摘要

基于拜占庭容错共识算法的可信数据存储系统的研究与实现

随着人们对数据的重视程度不断的加深，以及越来越完善的传感器与物联网技术，海量的数据被人类不断的制造出来。而近年来数据规模的爆炸式增长也同时带来了数据安全事件的频频发生，对数据的篡改、销毁、恶意泄露等行为几乎每周都有所报道，对于一些敏感数据如经济数据的恶意篡改甚至会造成严重的经济损失，传统的数据存储系统的安全性隐患逐渐浮出水面，同时大型分布式存储系统的单点故障问题也使得系统的稳定性难以保证。区块链技术自出现以来，受到了工业界和学术界的广泛关注，其天生拥有的去中心化、防篡改等属性也对一些传统的业务模式带来了天翻地覆的变化。作为一种去中心化的技术解决方案，已经在金融领域有了非常多的应用场景。人们越来越寄希望于能够引入区块链技术对存储系统带来新的改变。

当前被广泛应用的基于工作量证明（PoW）的共识算法能够应用于非常大规模的区块链节点网络之中，但是其所带来的计算量的浪费与较低的响应速度也使其难以应用于需要快速响应的基础应用系统中。而传统的分布式一致性算法PBFT（Practical Byzantine Fault Tolerance）虽然能解决工作量证明共识算法严重的资源浪费的问题，也依然存在通信复杂度高、大规模节点的网络性能较差、投票签名消息过大等问题。对于一些需要很高性能的应用场景下依然难以满足生产需求。本文针对上述的问题，进行了如下的工作：

1.提出了基于轻客户端节点和重客户端节点的改进拜占庭容错算法 WBFT，通过改进 PBFT 共识流程，将通信复杂度从$O(n^2)$降低到$O(n)$。并使用 Go 语言实现了 WBFT 共识算法。通过实验证明，相比较于 PBFT 算法至少有 10%的性能提升。

2.基于 WBFT 算法设计并实现了可信数据存储系统，通过对系统的测试和实验，证明系统能够在存在部分恶意节点的情况下，保证存储数据的安全可信。

关键字：

区块链，共识算法，拜占庭容错，分布式存储

Abstract

Research and Implementation of Trusted Storage System Based on Byzantine Fault-tolerance Consensus Algorithm

With the deepening of people's attention to data and the improvement of sensors and Internet of things technology, a massive of data is produced by human beings. Recently, because of the explosive growth of data scale, data security accidents have occurred frequently. Specifically, the behaviors of tampering, destruction, malicious leaking on data were reported almost every week. The behavior of maliciously tampering with some sensitive data like economic data may cause serious economic losses. According to the above description, the security vulnerabilities of traditional data storage system are emerging, and a single point of failure problem of large-scale distributed storage systems also makes the stability of the system is difficult to guarantee. With the advancement of technology, blockchain was proposed and has attracted increasing attention from industry and academia. The inherent attributes like decentralization and tamper-proof of blockchain technology have brought tremendous changes to some traditional business models. As a decentralized technical solution, it has been widely applied in many financial fields. Based on this, the hope that introducing blockchain technology to storage systems and realize the progress of data storage system is growing.

The currently popular consensus algorithm is proof-of-work (PoW), which can be applied to a very large scale blockchain node network. However, its waste of computation and low response speed makes it unable to be directly applied to basic application systems requiring fast response. While the traditional distributed consistency algorithm of practical byzantine fault toll (PBFT) can solve the problem of serious resource waste exist in the PoW consensus algorithm. But it still has the problems of high communication complexity, poor performance of large-scale network, and large number of voting signature messages. These disadvantages make the algorithm difficult to satisfy the production requirements of high performance for some applications. Based on the above problems, this paper carries out the following

research work:

1.Propose an improved Byzantine fault-tolerant (WBFT) algorithm based on light and heavy client nodes. This method improves the PBFT consensus process and reduces the communication complexity of algorithm from $O(n^2)$ to $O(n)$. We implement the WBFT consensus algorithm with Go language and the experiment results demonstrate that the average transaction throughput of algorithm is improved by 10%.

2.Design and implement a trusted data storage system based on WBFT algorithm. By testing and experimenting this system, we demonstrate that the system can guarantee the security and reliability of stored data in the case of some malicious nodes.

Keyword：blockChain, Consensus Algorithm, Byzantine Fault Tolerance, Distributed Store

目　　录

第一章 绪论

1.1 研究背景及意义

随着信息产业的蓬勃发展，人们逐步进入了大数据时代[1]。海量的数据在人们的生产生活中产生，互联网上的浏览记录、社交记录、乃至智能家居获取用户的习惯数据等等数据越来越多的被利用。人们对数据的保存提出了更高的要求，越来越需要更为方便快捷的手段进行数据的保存和处理。根据国际数据公司（IDC）的研究表明，全球每年至少会产生的 30ZB 的总数据量，而到 2025 年，这一数字甚至预计会达到 175ZB[2]。2018 年所有微信用户每日就发送 450 亿条微信，微博视频日均发布量为 150 万以上[3]。

数据的价值增长使人们认识到企业的重要资产其中也应该包括企业所掌握的数据。然而面对数据的爆炸产生，数据安全事件同样频频发生[4]。丢失、泄露甚至伪造数据等情况经常出现。根据 Risk Based Security(RBS)的 2019 年第三季度报告，仅仅 2019 年全世界就有 79.95 亿条记录被泄露[5]。2019 年 12 月，美国短信服务商 TrueDialog 保存有 10 亿条数据的数据库，其中包含用户短信内容、用户的电话号码甚至用户的用户名和密码等数据被泄露。有 1 亿多美国公民被波及[6]。

根据上述的事件可以看出，大量的数据被一些大公司采用中心化的存储方式进行保存。而这种数据的中心化存储方式并不能有效的防止数据被篡改。随着人们对数据本身所拥有的巨大潜力价值越来越重视，同时数据也越来越多的受到了不法分子的重点关注。对一些敏感数据的攻击并不仅仅会产生经济上的损失，甚至会对国家安全产生危害。我们认为如果一个数据的来源可信，处理可信，数据的访问可信。那么这个数据就是基本可信的[7]。数据来源可信指的是数据的采集过程是可信的，可以准确的追溯到数据的源头，数据的处理可信指的是数据的处理过程是可信的，我们对于数据的处理过程可以有十分明确的验证过程。访问可信指的是一个用户访问数据的权限是需要被准确的控制的，不会产生任何的越权访问情况。

区块链技术所拥有的透明可信、防篡改、可追溯等特性在金融行业的应用改变了行业传统生态，甚至在某种程度上重构了整个金融体系[8]。最早的区块链技术仅仅用来在不可信的网络中创建一个可以被用户信任的账本，直到以太坊技术

的成熟与大规模的使用，区块链变成了用来建立可信数据的优秀解决方案。非常多的行业都寄希望于能够利用区块链技术来对旧有问题提出新的解决方案[9]。是否能够使用区块链技术解决我们所面临的数据篡改丢失的问题便是一个很重要的研究方向。

1.2 国内外研究现状

1.2.1 存储系统的研究现状

目前存储系统的发展主要集中在两个方面上：对于单点存储系统效率的研究以及对于多点位的分布式的存储系统的技术研究。在单点位的存储系统的研究上，随着计算机硬件技术的稳步发展，单机存储系统性能上有着十分显著的进步。无论是近年来英特尔的持久内存技术，还是英特尔的Optane傲腾固态硬盘技术，都显著的改善了单机存储的性能瓶颈，进一步的指明了未来的单机存储发展方向[10]。

对于存储集群技术研究来说，无论是Google的GFS（Google File System）[11]所拥有的的快速恢复的能力[12],还是开源的Ceph文件系统对RUSH算法的改进[13]。都实现了完备的分布式存储系统的容错能力，通过对数据的切块与多备份处理，可以支持在大量的廉价且普通的机器上进行数据存储。用户可以灵活的在大型的存储集群上进行数据的存储工作，对于存储规模的提升有着显著的意义。

按照数据组织方式划分，我们把存储系统分为关系型数据库和非关系型数据库（No-Sql）。关系型数据库采用关系模型来组织数据，对数据采用行和列的形式来进行存储，支持对数据的检索和查询，同时支持事务等很复杂的业务逻辑，比如常见的关系型数据库MySQL就可以使用结构化查询语言（SQL）来对数据进行检索，SQL语言可以支持非常复杂的检索逻辑。

非关系型数据库并不采用关系模型来组织数据，常见的非关系型数据库有键值对数据库(Key-Value)数据库、列存储数据库、图形数据库等。键值对数据库中只有一个特定的键和一个特定的数据，二者一一对应，用户用键来获取对应的数据。而列存储数据库采用列的方式对数据进行组织，在海量数据存储中对磁盘和CPU比较友好。

关系型数据库比较复杂，特别是在并行查找的时候需要进行很复杂的加锁操作以保证数据的一致性，所以数据的读写性能通常不如非关系型数据库，但是由于支持很复杂的查询逻辑，所以可以适用于非常多的业务场景中。而非关系型数

据库数据模型通常比较简单，可以很方便的进行读写优化。

随着人们所掌握的数据量的不断增长，单一的单机存储越来越无法满足生产需求，我们很多时候需要大量数据存储集群来进行数据的管理。非关系型数据库数据模型简单，可以很方便的进行网络扩展，越来越受到学术界的关注。

1.2.2 区块链的研究现状

如何在不完全可信的环境中构建一个可以让用户达到信任的系统一直是学术界的重点研究方向。而随着区块链技术近年来快速的发展，使用分布式的系统架构，利用链式的存储结构，定义所有的忠诚节点都参与的共识算法就成为了一套很成熟的解决信任问题的方案。作为一种全新的分布式基础架构，区块链在未来有着广袤的发展前景[14]。

共识算法的创新推动了区块链技术的进步，最早由中本聪（Nakamoto Santoshi）提出比特币（Bitcoin）[15]，且在其中提出并使用了基于工作量证明（Proof of Work,PoW）的共识算法。工作量证明算法中提出了矿工的概念，算法要求网络中的所有矿工使用自己所拥有的计算力求解一个求解困难，但是验证简单的数学问题。矿工求解数学问题的过程就是整个比特币网络达成共识的过程，一旦有一个矿工完整的求解了数学问题，就拿到了下一个区块的建立权，而其他的矿工的全部计算都将毫无产出。基于工作量证明的算法需要消耗大量无意义的计算资源，造成大量的能源浪费，同时还使得最终交易的确认时间变得冗长[16]。如图 1.1 所示：2020 年的比特币交易平均确认时间达到了 51 分钟[17]。冗长的交易确认时间严重限制了比特币的交易吞吐量。

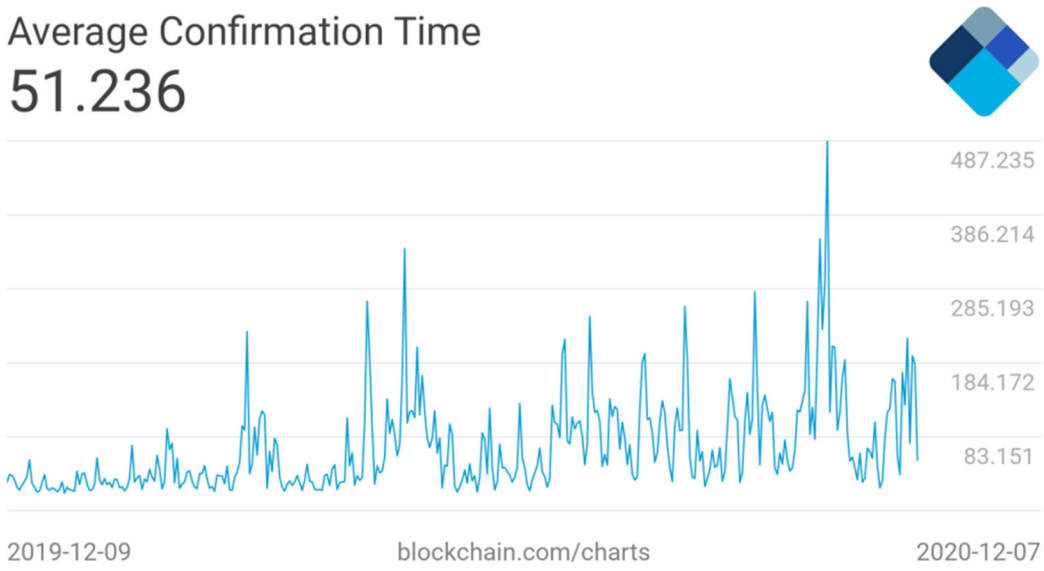

图 1.1 2020 年比特币交易平均确认时间统计

为了解决上述问题，更多的共识算法被人们提出。如权益证明（Proof of Stake，PoS）算法和拜占庭容错（BFT）算法。相比于 PoW 算法，权益证明算法中矿工工作量证明的难度与其所拥有的权益成正比。而拜占庭容错算法则可以在容忍少量拜占庭节点的情况下快速的达成网络中的共识。

从比特币被提出一直到如今，区块链技术日新月异，并不仅仅是不同的共识算法被提出，还有非常多的如以太坊、EOS、Fabric 等基于区块链的成熟的商业平台的出现[18]。这些成熟的商业平台大大的促进了区块链应用的长足发展。

1.2.3 存在的挑战

根据上文我们可以看出，存储系统在存储集群化，存储高速化上都有着长足的发展。也出现了诸如 Ceph 的分布式存储系统[19]。但是，如果一个分布式存储系统中存在恶意节点将会对整个存储系统有多大的影响现有的研究还有所不足。对分布式存储系统中数据的恶意篡改也难以防范。同时虽然区块链技术的发展日新月异，但由于其本身的性能问题，使得人们多聚焦于上层应用系统。如何将需要大量的计算量以及网络通信的区块链技术应用于基础存储系统依然是一个需要研究的问题。在当前的分布式存储的场景下，如何在保证存储效率的同时应用区块链技术保护数据的安全，如何设计适用于此场景下的共识算法同样也是摆在我们面前的问题。

1.3 论文研究内容以及创新点

1.3.1 研究内容

本文主要的研究内容是基于区块链技术，在保证分布式存储系统性能要求的前提下，设计一套在容忍一定量的恶意节点的情况下，依然能够保护数据不被篡改的存储系统。为了应对以上挑战，本文对 PBFT 共识算法进行了改进，提出了 WBFT 共识算法，同时基于 WBFT 共识算法建立了一个分布式的数据存储系统。在保证数据存储效率的同时，达到了保证所有的存储数据均不被篡改，所有上交的数据均可以溯源的目标。

1.3.2 创新点

1.文本针对 PBFT 算法通信复杂度高的问题，提出了重客户端和轻客户端节点角色，以及客户端域的概念。并基于此提出了改进算法 WBFT。使得算法的通信复杂度由 $O(n^2)$ 下降到了 $O(n)$，提高了算法的执行效率。同时最大限度的保证算法的可用性。

2.本文基于 WBFT 算法设计了分布式存储系统，系统能够保证存储数据的可靠与安全，保证数据不被恶意的篡改。

1.4 论文组织结构

本文共有个五章节，每个章节的内容如下所示：

第一章为绪论，介绍了本文的选题背景和选题意义。简单的介绍数据存储技术与区块链技术在当前国内外的研究现状，阐述了在保证数据安全性上存在的挑战，提出了利用区块链技术来保证存储系统中数据安全性的可能性与难点。最后介绍了本论文的创新点和研究内容。

第二章为相关的理论和技术，阐述了区块链的技术特点，介绍了密码学和共识算法等区块链关键技术，详细介绍了非对称加密和数字签名等密码学技术。最后对星际文件系统进行了简要介绍。

第三章针对 PBFT 算法存在的不足，提出了基于重客户端与轻客户端的改进算法 WBFT，降低了算法网络通信的复杂度。并对算法进行了实验，将 WBFT 算法和 PBFT 共识算法进行对比实验，分析了实验的结果。

第四章对基于 WBFT 算法的可信数据存储系统进行设计和实现，首先对可信

存储系统的需求进行了分析，之后阐述了系统的总体架构，然后对系统进行了实现与测试。以证明可信存储系统对数据的保护。

第五章为总结与展望，首先总结了本文的主要工作，同时阐述了 WBFT 算法的不足之处，以及可信数据存储系统的不足。对未来的发展做了进一步的展望。

第二章 相关理论和技术

在本章中将会详细的对本文所使用到的相关理论进行介绍。首先将会对本文使用到的数字加密技术进行介绍，之后会对常见的分布式系统进行介绍，最后会详细介绍区块链相关技术。

2.1 数据加密技术

数字加密技术是区块链技术的安全基础，包括哈希算法的不可反推性，数字签名的不可伪造性等多种特性共同支撑了区块链技术的安全。如果数字加密技术被破坏，区块链的安全性则自然失效。

2.1.1 哈希算法

哈希算法（Hash）或称为散列算法，可以将任意长度的一串数据作为输入，输出则为一段长度固定的数据。输出数据就是哈希值，或称为散列值。一般来说用于数据安全的哈希算法必须要符合几个特点：

1. 在有限的时间内必须要计算出给定明文的哈希值。
2. 如果算法的输入值发生了改变，则算法的输出值也必须随之改变，即使算法的输入值仅有极小的改变，算法的输出值也必须有很大的改变量。
3. 很难找到两段内容不同的明文使得输出的哈希值相同。
4. 通过算法的输出值无法通过逆运算获得算法的输入值[20]。

哈希算法常用来计算数字指纹，通过计算数字指纹可以在不获得原始数据的情况下验证数据的真实性与完整性。常用的哈希算法包括MD5和SHA系列算法[21]，基于安全性的考虑，在本文所设计的系统中使用SHA256算法。

2.1.2 加密算法

现有的加密算法一般按照加解密的密钥是否相同划分为对称加密算法和非对称加密算法。

对称加密算法是应用比较早的加密算法，甚至在还没有计算机的时代就已经被人们广为使用。在对称加密算法中，数据发送方将数据明文使用密钥进行加密处理后，使其变成密文后发送出去，收信方收到密文后，使用密钥和密文进行加密的逆运算对密文进行解密，就产生了明文。对称加密示意图如图2.1所示。

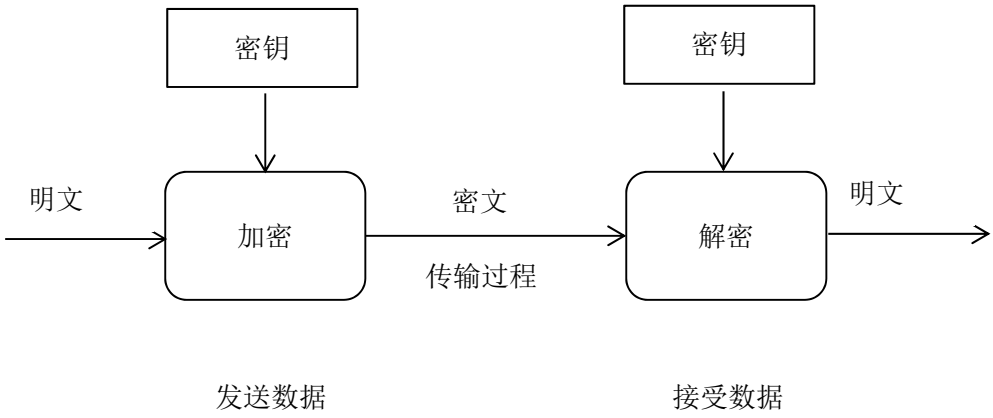

图 2.1 对称加密示意图

而在非对称加密算法中，加解密的密钥是不同的，用户首先使用私钥对数据明文进行加密产生密文，而收信方在收到密文后，需要使用公钥进行解密方能产生明文。非对称加密示意图如图 2.2.所示：

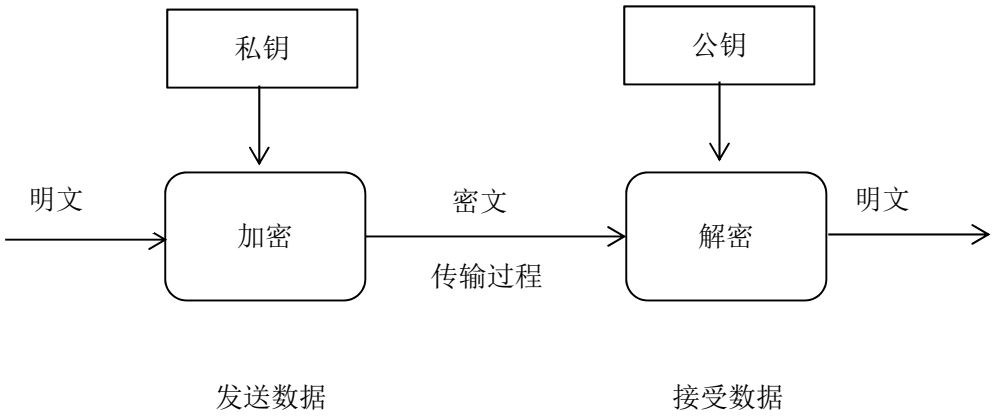

图 2.2 非对称加密示意图

一般来说，对称加密算法的计算效率要高于非对称加密算法，但是由于需要双方共享同一个密钥，比较容易产生密钥的泄露行为。而非对称加密在一定程度上解决了密钥泄露的问题，我们可以使用对方提供的公钥进行加密，而对方可以使用私钥进行解密，只要私钥不丢失则数据的安全性就有所保证。

2.1.3 数字签名

数字签名是一种利用电子签名技术对消息的摘要进行加密，防止消息被修改同时证明消息来源的技术。在一些情况下同纸质合同上的签名一样具有合法的法

律地位。椭圆曲线数字签名是一常见的数字签名算法，通过椭圆曲线加密来实现[22],其过程如下。

1.生成数字签名：

步骤一：利用公式 2.1 计算数字信息 m 的数字摘要；

$$H_m = h(m) \quad \cdots\cdots\cdots\cdots\cdots\cdots (2.1)$$

步骤二：通过私钥 pk 和数字消息 m 生成确定性随机数 k，出于安全性的考虑，随机数 k 的生成一般使用协议 RFC6979；

步骤三：计算 $R = k \times G$，则可得到 R 的横坐标 r，利用公式 2.2 计算 s

$$s = (H_m + r \times pk) / k \quad \cdots\cdots\cdots\cdots\cdots (2.2)$$

步骤四：得到 m 的数字签名 $Sig^m_{ECDSA} = < r, s >$；

2.验证数字签名：

通过计算公式 2.3 是否成立来进行数字签名的验证。

$$s \times R = H_m \times G + r \times P \quad \cdots\cdots\cdots\cdots (2.3)$$

2.2 分布式系统

一般来说，我们将多个都具有一定能力的通过网络连接的节点汇聚在一起的系统称为分布式系统。对于外部用户来说，他们并不需要了解系统中准确的网络拓扑结构。通过网络和统一的对外服务接口便可以使用分布式系统的功能。进入 21 世纪以来，摩尔定律逐渐失效，单机性能逐渐陷入瓶颈，人们越来越寄希望于使用分布式系统对原有系统进行升级改造。

分布式系统与传统的集中式系统有着很大的不同，可以解决集中式系统难以扩展的弊端，但同时也引入了新的问题。

对于一个分布式系统来说，首先面临的问题便是整个系统的一致性问题。在分布式系统中，所有的节点均是使用网络相互连接的，节点与节点之间的数据需要进行很复杂的同步工作以便节点之间的数据保持一致。由于网络存在很大的不稳定性，而且系统中的节点均存在离线的可能性。这就对分布式系统的开发造成了很严重的挑战。

正如著名的 CAP 原理所述：一个分布式系统最多只能在一致性

(Consistency)、可用性（Availability）和分区容错性（Partition tolerance）这三个属性中满足两个属性。其中一致性表示所有的节点需要保证对外的状态是一致的，可用性表示系统可以在一定的时间内对外界的请求作出回应。系统对于请求回应越快，则可用性越高。分区容错性表示系统可以在有一定的错误节点的情况下对外提供正确服务的能力。

在设计分布式系统时，需要对系统的能力进行取舍，以便设计出适用于当前业务场景的系统。

2.3 区块链相关理论和技术

一直以来，在不完全可信的环境中建立一个无需信任节点的分布式数据账本都是学术界的重要挑战。直到区块链技术的发明且在比特币中成功的实现与应用才使得这一设想变得可能。区块链是一个去中心化的数据库，是一串使用密码学方法产生了相互关联的数据[23]。其简化模型图如图 2.3 所示。作为比特币的底层技术，区块链中的每一个数据块都负责保存一个批次的比特币网络交易信息，这些数据一方面用来验证交易的真实有效性，一方面用来使用密码学方法生成下一个批次的区块。

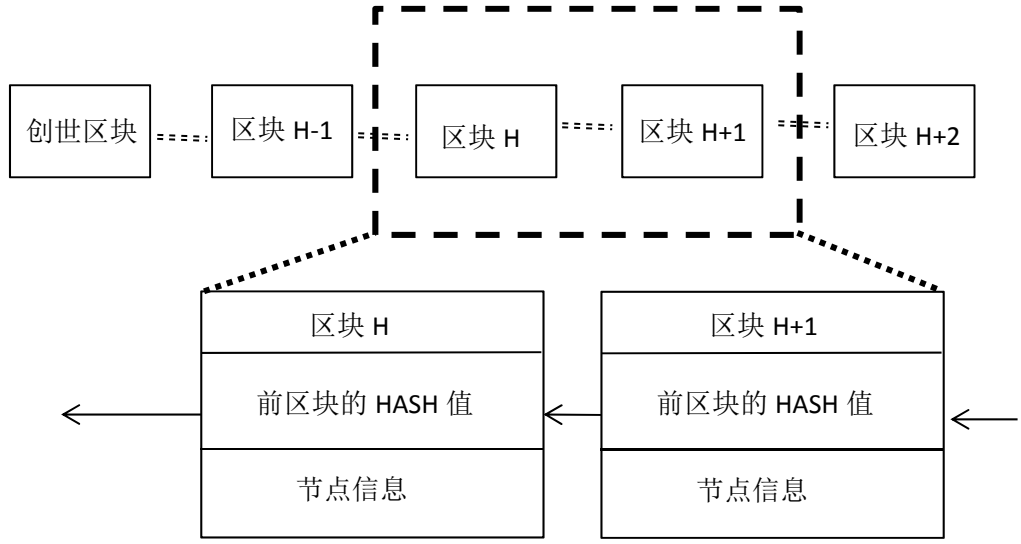

图 2.3 简化的区块链基础模型示意图

2.3.1 区块链的分类

区块链的分类方法非常的多种多样，业界并没有统一的分类方案。本文按照

节点的加入方式进行划分，将区块链划分为许可链和非许可链。

1.许可链

许可链中每个结点都需要经过许可链发行联盟的许可，没有经过许可的节点不可接入系统之中，许可链有时又称为联盟链，网络中所有的节点相互了解彼此的身份，共识由所有节点共同形成。

2.非许可链

非许可链又称为公有链，任何的节点都可以随时的加入链或者退出链，系统中的任何节点都可以参与共识的过程。由共识算法来防止节点作恶。

一般来说，公有链需要使用数字货币来鼓励链上的所有节点进行竞争记账，需要耗费计算资源来形成共识，共识形成的速度比较慢，而有的许可链可以脱离数字货币的形式单独出现。

2.3.2 区块链架构

业界并没有对区块链的架构有着十分清晰且明确的定义，本文依照区块链系统中各个组成部分的功能将其分为以下四个部分，分别为数据存储系统、网络传输协议、共识算法和智能合约系统。

1.数据存储系统

数据存储系统主要解决区块链中的数据如何保存的问题，在一个单一的节点上，区块链数据是链式存储的，一个节点中包含着前一个节点的哈希指纹。在区块内部区块链普遍使用默克尔树来对非常多的交易数据进行保存以及验证。

默克尔树（Merkle Tree），又称为哈希树，是一种用来做快速归纳和大规模数据完整性校验的树形数据结构，由 Ralph Merkle 在论文中提出[24]。默克尔树具有树结构的所有特点。在比特币（Bitcoin）就中采用了简单的二叉默克尔树。一个简单的默克尔树如图 2.4 所示：

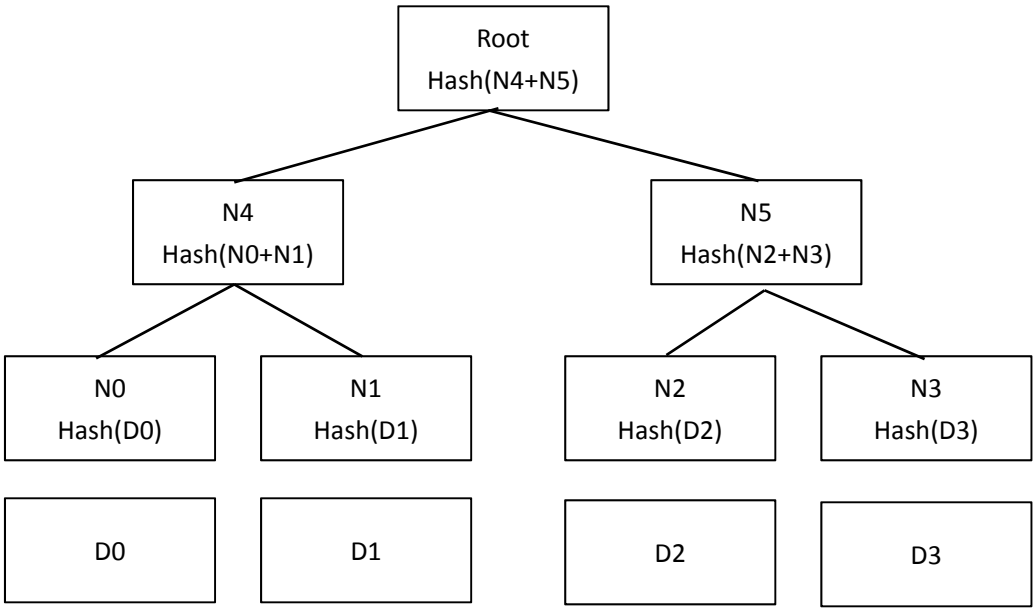

图 2.4 默克尔树示意图

由图 2.4 可以看出，默克尔树的叶节点中保存基础数据，而叶节点的上一层保存有基础数据的哈希值。每个结点都保存有两个孩子节点的哈希值进行哈希运算得出的哈希值。默克尔树的生成由最底层开始，一直到最顶层的根节点为止。

在区块链系统中，如果一个节点希望对数据 D1 进行验证，则无需得到所有数据，只需得到数据 N0 数据 N5 即可计算出默克尔树的根节点的哈希值，只要根节点的哈希值相同，则表明数据相同。

2.网络传输协议

区块链是一个 P2P 拓扑结构的网络系统，节点之间两两通过网络进行连接。而网络传输协议主要负责定义网络之间如何完成数据的交互过程。

Gossip 是一个常见的区块链网络协议[25]。在 Gossip 协议下，节点可以随时的退出与加入区块链网络，新增加的节点一定会在有限的时间内和旧节点达成相同的状态，节点中的网络通信速度影响达成状态的时间。任何节点的离线、宕机都不会对协议的进行产生任何的影响。在 Gossip 协议中，无需任何特殊的中心节点，所有的节点都是完全平等的对等节点。任何一个节点都无需知道网络的整体情况，只需要知道网络的部分拓扑信息。一个节点只要与网络之间是连通的，那么这个节点最终就一定能收到消息。

3.共识算法

共识算法是一个区块链系统的核心部件[26]，是共识节点决定整个区块链系统

的下一步共识过程的算法。共识算法的唯一功能就是规定整个区块链网络中的共识节点的行为。包括在分布式环境下，如何产生相同的节点数据。如何在完全无信任的环境下让所有的节点信任某些数据的真实性。简单来说，共识算法就是为了解决如何让节点对交易数据达成共识的问题[27]。各个节点根据事先协商好的算法来确定分布式账本的记账规则，所有的节点都遵从这一规则产生区块数据，从而保证整个分布式系统的一致性和真实性。

公有链的共识算法一般使用代币机制，如果一个节点对于整个区块链的运行是有益的则会获得代币，如果是有害的则会失去代币。使用奖励机制来激励节点参与到共识过程中。

4.智能合约系统

智能合约是一种在区块链上自动执行的由计算机程序语言编写的合同。代码和数据被存储在区块链网络中的所有节点上，当智能合约达到触发条件自动触发的时候会自动触发执行，任何人或者节点均无法改变或者停止智能合约的执行。常见的智能合约语言如类似 JavaScript 的高级语言 Solidity，在 2014 年由 Vitalik Buterin 提出，并在论文中对其进行了图灵完备的证明[28]。

2.3.3 区块链所涉及的概念

在区块链系统中存在一些基础概念，本节将对这些基础概念进行解释，以防止后文描述的混乱。

1. 节点（node）：一般在一个区块链系统中，节点指的是一个独立运行的最小的单位，根据共识算法的定义，节点一般拥有不同的角色，在共识网络中承担不同的责任。

2. 交易（Transaction）：由于最早的区块链系统是基于数字货币的分布式记账数据库，所以对区块链的操作被称之为交易。虽然现在存在不依存于数字货币的区块链技术，且区块链中保存的数据也并不一定是账本，但我们依然使用交易来指代对区块链中数据的操作。

3. 链（Chain）：链一般指区块链，我们使用上链来表示将数据经过共识过程保存在区块链的分布式数据库中。

4. 矿工（Miner）：在区块链系统中提供计算量的节点被称为矿工节点，其进行计算的过程即为挖矿（mining）。

5. 创世区块（Genesis Block）：区块链是链式的存储结构，整个链的第一个节点即为创世区块。

2.3.4 对区块链的攻击手段

一个区块链系统需要能够面对多种攻击才可以保证安全性。

1. 女巫攻击：攻击者通过伪造身份的方式进行攻击，由于区块链是一个分布式的数据库，女巫攻击即为一个攻击者伪造出多个节点参与共识行为，这样就使得网络中的忠诚节点的数量下降，同时会降低区块链的网络节点效率。对于女巫攻击，许可链采用认证的方式进行防备，任何节点均需要经过联盟链的认证才可以参与到共识中来，攻击者无法伪造节点。公有链则采用工作量证明的方式阻止女巫攻击，每个节点都需要进行工作量的证明，以证明自己确实是一个真正的节点。

2. 双花攻击：由于网络的不稳定，我们无法保证区块链中的所有节点同时达到一致的状态，这就有可能导致区块链中的一部分节点与另一部分节点的区块产生不相同的情况。如果一部分节点接受了区块 1，而另一部分节点接受了区块 2，我们就称这个区块链网络中产生了分叉现象。双花攻击正如其字面意义一样，就是将一笔钱花两次，攻击者需要故意使区块链网络中产生分叉现象，而两个分叉区块中的交易不同，就达到了双花攻击的效果。对于双花攻击，比特币提出了最长链优先原则，即如果一个节点发现存在一条叉链比当前的链更长，则需要将当前链切换到更长的链。同时对每笔交易规定需要产生出多个块之后才能认定交易成立。

2.4 星际文件系统

星际文件系统（IPFS）发明的目的就是为了创建一个持久的且能够分布式存储的网络传输协议[29]。IPFS 存储采用分布式的存储方式，IPFS 节点通过网络相互连接，将文件进行分割保存。在存储文件时，IPFS 首先会根据文件的内容进行计算，得出一个 IPFS 哈希值。此哈希值即为文件在 IPFS 系统中的索引。在查找文件的时候，IPFS 会在全局的哈希表中进行查找，找到对应的节点，之后从节点中获取数据，将数据组合之后返回给用户。本文使用 IPFS 集群进行原始数据的永久性存储。

2.5 本章小结

在本章中对区块链技术的定义、分类、分层等多种属性进行了详细的介绍。对区块链技术去中心化防篡改的属性进行了分析。同时对区块链技术所使用的数字加密技术，包括哈希算法，非对称加密，数字签名等技术的应用与原理进行了介绍。最后对星际文件系统进行了介绍。

第三章 WBFT 共识算法的研究与设计

在第二章中对区块链技术进行了简要的介绍，在本章中将会对区块链技术的共识算法进行分析，来确定在本文提出的业务场景下最适合的共识算法。之后针对 PBFT 算法的不足，提出了客户端域的概念，并基于此提出了 WBFT 共识算法。同时对 WBFT 算法进行了实验。

3.1 常见共识算法的讨论

自从区块链技术蓬勃发展以来，多种不同且成熟的共识算法被人们提出，这些算法有着不同的特性，适用于不同的业务场景，本节将对一些常见的共识算法进行讨论和对比，以便于后续技术的选择。

3.1.1 工作量证明类共识算法

工作量证明（PoW，Proof of Work）类共识算法最早被中本聪在比特币（Bitcoin）中采用。PoW 算法要求矿工使用自身的计算能力求解一个计算非常困难，但是验证非常简单的数学问题。即找到一个随机数 x，使得公式 3.1 成立：

$$h(x \| h(b)) \leq D \cdots\cdots\cdots\cdots\cdots\cdots \quad (3.1)$$

在公式 3.1 中，随机数 x 和区块 b 产生的哈希值需要小于 D，D 是一个难度阈值，由共识算法规定。由于哈希算法的不可逆性，一个矿工为了找到随机数 x 需要不停的进行 $h(x \| h(b))$ 计算。

共识算法规定，如果一个矿工找到了一个合适的随机数 x，则打包第 $b+1$ 区块，并将随机数 x 包含在第 $b+1$ 区块中[30]，其余的矿工可以经过非常简单的计算验证随机数 x 的真实性，如果验证成功，则表示认可第 $b+1$ 区块，将区块加入链中，开始进行下一个区块的运算工作。

打包 $b+1$ 区块的矿工可以获取一定量的代币奖励，奖励直接在第 $b+1$ 区块中体现，此奖励的大小由共识算法进行指定，同时区块中的每笔交易也需要给矿工一定的奖励以便于让自己的交易更快的被确认。其余的矿工在验证 $b+1$ 区块真实性的时候就可以检查区块中的交易以及矿工收益是否符合共识算法的规定。

对于一个节点来说，越早的认可一个合法的区块就能越早的进行下一个区块的挖掘工作，获得下一个区块记账权的可能性越高。所以节点会很快的对合法区块进行确认同时将其广播进入网络中。

工作量证明类共识算法需要矿工进行大量复杂的计算以便于确定下一个记账矿工的产生，所以其产生一个区块的时间非常的冗长。但是在十分冗长的共识时间内，可以有非常多的节点通过网络获得共识信息参与共识过程。所以工作量证明类的算法非常适合于有大量节点的环境中，而且这些节点可以广泛分布于全世界。在 2020 年，使用工作量证明类共识算法的比特币在全世界就拥有一万个节点。冗长的共识时间也导致了工作量证明类共识算法的每秒交易吞吐量非常低，比较适合于对交易吞吐量不敏感的环境。

3.1.2 权益证明类共识算法

由于工作量证明类算法计算力的浪费，人们不断的对其提出各种改进。2012 年，一位爱好者首次提出了权益证明(PoS，Proof of Stake)共识算法，且第一个基于权益证明共识机制的数字货币点点币（Peercoin）于同年上线[31]。在算法中，一个节点对数字货币持有的天数称为币龄（Coin Days）。一个节点所拥有的币龄越长，则产生的区块越容易被承认是合法的区块。我们用函数 $cd(\bullet)$ 来表示一个节点的币龄，则 PoS 类共识算法如公式 3.2 所示：

$$h(n \| h(b)) \le cd(S) \times D \quad \cdots\cdots\cdots \quad (3.2)$$

从公式中可以看出，如果一个矿工的币龄越长，则越容易找到满足公式的随机数 n。虽然 PoS 算法可以在一定程度上解决 PoW 算法所带来的资源浪费，但是其依然会带来新的问题。比如矿工可以对多个分叉进行同时下注，而下注的过程完全不需要消耗资源。为了改进原生 PoS 算法的问题，产生了非常多的基于 PoS 的改进共识算法，比如 Gasper 算法就要求节点缴纳一定数字货币作为保证金，如果节点作恶就会扣除全部的保证金，目前 Casper 算法已经在以太坊中得到了应用。

权益证明类共识算法对工作量证明类共识算法进行了优化，大大减少了挖矿时所造成的的计算力浪费。但是其本质还是基于工作量证明的共识算法，并没有从根本上解决工作量证明类算法极低的交易吞吐量。

3.1.3 传统分布式共识算法

为了维护分布式数据库的系统一致性，实现节点的状态机复制所提出的共识算法为传统的共识算法。主要分为 CFT 算法和 BFT 算法。一般来说，CFT 算法只

能在无拜占庭节点的网络上运行，对于节点的崩溃故障等问题可以很好的解决，但是对于篡改消息或散播错误消息的恶意节点则无法解决。BFT 算法则可以容忍拜占庭错误，但是相比较于 CFT 算法来说牺牲了一定的性能。常见的 CFT 算法有 Paxos 算法和 Raft 算法。

Paxos 算法是一种 CFT 算法，在一个 $n=2f+1$ 的节点中可以容忍 f 个崩溃节点[32]。Paxos 支持多个提案者同时提案，采用消息序列号来解决顺序问题，Paxos 对崩溃节点有一定的容错性，但是对于拜占庭节点无能为力。

Paxos 作为一种分布式算法的标准，虽然非常的优秀，但是难以理解且难以实现。所以在实际的工程中更多采用的是 Raft 算法。

Raft 算法和普通的一致性算法不同[33]，Raft 算法采用强领导模式，Leader 节点具有更大的权利。同时采用了一个简单快捷的机制来进行领导者（Leader）选举，使用了一种统一一致的方法来处理集群成员变换的问题，使得在集群成员变换的时候依然可以继续工作。

作为 Paxos 算法的简化，Raft 算法广泛的应用于分布式系统中，但是存在和 Paxos 算法相同的问题，即无法应对恶意的拜占庭节点。

CFT 类算法可以带来极高的交易吞吐量，节点无需任何的额外证明以驱动共识算法的执行，但是同样导致 CFT 类算法对作恶节点完全无法应对。

3.1.4 改进的传统分布式一致性共识算法

“拜占庭将军问题”一直都是十分经典的计算机通信领域的基本问题[34]。而为了应对伪造信息的拜占庭错误（BFT，Byzantine Fault Tolerance），人们提出了不同的共识算法，其中比较经典的就是在 1999 年提出的 PBFT（Practical Byzantine Fault Tolerance）算法[35]。PBFT 算法引入了多个阶段的共识过程，可以将算法的复杂度由指数级降到多项式级别，并且可以容忍一定量的恶意节点。

PBFT 算法适用于联盟链，有着很严格的准入制度以防止恶意节点的攻击，所有的共识节点需要经过其他节点的认证才能加入共识网络中。如果共识网络中的绝大多数节点达成了共识，就认为通过了这一共识。算法采用三段确认的共识模式，在每一段的确认模式中都需要大多数节点的确认。

同时 PBFT 算法可以不依存于数字货币而运行，对于整个网络的多方参与者

比较友好。

PBFT 算法在允许网络中存在少数的恶意节点的情况下保证系统有较高的交易吞吐量，可以保证 100 个左右的节点达成共识。比较适合于需要较快的交易吞吐量以及存在一定数量的拜占庭节点的网络环境。

3.2 PBFT 算法简介

通过前一节的分析我们可以看出，PBFT 算法比较适合于小规模的分布式共识网络中，同时可以允许整个网络中存在一定的作恶节点，比较适合可信数据存储的业务场景中，下面将对 PBFT 算法进行详细的介绍。

为了便于后文的理解，表 3.1 列出了后文描述算法中需要用到的符号。

表 3.1 算法符号表

符号	定义
<MESSAGE,o>	在共识网络中传递的消息，MESSAGE 即为消息的种类。o 为消息所携带的参数。
σ_i	表示某个节点 i 对消息进行了数字签名
tx	表示一次交易
t	时间戳
v	表示一次共识的视图
n	请求序列号
$< P, pk >$	表示一个公钥私钥对
f	拜占庭节点
PN	主节点
RN	备份共识节点
CN	客户端节点
SN	存储节点
B	区块
h,H	算法中的高低水位
CP	算法检查点
d	消息的摘要

3.2.1 PBFT 算法描述

PBFT 中包含客户端结点(Client)、主节点(Primary)以及备份节点(Backups)三种角色。在网络初始化之时产生主节点，只要主节点并未出现故障，或者其余共识节点认为主节点是恶意节点的情况下，主节点就将会一直担任主节点。如果需要发生主节点更换，将会触发视图切换（View Change）来进行主节点的切换。

PBFT 算法中一共分为三个不同的阶段，分别是预准备阶段（pre-prepare）、准备阶段（prepare）和确认阶段（commit），算法的流程图如图 3.1 所示。详细步骤如下：

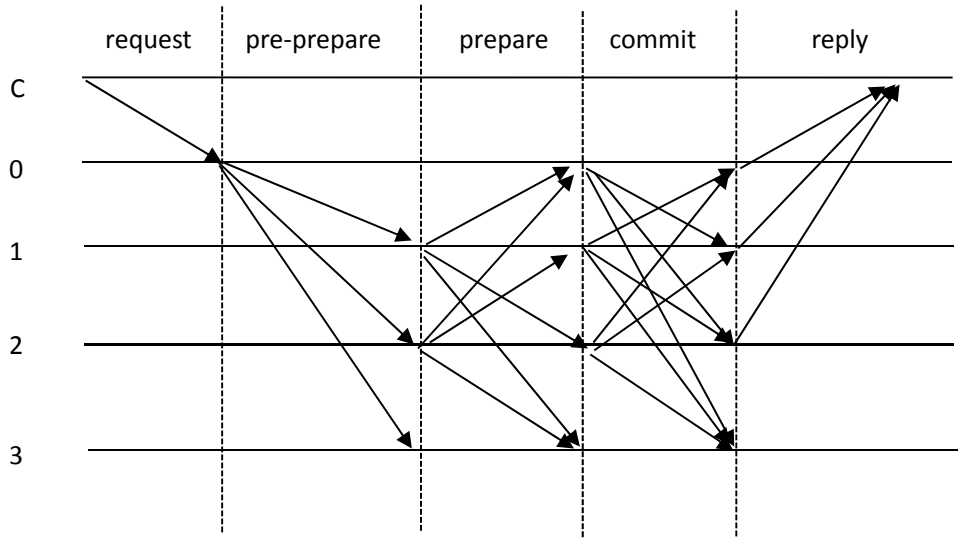

图 3.1 PBFT 共识算法流程图

步骤一 客户端发送请求：客户端c向系统中请求执行操作o,并向主节点发送请求$<REQUEST,o,t,c>_{\sigma_c}$， t是请求的时间戳， σ_c是客户端对消息的签名。

步骤二 预准备阶段：在视图v中，主节点PN接收到来自客户端的请求R后，首先分配一个整数序号n，之后计算出请求R的哈希值H，构建预准备消息$<<PRE-PREPARE,v,n,H>_{\sigma_{PN}},R>$，并将此消息发送给网络中的其他节点。其中$\sigma_{PN}$是主节点$PN$对于该消息的签名。之后主节点进入准备状态。备份节点收到主节点构建的预准备消息之后，验证消息的签名和时间戳。如果通过则进入准备阶段。

步骤三 准备阶段：当一个节点 j 进入准备状态，则需要构建准备消息 $<PREPARE,v,n,H,j>_{\sigma_j}$，并在网络中对准备消息进行广播。当一个节点收到了大于 $2f$ 个节点发送的准备消息时，就进入确认阶段。

步骤四 确认阶段：当节点 j 进入确认阶段之后，则需要构建确认消息 $<COMMIT,v,n,H,j>_{\sigma_j}$，并在网络中对确认消息进行广播。

步骤五 回复客户端：收到足够多（大于 $2f$ ）的确认消息的节点需要构建回复消息 $<REPLY,v,t,c,j,RESULT>_{\sigma_j}$，并将其发送给客户端。当客户端节点收到足够多（大于 $2f+1$ ）的结果相同的执行结果时，就可以认为一个共识流程结束。

通过上文的描述我们可以看出，PBFT 算法的通信复杂度为 $O(N^2)$。共识网络中的恶意节点数量可以满足 $3f+1\le n$，其中 f 为恶意节点数量，n 为网络中所有节点的数量。

3.2.2 PBFT 算法的改进

PBFT 算法虽然将算法的复杂度由指数级降到多项式级别，但是多项式级别的通信复杂度依然会对节点的数量产生限制。算法中每个结点在共识过程中都需要向共识网路中的所有节点发送消息多次，当整个节点中的节点数量大于 100 时就会产生很严重的网络拥堵现象，使得共识行为难以进行下去。同时，PBFT 算法对于网络的性能要求很高，如果有部分节点的网络通信速度很慢，也会严重的对 PBFT 算法的共识速度产生影响。

对于上述问题，学术界对于 PBFT 算法有着各种的优化方案。

Dynamic PBFT 算法[36]解决 PBFT 算法中节点在初始化之后不能动态的进入退出网络的问题，提出了不需要将共识网络重新初始化就能增加新节点的功能。同时对节点提出了信誉评价标准，算法可以将作恶的节点剔除网络。但是从本质上来说，这种方案并不能解决 PBFT 算法的通信复杂度问题，只是增加了共识网络的扩展性，方便节点的加入与退出。

SBFT 算法是一种基于 PBFT 算法的改进算法，由于 PBFT 算法本身对于网络的需求比较高，SBFT 算法提出目的就是将 PBFT 算法的节点扩充到至少能允许 200

个节点同时完成共识[37]。相比于 PBFT 算法，SBFT 算法引入了负责用来进行收集的两个角色，将 PBFT 算法中网状的网络通信模型改为了星型模型，降低了算法的通信复杂度，从而扩展了 PBFT 算法的最大结点数量。但是对于一个分布式网络来说，改为星型模型的代价就是降低了整个系统的稳定性，丧失了去中心化的优势，如果两个收集角色离线，就会影响整个网络的功能。对此，SBFT 算法提出了多备份的收集者节点，如果其中某个收集者节点丧失了作用，则转换为其他能够正常运行的收集者节点继续进行共识。这种设计并不能从根本上解决中心化的问题，收集者节点的数量、如何进行负载均衡等问题依然对于算法在工程实践中的使用产生限制。

Zyzzyva 算法[38]同样是一种采用中心收集者节点模式的算法，与 SBFT 算法不同的是，Zyzzyva 将客户端节点引入了共识过程之中，由客户端节点负责进行备份节点消息的收集工作，同样将网状的通信模型改为了星型。但这就限制了所有的客户端节点必须拥有一定的网络与计算资源，能够保证整个共识的进行，没有网络与计算资源的客户端将无法参与到共识中，同时由于各个客户端节点的网络与计算资源无法统一，也使得算法的共识过程变得不稳定。

HotStuff 将收集者的角色委派给了主节点，同时设计了简单的视图切换方式，在主节点失效的时候整个系统可以快速的产生新的主节点，使得共识网络从无法共识的状态中恢复[39]。同时设计了共识流水线的功能使得共识速度更快，但依然无法避免中心节点失效带来的无法共识的问题。

同时也有基于上述思想的对 PBFT 改进算法，如在论文[40]中提出的 FBFT 算法，采用了节点状态表的形式，即可以允许节点的动态加入与退出，又基于 HotStuff 算法将算法的时间复杂度降低至 $O(n)$，但是由于需要对节点的动态加入与退出进行兼容，反而降低了共识网络的可用性，对于单一节点的宕机变得十分敏感。

本文提出了一种基于分域的轻重客户端节点模型，在整个网络初始化之时就将客户节点分域，每个域内的节点使用相同的收集器角色参与节点共识。能够将算法的复杂度由 $O(N^2)$ 降低至 $O(n)$，同时最大限度的降低网络中单个节点出现问题对整个网络所造成的影响。

3.3 WBFT 算法简介

3.3.1 算法问题假设

我们首先对算法面对的问题进行形式化的描述，以及对于一些问题进行假设，以便后文对算法的描述。

在分布式网络中，网络是不可靠的，一个节点向另一个节点发送的任何消息都存在丢失的可能性，节点也可能出现宕机、离线、甚至发送错误消息，节点向其余节点发送的消息到达时间也难以预计。

基于上述的问题，我们可以假设共识网络中的所有节点都是独立存在的，即一个节点的宕机不会对其他节点造成影响，任意两个节点之间的通信并不依存于其余节点的正常运行。正常的节点均会按照共识算法稳定的向下运行。

在一个联盟链共识网络中，我们可以把参与其中的角色分为三类，分别是客户节点，共识节点，存储节点。其中客户端是交易的发起者，也是交易结果的接受者。共识节点是一个区块链系统的主要功能节点，和其他的共识节点共同组成共识网络，依照共识算法来进行共识行为，多个共识节点可以对交易数据的执行结果达成一致性。存储节点负责保存区块链中产生的数据。一个节点可以同时担任多种角色。

我们首先设定一个区块链系统，参与共识的共识节点集合可以表示为 N，则所有的共识节点的数量可以用 $|N|$ 来表示，我们可以简单的将 $|N|$ 个节点分别编号为 $\{1,2,3,\cdots,|N|\}$，用 RN_i 来表示第 i 个共识节点，用 PN 表示主节点。

假设在节点集合 N 中存在 f 个拜占庭节点，所有拜占庭节点的集合表示为 F，即 $f \in F$，这些拜占庭节点将会在共识网络中广播错误消息。

存在存储节点集合 S，则可以用 SN 来表示一个存储节点。存在客户端节点集合 C，则可以用 CN 来表示一个客户端节点。

同时，我们有如下的功能性假设：

1. 共识节点总数满足 $3f+1 \le |N|$。

2. 如果一个共识网络中主节点是忠诚的，则所有的备份节点在收到主节点发出的消息后一定会得出正确的结论。

3. 网络中传输的消息只可能丢失而不会被篡改，在有限的网络通信时间之