第5章

区块链分片

教师：“同学们，我们先来想象一下一个大型图书馆。最初，图书馆的书籍数量不多，读者也少，所有借阅和还书手续都由一个总服务台处理，效率非常高。但随着图书馆的规模不断扩大，藏书量达到几百万、几千万册，读者也越来越多，如果所有人都涌向这个总服务台，会发生什么？”

学生：“老师，那肯定会排长队，效率变得非常低！甚至可能造成服务台系统崩溃。”

教师：“在区块链世界里，也存在类似的问题。早期的区块链系统就像这个只有一个总服务台的图书馆，所有的交易数据都必须经过网络中所有节点的验证和记录。当交易量激增时，整个系统就会变得非常拥堵，处理速度大打折扣，这就是我们常说的低吞吐量。在现实世界中，面对图书馆人流量大、藏书多的问题，我们通常会怎么做呢？”

学生：“图书馆会设立不同的楼层或者区域，比如一楼是文学区，二楼是科技区，每个区域都有自己的服务台。这样，读者就可以去对应的区域办理手续，不用都挤到一起了。”

教师：“同学的这个例子非常形象地描绘了我们今天课程的主题——区块链分片（Sharding）技术的核心思想。分片技术就是将整个区块链网络“切分”成多个独立的、能够并行处理交易的“分片”。每个分片就像图书馆的一个独立区域。这样一来，整体的处理能力就能大大提高，就像图书馆分散了服务台，能同时服务更多读者一样。”

学生：“听起来好像能大幅提升交易速度。老师，那具体要怎么“切分”呢？不同的分片之间又是怎么协同工作的呢？”

随着区块链技术的飞速发展，其在去中心化和安全性方面的优势日益凸显。然而，如同任何新兴技术一样，区块链也面临着不小的挑战，其中可扩展性（Scalability）无疑是当前限制其大规模应用的核心瓶颈。为了解决这一痛点，分片技术应运而生，它旨在通过“分而治之”的策略，大幅提升区块链网络的处理能力。

为了帮助各位读者系统地了解区块链分片理论，本章将首先带大家回顾一个我们都比较熟悉的领域——数据库分区技术。通过了解传统数据库是如何通过分区来提升性能的，我们将更好地理解为什么这些传统方法无法直接应用于去中心化的区块链网络，从而引出区块链分片的独特需求和挑战。接下来，我们将深入探讨区块链分片的核心概念、不同分片配置方案的实现细节，并剖析分片后不可

避免的复杂性——跨片交易及其解决方案。最后，我们还会介绍如何通过时间片重置方案来保障分片系统的长期安全性和动态适应性。

5.1 区块链与传统数据库：异同与借鉴

区块链是一种分布式账本技术，它以透明、安全且不可篡改的方式记录数据。从本质上看，区块链与传统数据库技术都承担着数据存储的核心功能。然而，数据库技术的发展历史远比区块链悠久，积累了丰富的理论与实践经验。相比之下，区块链技术作为相对新兴的领域，虽然在节点互不可信的场景下展现出强大潜力，但在处理大规模数据和高并发交易时，其扩展性问题已成为其广泛应用的关键瓶颈。例如，许多主流的公链每秒能处理的交易数量远低于传统支付系统。因此，当区块链技术在扩展性方面遭遇挑战时，我们便可从数据库这一传统且成熟的领域中寻求潜在的解决方案和设计思想。

5.1.1 数据库分区的基本原理

数据库技术作为计算机领域的核心组成部分，其发展历程与计算机硬件及应用需求的演进紧密相连。早期数据库系统主要面向有限的数据存储和处理能力。然而，随着互联网的普及、大数据时代的到来以及企业级应用对数据处理能力需求的激增，传统的单一数据库系统在处理海量数据和高并发访问时，其性能瓶颈、存储容量限制以及扩展性不足等问题日益凸显。举例来说，一个大型电商平台可能需要存储数十亿条用户交易记录，并同时处理每秒数万次的并发查询请求。这对于非分布式的数据库系统而言，无疑是巨大的挑战。为了有效应对这些挑战，并确保数据库系统能够持续高效地支持业务发展，分区技术应运而生。这项技术最早且最广泛的应用便是在数据库领域，形成了我们现在所熟知的数据库分区（Database Partitioning）技术。

数据库分区是指将一个表按照行或列分割成多个不同的子表，每个子表称为一个分区，每个分区保存的数据彼此独立，尽量不重叠。分区技术包括垂直分区（Vertical Partitioning）和水平分区（Horizontal Partitioning）。垂直分区是对数据表按列进行划分，各分区中的行数均与原数据表相同，可以减少数据表的宽度。水平分区是指对数据表按行进行划分，各分区中的列数及列标签均与原数据表相同，可以保持原数据表的特性。我们通过图5.1来介绍水平分区和垂直分区。

（1）垂直分区：如图5.1右上部所示，原始总表根据列进行划分。在本例中，“总表”被垂直分割为“分区1（包含序号和名字）”以及“分区2（包含序号和颜色）”。每个分区保留了原始表的全部行，但仅包含部分列，旨在减少单表宽度，优化特定列的查询效率。

（2）水平分区：如图5.1右下部所示，原始总表根据行进行划分。在本例中，“总表”被水平分割为“分区1（包含序号A、B的行）”和“分区2（包含序号C、D的行）”。每个分区具有与原始表相同的列结构，但仅包含部分行，旨在分散数据，提高并发处理能力和查询性能。

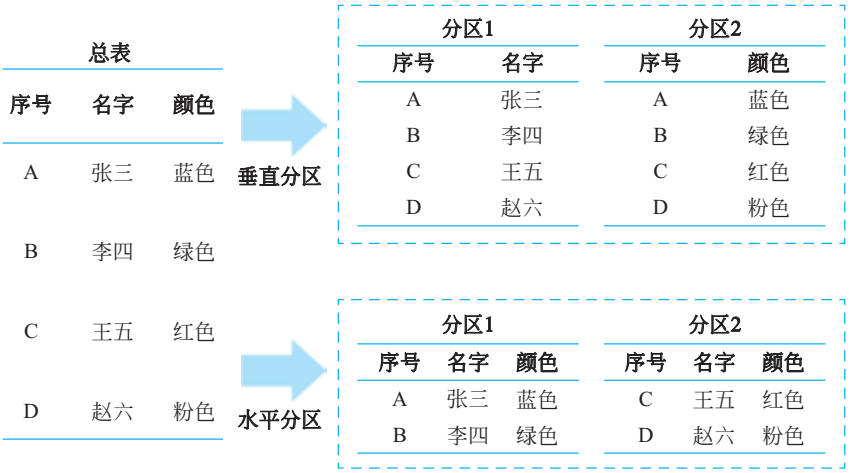

图 5.1 数据库分区技术示例

小提示

术 语 解 析

在数据库领域，分区与分片是两个密切相关且常被交替使用的概念。尽管在不同数据库系统中命名各异，但其核心思想和目标是共通的。具体而言，虽然本书统一采用“数据库分区”这一术语，但读者需了解，在多种数据库系统中，实现类似功能的机制可能被称为“分片”或其他特定名称。在MongoDB、Elasticsearch和Solr Cloud中被称为分片（Shard），在HBase中被称为区域（Region），在Bigtable中则被称为表块（Tablet），在Cassandra和Riak中被称为虚节点（Virtual Node，vNode），在Couchbase中被称为虚桶（Virtual Bucket，vBucket）。

垂直分区相对简单，它根据数据表的列属性进行划分，将不同的列存储在不同的物理存储节点上。例如，一个包含用户基本信息和用户交易记录的表，可以将基本信息和交易记录分别存储在不同的分区中。这种方法有助于优化特定类型查询的性能，因为查询时只需读取所需列的分区，减少I/O开销。然而，垂直分区的主要局限性在于，尽管可以将不同列分布到不同存储节点，但对于每个独立的分区而言，如果其所包含的行数不断增加，单个物理存储节点的存储容量和处理能力最终仍会达到上限。这意味着垂直分区在应对数据总量持续增长的场景时，其扩展能力有限，无法根本解决单点瓶颈问题。

相比之下，水平分区更为复杂，但其在应对海量数据和高并发访问方面具有显著优势。水平分区代表了一种无共享（Shared-Nothing）架构，这意味着各分区是独立自治的，它们之间不共享相同的数据或计算资源（如中央处理器、内存、磁盘）。这种架构的主要优势在于可以实现数据库系统的水平扩展。水平扩展是指通过向现有集群中添加更多的存储节点来分散数据存储负载，从而有效提升数据库的整体吞吐量和响应速度。这正是分布式系统设计的核心目标之一。选择一个正确且高效的水平分区方案至关重要。不当的水平分区方法可能导致数据分布不均匀、查询效率低下甚至数据丢失或读写操作缓慢等问题。接下来，我们将简单介绍一些常见的水平分区方法：

（1）基于键值的分区（Key-Based Partitioning），最常见的是基于哈希的分区，通过对数据中的特定键值（例如，客户ID、应用程序IP地址、邮政编码等）应用一个哈希函数来确定数据所属的分区。如图5.2所示，哈希函数的输出作为分区唯一标识符，用于将传入数据存储到相应的物理分区。这种策略的优势在于哈希函数通常能够使数据在各个分区之间分布相对均衡，从而有效避免部分分区负载过高的问题，提高整体负载均衡，并依赖哈希函数的防碰撞特性防止不同数据的索引冲突。然而，其主要弊端在于对范围查询的效率低下，因为哈希会打乱数据的顺序性，导致范围查询可能需要扫描多个甚至所有分区。此外，一旦选定哈希函数并开始数据写入，后期更改哈希函数或进行大规模数据重分布的成本较高，且分区依据与应用逻辑通常不直接相关。

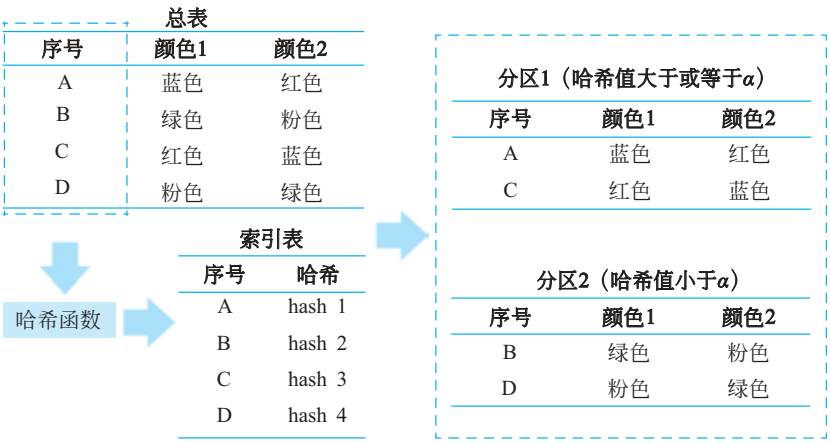

图 5.2 基于键值的分区

（2）基于范围的分区（Range-Based Partitioning）依据数据中某个键的值域范围对数据进行分区，例如按照用户ID的数值范围、时间戳范围或地理位置范围等来划分数据。如图5.3所示，每个分区负责存储特定范围内的数据。这种方法的优点在于实现相对简单直观，分区规则明确，且对范围查询的效率很高，因为相关数据通常存储在相邻或同一个分区，减少了跨分区查询的开销。它也易于根据数据的自然增长趋势添加新的范围分区。然而，其主要缺点是数据分布容易不均衡。如果数据增长趋势集中在某个特定范围（例如，在某个时间段内新用户激增），可能导致该分区的数据量和访问压力过大，影响系统性能和扩展性。因此，分区键的选择至关重要，特别是避免选择随时间单调递增或具有集中趋势的键。

（3）基于目录的分区（Directory-Based Partitioning，如图5.4所示）通过维护一个独立的目录表来实现数据到分区的映射。每次数据读写操作之前，系统都需要查询这个目录表，以确定目标数据位于哪个分区。这种方法的优势在于其极高的灵活性，不受限于固定的范围定义或哈希函数，可以采用任何自定义的逻辑或算法为分区分配数据条目。在系统扩展时，通过更新目录表可以非常灵活地添加新分区或调整现有分区的映射关系，实现更精细的数据负载均衡。然而，其主要缺点在于引入了额外的性能开销，因为每次查询或写入操作都需要额外一次对目录表的查找，这会增加网络延迟和计算负担。同时，如果目录表本身未实现高可用，它将成为整个系统的潜在单点故障风险，且维护目录表的复杂性也增加了系统的负担。

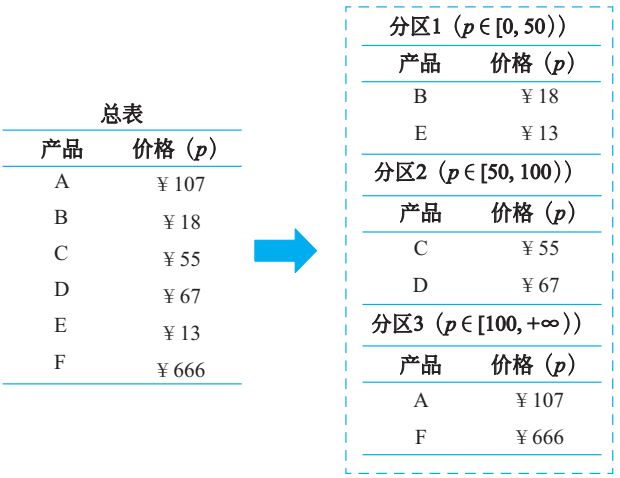

图 5.3 基于范围的分区

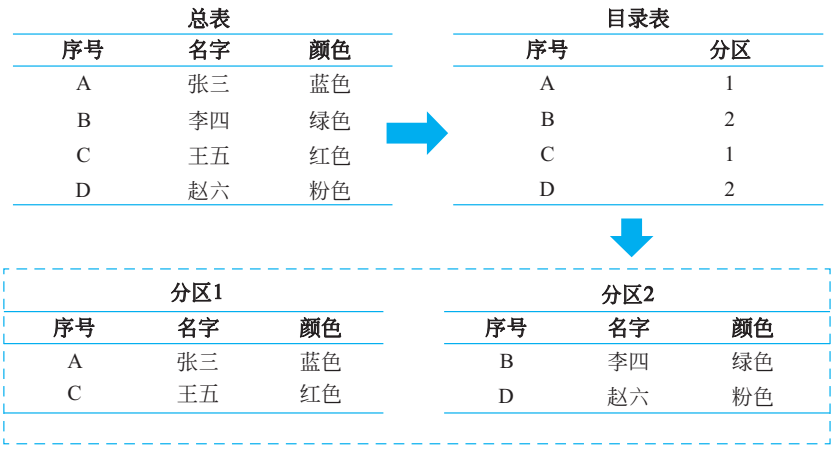

图 5.4 基于目录的分区

从上述对三种常见水平分区方法的分析中可以发现，根据场景特定需求来选择一个合适的“分区依据”是实现有效数据划分的核心思路。无论是基于键值的哈希运算结果、基于数据值的预设范围，还是通过独立的目录表，其本质都是建立数据与特定分区之间的映射关系。这一核心思想在未来的区块链分片技术设计中亦将得到延续和发展。

5.1.2 为什么数据库分区不能直接用于区块链？

尽管数据库分区技术为解决传统中心化数据库的扩展性问题提供了宝贵的经验，且其核心思路——选取分区依据——对区块链分片具有重要的借鉴意义，但数据库分区技术并不能直接应用于区块链系统。这主要源于两者在敌手模型上的根本性差异。

传统数据库系统运行于一个可信环境中。系统假定数据库管理员及承载数据库服务器的物理节点是可信的、非恶意的。系统的安全性主要通过访问控制、权限管理等中心化机制来保障。尽管数据被分区存放于不同节点，但这些节点仍处于统一的、受信任的管理域内。区块链则为在不可信的环境中协作而设计。它从根本上假定网络中的任何参与者都可能是不可信或潜在恶意的（即拜占庭节点）。系统的安全性与数据一致性并非依赖于对参

与者的信任，而是通过共识机制、激励机制等手段来确保。数据库分区方案并未考虑“拜占庭容错”问题。若将其直接应用于区块链，无异于将部分账本数据全权委托给不可信的节点。由于缺少固有的安全机制，恶意节点可以轻易地合谋控制某个分片，从而篡改数据、进行双花攻击或拒绝服务，而整个系统缺乏有效的发现与惩罚机制。区块链分片则必须设计合理的随机抽样、跨片交易验证和节点重组等机制，以抵御此类针对特定分片的恶意行为。同时，这些机制也是我们后续几节的重要内容。

尽管数据库分区提供了数据划分的有效思路，但由于区块链的敌手模型，直接移植数据库分区技术是不现实的。区块链的分片技术必须在继承数据划分理念的基础上，重新设计一套能够适应其分布式共识和安全模型的复杂方案。这些挑战主要体现在以下几方面，它们也是本章后续内容将深入探讨的核心议题：

- 5.2节“区块链分片：解决扩展性困境”将简要介绍区块链扩展性问题在当前系统中的具体表现，并阐释区块链分片的相关基本概念。本节旨在帮助读者理解区块链分片技术的必要性及其所期望达成的目标，为后续内容奠定基础。
- 5.3节“区块链分片的核心机制”：详细阐述分片配置方案，即如何在去中心化环境下确定数据的分区依据，并介绍随机数生成方法在分片中用于节点分配的关键作用。
- 5.4节“跨片交易：分片世界的协同难题”将聚焦于分片化区块链中面临的一大挑战。本节将首先明确原子提交作为跨片交易的准则与目标，随后深入分析常见的跨片交易处理机制，探讨如何在无须信任的去中心化环境中，确保涉及多个分片交易的原子性、一致性、最终性以及安全性。
- 5.5节“分片重置：应对节点变化与安全威胁”将分析分片化区块链在动态环境下的稳定性问题。本节将解释为什么需要分片重置，即如何应对网络中节点的动态加入或退出以及潜在的安全威胁，并探讨主流的分片重置机制，以确保整个分片化区块链系统的鲁棒性与持续安全性。

5.2 区块链分片：解决扩展性困境

重点：区块链分片主要流程

区块链扩展性问题都有哪些具体表现？区块链分片协议需要满足什么样的特性？一个完整的区块链分片协议通常包括哪几个关键部分？

5.2.1 区块链扩展性问题的表现

我们已初步探讨了区块链系统在面对大规模应用时所面临的扩展性挑战。为更深入地理解并有效设计符合区块链特性的分片方案，本节将首先详细剖析区块链扩展性问题的具体表现。通过精准识别这些性能瓶颈，我们将能更清晰地把握分片技术致力于解决的核心痛点，从而为理解后续的分片方案设计提供坚实的基础。

区块链的可扩展性问题，从其技术性能指标来看，主要体现在交易吞吐量瓶颈和交易延迟两方面。交易吞吐量是衡量区块链系统在单位时间内处理交易数量的关键指标。当前

主流公有链，例如比特币网络，其处理能力约为每秒7笔交易（Transactions Per Second，TPS），而以太坊的吞吐量虽有所提升，但也仅能达到每秒15~30笔交易。这与传统金融系统每秒处理数万笔交易的能力，以及诸多商业应用对高并发处理的需求相比，存在显著差距。与此同时，交易延迟则反映了从交易发起至被网络确认并记录于区块链的时间间隔。例如，比特币网络的交易确认延迟通常在10分钟左右，而以太坊的确认时间约为15秒。尽管相较于比特币有所改善，但此类延迟对于高频交易和实时性要求较高的应用（如金融交易或即时支付场景）而言，仍难以接受，显著影响用户体验和业务流程效率。

除了上述直接影响用户感知的性能指标外，区块链的扩展性问题还体现在其底层资源消耗与固有限制。随着区块链数据的持续增长，每个全节点必须同步并存储不断累积的全部历史交易记录。这种机制对节点的网络带宽和本地存储容量提出了严峻挑战，可能导致显著的存储负担，并制约交易在网络中的传输速率。更深层次地，区块链的扩展性问题还涉及去中心化与可扩展性之间的内在权衡。当前大多数区块链系统通过限制区块大小或延长出块时间来确保所有节点能够同步和验证所有交易，从而维护网络的去中心化程度和安全性。然而，这种策略却直接导致了吞吐量的下降和交易延迟的增加。若无法在不牺牲去中心化特性的前提下有效提升可扩展性，区块链系统在面对大规模用户或高频交易时将难以满足实际需求，从而严重限制其商业化进程和普适性应用。

区块链的可扩展性问题已成为打破其广泛应用的关键瓶颈。解决此问题需要在提高交易吞吐量、缩短交易确认时间、优化网络带宽与存储利用效率的同时，确保区块链固有的去中心化、安全性和不可篡改性等核心特性不被削弱。这些挑战正是分片技术应运而生并持续研究的核心驱动力。

5.2.2 区块链分片技术的简介

在5.2.1节中，我们详细剖析了当前区块链系统所面临的扩展性问题。这些表现形式共同构成了区块链技术大规模应用的主要障碍。为了系统性地解决这些问题，本节将进一步深入，首先对区块链分片进行形式化定义，明确其在安全和性能方面的核心要求。在此基础上，我们将阐述为实现这些目标，分片系统需要完成的各项关键任务，从而为读者构建对区块链分片协议设计原理的全面认知。

定义 5.1

假设在区块链网络中共有n个节点，它们具有相同的算力，其中有f个拜占庭节点。对于区块j上的一笔交易i，可以表示为整数$x_i^j \in \mathbb{Z}_N$，其中$\mathbb{Z}_N$是模N的整数环。所有节点都可以通过访问一个指定的约束函数C：$\mathbb{Z}_N \to \{0,1\}$来确定每笔交易的有效性。对交易进行分片需要寻找一个运行在节点间的协议Π，该协议的输出结果是一个集合X，其中包含k个独立的片$X_i = \{x_i^j\}(1 \leqslant j \leqslant |X_i|)$。集合$X$满足以下条件：

- 一致性。对于给定的安全参数λ，诚实节点认同X的概率至少为$1-2^{-\lambda}$。
- 有效性。集合X满足指定的约束函数C，即对任意$i \in \{1,2,\cdots,k\}$、任意$x_i^j \in X_i$，都有$C(x_i^j)=1$。
- 可扩展性。k可以随网络规模的增长近似线性地增长。

区块链分片的核心思想在于将整个网络划分为多个规模较小的并行处理单元，每个单

元独立负责处理一个特定的交易子集，这个独立并行的单元称为一个“片”（Shard）。这种设计旨在通过并发处理来提升系统整体性能。区块链分片定义5.1[1]中提及的三个核心属性——一致性、有效性与可扩展性——可分别归类为安全和性能两方面的考量。从安全方面来看，分片协议的设计必须严格遵循一致性和有效性两大原则。一致性要求即使在存在一定比例恶意节点（拜占庭节点）的情况下，诚实节点也必须对分片处理的交易结果达成高度一致意见。这确保了分片系统在去中心化环境中的可靠性和安全性，防止恶意行为导致账本分歧。而有效性则强调所有被分片处理并最终记录在区块链上的交易，都必须严格遵守系统预设的规则和约束条件（如交易格式正确、签名有效、资金充足等）。这意味着每笔交易在被纳入区块链之前，都必须经过严格的验证，从而保证了链上数据的正确性和可信度。从性能方面来看，可扩展性是分片技术最重要的设计目标。它旨在解决传统区块链因所有节点处理所有交易而导致的性能瓶颈。通过允许分片数量k随着网络总节点数n的增长而近似线性增长，分片技术期望能按比例提升系统的总吞吐量，使区块链能够支撑更多用户和更高频率的交易，从而满足主流商业应用的需求。

为了实现上述区块链分片所定义的一致性、有效性和可扩展性这三个关键属性，一个功能完整的分片方案必须包含五个关键组件。这些组件涵盖了从区块链节点的身份管理，到分片的划分，再到分片内部及分片之间的协调与共识达成，直至应对网络动态变化的分片重置与安全保障。理解这些核心组件对于把握分片协议的运作机制至关重要。区块链的分片协议通常由以下五个关键组件组成：

（1）**节点身份的建立**：为了使节点能够安全、有序地加入并参与分片协议，每个节点首先需要建立一个唯一的身份标识。这通常涉及生成加密密钥对，其中公钥可作为节点的公开身份，并通过类似IP地址等网络标识进行关联。在开放的公有区块链网络中，系统必须设计有效的机制来防范女巫攻击，即防止恶意实体伪造大量虚假身份以获取不成比例的控制权。若节点在接入区块链网络时已具备唯一的身份标识，则无须在分片流程中另行创建独立的身份验证体系。

（2）**分片的生成**：在节点身份建立后，分片系统需要根据预设的分片算法（通常基于节点的身份标识或随机分配结果），将网络中的全部节点逻辑划分为多个独立的分片。每个分片负责处理一部分交易。分片形成后，其内部的节点需要通过通信协议来发现并识别同一分片内的其他成员。对于区块链而言，每个分片的覆盖网络通常被设计成一个包含其内部所有成员的完全图或高效的点对点拓扑，以确保分片内部成员间通信的高效性和可靠性，为后续的片内共识奠定基础。值得注意的是，节点分配机制的设计直接关系到分片系统的整体安全性。具有强规律性的分片算法会导致系统产生严重的安全漏洞：拜占庭节点可能利用规律对特定分片实施定向渗透。鉴于单分片的节点规模及资源权重远低于全网总量，攻击者能够以较低成本在局部形成优势，从而发起针对性攻击（如“日蚀攻击”）。因此，分片防御的核心在于引入随机性，确保节点分配过程具备不可预测性与抗操纵性，从而有效瓦解有预谋的协同攻击。

（3）**片内交易处理**：分片形成并完成内部通信拓扑构建后，其核心任务便是对所分配的交易子集达成共识。具体而言，分片内部的每个节点都将运行同一个标准共识协议，以对一组提交到该分片的交易达成最终一致性确认，并打包成区块。在此过程中，所有诚实的成员都必须在该分片内部就提议的区块达成一致性共识，确保即使存在一定数量的恶意

节点，也无法破坏该分片账本的完整性和唯一性。

（4）**跨片交易处理**：在分片化区块链系统中，一笔交易可能涉及多个分片（例如，发送方账户在一个分片，接收方账户在另一个分片）。在这种情况下，系统必须保证这笔跨片交易的原子性，即要么所有涉及分片的操作全部成功，要么全部失败，不能出现部分成功的情况。此外，所有相关分片的数据必须保持最终一致。通常情况下，该过程需要设计特定的跨片交易处理机制，例如使用中继（Relay）交易来实现各跨片交易相关分片的状态同步。

（5）**时间片重置**：时间片通常是指一个固定的时间周期，在一个时间片内，同一分片的节点将负责维护该分片，并按照一定的共识规则来确认和打包交易。为了保障分片系统的长期安全性和鲁棒性，尤其是在面对潜在的自适应攻击（如攻击者长期集中攻击某一个或少数几个分片）时，分片的划分不能是永久不变的。为了抵抗此类攻击并增强安全性，分片的划分需要周期性、随机地进行重置和重组。此次随机重置将应用于下一个时间片，打乱原有节点与分片的对应关系，从而降低攻击者长期控制某个分片的可能性，确保系统的整体安全性。

上述五项组件是构建一个完整、高效且安全的区块链分片系统所必须解决的核心挑战。节点身份的建立所依赖的密码学基础已在第2章的密码学工具中详细介绍，片内交易处理的本质是片内节点执行共识算法，共识算法是本书第3章的核心内容。分片的形成、跨片交易处理机制以及分片重置这三个组件将在5.3节、5.4节、5.5节详细介绍。通过对这些组件的逐一剖析，读者将能全面理解区块链分片技术的运作原理及其复杂性。

5.3 区块链分片的核心机制

重点：掌握主流的分片配置方案

主流的分片配置方案分成哪几类？不同分片配置方案分别适用于什么样的场景？什么样的随机数生成方法能够满足区块链分片技术的需求？主流的随机数生成方法有哪些？

5.3.1 主流的分片配置方案

区块链分片技术旨在通过并行处理提升系统吞吐量，其核心在于将整个网络有效地划分为多个独立的、可并行运行的分片。在分片协议的设计中，分片的生成是保障分片过程去中心化和安全性的关键环节。具体而言，区块链的一个分片由一定数量的验证节点构成，这些验证节点遵循预设的共识算法对分片内的交易进行确认和区块打包。为确保网络的整体安全性，每一个分片内的节点构成需具备充分的去中心化程度，以有效规避单个验证器或少数验证器对整个网络的潜在控制风险。因此，分片配置方案的设计显得尤为重要，它不仅决定了网络节点如何被分配到不同的分片，还直接影响着分片网络的安全性、去中心化程度和整体性能。一个安全的分片配置方案是区块链分片架构的基石，其设计优劣直接关系到整个分片系统能否成功应对扩展性挑战并保持区块链的固有特性。根据区块链系统对

性能和安全的需求，常见的分片配置方案主要包括四种：静态配置（Static Configuration）、动态配置（Dynamic Configuration）、单成员轮换（Single-member Rotation）和多成员轮换（Multi-member Rotation）。为了清楚地对比这四种分片配置方案的优缺点，表5.1对这几种配置方式进行了对比。

表 5.1　分片配置方案的优劣比较

配置方式	优　点	缺　点	适用场景
静态配置	高效、简单、无须频繁更新	缺乏灵活性，不适合公有链	联盟链、需要可信机构验证节点身份
动态配置	随机性强，抗攻击能力高	计算资源消耗较大	公有链、强去中心化场景
单成员轮换	平滑更新，兼顾稳定性与动态性	更新速度较慢，可能积累不可信成员	部分去中心化场景
多成员轮换	更新效率高，抗攻击能力强	实现复杂，可能增加同步开销	高动态性、高安全性需求场景

（1）**静态配置**是一种最为简单的分片配置方案，其特点是分片内的节点不会定期更改。这种方式在联盟链中较为常见，主要因为联盟链的参与方通常是可信的实体，例如已知的企业或组织。在静态配置中，分片内的节点彼此了解身份，且是经过预先设定的可信节点。在这种情况下，由于所有参与者均已被信任，并且身份明确，系统通常不需要考虑诸如女巫攻击等安全威胁。代表性的使用静态配置方案的案例有Hyperledger Fabric[2]和RSCoin[3]。作为一种面向企业的联盟链，Hyperledger Fabric采用了静态配置方案，确保参与各方在系统内维持高效协作的同时，降低了安全风险。RSCoin是一种基于中心化信任的数字货币系统，利用可信机构（例如央行）来保证系统的安全性和效率，因此它可以通过简单的静态配置来实现其共识。

（2）**动态配置**是一种基于随机选择分片内节点的机制。在这一机制中，分片内节点是在每个时间片开始时，通过随机数生成算法来进行分片节点配置。这样可以确保分片内节点动态变动，从而增强系统的安全性和抗攻击能力。动态配置比较适合公有链，因为它能够降低某个单一实体控制分片的风险，从而提高去中心化程度。动态配置的代表性案例有Algorand[4]和SnowWhite[5]。Algorand通过加密随机数生成算法选择分片节点，确保片内节点选择的公平性和随机性，从而防止攻击者事先预测或控制分片节点的组成。SnowWhite在随机选择分片节点的基础上，还确保了系统的安全性和高效性，适用于去中心化网络中的动态参与者场景。

（3）**单成员轮换**是分片内节点更新的一种滑动窗口机制。在这种方式下，分片每次更新时，会有一个新节点加入，同时最早加入的一个节点离开。这种机制以渐进方式替换分片内节点，从而在保持分片稳定性的同时，允许其不断演进。这种方式适用于既需要动态更新，又希望在一定程度上保持分片一致性的场景，例如混合型链或部分去中心化的网络环境。ByzCoin[6]采用单成员轮换方式，通过渐进式节点更新，提升了系统在大规模分布式网络中的容错性和性能。

（4）**多成员轮换**是单成员轮换的扩展版本。在这种方式下，每次分片更新时，会有多个

节点同时被替换。这种机制的目标是在动态性和性能之间找到更好的平衡，并提高分片更换的效率。通常，多成员轮换采用更为复杂的机制，例如密码抽签（Cryptographic Sortition）来选择需要更换的节点。这种方式能够快速更换大量分片内节点，同时保证系统的随机性和安全性。OmniLedger[7] 通过密码抽签机制，从所有分片中选出需要交换的节点，并动态更新分片组成，以此提高区块链系统的并行处理能力和安全性。

上述四种分片配置方案各自适用于不同的区块链网络特性与安全需求。静态配置因其简单性和信任假设，多见于联盟链等节点身份明确且可信的场景，强调效率和稳定性。与之相对，动态配置、单成员轮换和多成员轮换则更侧重于通过引入随机性和动态性来增强公有链的去中心化程度和抗攻击能力，它们在保证性能的同时，致力于降低恶意实体控制分片的风险。无论是哪种分片配置方案，尤其是在去中心化程度要求更高的公有链环境中，随机数生成方法都扮演着不可或缺的核心角色。分片配置方案的有效性在很大程度上取决于其是否能够生成高质量、不可预测且可验证的随机数，以确保节点被公平、均匀地分配到各个分片，并周期性地进行重组。安全的随机数生成可以有效抵御恶意行为，从而保障整个分片化区块链系统的安全性。鉴于此，下一节将深入探讨各种随机数生成方法及其在区块链分片配置中的具体应用与安全考量。

5.3.2 分片配置的核心：随机数生成方法

在 5.3.1 节对分片配置方案的探讨中，我们已明确认识到随机数生成在区块链分片中扮演着极其重要的角色。它不仅是实现分片节点动态、公平分配的关键机制，更是保障整个分片系统去中心化程度与安全性的核心要素。然而，并非所有的随机数生成方法都能满足区块链分片这一复杂分布式环境的严格需求。传统的伪随机数生成器往往因其可预测性而无法在不可信的区块链环境中提供足够的安全性，而完全依赖物理随机源又面临可验证性和效率的挑战。因此，分片协议所依赖的分布式随机数生成机制通常需要满足一系列严苛的关键特性，以确保其在去中心化网络中的安全性、公平性和可用性。下面，我们将首先详细阐述这些至关重要的特性，随后再深入分析目前主流的分布式随机数生成方法及其在区块链分片中的应用。

- 公开可验证性（Public-Verifiability）：一旦生成新的可用随机信标值（Random Beacon Value），各方均可以通过公开信息来验证该值的正确性。
- 无偏性（Bias-Resistance）：任何单个参与者或者串通的恶意节点都不能通过影响未来的随机信标值来使自己受益。
- 不可预测性（Unpredictability）：所有的参与者都无法计算或预测未来的随机信标值。
- 可用性（Availability）：任何单个参与者或者串通的恶意节点都不能阻止进程的正常执行。

在第 2 章中，我们对可验证秘密共享有了初步的了解。本节将在此基础上，对目前应用于区块链分片中的主流分布式随机数生成方法进行详细介绍。这些方法旨在解决去中心化环境中生成安全、公正且可验证随机数的难题，以满足分片配置对高强度随机性的需求。我们将重点探讨以下几种代表性技术：可验证随机函数 [8]、可验证秘密共享 [9]、公开可验证秘密共享 [10] 和可验证延迟函数 [11]。这些技术各有侧重，共同构成了区块链分片安全随

机数生成的核心支撑。

（1）**可验证随机函数**（Verifiable Random Function，VRF）。简单来说，Alice 向 Bob 发送一个输入 x，并要求 Bob 使用特定的函数 f_s 进行计算并返回计算结果 $v = f_s(x)$。该结果依赖于一个只有 Bob 知道的秘密值 s。结果 $v = f_s(x)$ 是唯一的，并且与同样长度的真随机数 v' 在计算难度上是不可区分的。Alice 想要确定 Bob 提供的是唯一正确的计算结果，而 VRF 就可以实现 Alice 的验证需求，同时保证 Bob 的秘密值 s 不公开。从形式上来说，VRF 解决了伪随机函数不可验证的问题。为什么这么说呢？举个例子，计算 $f_s(x_1), f_s(x_2), \cdots, f_s(x_n)$ 的一方声称其相应的输出分别为 $o_1, o_2, \cdots, o_n$。然而，在不知道 s 的情况下，验证者无法验证 $f_s(x_i) = o_i$。若是将 s 公布出来，那么函数 $f_s\ x_i$ 的输出就会变得可以预测，所有人都可以正确计算出该结果，不再与真正的随机数难以区分。那么，VRF 是如何做的呢？VRF 要求知道秘密值 s 的一方在公开结果 $v = f_s(x)$ 的同时也公开一个证明 proof_x，这个证明可以验证结果 v 的确是由 $f_s(x)$ 计算出来的，而不用公开 s。重要的是，知道 s 的一方对于每一个 x 只能构造一个唯一的有效证明，来证明 v 是输入 x 通过函数 f_s 计算出来的结果。当然，对于证明本身而言，并没有唯一性的要求，通常情况下该证明过程是通过交互式零知识证明实现的。

（2）**可验证秘密共享**（Verifiable Secret Sharing，VSS）。秘密共享（Secret Sharing）是一种在一定数量的参与者中分发秘密 S 的方案，每个参与者都会得到该秘密的一部分。这些参与者可以通过将各自持有的秘密碎片进行合并，从而恢复秘密 S。(t, n)-秘密共享是指对秘密 S 生成 n 个秘密碎片，分别分发给 n 个参与者，只需要收集不少于 t 个参与者所持有的秘密碎片，即可恢复出秘密 S。Shamir 的秘密共享方案是基于多项式插值的，其关键思想是：对于给定的 t 个 x 坐标各不相同的点 $(x_1, y_1), (x_2, y_2), \cdots, (x_t, y_t)$，只有一个度为 $t-1$ 的多项式 $p(x)$ 经过所有的点。然而，Shamir 的秘密共享方案基于一个重要的假设：所有的参与者都假设自己获得了正确的秘密碎片。这就限制了该方案的应用场景，使其不适用于具有拜占庭节点的系统。因此需要一个可验证的秘密共享方案来防范恶意参与者。

（3）**公开可验证秘密共享**（Publicly Verifiable Secret Sharing，PVSS）。PVSS 方案可以使任何一方验证秘密碎片的正确性而不泄露任何关于秘密 S 或碎片的信息。方案中包含受托人（Trustee）和经理人（Dealer）两种角色。在秘密碎片分发阶段，对于每个受托人 i，经理人生成一个加密的碎片 $E_i(s_i)$ 以及一个非交互式零知识证明，用于证明 $E_i(s_i)$ 正确地加密了秘密值 S 的碎片 s_i。在秘密重建阶段，受托人通过收集 t 个解密后的碎片来恢复 S。然后受托人公开 S 和所有的碎片，也公开所有的解密证明以表明这些碎片都被正确解密了。通常来说，PVSS 的流程分为三个步骤：①经理人选择一个度为 $t-1$ 的秘密共享多项式 $s(x) = \sum_{j=0}^{t-1} a_j x_j$，并且为每一个受托人 $i \in \{1, 2, \cdots, n\}$ 创建一个碎片 $S_0 = G^{s(0)}$ 对应的加密碎片 $\hat{S}_i = X_i^{s(i)}$。同时，经理人创建承诺 $A_j = H^{a_j}$，其中 $H \neq G$。并为每个碎片创建一个非交互式零知识证明 $\hat{P}_i$。之后，经理人公开 $\hat{S}_i, \hat{P}_i, A_j$。②每个受托人 i 使用 $\hat{P}_i$ 和 A_j 来验证加密碎片 $\hat{S}_i$，如果验证成功，则公开解密后的碎片 $S_i = \hat{S}_i^{x_i^{-1}}$ 与解密证明 P_i。③经理人检查 S_i 和 P_i 的有效性，将验证不通过的碎片丢弃；如果剩下的解密碎片不少于 t 个，那么就通过拉格朗日插值法恢复秘密 S_0。

VRF、VSS与PVSS在密码学协议中扮演着各不相同的角色：VRF侧重于由单一实体生成可公开验证的随机数；而VSS和PVSS则要求多方共同生成并共享秘密。对VSS与PVSS进行深入比较，我们可以发现VSS旨在防止恶意参与方持有秘密碎片，为此，每个参与者都拥有一套验证机制来确认其所持碎片的有效性。相较之下，PVSS不仅允许参与者验证自身的碎片，而且任何人都能够公开证明该参与者接收到了正确的碎片。然而，当前绝大多数PVSS方案在计算上都显得复杂且效率低下。尤其值得注意的是，PVSS通常被视为“一次性”方案，而VSS方案及其衍生出的分布式密钥生成算法则能够生成可重复使用的分布式阈值密钥对。

（4）**可验证延迟函数**（Verifiable Delay Functions，VDF）。VDF需要按照特定数量的顺序步骤进行计算，同时产生一个可公开验证且唯一有效输出。VDF在分布式系统中有许多应用，包括随机信标（Randomness Beacons），共识协议中的领导人选举算法（Leader Election）和复制证明（Proofs of Replication）。VDF是一个函数$f:\mathcal{X}\to\mathcal{Y}$，它需要规定的时间来进行计算，即使在并行计算机上也是如此。一旦计算出来，输出就可以被任何人快速验证。对于每一个输入$x\in\mathcal{X}$，必须有唯一一个有效的输出$y\in\mathcal{Y}$与之对应。具体来说，VDF实现包括三个算法：① $\text{Setup}(\lambda,T)\to pp$。对于给定的一个安全参数$\lambda$和时间边界$T$，并输出公共参数$pp$。② $\text{Eval}(pp,x)\to(y,\pi)$。该算法对于给定的输入$x\in\mathcal{X}$，输出$y\in\mathcal{Y}$和一个证明$\pi$。③$\text{Verify}(pp,x,y,\pi)\to\{\text{accept,reject}\}$。如果$y$是通过VDF对输入$x$的正确计算，那么输出accept。除此之外，一个VDF还必须满足三个属性：ϵ计算时间（ϵ-Evaluation）、顺序性（Sequentiality）和唯一性（Uniqueness）。

与前面三个随机数生成方法相比，VDF的主要区别在于，它不是为了生成随机数本身（尽管其输出可以作为随机源），而是为了强制性地引入时间维度上的顺序性并提供可验证的延迟。这使得VDF可以用于构建无须预设信任的延迟机制，防止攻击者通过暴力计算提前预知随机数，或者用于抵御时间同步攻击。因此，VRF、VSS和PVSS更多关注如何安全、公平地生成和分发随机数，解决的是随机性来源和分布式验证问题。而VDF则关注如何通过引入可验证的计算延迟来确保随机性的不可预测性（在某个时间点之前）和公平性。在区块链分片中，VDF常与VRF结合使用，以提供更强的随机性和安全性保障。

在分片配置方案中，随机数生成方法的选择非常重要，它可以直接影响分片协议的安全性和可靠性。因此，分片协议的设计者需要仔细考虑随机数生成方法，并根据实际情况进行合理的选择。除了上述的随机数生成方法之外，实践中还有其他方法，例如Random Zoo、确定性门限签名（Deterministic Threshold Signatures，DTS）、分布式密钥生成（Distributed Key Generation，DKG）。限于篇幅，本书不再逐一介绍，有兴趣的读者可自行查阅相关资料。

5.4 跨片交易：分片世界的协同难题

重点：掌握主流的跨片交易机制

跨片交易需要满足什么样的特性？主流的跨片交易主要分成哪几类？

一项典型的区块链交易通常由一个或多个输入（Input）与输出（Output）构成。鉴于分片技术的特性，一项交易的输入与输出可能被分配到不同的分片上进行处理，此时便会产生跨片交易。由于分片协议中的交易分配策略（例如，基于哈希的随机分配）通常无法保证一项交易的所有关联组件都位于同一分片内，因此跨片交易是分片系统中普遍存在且必须高效、安全处理的问题。解决这一问题的关键在于保障此类交易满足事务的ACID（Atomicity，Consistency，Isolation，Durability）特性，而原子提交协议正是实现这种特性的核心机制。数据库事务遵循经典的ACID原则；而区块链跨片交易中的ACID特性，则是对传统数据库理论在分布式环境下的深度重构与演进。本节将深入探讨跨片交易对ACID特性的需求，详细阐述原子提交协议如何作为保障这些特性的关键手段，并介绍一些主流区块链中的跨片交易处理机制。

5.4.1 ACID特性：跨片交易的保障

区块链本质上是一个分布式账本系统，跨片交易实质上构成了典型的分布式事务。因此，保障这类跨片交易的关键是严格遵循事务的ACID特性。在分布式系统和数据库领域，事务是指由一系列操作组成的逻辑工作单元。为了确保数据在并发访问和系统故障情况下的完整性与可靠性，事务必须满足以下四个核心特性，通常统称为ACID特性：

- 原子性（Atomicity）：强调事务是不可分割的最小工作单元。这意味着一个事务中的所有操作要么全部成功执行并持久化到系统中，要么全部失败并完全回滚到事务开始前的状态。事务没有中间状态，确保了“全有或全无”（all-or-nothing）的原则。
- 一致性（Consistency）：指事务执行前后，系统必须始终保持在有效且一致的状态。这意味着事务完成后，所有数据必须满足预定义的业务规则、完整性约束和逻辑关系。
- 隔离性（Isolation）：指并发执行的多个事务之间互不干扰，每个事务都感觉自己是系统中唯一运行的事务。一个事务的中间状态对其他并发事务是不可见的，从而有效避免了脏读、不可重复读和幻读等并发控制问题。
- 持久性（Durability）：指一旦事务提交成功，其所作的所有更改都是永久性的，即使系统发生故障（如电源中断、硬件崩溃等），这些更改也不会丢失。数据通常会被写入非易失性存储介质中，并经过冗余备份以确保其可靠性。

简而言之，对于跨片交易而言，满足ACID特性是确保其在多个分片中正确执行的基础。这意味着即使一笔转账涉及两个分片，其扣款和加款操作也必须原子性地同时成功或同时失败，绝不会出现资金在其中一个分片更新而另一个分片未更新的情况，从而保证整个区块链账本始终处于正确且一致的状态。同时，当多个跨片交易或常规交易并行发生时，它们不会相互干扰，每个交易的执行结果都是确定且可预测的，不会因为并发而导致数据混乱。此外，一旦跨片交易被确认并记录到区块链上，其状态变更就是永久的，即使系统发生意外死机，交易结果也不会丢失，保障了数据的可靠性。

5.4.2 协调一致的“锁定—提交”方案

这类方案的核心思想类似于现实世界中的“托管交易”。在所有相关方都确认交易条件无误之前，先把钱“锁定”在一个中间状态。只有当所有相关方达成共识后，交易才最

终完成。否则，就取消交易，把锁定的钱退回。

最直观的想法是，既然交易是客户端发起的，那就让客户端来充当“协调者”，确保交易在所有相关分片上要么都执行，要么都不执行。RSCoin[3] 是这类架构的一个代表性协议。该加密货币框架的设计理念是中央银行对货币供应保持完全控制，同时依赖一组分布式分片来防止双重支付问题。在该框架中，分片负责处理底层的交易区块，并构建一个交叉引用链。分片之间通过客户端进行通信，并依赖客户端来确保交易能够成功完成。具体而言，客户端首先从管理交易输入的大多数分片中获取签名许可，这相当于对交易输入进行锁定。随后，客户端将交易信息和已签名的许可发送给负责交易输出的分片。接收分片会验证交易的有效性，并核实来自输入分片的签名，以证明该交易的输入尚未被双重支付。如果验证通过，该分片将把该交易添加到下一个区块中。RSCoin 以时间片为单位运行：在每个时间片结束时，所有分片将其已清算的交易发送给中央银行，由中央银行负责对这些交易进行整理、打包成区块并最终添加到区块链上，从而确保了交易的最终性和全局一致性。

然而，客户端驱动的原子提交协议存在固有的脆弱性，尤其容易受到拒绝服务（Denial of Service，DoS）攻击的影响。如果客户端在协议执行过程中停止参与，可能导致交易输入被永久锁定，从而使系统陷入僵局。这类系统通常假设客户端会基于某种激励机制而进入解锁阶段。例如，在加密货币应用中，如果交易输入被永久锁定而客户端不响应，其自身将蒙受资产损失，这构成了客户端继续参与的内在激励。但这种激励机制对于那些可能具有共享所有权的通用平台而言并不适用，因为在共享所有权的场景下，单一客户端的损失可能不足以促使其完成整个解锁过程。此外，RSCoin 框架内部依赖于每个分片执行的两阶段提交协议，但该协议本身不具备拜占庭容错能力。这意味着在存在恶意或结盟敌手的情况下，系统可能遭受双花攻击，从而破坏系统的安全性和可靠性。

鉴于对客户端完全依赖带来的脆弱性，协议设计者开始探索更强大的协调机制。一条自然的演进路径是，将协调的重任从不稳定的客户端转移到更为可靠的分片网络。Chainspace[12] 便是这一方向的典范。其核心思想在于，针对一笔涉及多个分片的交易，所有相关的分片会动态地组成一个临时共识组，并在组内部运行一个拜占庭共识协议，共同决定该笔交易是最终“提交”还是“中止”，从而确保交易的原子性。这种方法的主要优势在于不再依赖脆弱的客户端作为协调者，有效避免了因客户端离线、恶意行为或拒绝服务攻击导致的问题。同时，由于分片间运行的是具备拜占庭容错能力的协议，Chainspace 能够容忍一定数量的恶意节点，显著提升了系统的安全性和鲁棒性。

尽管将协调职责转移至分片网络内部提升了系统的鲁棒性，但另一条演进路径则认为，不应完全抛弃客户端驱动模型的简洁性，而是应该对其进行“加固”，通过引入更强大的协议来增强其可靠性。OmniLedger[7] 采取的正是这种改良策略的典范。其核心思想在于，在保留客户端驱动灵活性的同时，通过引入拜占庭容错机制和巧妙的解锁机制来克服传统客户端驱动模型的脆弱性。这种方法的优势在于，它既能像早期客户端驱动模型一样有效减少不必要的片间通信，又能通过引入拜占庭分片原子提交协议（Byzantine Shard Atomic Commit Protocol，Atomix）来处理跨片交易，确保每笔交易都能被原子性地提交或中止，从而解决了 RSCoin 等协议面临的安全性问题。此外，OmniLedger 还设计了一个“锁定然后解锁”的过程，增强了系统的活性：即使原始发起交易的客户端在锁定资产后“消失”，

任何其他参与方（例如验证者或其他客户端）都可以根据链上已有的“锁定证明”，继续推进后续的解锁或提交步骤，从而有效避免了资产被永久锁定的问题。

5.4.3 另辟蹊径——跳出“原子提交”的思维定式

至此，我们讨论的方案都在围绕“如何实现一个完美的原子提交”进行博弈，无论是依赖客户端还是分片本身。这些方法试图通过复杂的同步协调机制来确保跨片交易的原子性。然而，更高明的思考者会提出疑问：我们是否必须陷入这种复杂的原子提交中？于是，一些跳出原有框架的新范式应运而生，它们尝试以非传统的方式解决跨片交易的原子性问题，从而简化协议设计或提升系统性能。

RapidChain[13] 提出了一种截然不同的思路，其核心思想在于：不直接执行复杂的跨分片原子提交，而是将涉及跨分片的资产首先“转移”到同一个目标分片上，随后在该目标分片内部执行一个普通的单分片交易。这种方法的优势在于极大地简化了跨分片交易的复杂性，避免了烦琐的分布式协调协议，从而提高了交易处理效率和系统的整体可扩展性。下面我们通过一个简单的例子来说明 RapidChain 的巧妙之处。假设 Alice 在分片 A 上有 10 个代币，想要转账给 Bob 在分片 B 上的账户。RapidChain 不会尝试让分片 A 和分片 B 之间进行复杂的同步协调来完成这笔跨分片交易。相反，它会将这笔交易分解成两个简单的“子交易”。首先，在分片 A 上，Alice 的 10 个代币被销毁，同时在目标分片 B 上会生成等价的 10 个“临时币”给 Alice。这个过程就像是 Alice 把分片 A 的代币存入了分片 B 的一个临时账户。接着，在分片 B 上，Alice 就可以使用这 10 个“临时币”转账给 Bob，这是一个普通的片内交易，由分片 B 独立处理。这两个子交易都是在各自的分片内独立执行的，它们之间没有复杂的原子提交协议。如果第一个子交易成功，但第二个子交易因为某种原因失败了（比如 Bob 的账户有问题），Alice 的资产并没有损失。她的 10 个代币只是从分片 A“移动”到了分片 B 的临时账户。Alice 可以在分片 B 上重新尝试转账给 Bob，或者选择将这 10 个代币从分片 B 再“移回”到分片 A。这种机制确保了交易的一致性和原子性，即资产要么最终到达 Bob 的账户，要么安全地回到由 Alice 控制的某个分片之中，避免了资产丢失的风险。

Monoxide[14] 提出了一种创新的“最终原子性”技术来处理跨分片交易，其核心思想在于：我们不必追求所有分片在“同一瞬间”完成状态更新，只要能保证最终结果是正确的，短暂的延迟是可接受的。这种模式被称为“最终原子性”，它不是同步的、全有或全无的操作，而是异步的、保证最终完成的流程。这极大地简化了分片间的协调逻辑，从而提升了系统的可扩展性和效率。具体来说，对于一个涉及付款人 A 和收款人 B 的支付交易，如果 A 和 B 位于同一分片，Monoxide 可以直接在该分片内部完成交易处理。当 A 和 B 属于不同分片时，Monoxide 则采用一种双阶段事务处理机制，其流程更像是“接力棒”。在第一阶段，付款人 A 所在的分片（A 区域）的矿工在构造新区块时，如果接收到一个未确认的跨分片交易，会首先验证付款人 A 的余额是否满足花费要求，若满足则直接扣款并通过该交易验证。此时，分片 A 的账本已经更新。随后，矿工会创建并广播一个包含已验证交易列表的区块，并为跨分片的部分生成一个新的中继交易，该中继交易携带相关存款操作信息，并将其发送到收款人 B 所在的目标分片（B 区域）。在第二阶段，收款人 B 所在的分片（B 区域）的矿工接收到中继交易后，会根据其来源信息进行验证，若验证有效则构造

新的区块记录中继交易信息。最终，通过B区域的共识机制确认后，交易结果将被广播到B区域并完成存款操作，从而结束整个跨分片交易过程。这种分解交易并采用异步接力的方式，极大地降低了跨分片协调的复杂性，提供了一种高效且具有最终一致性保障的跨分片交易解决方案。

5.5 分片重置：应对节点变化与安全威胁

学生：老师，经过前几节的学习，我们已经学会了如何通过分片协议来提升区块链的吞吐量。但是，我不理解为什么精心设计的分片配置方案还要重组呢？因为5.1节数据库分区的学习告诉我们，重新分区会消耗非常大的成本。

教师：在区块链分片过程中，将节点分配到各个分片时存在诸多潜在的安全隐患。例如，恶意节点可能采取策略性行为，频繁离开并重新加入不同的分片，最终可能完全控制某个特定的分片，从而破坏区块链协议的安全保障。此外，即使没有节点主动加入或重新加入分片，攻击者亦可在一定时间段内攻陷一定数量的正常节点。

学生：老师，我明白了。这是因为传统数据库和区块链安全模型不同，它们的设计目标也不同。这也回应了我们刚开始的问题：为什么数据库分区不能直接用于区块链？

教师：你的理解非常正确。区块链分片技术将区块链分成多个分片，为了应对节点变化和安全威胁，这些分片只能存在一段特定的时间，这段时间称为时间片（Epoch）。时间片重置（Epoch Reset）协议用于分片重置，是维护区块链系统安全性的关键机制。

学生：老师，我明白了。我有一个非常简单的想法，只要设置一个固定的时间，每当到达该时间后，整个区块链网络就进行全网重分片操作，这样似乎能够解决问题。

教师：不错，你已经学会举一反三了。早期区块链分片协议就是这么做的，但是后来人们发现对整个区块链网络重新分片会消耗很多资源，而且固定时间长短不好设置。

为了帮助读者系统学习并深入理解分片重置的核心概念及其在区块链分片协议中的重要作用，本节将首先介绍早期研究人员所设计的简易分片重置协议。通过剖析早期协议的基本原理，我们将进一步探讨它们在安全性和效率方面可能存在的问题与局限性。在此基础上，本节的后续部分将介绍为解决这些挑战而提出的更为复杂且更稳健的分片重置协议，展示分片安全设计理念的演进与完善。

5.5.1 分片重置初步尝试

在分片重置机制的研究初期，研究人员设计了一些相对简化的方案以实现分片节点的周期性更新。其中，Elastico[15]提供了一种早期且具代表性的分片重置方法，其核心理念是定期重新选举所有的分片成员。Elastico的重置过程依赖于一个新的全局共识组，该共识组在每个时间片的末尾，专门负责生成用于下一个时间片节点分配的随机字符串集合。Elastico的时间片重置过程通常包括两个主要阶段：

（1）**哈希值承诺：**在重置的第一阶段，共识组的每个成员会独立选择一个长度为r比特的随机字符串，并计算其所选随机字符串的哈希值，随后将这个哈希值广播给该共识组内的所有其他成员。为了防止串通或操纵，所有成员会共同运行一个交互式的一致性协议（例如拜占庭容错共识协议），以就一个包含所有成员所提交的哈希值集合达成最终一致。一旦达成共识，这个哈希值集合会被广播到网络上的每个节点。该集合被视为对随机字符

串的承诺，必须包含至少$2c/3$个（其中c是共识组的大小）诚实成员的有效哈希值承诺。这一阶段的目标是确保所有诚实成员都承诺了一个未来的随机值，并且这个承诺是公开可验证的，任何恶意成员都无法在后续阶段改变其承诺。

（2）**随机字符串揭示与聚合：**在对哈希值集合达成一致并广播之后，重置过程进入第二阶段。在此阶段，共识组的每个成员都会向全网节点广播一条消息，该消息包含其在第一阶段承诺的随机字符串本身。此阶段的开始严格依赖于第一阶段哈希值集合的共识达成（即必须有对同一个哈希值集合的至少$2c/3$个有效签名），这保证了诚实的成员只有在确定该共识组已对哈希值集合达成一致且敌手不能改变其承诺后，才会揭示他们的随机字符串。在第二阶段结束后，Elastico系统中的每个节点都将从共识组的成员那里接收到一定数量的随机字符串及其对应的哈希值对。由于诚实的成员一定会遵守协议，而恶意节点则未必会揭示其承诺，因此每个节点通常会收到至少$2c/3$对、至多$3c/2$对有效的随机字符串和哈希值。节点会丢弃任何与先前承诺的哈希值不匹配的随机字符串，以确保数据的完整性。最终，这些有效且聚合的随机字符串集合将被用来配置下一个时间片的分片结构和成员组成。

然而，尽管Elastico提出的分片重置方案具有开创性，但也暴露出几个显著的问题，这促使后续研究的深入，其核心问题主要集中在两个方面。首先是效率问题，Elastico在每个时间片结束时重新生成所有分片，这种全局性的重置方式，尤其是在大型网络中，其巨大开销难以承受，严重影响了系统的整体效率和响应速度。同时，在分片成员替换时如何平滑、无缝地维护账本也增加了协议设计的复杂性。其次是安全性问题，在每个时间片中用于选举新分片成员的随机性可能会被潜在敌手影响，如果攻击者能够提前预测或轻微操纵随机数生成过程，将可能威胁分片选举的公平性和随机性，甚至导致恶意节点掌握特定分片的控制权。此外，Elastico需要一个可信第三方来生成一个初始的公共随机数并向各方公开。这种对初始可信第三方的依赖，与区块链去中心化和无须信任的原则相悖，降低了系统的整体去中心化程度。

这些问题表明，虽然Elastico为分片重置提供了初步的解决方案，但要构建一个真正高效、安全且完全去中心化的分片系统，还需要在效率和安全性方面进行更深入的研究和改进。为此，我们在5.5.2节将深入探讨提升分片重置效率的方案，在5.5.3节则会聚焦于增强分片重置安全性的方法。

5.5.2 更高效的分片重置

鉴于早期分片重置协议在效率和开销方面存在的挑战，后续研究致力于开发更为高效、渐进式的重置机制。RapidChain[13]采取了一种不同于Elastico的创新方法来实现分片重置，该方法的核心是引入了被称为布谷鸟规则（Cuckoo Rule）的动态调整机制。与Elastico每次重置所有分片成员不同，RapidChain的布谷鸟规则只重置分片成员的一个子集。当新节点加入或现有节点离开网络时，这种机制能够动态调整分片内部的成员构成，同时保持各分片的大小相对稳定，从而显著降低每次重置所需的开销和复杂度。

RapidChain实现分片局部重置的关键在于其独特的节点分配和动态调整方式。它首先通过一个哈希函数将每个节点的身份映射到$[0,1)$区间内的一个随机位置。随后，这个$[0,1)$的区间被逻辑划分为k个大小约为$1/k$的子区域，每个子区域对应一个分片，而该分片中的节点即是其对应区域内的一组节点。布谷鸟规则在节点分配和维护分片鲁棒性方面发挥

了核心作用。基于此规则，当一个新节点想要加入该网络时，它首先会被放置在一个随机位置$x, x \in [0,1)$。为了保持分片的均衡性和避免过度集中，围绕x的固定间隔内的现有节点（即x附近一定范围内的节点）会被移动到$[0,1)$中的新随机位置，从而为新节点腾出空间并重新平衡周围的分片成员，实现分片成员的动态、高效且局部性的重置。

这种局部调整机制带来了显著的效率优势。首先，它具备低开销的特点，因为每次重置仅涉及网络中的一小部分节点，显著减少了节点重组产生的网络通信与计算开销。其次，由于大部分分片成员保持不变，这种机制确保了状态稳定，并使分片状态的交接问题大大简化，避免了大规模数据迁移和同步带来的复杂性。最后，它能动态维持规模稳定，可确保在节点加入或离开网络时，各分片的大小能动态维持在预期范围内，从而保证分片处理能力的均衡性。

除了布谷鸟规则带来的局部重置效率外，RapidChain还通过其他机制进一步提升了分片重置的整体效率。其中一项是其独特的工作量证明机制。为了实现高效且抗女巫攻击的节点选择，RapidChain要求每个参与协议的节点解决一个计算难题，并在每个时间片开始时，基于当前时间片的随机性产生一个新的PoW难题。尤其值得注意的是，RapidChain中所有的PoW计算都在链下进行。这意味着区块链的主链或共识流程无须停下来等待PoW计算结果，因此PoW的计算延迟不会成为整个区块链系统进程的瓶颈，极大地提升了系统的吞吐量和响应效率。

5.5.3 更安全的分片重置

在回顾了早期分片重置协议（如Elastico）所面临的效率挑战后，我们需要进一步探讨其在安全性方面的关键问题。正如5.5.1节中所指出的，早期方案的一个显著安全问题在于如何确保随机性生成的公平性与抗操纵性，以及避免对可信第三方的依赖。如果随机数能够被攻击者预测或影响，那么分片成员的分配将不再是真正的随机，从而可能导致恶意节点集中控制特定分片，威胁整个系统的安全性。

针对这一核心安全挑战，OmniLedger[7]协议设计了一套基于强密码学原语的高安全随机性生成协议，旨在从根本上杜绝随机性偏见的风险，并消除对初始可信设置的依赖。OmniLedger同样采用全局重置的框架，即在每个时间片（Epoch）开始时对所有分片成员进行重新分配，但其核心创新和主要优势正是体现在其安全、去中心化的随机性生成过程上。它引入了可验证随机函数（Verifiable Random Function，VRF）来实现不可预测的领导者选举，确保了随机数的公平性和不可操纵性。

OmniLedger的重置协议首先负责生成和管理节点身份，并通过其身份区块链（Identity Blockchain）来维护所有已注册参与者的信息。该身份区块链假设在同步通信通道下运行，并在此基础上将验证者（参与者）分配到各个分片。其安全增强的核心在于时间片随机数的生成：协议依赖于VRF，实现了不可预测的领导者选举。随后，该协议将通过RandHound协议选出的领导者作为“客户端”，负责协调并生成最终的时间片随机性，进一步增强了随机性的去中心化和抗审查性。

具体而言，在一个时间片开始时，所有参与网络的验证者都会计算一个票据（Ticket）。该票据中包含当前时间片内所有正确注册的验证者身份信息（这些信息存储在身份区块链中）以及一个计数器。验证者在设定的时间窗口内将这些票据在网络中广播。在Δ时间结束后，每个验证者会锁定迄今为止看到的最低哈希值（或最低值的有效票据），并将持有该

票据的节点认定为RandHound协议的领导者。一旦验证者成功完成一次RandHound协议的运行，并由选出的领导者广播生成新随机性及其相应的正确性证明，那么每个注册的验证者都可以在本地独立验证该随机性的有效性。验证通过后，这些验证者便可以使用这种经验证的随机性来计算一个序列，并将计算结果细分为大小相同的“桶”（Buckets），从而最终确定如何将节点公平且随机地分配到各个分片中。

OmniLedger的这种设计，尤其是在随机性生成方面的创新，带来了显著优势：首先，通过采用强密码学原语和去中心化的RandHound协议，它极大地增强了随机性的不可预测性和抗操纵性，从而有效抵御恶意节点试图预测或影响分片分配以集中攻击的风险，提升整体系统安全性。其次，它消除了对可信设置或中心化第三方的依赖，进一步强化了协议的去中心化特性。最后，这种方案确保了分片成员分配的公平性，保障了分片内部的活跃度和健康性，这对于分片化区块链的长期稳定运行至关重要。

5.6 总　　结

经过本章对区块链分片协议相关概念的介绍，我们对区块链分片协议有了基本了解。在5.1节，我们首先学习了传统数据库分区的基本原理，了解到数据库和区块链在安全模型方面存在本质差异，这导致传统数据库中的技术手段不能直接应用于区块链。但是传统数据库领域在分区方面的发展，仍然给区块链的分片协议设计带来了不小的启发。在5.2节，我们系统地了解了区块链扩展性问题的具体表现，这引导我们设计相关协议以避免这些问题。随后，我们了解了完整的区块链分片协议所具有的关键流程。在5.3节，我们正式开始学习区块链分片的核心机制，了解了分片协议中的随机数生成方法。在5.4节，我们学习了分片协议中必须面对的难题——跨片交易。共学习了两类跨片交易处理方案，认识到处理问题不一定拘泥于传统思维，从新的角度跳出原来的思维定式，也许会有更巧妙的解决方案。最后，我们在5.5节认识到区块链分片协议并不是一成不变的，这与传统数据库领域存在本质差异，而这也是区块链为了应对节点变化和安全威胁所必须采取的措施。

5.7 课 后 习 题

1. 根据本书分析，传统数据库分区和区块链分片虽然都采用了“分而治之”的思想，但其最根本的设计出发点差异在于（　　）。

A. 数据库分区主要为了处理海量数据存储，而区块链分片主要为了提升加密计算速度

B. 数据库分区旨在优化特定类型的查询（如范围查询），而区块链分片旨在实现网络通信的匿名性

C. 数据库分区是在一个可信环境下为突破单一服务器的I/O和存储瓶颈，而区块链分片是在不可信环境下为突破全网冗余验证导致的计算和共识瓶颈

D. 数据库分区是物理层面的技术，而区块链分片是应用层面的技术

2. 传统数据库分区技术不能直接应用于区块链，其根本性的差异在于敌手模型的不同。请详细阐述这一根本性差异，并说明若将数据库分区方案直接用于区块链，会产生什么样的安全风险？

3. 如果一个分片协议没有时间片重置机制，请问会产生怎样的后果？以及时间片重置

机制是如何避免这种后果的。

4. 假设你是一个分片协议的设计者。在“分片的生成”阶段，你需要在两种分配方式中选择：

- 基于账户地址：将地址前缀相同的用户账户分配到同一个分片。
- 基于随机分配：完全随机地将节点分配到不同分片，并通过随机哈希将账户映射到分片。

请分析这两种方式各自的优缺点，以及它们分别适合什么样的应用场景。

5. 为什么说“静态配置”更适用于联盟链，而“多成员轮换”更适用于对安全性和动态性要求高的公有链？

6. 相比于简单的VSS，PVSS的“公开可验证”特性提供了什么额外的关键优势？

7. 从RSCoin到Chainspace，跨片交易的协调机制发生了什么核心演变？这种演变解决了什么关键问题，但又可能引入了什么新的开销？

8. Alice希望用她在分片A上的1个ETH，兑换Bob在分片B上的100个DAI。这是一个典型的跨片资产交换。回顾文中几种跨片交易方案，哪一种方案在执行时，会先在分片B上给Alice生成一个“ETH收据”，然后Alice再用这个“收据”与Bob进行一次纯粹的分片B内部交易？

A. RSCoin方案

B. Chainspace方案

C. RapidChain方案

D. Monoxide方案

9. 在5.5节的师生对话中，老师提到即使没有节点主动加入或退出，攻击者也可能“主动腐蚀一定数量的正常节点”。请解释分片重置机制是如何对抗这种攻击的？

10. 作为一名协议设计师，你现在有两个分片重置方案可选：

方案A：全局重置（类似OmniLedger），每小时一次，使用强密码学工具保证极高的安全性、无偏性和不可预测性，但每次重置耗时5分钟，其间系统吞吐量显著下降。

方案B：局部重置（类似RapidChain的布谷鸟规则），实时响应节点变化，效率极高，对系统性能影响微乎其微，但其随机性根植于节点标识符ID的哈希，理论上不如方案A的密码学随机性“完美”。

请分别描述一个最适合采用方案A的应用场景和一个最适合采用方案B的应用场景，并阐述你的设计权衡。

参考文献

[1] Wang G, Shi Z J, Nixon M, et al. Sok: sharding on blockchain[C]//Proceedings of the 1st ACM Conference on Advances in Financial Technologies. New York: ACM, 2019: 41-61.

[2] Androulaki E, Barger A, Bortnikov V, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains[C]//Proceedings of the Thirteenth EuroSys Conference. New York: ACM, 2018: 1-15.

[3] Danezis G, Meiklejohn S. Centrally banked cryptocurrencies[C]//Proceedings of the 23rd Annual Network and Distributed System Security Symposium. San Diego, CA, USA: Internet Society, 2016.

[4] Gilad Y, Hemo R, Micali S, et al. Algorand: Scaling Byzantine agreements for cryptocurrencies[C]//Proceedings of the 26th Symposium on Operating Systems Principles. New York: ACM, 2017: 51-68.

[5] Daian P, Pass R, Shi E. Snow white: robustly reconfigurable consensus and applications to provably secure proof of stake[C]//Financial Cryptography and Data Security. Cham: Springer, 2019: 23-41.

[6] Kokoris-Kogias E, Jovanovic P, Gailly N, et al. Enhancing bitcoin security and performance with strong consistency via collective signing[C]//25th USENIX Security Symposium (USENIX Security 16). Berkeley: USENIX Association, 2016: 279-296.

[7] Kokoris-Kogias E, Jovanovic P, Gasser L, et al. Omniledger: a secure, scaleout, decentralized ledger via sharding[C]//2018 IEEE Symposium on Security and Privacy (SP). Los Alamitos: 2018: 583-598.

[8] Micali S, Rabin M, Vadhan S. Verifiable random functions[C]//40th Annual Symposium on Foundations of Computer Science. Los Alamitos: IEEE, 1999: 120-130.

[9] Feldman P. A practical scheme for non-interactive verifiable secret sharing[C]//28th Annual Symposium on Foundations of Computer Science (SFCS 1987). Los Angeles: IEEE, 1987: 427-437.

[10] Stadler M. Publicly verifiable secret sharing[C]//International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 1996: 190-199.

[11] Boneh D, Bonneau J, Bünz B, et al. Verifiable delay functions[C]//Annual International Cryptology Conference. Cham: Springer, 2018: 757-788.

[12] Al-Bassam M, Sonnino A, Bano S, et al. Chainspace: a sharded smart contracts platform[C]//Network and Distributed System Security (NDSS) Symposium. Reston: Internet Society, 2018.

[13] Zamani M, Movahedi M, Raykova M. Rapidchain: scaling blockchain via full sharding[C]//Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM, 2018: 931-948.

[14] Wang J, Wang H. Monoxide: scale out blockchains with asynchronous consensus zones.[C]//USENIX Symposium on Networked Systems Design and Implementation. Berkeley: USENIX Association, 2019: 95-112.

[15] Luu L, Narayanan V, Zheng C, et al. A secure sharding protocol for open blockchains[C]//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM, 2016: 17-30.