available Distributed Systems* 中首次提出了解决 Paxos 问题的算法。论文中为了描述问题编造了一个虚构故事：在古代爱琴海的 Paxos 岛，议员们通过信使传递消息来对议案进行表决。但议员可能离开，信使可能走丢，甚至重复传递消息。1990 年，Leslie Lamport 在论文 *The Part-time Parliament* 中提出了 Paxos 共识算法，从工程角度实现了一种能够最大限度地保障分布式系统一致性（存在无法实现一致的极小概率）的机制。Paxos 算法在本质上与前者相同，广泛应用在 Chubby、ZooKeeper 这类分布式系统中。Leslie Lamport 作为分布式系统领域的早期研究者，凭借相关杰出贡献获得了 2013 年度图灵奖。

Paxos 是首个得到证明并被广泛应用的共识算法，其原理类似两阶段提交算法，并进行了泛化和扩展，通过消息传递来逐步消除系统中的不确定状态。作为后来很多共识算法（如 Raft、ZAB 等）的基础，Paxos 算法的基本思想并不复杂，但最初论文中描述的比较难懂，甚至发表时也几经波折。2001 年，Leslie Lamport 还专门发表论文 *Paxos Made Simple* 进行重新解释。

Paxos 将系统中的角色分为提议者（Proposer）、决策者（Acceptor）和最终决策学习者（Learner）。

（1）Proposer：提出提案（Proposal）。Proposal 信息包括提案编号（Proposal ID）和提议的值（Value）。

（2）Acceptor：参与决策，回应 Proposers 的提案。收到 Proposal 后可以接受提案，若 Proposal 获得多数 Acceptor 的接受，则称该 Proposal 被批准。

（3）Learner：不参与决策，从 Proposer/Acceptor 学习最新达成一致的提案（Proposal）。

算法需要满足安全性（Safety）和存活性（Liveness）这两方面的约束要求。

如图 3.1 所示，Paxos 算法通过一个决议的过程可以分为 3 个阶段(前 2 个阶段在 Learn 阶段之前决议已经形成)：

(1) 第一阶段：Prepare 阶段。Proposer 向 Acceptors 发出 Prepare 请求，Acceptors 针对收到的 Prepare 请求进行 Promise(承诺)。

(2) 第二阶段：Accept 阶段。Proposer 收到多数 Acceptors 承诺的 Promise 后，向 Acceptors 发出 Propose 请求，Acceptors 针对收到的 Propose 请求进行 Accept 处理。

(3) 第三阶段：Learn 阶段。Proposer 在收到多数 Acceptors 的 Accept 之后，标志着本次 Accept 成功，决议形成，将形成的决议发送给所有 Learners。

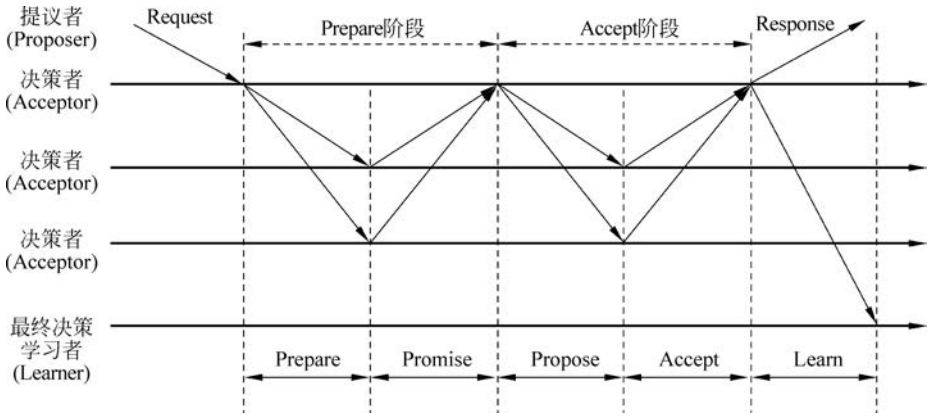

图 3.1 Paxos 算法通过决议流程

Paxos 并不保证系统总处在一致的状态。但由于每次达成共识时至少有超过一半的节点参与，这样最终整个系统都会获知共识结果。一个潜在的问题是提案者可能会在提案过程中出现故障，这可以通过超时机制来缓解。若在极为凑巧的情况下，每次新一轮提案的提案者都恰好故障，又或者两个提案者恰好依次提出更新的提案，则导致活锁，系统将永远无法达成共识(实际发生概率很小)。

Paxos 能保证在超过一半的节点正常工作时，系统总能以较大概率达成共识。读者可以试着自己设计一套非拜占庭容错下基于消息传递的异步共识方案，会发现：在满足各种约束的情况下，算法过程总会十分类似于

Paxos 的过程。这也是为何 Google Chubby 的作者 Mike Burrows 说："这个世界上只有一种一致性算法，那就是 Paxos(There is only one consensus protocol，and that's Paxos)"。

Paxos 算法虽然给出了共识设计，但并没有讨论太多实现细节，也并不重视工程上的优化，因此后来在学术界和工程界作了一些改进工作，包括 Fast Paxos、Multi-Paxos、Zookeeper Atomic Broadcast(ZAB)和 Raft 等。这些算法重点在于改进执行效率和提高可实现性。

其中，Raft 算法由斯坦福大学的 Diego Ongaro 和 John Ousterhout 于 2014 年在论文 *In Search of an Understandable Consensus Algorithm* 中提出，基于 Multi-Paxos 算法进行重新简化设计和实现，提高了工程实践性。Raft 算法的主要设计思想与 ZAB 类似，通过先选出领导节点来简化流程和提高效率。实现上分解了领导者选举、日志复制和安全方面的考虑，并通过约束减少了不确定性的状态空间。

如图 3.2 所示，Raft 算法包括 3 种角色——领导者(Leader)、候选者(Candidate)和跟随者(Follower)，每个任期内选举一个全局的领导者。领导者角色十分关键——决定了日志(Log)的提交。每个日志都会路由到领导者，并且只能由领导者向跟随者单向复制。

典型的过程包括两个主要阶段。

(1) 领导者选举：开始所有节点都是跟随者，在随机超时发生后，如果未收到来自领导者或候选者消息，则转变角色为候选者(中间状态)，提出选举请求。最近选举阶段(Term)中得票超过一半者被选为领导者；如果未选出，随机超时后进入新的阶段重试。领导者负责从客户端接收请求，并分发到其他节点。

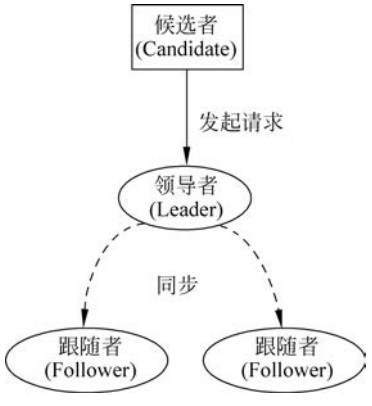

图 3.2　Raft 算法运行模式

(2) 同步日志：领导者会决定系统中最新的日志记录，并强制所有跟随者来刷新到这个纪录，数据的同步是单向的，确保所有节点看到的视图一致。

此外，领导者会定期向所有跟随者发送心跳消息（Heartbeat Message），跟随者如果发现心跳消息超时未收到，则认为领导者已经下线，尝试发起新的选举过程。

3.4 其他新型共识算法

3.4.1 其他竞争类共识算法

1. 所用时间证明（Proof of Elapsed Time，PoET）共识算法

PoET 的工作机制如下：网络中的每个参与节点都必须等待一个随机选取的时期，首个完成设定等待时间的节点将获得一个新区块。区块链网络中的每个节点会生成一个随机的等待时间，并休眠一个设定的时间。最先醒来的节点，即具有最短等待时间的节点，唤醒并向区块链提交一个新区块，然后把必要的信息广播到整个对等网络中。同一过程将会重复，以发现下一个区块。

在 PoET 网络共识机制中，需要确保两个重要因素。第一，参与节点在本质上会自然地选取一个随机的时间，而非某一个参与者为了胜出而刻意选取的较短的时间。第二，胜出者的确完成了等待时间。

这种内在机制允许应用在受保护的环境中执行受信任的代码，它确保了上面提出的两个要求得到满足，即随机选择所有参与节点的等待时间，以及胜出参与者真正完成了等待时间。这种在安全环境中运行可信代码的机制也同时考虑到了其他一些网络的需求。它确保了受信代码运行在安全环境中，并不可被其他外部参与者更改。它也确保了结果可被外部参与者和实体验证，进而提高了网络共识的透明度。

PoET 通过控制代价实现了共识过程，该代价依然是与从过程中获得的价值成正比。这是保证加密货币经济持续繁荣的一个关键需求。其优点为：参与代价低。更多人可轻易加入，进而达到去中心化；对于所有参与者而言，更易于验证领导者是通过合法选举产生的；控制领导者选举过程的代价，与从中获得的价值成正比。但也存在一些不足，例如，尽管 PoET 的代

价低，但是必须要使用特定的硬件。因此不会被大规模采纳，且并不适用于公有区块链。

PoET 共识机制算法通常用于许可区块链网络，它可决定网络中获得区块者的挖矿权利。许可区块链网络需要任何预期参与者在加入前验证身份。根据公平彩票系统的原则，每个节点成为胜出者的可能性相同。PoET 机制赋予大量可能的网络参与者以平等胜出的机会。

2. 空间证明（Proof of Space，PoSpace）共识算法

与大多数基于计算能力或质押代币授予记账权的区块链系统不同，空间证明（PoSpace），也称为容量证明（Proof of Capacity，PoC），其共识算法基于节点硬盘驱动器中的可用空间量。

在 PoSpace 中，矿工在称为“绘图”的过程中预先生成所有可能的哈希列表，然后将这些图存储在硬盘驱动器上。矿工拥有的存储容量越大，可能的解决方案就越多。解决方案越多，拥有正确的哈希组合并赢得奖励的机会就越高。

由于不需要昂贵或专门的设备，PoSpace 为普通人提供了参与网络的机会。因此，它是一种能耗更低、更分散的替代方案。然而，到目前为止，选择采用该系统的开发人员并不多，而且人们担心它容易受到恶意软件攻击。该机制目前由 Signum（SIGNA）（以前的 Burstcoin（BURST）、Storj（STORJ）和 Chia（XCH）等）使用。

PoSpace 共识算法使用存储空间代替计算，以节约电力资源；同时，在该共识协议下，节点初次接入网络时确定存储空间大小，之后不能扩容，这防止了 PoW 共识算法中“矿池”（将不同节点的算力集合成一个大的算力节点）的出现，在一定程度上避免了中心化程度增强，同时降低了安全风险。

3.4.2 其他选举类共识算法

授权拜占庭容错（delegated Byzantine Fault Tolerance，dBFT）算法根据权益选出记账人，然后记账人之间通过拜占庭容错算法来达成共识。该算

法由小蚁(NEO)团队提出,与 PBFT 相比,白皮书中说明的改进包括:

(1) 将 C/S 架构的请求响应模式,改进为适合 P2P 网络的对等节点模式;

(2) 将静态的共识参与节点改进为可动态进入、退出的动态共识参与节点;

(3) 为共识参与节点的产生设计了一套基于持有权益比例的投票机制,通过投票决定共识参与节点(记账节点);

(4) 在区块链中引入数字证书,解决了投票中对记账节点真实身份的认证问题。

该机制将网络的参与者分为两类:专业记账的记账节点和普通用户。普通用户基于持有权益的比例进行投票,选择记账节点,当需要通过一项共识时,在记账节点中选定一个发言人进行方案的拟定,其他记账节点根据拜占庭容错算法进行表态,如果存在超过指定比例的节点同意该方案,则方案达成,否则重新选择发言人,进行方案的拟定。

这种方法的优点是有专业化的记账人;能够容忍任何类型的错误;记账由多人协同完成;每一个区块都有最终性,不会分叉;算法的可靠性有严格的数学证明。但白皮书中也坦言了现有算法的缺陷:当有 1/3 或以上记账人停止工作时,系统将无法提供服务;当有 1/3 或以上的记账人联合作恶,且其他所有记账人被恰好分割为两个网络孤岛时,恶意记账人可以使系统出现分叉,但是会留下密码学证据。综上来说,dBFT 机制最核心的一点就是,最大限度地确保系统的最终性,使区块链能够适用于真正的金融应用场景。

这符合 NEO 团队的定位:用户可以将实体世界的资产和权益进行数字化,通过点对点网络实现登记发行、转让交易、清算交割等金融业务的去中心化。目标市场不仅是数字货币圈,还包括主流互联网金融。NEO 可以被用于股权众筹、P2P 网贷、数字资产管理、智能合约等。

3.4.3 基于 DAG 共识算法

DAG(Directed Acyclic Graph,有向无环图)原本是计算机领域一种常

用数据结构，因为独特的拓扑结构所带来的优异特性，经常被用于处理动态规划、导航中寻求最短路径、数据压缩等多种算法场景。传统区块链和 DAG 的主要区别如下。

（1）单元：区块链的组成单元是 Block（区块），DAG 的组成单元是 TX（交易）。

（2）拓扑：区块链是由 Block 区块组成的单链，只能按出块时间同步依次写入，这种操作类似于单核单线程 CPU；DAG 是由交易单元组成的网络，可以异步并发写入交易，这种操作类似于多核多线程 CPU。

（3）粒度：区块链每个区块单元记录多个用户的多笔交易，DAG 每个单元记录单个用户的交易。

最早在区块链中引入 DAG 概念作为共识算法的是在 2013 年，在网络 bitcointalik.org 上，由 ID 为 avivz78 的以色列希伯来大学学者提出 GHOST 协议作为比特币的交易处理能力扩容解决方案。而 Vitalik 在以太坊紫皮书中描述的 POS 共识协议 Casper，也是基于 GHOST POW 协议的 POS 变种。后来，NXT 社区有人提出用 DAG 的拓扑结构来存储区块，以解决区块链的效率问题。区块链只有一条单链，若打包出块将无法并发执行。如果改变区块的链式存储结构，变成 DAG 的网状拓扑，则可以并发写入。在区块打包时间不变的情况下，网络中可以并行打包 N 个区块，网络中的交易就可以扩大成 N 倍。此时，DAG 跟区块链的结合依旧停留在类似侧链的解决思路，交易打包可以并行在不同的分支链条进行，达到提升性能的目的。此时 DAG 还是有区块的概念。

2015 年 9 月，Sergio Demian Lerner 发表了 *DagCoin*：*a cryptocurrency without blocks* 一文，提出 DAG-Chain 的概念，首次把 DAG 网络从区块打包这样粗粒度提升到了基于交易层面，但这一篇论文没有代码实现，其思路是让每一笔交易都直接参与维护全网的交易顺序。交易发起后，直接广播全网，跳过打包区块阶段，达到所谓的 Blockless。这样省去了打包交易出块的时间。如前文提到的，DAG 最初跟区块链的结合就是为了解决效率问题，现在不用打包确认，交易发起后直接广播网络确认，使效率得到了质的飞跃。DAG 进一步演变成了完全抛弃区块链的一种解决方案。

2016 年 7 月，基于 Bitcointalk 论坛公布的创世帖，IOTA 横空出世，随后 ByteBall 也闪亮登场，IOTA 和 Byteball 是 DAG 网络第一次从技术上得到实现；此时，号称无块之链(Block Less)、独树一帜的 DAG 链家族的雏形基本形成。

可以说，DAG 是面向未来的新一代区块链，宏观地从图论拓扑模型来看，从单链进化到树状和网状、从区块粒度细化到交易粒度、从单点跃迁到并发写入；这是区块链从容量到速度的一次革新。如图 3.3 所示，现有基于 DAG 的共识机制分为基于主干链、基于平行链和基于朴素 DAG 的共识协议。区别在于基于主干链的 DAG 共识协议，首先在 DAG 中确定主链，进而确定交易顺序，基于平行链的 DAG 共识协议，网络中各实体或实体集合分别维护一条链，链间通过相互引用构成平行链结构，实体间利用此引用关系进行共识，基于朴素 DAG 的共识协议，除基本引用规则外无特殊限制，在 DAG 拓扑结构中利用某种投票机制进行共识。

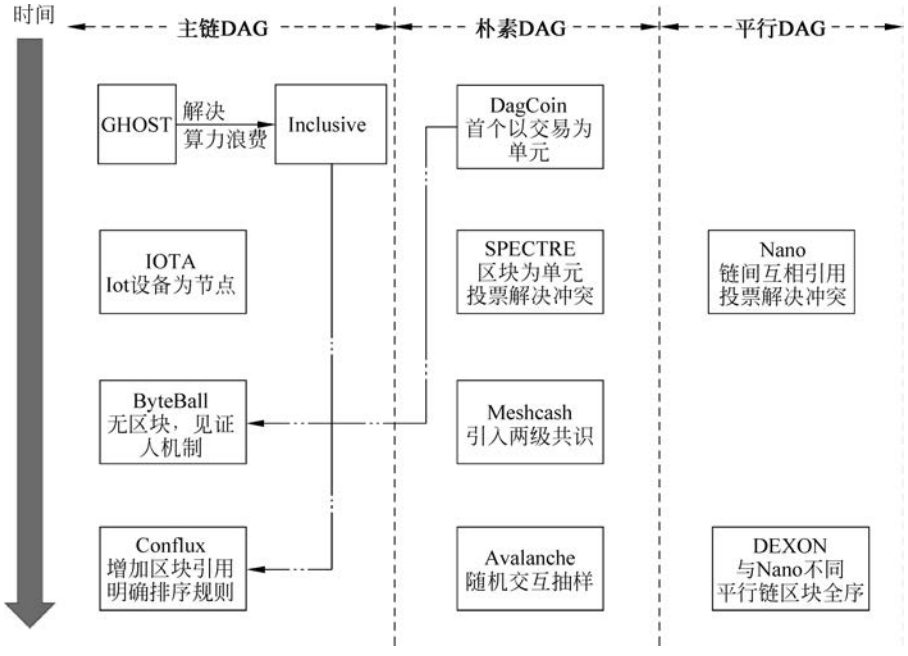

图 3.3　基于 DAG 的共识机制

本章小结

本章主要介绍了区块链的共识机制与相关算法。首先对区块链技术中常见的一致性问题与拜占庭将军算法问题进行了阐述，并引出目前各种类型的共识机制与代表算法；然后对工作量证明、权益证明、委托权益证明三种算法进行介绍，并引入 Paxos 与 Raft 算法框架与算法原理；同时，本章还对其他新型的竞争类、选举类共识算法进行了介绍，包括基于 DAG 的共识算法的原理与演进。