Insight Chain 技术白皮书 V1.0

无限扩展的数据生态公链

Insight Chain 创始团队

2019 年 2 月 1 日

摘要

由于 BTC、ETH 等公链的 TPS 不高，很难支持应用的大规模使用，2017 和 2018 两年，越来越多的公链为了提高 TPS 而做技术上的探索，例如 EOS 采用 DPoS 共识算法，将 TPS 提高到数千量级，但是牺牲了一定程度的去中心化特征，公链的去中心化、安全性和可扩展性之间的不可能三角问题亟待解决。同时现有的公链主要是将交易数据上链，并没有实现业务数据的结构化上链，使得互联网和传统商业的场景落地有很大的困难。所以我们把 Insight Chain 的核心目标定义为两个：在满足去中心化和安全性的前提下提高可扩展性、满足更多应用的业务数据上链需求，实现每秒 10 万级以上(100,000+TPS)的交易数据处理能力，以及每秒 100 万级以上(1,000,000+TPS)的业务数据处理能力，Insight Chain 将通过以下核心技术实现这两个目标：

- VDPoS 共识算法，引入验证节点增加去中心化和安全程度；
- 多主链+多子链架构，实现近乎无限的垂直扩展和水平扩展能力；
- 公链利用率模型(IUM)，利用自适应分片技术增加公链处理能力；
- 业务数据结构化上链，支持数据的保存、加密解密、交易等场景；
- 可视化的智能合约，引入交易引擎简化 Token 的发行和交易；
- 利用 Floyd 算法改进 Kademlia 网络；
- 基于 VRF 的随机验证机制；
- 通过状态通道智能合约、委托执行和公证人等机制支持跨链。

目录

1. 背景

1.1 从加密数字货币到公链数据生态

自 2008 年比特币[2]诞生以来，货币类项目一直是区块链行业的主体。货币类项目以比特币、莱特币等为代表，以一维性的价值传递为基本特征。以太坊[3]的诞生意味着区块链行业突破了单纯的数字货币概念，开启了向更复杂的通用开发平台发展的行业趋势。随着区块链技术的日渐发展，产业中对区块链的大量需求将逐步得到满足，将区块链与具体商业应用场景的结合也在快速推进，通用开发平台的发展为此奠定了基础。

我们认为，考虑到实际商业场景的复杂性，“区块链+具体产业”的落地使用将必然催生出一个相应的公链数据生态，并以公链数据生态内部的多维性价值传递为数字货币市场提供更高层次、更多元化的真实价值支撑，从而完成区块链行业从加密数字货币到公链数据生态的 2.0 阶段的转变。

1.2 公链数据生态的落地与运转

在数字货币阶段，交易上链已经能够满足资产流转的大部分需求，但在公链数据生态内，由各角色的交互行为产生的数据是维持系统运转、合理运行经济分配机制的必要材料，在某些场景下数据信息本身亦是价值的载体，因此从单纯的交易上链到数据上链是公链数据生态发展的必然趋势。除了对数据真实及数据价值保护的需求，生态体系内产生的大量数据对区块链的处理能力提出了更高的要求，现有多数公链在区块打包速度及确认速度等方面均落后于实际需求。

2. Insight Chain 技术架构

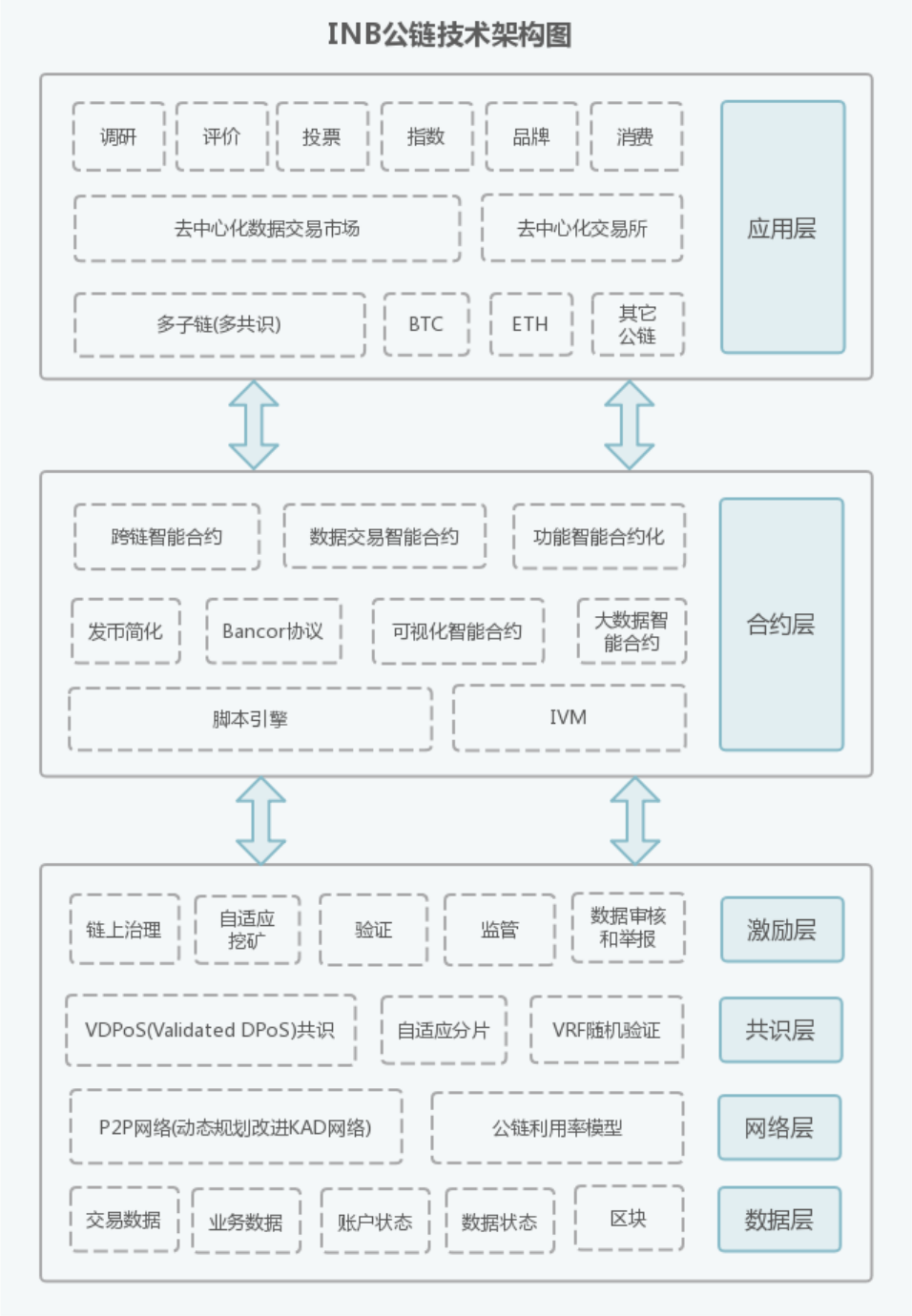
Insight Chain 简称 INB 公链，为了应对当下对区块链日益增长的需求，我们把 INB 公链的核心目标定义为两个：在满足去中心化和安全性的前提下提高可扩展性、满足应用的业务数据上链需求，实现每秒 10 万级以上(100,000+TPS)的交易数据处理能力，以及每秒 100 万级以上(1,000,000+TPS)的业务数据处理能力，为众多实际场景的区块链落地奠定技术基础。

INB 公链提出 VDPoS 共识算法，通过在传统的 DPoS[4]共识机制中引入验证节点解决去中心化和安全性问题，通过多主链+多子链架构实现公链的垂直和水平扩展，并对数据进行分层存储，保存全局数据状态，支持业务数据的结构化上链和交易。目标是建立全球首条可无限扩展的数据生态公链，真正支持高并发互联网应用的链上运行。

INB 公链中涉及到的术语解释如下表所示：

术语	解释
主链	公链的主要链，保存交易、子链验证数据等高价值数据
子链	公链的侧链，保存了交易数据以及业务数据等价值相对低的数据
超级节点	主链的出块节点和验证节点
验证节点	主链的验证节点
监管节点	负责对公链以及数据进行监督、反馈
普通节点	同步公链数据，并对外提供公链 API
业务数据	子链上保存的除了交易以外的数据信息
子链服务商	为子链提供节点的服务商
智能合约服务商	编写大数据等算法的智能合约，并对外收取智能合约的使用费
可信数据源	用户可以保存到链上的可信数据源

INB 公链的技术架构图如下图所示：



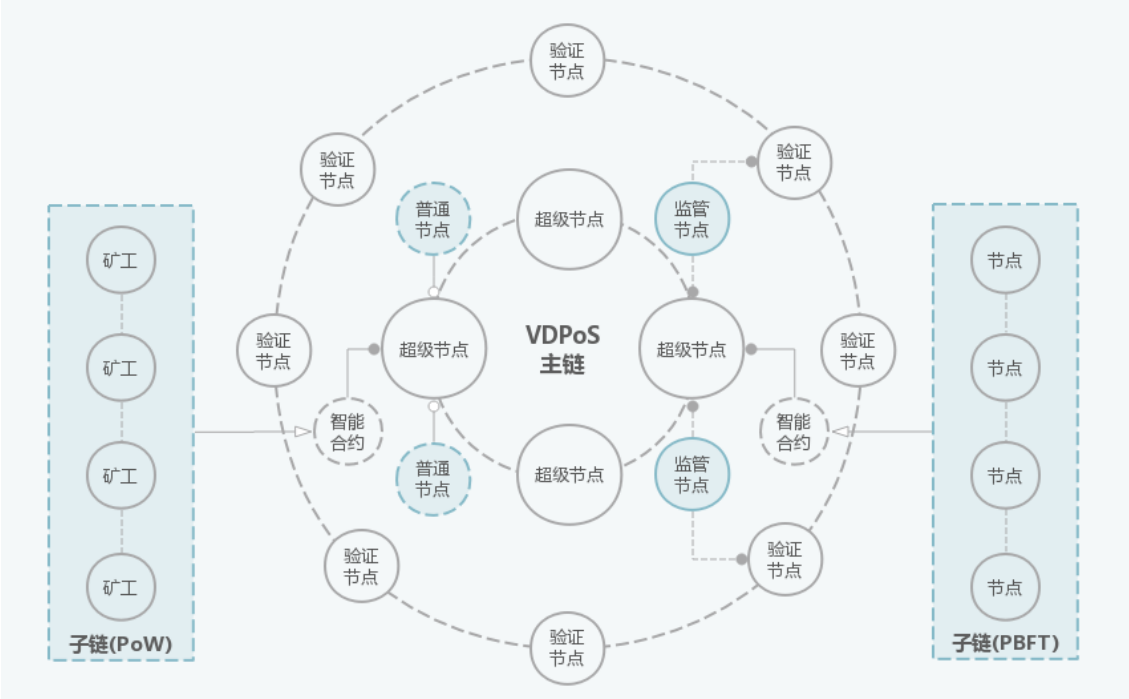
INB 公链技术架构图

2.1 共识算法：VDPoS

为了解决区块链不可能三角的矛盾，INB 公链提出了一个全新的共识算法：

VDPoS(Validated DPoS)算法，该算法是 DPoS+BFT[7]+验证节点的有机结合，公链中的每条主链都会采用该共识算法。DPoS 算法中的投票解决了 PoW 算法资源被大量无用消耗的问题，并且使用 INB 的抵押和惩罚机制，很大程度上限制了节点的作恶。区块产生以后，首先在超级节点内部利用 BFT 算法进行快速验证，同时超级节点利用 VRF 算法寻找多个随机验证节点，验证节点也使用 BFT 算法对块数据进行异步验证，以防止超级节点和验证节点的联合作恶，极大地提高了公链的去中心化程度和安全性。

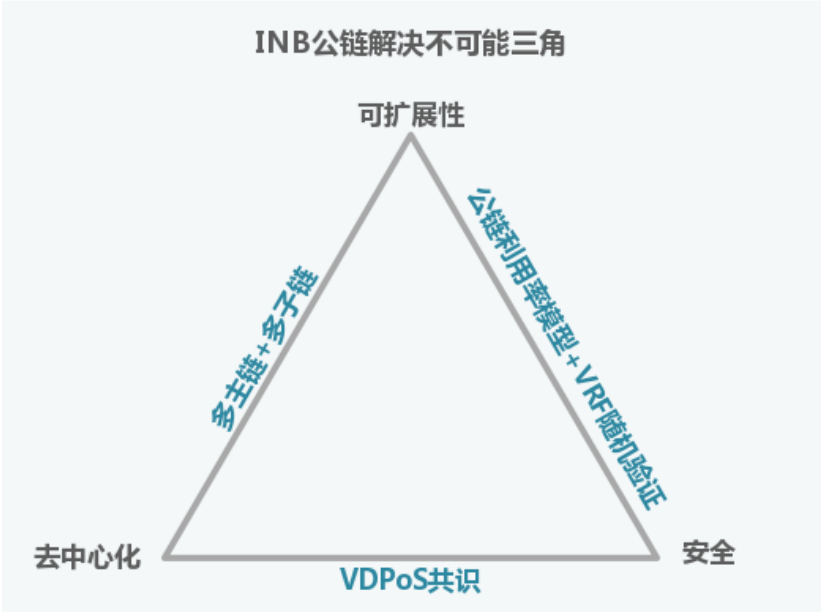
子链是附属于主链之上的子区块链网络，每条子链都有自己的共识算法，子链可以采用 BFT 类联盟链的共识机制，也可以采用 PoW、PoS[3]、DPoS 等公链共识机制，根据不同的 DApp 对数据上链的效率、安全性等不同的需求而采用不同的共识算法，公链不对子链的共识机制作限制。子链在产生一定数量的区块以后，会将该部分区块对应的默克尔树的树根保存到主链的区块上，以保证子链的安全性，对应的子链上的区块称为验证区块。验证区块的保存高度间隔会根据主链的资源利用情况来自动调整，当主链的资源利用率低的时候，可能每个区块的哈希值都保存到主链上，但是当主链的资源利用率高的时候验证区块的高度间隔会比较大。



主链和子链的节点生态图

2.2 区块链不可能三角的平衡

INB 公链从设计上就充分考虑解决区块链中不可能三角的平衡问题，下面将进行详细论述。



不可能三角示意图

2.2.1 可扩展性：垂直和水平扩展

很多的公链和媒体将区块链的可扩展性和 TPS 混为一谈，其实是不严谨的，在软件工程领域，可扩展性是指“在系统扩展成长过程中，软件能够保证旺盛的生命力，通过很少的改动甚至只是硬件设备的添置，就能实现整个系统处理能力的线性增长，实现高吞吐量和低延迟高性能”。一般可扩展性分为：水平扩展和垂直扩展两种，前者指可以通过添加新设备来增加系统的处理能力，后者是通过提高现有设备的处理能力来增加整个系统的处理能力。

INB 公链引入公链利用率模型来衡量整个公链的使用情况，当公链的利用率不足或者公链资源增强时，例如：超级节点的处理能力增强、网络速度的提高等，公链将自动增加每个超级节点的分片个数，以增加主链的条数，从而增加主链的出块速度，实现了公链的垂直扩展。当公链上的 DApp 数量增加，对数据上链的需求增加的时候，DApp 可以向超级节点申请启动新的子链，超级节点投票通过以后将启动子链对新的 DApp 的数据进行保存，从而实现了公链的水平扩展。

2.2.2 去中心化：引入验证节点

DPoS 共识算法，由于超级节点的数量比较少，经常受到去中心化程度不够、安全性不高的诟病，为了弥补这些缺陷，INB 公链引入验证节点角色，通过 INB 的激励刺激大量验证节点的加入，对公链上的数据使用 BFT 算法进行异步验证，防止超级节点联合作弊，极大地增加了公链的去中心化程度和安全性。同时，INB 公链通过 VRF(Verifiable Random Function)[8]随机抽签机制选择验证节点，保证了验证节点的随机可验证性，防止超级节点和验证节点联合作恶。

2.2.3 安全性：资源使用规则和 VRF 随机验证

对于 INB 公链，安全性体现在两个方面：传统公链的安全性和链上数据的安全性。

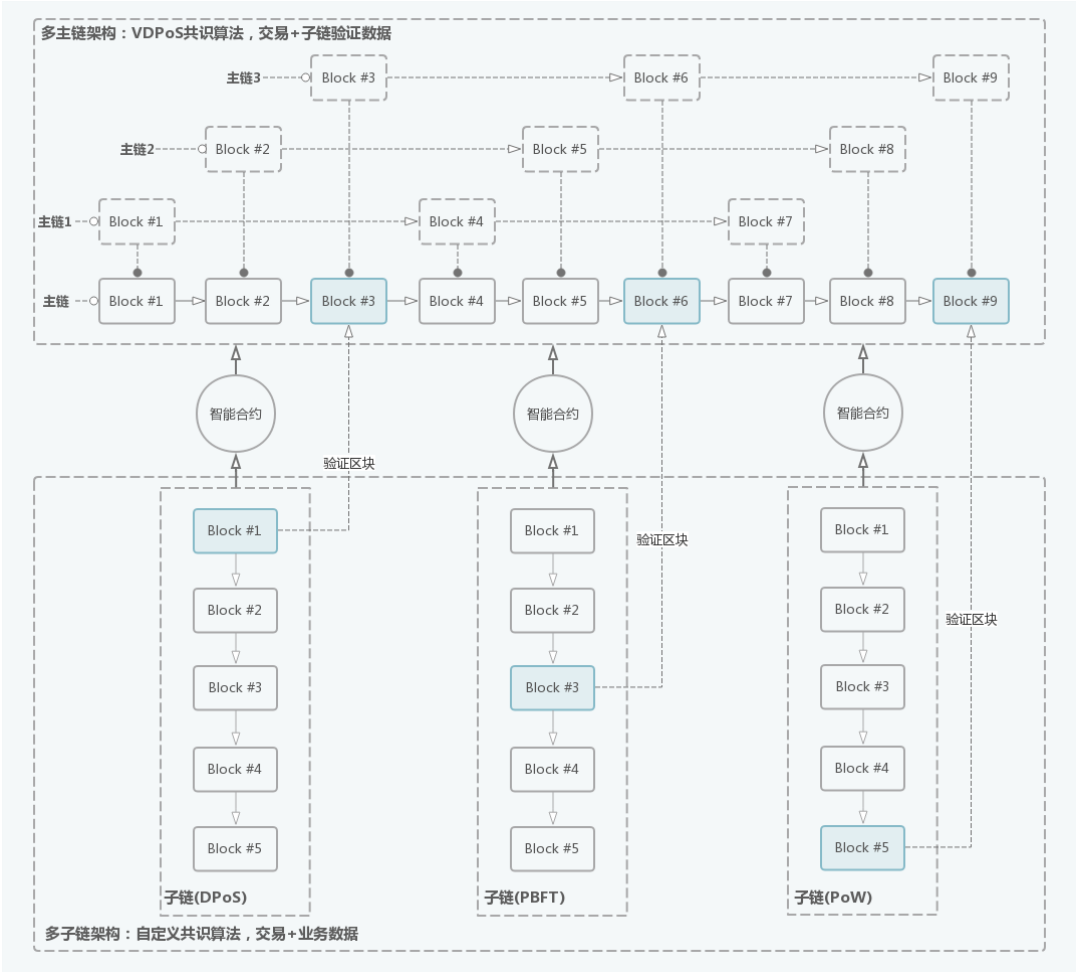
INB 首先通过使用哈希算法[11]、非对称加密[16]、MPT(Merkle Patricia Trie)等算法和机制确保公链的正常逻辑安全。对于防止恶意攻击，INB 公链通过抵押链上发行的 INB 来换取主链上资源的使用权：包括 CPU 的使用权和网络的使用权，避免了黑客通过 DDoS[28]等方式对公链网络进行攻击。同时对于子链的数据存储资源，DApp 需要根据存储的数据量进行付费，也防止了恶意攻击。另外，INB 公链使用基于 VRF 的随机验证机制，用于验证节点的选择、分片的验证等机制上，防止超级节点、验证节点等的作恶。

对于链上数据的安全性，INB 公链将对用户需要保密的数据进行加密存储，需要用户的私钥或者授权的其它账户的私钥才能解密查看。同时，数据的存储和属性的更改都需要拥有者或者授权用户的私钥进行加密。

2.3 核心架构：多主链+多子链

INB 公链使用多主链+多子链(Multi-Main Chain + Multi-Child Chain，简称 MMC + MCC)的混合架构，为全球第一个使用此种混合架构的公链。根据整个公链的资源利用情况自动对超级节点进行分片，形成多主链并行出块模式，充分利用网络资源和超级节点资源，极大的提高出块速度，增加公链的垂直扩展性；同时在主链之上可以根据应用的使用情况，启动多条子链，以支持更多的 DApp 业

务数据上链，增加公链的水平扩展性。INB 公链由超级节点、验证节点、监管节点、普通节点、子链节点组成。



多主链+多子链架构图

INB 引入公链利用率模型(IUM)来衡量整个公链的资源利用情况，包括了超级节点的 CPU 和内存资源使用情况、公链内网络的资源使用情况等对整个公链的资源利用情况做量化评估。根据资源利用率，INB 公链将自动对超级节点进行分片，超级节点分片以后形成多条平行的主链，系统将控制平行主链中位于不同节点的分片轮流出块，形成多主链并行出块模式，充分利用了公链和超级节点的资源，出块以后块的分发和验证也分别占用了不同的节点之间的网络连接，使得

整个公链的资源利用率最大化。当节点或者网络升级的时候，分片数量会自动增加，从而自动实现公链的垂直扩展性。

INB 引入数据分层模型，将交易和业务数据进行分层管理，价值大的交易数据和子链验证数据存储于主链之上，业务数据存储在子链上，同时引入全局的数据状态，将数据的版本、价格、购买次数、浏览次数等数据结构化的保存于子链之上。子链由节点或者社区发起，经过超级节点投票以后可以启动，不同的 DApp 可以使用不同的子链对数据进行存储，形成多子链并行模式，将公链的水平扩展能力提高到近乎无限。

2.4 垂直扩展：多主链架构

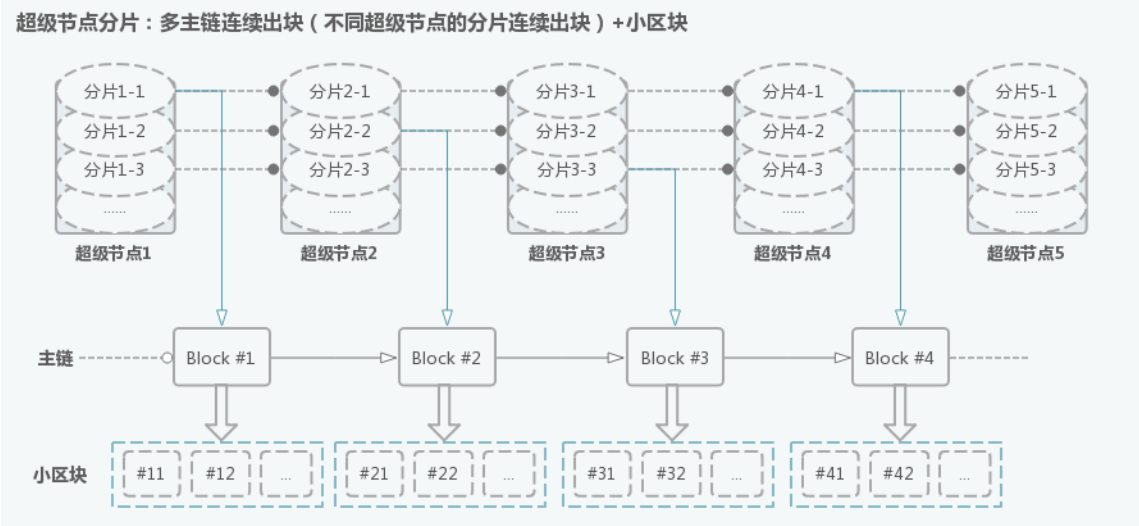
INB 公链的主链采用 VDPoS(DPoS+BFT+验证节点)共识算法，其中 21 个超级节点运行 DPoS 算法按照固定的顺序出块，固定顺序的原则为距离近的节点相邻，即网络传输时间短的节点相邻，以最小化每次出块后的网络传输时间，该顺序会在每一轮出块开始前确定。BFT 算法负责对主链的出块进行快速的一级验证。同时，INB 公链将根据公链的资源利用率对超级节点进行自动分片，分片的个数最多和超级节点的数量相等，分片以后每个分片相当于一个单独的超级节点，称为超级节点分片，21 个超级节点中各取一个分片构成了一个主链，从而形成了多主链平行运行的架构，用公式表达如下：

$$Num(M) = Num(S) = 1 / Max(IU(Net), IU(SNode)) <= 21$$

其中 Num(M)代表主链的个数，Num(S)代表分片个数，IU 代表利用率，IU(Net)

代表网络的利用率，IU(Node)代表超级节点的利用率。

在 INB 公链中，每一条主链的出块时间为 0.5 秒，为了节约网络的使用，每个分片超级节点会连续出一定数量的块(例如连续 6 个块)，再轮到下一个节点出块，这种连续出块的机制也会阻止分叉的产生。多主链都会以同样的机制出块，但是出块时间有 0.5/Num(M)的差距，并且在 0.5 秒内，每个主链的超级节点分片都会出块一次，形成多主链连续出块机制，在此 0.5 秒内出块的分片均不在同一个超级节点上，从而最大化的利用了网络和超级节点资源。例如在每个超级节点进行 20 分片的情况下，每隔 0.025 秒整个主链就会出块一次，比传统的 DPoS 算法 TPS 提高了 20 倍，并且大大缩短了等待时间，示意图如下所示。采用此种机制主链在现有条件下将实现每秒 10 万级以上(100,000+TPS)的交易数据处理能力，并且会随着节点和网络能力的提升而逐步提升。



超级节点分片示意图

2.4.1 动态规划确定出块顺序

前文提到，在每轮出块之前，都要提前确定出块的超级节点分片的顺序，以

能最快的同步区块数据，此问题可以描述为：已知任意两个节点的距离，求如何从一个节点遍历所有节点路径最短的问题，也就是 TSP[34]问题(Traveling Salesman Problem，旅行商问题)，INB 公链引入动态规划[29]算法解决该 TSP 问题。

2.4.2 自适应的多主链架构

2.4.2.1 超级节点的自适应分片

超级节点的自适应分片算法(Auto-Sharding Algorithm)会每隔一段时间根据超级节点的利用情况、网络利用情况自动执行一次，以最大化利用网络和超级节点资源，称为 INB 公链的自适应垂直扩展机制。一般情况下，分片的数量将大于 10，也就是将传统 DPoS 的 TPS 提高了一到两个数量级。

2.4.2.2 节点自适应奖励机制

除了主链自适应分片机制，INB 公链会根据节点的处理速度、网络延迟等情况，奖励不同数量的 INB，称为节点自适应奖励机制，以激励节点随着公链的负载增加，自发的增加配置和改善网络情况，从而实现了整个公链的垂直扩展，而不用早期投入非常多的资源，这在 INB 公链中称为自适应的多主链架构。这一点是非常有意义的，例如 CPU 和内存的不断升级、5G 的普及等，都将给主链的性能带来极大的提升。

2.5 水平扩展：多子链架构

INB 公链将交易数据保存于主链之上，将业务数据结构化的保存在子链上，

而普通非交易的业务数据的上链是众多 DApp 可以落地的关键因素。INB 公链采用多子链的结构将业务数据上链，并对该数据进行必要的加密和解密，是第一个实现业务数据结构化上链的公链。

2.5.1 子链的机制

子链由 DApp 或者社区发起启动子链的提案，并选择合适的共识算法，子链可以采用 BFT 类联盟链的共识机制，也可以采用 PoW、PoS、DPoS 等公链共识机制，根据不同的 DApp 对数据上链的效率、安全性等不同的需求而选择不同的共识算法,公链不对子链的共识机制作限制。

子链启动提案由超级节点进行投票，投票同意启动以后，公链将自动启动一个子链智能合约，用来对子链进行管理，然后等待满足条件的子链节点的加入以启动子链。

2.5.2 子链服务商

DApp 使用子链的资源将用 INBC 付费，INBC 是 INB Child Chain Coin 的缩写。在 INB 公链上有一个基于 Bancor[37]协议的智能合约，该智能合约可以进行 INB 和 INBC 的兑换，INB 可以通过该智能合约兑换成 INBC，同样，把 INBC 返还给该智能合约将以时价获得 INB。这种子链资源付费机制将刺激专门的个人或者机构加入整个公链生态，为 DApp 提供子链节点，称为子链服务商，要成为服务商必须要有一定数量的 INB 抵押到公链上，以防止作恶。

2.5.3 子链的数据

子链上可以保存的数据分为三种类型：INB 的交易数据、Token 的交易数据和业务数据。其中前两种统称为交易数据，跟现有公链保存的数据类似。第三种为 INB 公链创新的业务数据结构化存储方式，除了业务数据的各种版本以外，还将把业务数据的各种属性结构化的保存下来。

对于业务数据，数据本身可能会更改、属性会变更，这些改动会以数据版本和属性变化的形式保存下来，保证记录的可查、可追踪，真正实现了数据的结构化上链，为 DApp 的大规模应用奠定了技术基础。

2.5.4 子链数据的验证

子链会以某种共识算法运行，数据保存在子链的节点上，并且为了保证安全、可信，子链会把数据的验证信息保存到主链上用来做子链数据的验证。子链在产生一定数量的区块以后，会将该部分区块对应的默克尔树的树根保存到主链的区块上，以保证子链的安全和可信，对应的子链上的区块称为验证区块。

子链上的区块不一定每个都是验证区块，可能会有一定的高度间隔，此间隔 INB 公链会根据主链的资源利用情况来自动选择，当主链的资源利用率低的时候，可能每个区块的哈希值都保存到主链上，但是当主链的资源利用率高的时候会间隔的高度比较多，间隔的多少并不影响子链本身的效率，只是当使用主链对子链区块进行验证的时候效率会有一定的影响。主链会每隔一段时间对子链上的数据进行验证。

2.5.5 子链的反欺诈

一条子链由多个服务商开启，可能会由于去中心化程度较低，出现作弊的风险，监管节点有责任和义务对子链的运行情况进行监督，一旦发现有节点作弊，可以将作弊证明提交给主链进行投票验证，投票通过以后监督节点将获得 INB 作为奖励，并对作弊的服务商做出惩罚。

2.6 公链利用率

提高 TPS 的一个重要手段是在现有共识体系下，找到消耗资源的瓶颈，引入或者优化现有的算法和逻辑以提高整个公链资源的利用率，INB 公链针对公链的现有问题提出了几个提高资源利用率的手段，下面将进行论述。

2.6.1 公链利用率模型(IUM)

为了衡量整个公链的资源利用情况，INB 公链提出公链利用率模型的概念：INB Utilization Model，简称 IUM，该资源包括：超级节点的 CPU 和内存资源、网络资源。IUM 模型将给出公链资源利用率：INB Utilization，简称 IU，包括了公链网络利用率和公链超级节点利用率，分别用：IU(Net)和 IU(SNode)来表示。

我们设置任意两个超级节点之间的距离为 D(i, j)，网络带宽为 B(i, j)，当前的网络占用为 O(i, j)，则公链网络利用率 IU(Net) = Max(O(i, j)/B(i, j))。

我们设置每个超级节点的资源利用情况为 RU(i)，则公链超级节点利用率 IU(SNode) = Max(RU(i))。

2.6.2 IUM 的应用：超级节点自适应分片

由于整个公链的运行效率跟 IU 正相关，当 IU 很小时，应该充分利用公链资源以提高可扩展性。基于此概念，INB 公链提出根据 IU 的大小自动调整超级节点的分片数量和子链验证区块的高度间隔，以充分利用整个公链的资源，提高 TPS。

使用 VDPoS 共识算法，一条主链在满 TPS 运行的情况下，IU 仍然不能达到最大，所以 INB 公链提出多主链运行的架构，以充分利用公链的资源，提高公链的 TPS 和减小出块时间。多主链运行的时候避免一个节点的多个分片连续出块，防止相同节点之间的网络被多次占用，提高网络利用率。

同时 INB 公链提出一系列的机制来降低公链资源的使用，提高公链的处理能力，下面将从提高网络效率和降低节点资源消耗两个方面进行论述。

2.6.3 提高网络效率： Floyd 算法改进 Kademlia 网络

在 INB 公链中，主链将由 21 个超级节点构成，这些超级节点构成了一个 Kademlia[43](简称 KAD)网络，在所有超级节点均保存着其它超级节点的位置等信息，和传统的 KAD 不同的是，INB 公链中保存了任意两个节点的距离信息，并且引入 Floyd[32]算法提高节点之间数据传输的效率。

节点之间的数据传输在公链中是十分耗时的操作，直接影响了公链的出块时间，所以 INB 公链通过引入 Floyd 算法来提高网络的效率。当一个节点向其他节点传输数据的时候，不再是直接选择和其它节点通信，而是利用 Floyd 选择此节

点到其它节点的最短路径进行通信，从而提高网络效率。

我们设置任意两个超级节点之间的距离为 D(i, j)，则从一个节点向其它所有节点传输数据的问题就变为了一个求全局最短路径的问题，Floyd 能很好的解决这个问题。

INB 公链每隔一段时间都要重新计算任意两个超级节点之间的距离 D(i, j)，并更新 KAD 网络中相应的距离信息，当进行数据传输的时候，利用 Floyd 算法找到最短路径，按照最短路径进行传输，和“与所有节点直接连接传输数据”相比能很大程度上减少数据传输的时间。

2.6.4 降低节点资源消耗

节点的 CPU、内存等资源的消耗也是公链利用率的一个重要指标，在 INB 公链中，首先，利用 VDPoS 共识算法，不需要像 PoW 一样挖矿，很大程度上降低了资源消耗；然后，INB 公链给出了以下两种降低节点资源消耗的方法。

2.6.4.1 公链交易引擎：简化 Token 的发行和交易

INB 公链提出另外一个大幅度降低节点资源消耗的方式：将传统公链中 Token 的智能合约从智能合约业务中剥离出来，利用简洁的脚本语言的方式来执行，称为 INB 交易引擎(ITE, INB Transaction Engine)。在以太坊上，有很大比例的智能合约是 Token 的发行合约，至少有一半以上的交易是 ETH 和 Token 的交易，通过启动虚拟机调用智能合约的方式来执行，消耗的系统资源非常多，如果使用 INB 交易引擎来执行，将能极大的降低资源消耗，在 Token 的交易、验证的

时候都将节约很多时间。

INB 公链将默认支持 ERC20、ERC721、ERC223、ERC621 和 ERC827 等协议，用户可以通过可视化的方式申请符合这些协议的 Token，这样这些 Token 在交易等执行过程中将不再调用智能合约，从而消耗很少的资源。同时 INB 公链还会对支持的 Token 发行协议进行更新，以期能支持绝大多数的协议，彻底简化 Token 的发行和执行。

2.6.4.2 区块打包和验证的分片

INB 公链上将应用一种区块打包和验证的分片方法，如果每个区块的数据量很大，区块的打包和验证都很耗时，会将每个块利用 VRF 随机抽签机制进行分片并发给不同的节点进行打包和验证，以最大化的提高节点的利用率，降低处理时间。同时，此种随机抽签将由其它节点进行验证，防止作弊发生。

2.7 智能合约

INB 公链的智能合约分为两种: 可视化的智能合约和基于 IVM 的图灵完备的智能合约。

2.7.1 可视化的智能合约

INB 公链的设计思维是简化用户使用智能合约的场景以及减少漏洞的发生，尽量将通用的智能合约模块化，基于此理念 INB 公链将提供一种可视化的智能合约创建方法，包括:

- 简化发币：通过填写关键参数的方式就可以生成一个 Token，最大化的方便了用户的发币以及以后的交易、查看余额等操作。生成的 Token 可以是内置的 ERC20[36]等各种协议，并且原生支持基于 Bancor 协议[37]的 Token 发行方式，支持存储准备金制的发行 Token，即需要抵押一定数量的 INB 才可以发行 Token。

- INB 公链将会提供和数据的产生、处理、分析、结果展现等和数据相关的智能合约，用户可以通过选择的方式设计一个包含多个模块智能合约的大的智能合约。

- INB 公链推荐将 DApp 的功能智能合约化，每个功能都应该对应智能合约，保证流程的可信。

2.7.2 基于 IVM 的智能合约

同时 INB 公链也会提供图灵完备的智能合约，以供不通用的、更复杂的逻辑调用。INB 公链将通过全新的虚拟机(IVM, INB Virtual Machine)来支持智能合约，IVM 将使用 WebAssembly(WASM)[38]方案来实现，这意味着开发者可以使用任何熟悉的编程语言来开发智能合约，并且有着更优越的性能。

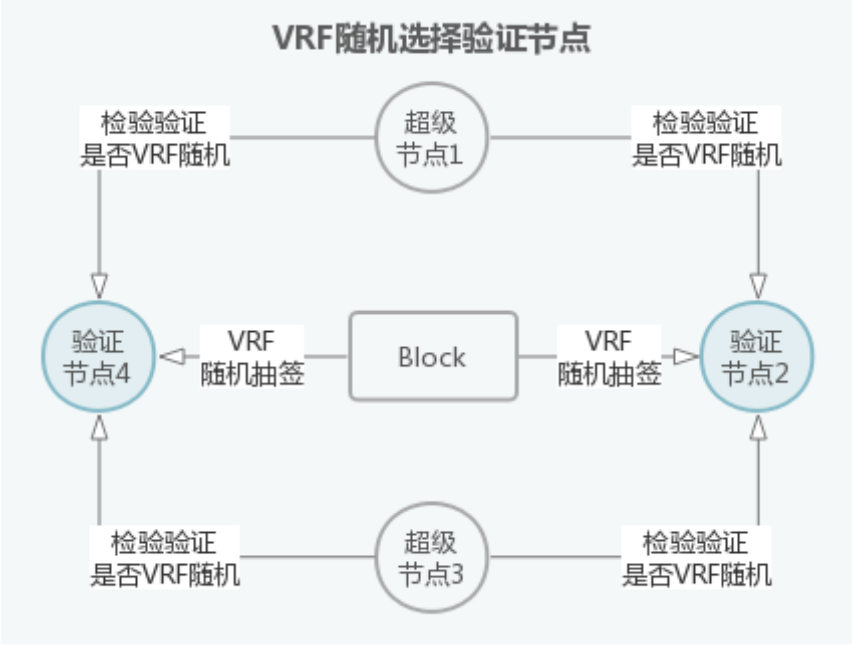
2.7.3 智能合约服务商：大数据算法的落地

INB 公链会将智能合约的所有权和使用权进行分离，引入智能合约服务商概念，刺激服务商提供一些大数据处理、分析相关等的智能合约，在智能合约中集成大数据算法，供第三方付费使用。

2.8 基于 VRF 的随机验证机制

INB 公链引入可验证随机函数：VRF，充分利用 VRF 的随机性和可验证的特点，实现可验证的随机选择逻辑，防止作弊的发生。有两个地方使用了 VRF：验证节点的选择与出块和验证的分片逻辑。

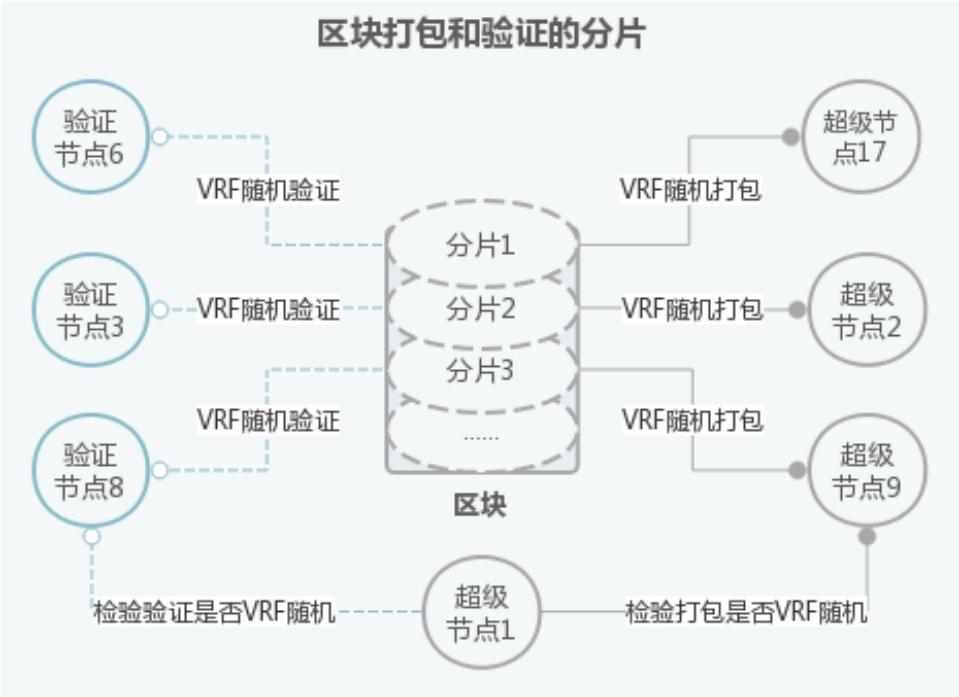
在超级节点出块以后，首先超级节点要对块进行验证，同时，为了防止超级节点的联合作弊，需要随机选择一组验证节点对快进行验证，此时，使用 VRF 的随机抽签机制来选择验证节点，有效防止了超级节点和验证节点的联合作恶。此种随机抽签将由其它节点进行验证，防止作弊发生。



基于 VRF 随机选择验证节点示意图

INB 公链上将应用一种创新的区块打包和验证的分片方法，如果每个区块的数据量很大，区块的打包和验证都很耗时，会将每个块利用 VRF 随机抽签机制进行分片并发给不同的节点进行打包和验证，以最大化的提高节点利用率，降低

处理时间。此种随机抽签将由其它节点进行验证，防止作弊发生。



区块打包和验证的分片示意图

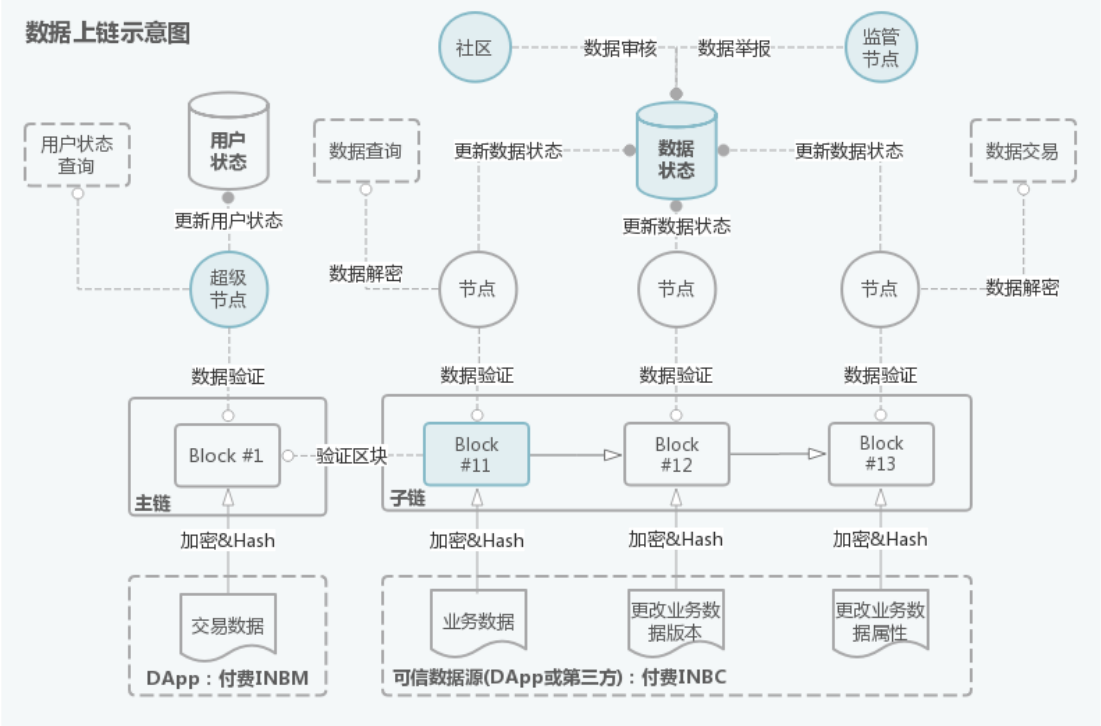
2.9 数据的上链机制

为了支持更多的应用落地，真正的实现数据的结构化上链，INB 公链将链上保存的数据分为几种：INB 的交易数据、Token 交易数据和业务数据，其中前两种统称为交易数据，跟现有公链保存的数据很类似，第三种为非交易数据的业务数据结构化上链。

业务数据的结构化上链是 INB 公链的一大优势，不同于传统公链项目仅将数据内容上链，并没有将数据的属性和过程上链，无法真正保证数据的可信。同时上链的数据并不是结构化的，不能表达数据之间的关联、属性和数据之间的关联等信息，在 INB 公链中使用业务数据的结构化上链方式，业务数据可以通过链

本身进行自解释，不再需要通过第三方应用对非结构化的数据进行解释，真正实现了数据的可信、确保了数据的价值传递。

整个数据上链的业务逻辑示意图如下：



数据上链示意图

2.9.1 数据的分层存储模型

INB 公链将对前述的数据进行分层存储，以满足各种数据的需求。主链上将保存 INB 和 Token 的交易数据，满足交易数据对高可信和不可篡改的需求；业务数据保存于子链之上，满足对高可用的需求。对于一些低价值或者过大的业务数据上链需求，INB 公链建议将数据的哈希值保存到链上，原始数据在链下保存。

INB 公链数据分层存储模型图

2.9.2 业务数据的上链

INB 公链定义了两种业务数据：主业务数据和关联业务数据，主业务数据是最初创建的业务数据，例如调研问卷的原始信息。关联业务数据是和主业务数据相关联的过程业务数据，一般对应了用户的某些操作，有时也称为操作数据，例如用户回答某个问卷的数据。在 INB 公链的区块中保存了业务数据的创建记录、业务数据的修改记录、业务数据普通属性的修改记录、账户的业务数据操作记录、业务数据的交易记录等信息，完成业务数据的上链过程，达到了不可篡改、可追溯的目的。

在很多应用场景中，业务数据的上链有助于提高可信度和可追溯性，例如：用户的调研数据、商品的购买数据、品牌的评价数据等，并且这些数据的购买量、

评价值等都有非常强的上链需求，而这种需求在现有的公链中是没办法很好的支持的。基于此，INB 设计了业务数据的结构化上链机制，业务数据是 INB 公链创新的数据结构化存储方式，将数据的各种版本、数据的各种属性、数据的关联关系都结构化的存储到链上，并进行可追溯的更新和维护。业务数据的上链有三个重要的特征：

- 数据是支持更改的，在 INB 公链中体现为数据的版本，每次更改都将以版本的形式保存在链上，可以追溯到数据的所有更改历史。

- 数据的属性被结构化的保存下来，数据的属性又分为两种：一种称为关联属性，一种称为普通属性。前者属性本身并不能被直接更改，而是关联链上的另一条数据被自动更改，例如购买数据，当有用户的购买数据保存到链上才能被自动增加；后者可以被直接更改，例如数据的价格数据，由数据的拥有者进行定价，和其它链上的数据关联性并不是很大。同样所有属性的更改历史都将被保存下来。

- 数据的关联，数据之间的关联关系是数据产生或者变化的过程，是数据真正可信的基础，INB 公链对该关联关系进行了保存，真正实现了数据的可信。例如上条所述的用户购买数据和商品购买数量的关联，都通过公链自动完成关联和更新。

- 加密验证，数据和属性的更改都需要使用用户的私钥进行加密，经过公链验证以后才可以保存和更改。

2.9.3 业务数据的安全

在数据上链之前，对于敏感的数据、用户希望进行交易的数据，可以设置为加密数据，这样 INB 公链将会使用用户的私钥对数据的一部分进行加密，并单独保存到属性：加密字段中，其它用户查看数据的时候将只能看到加密以后的数据，确保了数据的安全。当用户想查看或者交易数据的时候需要使用自己的私钥或者授权的私钥解密以后才可以进行查看或者交易。加密、解密的过程由超级节点完成，确保公平可信。

2.9.4 业务数据的状态

传统的公链在链上保存了账户的全局状态：账户余额等信息，INB 公链除了保存账户的全局状态信息，还保存了上述业务数据的全局状态信息，如下表所示：

数据哈希	不可更改
原始数据哈希	不可更改
数据的版本	业务数据内容
关联的数据	不可更改
价格	普通属性
浏览次数	关联属性
喜欢次数	关联属性
不喜欢次数	关联属性
收藏次数	关联属性
购买次数	关联属性
级别	普通属性
是否可以交易	普通属性
权限	普通属性
加密字段	普通属性
审核者	关联属性
举报者	关联属性
是否有问题	普通属性
分类	普通属性
所有者	关联属性
创建者	关联属性

INB 公链保存了数据的全局状态信息，可以很方便、快速的对业务数据的全局状态信息进行查询，方便更多的应用数据上链，例如：调研数据、商品购买数据、评价数据、评论数据、以及内容的浏览信息等。INB 公链并不推荐将多媒体文件或者过大的业务数据上链，并将对每次上链的数据大小作出限制，超过限制的数据推荐将数据的哈希值保存到链上，原始数据在链下保存。

2.9.5 业务数据的交易

INB 公链将提供业务数据的交易模块，对于可以交易的数据提供数据交易的原生支持，加密数据在收到用户的交易请求并获得数据所有用户或者授权用户的同意以后，将会把该数据解密并把数据的查看权给新用户，形成一个新的全局数据状态，同样的加密字段将被新用户的私钥加密以后保存。数据交易都将使用 INB 或者 INBC 进行，并按照数据属性中的价格收取，同时公链将收取一定的手续费作为矿工费。

2.10 可信数据源机制

现在有的很多平台已经保存了用户的很多数据，例如京东中的购物数据、微博中的兴趣数据、微愿中的投票数据等，用户有很强的需求将存量数据上链并且进行交易。为了适应这种场景，INB 公链提供了一种可信数据源机制，由社区发起可信数据源申请，超级节点进行投票，投票通过的可信数据源将被写入公链中，然后用户在可信数据源产生的数据可以保存到公链上，以进行交易或者查看。同样，INB 公链上的 DApp 都是原生的可信数据源，用户在 DApp 中产生的业务数据都可以保存到链上。

2.11 数据的审核和举报

业务数据上链以后，由于数据的所有者和产生过程需要依托于第三方的DApp，并不能做到过程的完全可信，所以 INB 公链提供了原生的数据审核和举报功能。

上链以后的数据，可以由其他用户进行审核，公链将把审核的过程上链，审核通过和不通过的用户都将被保存下来，接下来具体的处理逻辑将根据 DApp 不同处理逻辑不同，例如一个 DApp 可以认为必须 100%的用户审核通过才能将该数据展现并进行交易，另外一个 DApp 可能认为只需要 60%的人审核通过就可以进行交易。同时，DApp 可以给审核的用户以 INB 奖励。

同样，上链以后的数据，可以由其他用户进行举报，举报过多的数据，DApp 可以进行下架处理，不允许再查看和交易，具体的数量要求也是由 DApp 根据内部逻辑指定。同时，DApp 可以给举报的用户以 INB 奖励。

2.12 引入 DApp 角色

为了适应应用的区块链落地，INB 公链引入 DApp 角色，将 DApp 角色和账户区分开，众多 DApp 组成了 INB 的数据生态。DApp 由多个账户和多个智能合约组成，多个账户用于 DApp 拥有的币/Token 和用户以及公链的币/Token 之间的流转，多个智能合约是 DApp 的功能智能合约化的体现。

同时，DApp 将和 INB 公链上的业务数据和操作相关联，对于链上保存的业务数据和操作，将存储数据产生的 DApp。

2.12.1 公链资源使用的代付费

同时 INB 公链支持代付费功能，DApp 将可以选择代替用户进行公链资源使用的付费，将普通用户与底层链的存储和使用分离，避免用户过多的理解区块链的存储和付费逻辑，为区块链的大规模应用和普及奠定基础。

2.13 链上资源的使用规则

为了应对不同的使用场景，INB 公链将提供两种链上资源的使用方式：基于 INBC 的业务数据相关资源使用方式和基于 INBM 的交易数据相关资源使用方式。

2.13.1.1 业务数据相关资源的使用

业务数据相关资源为子链相关资源，由于业务数据的存储量比较大，需要在每次数据的产生、交易进行单独付费，INB 公链将会发行一个内部的代币：INBC，该代币基于 Bancor 协议发行，使用 INB 进行兑换，用于业务数据的产生、查看、交易等操作的付费，同时用户如果有剩余的 INBC 没有使用，可以返回给 INB 公链智能合约以时价兑换为 INB。

2.13.1.2 交易数据相关资源的使用

交易相关资源包括超级节点的资源以及超级节点之间的网络资源，对于交易数据，INB 公链也会发行一个内部的代币：INBM 和一个抵押 INB 的智能合约，该代币在 INB 抵押到智能合约后每天产生一次，可以在创建智能合约、交易、调用智能合约等和交易相关的场景中使用。同时如果不再需要 INBM，用户可以将

抵押的 INB 全部赎回。

2.14 跨链

随着越来越多的公链上线，对于跨链[42]的需求已经十分强烈，能使不同币和 INB 公链的币进行原子级别的交互是 INB 公链的又一个特点。同时 INB 公链本身的主链和子链之间也需要交互，接下来将从这两个方面进行论述。

2.14.1 INB 公链和其它公链的跨链：状态通道合约

对于和其它公链的跨链，INB 公链使用状态通道的解决方案，INB 公链提供一系列原生的状态通道智能合约，每个智能合约对应一条公链以及公链上对应的币的映射。

例如对于 BTC，INB 公链将提供一个 BTC 跨链状态通道智能合约、一个接收 BTC 的账户和 IBTC 代币，其中的 IBTC 代币为 BTC 在 INB 公链的 1:1 映射。当用户将 BTC 存入该接收 BTC 的账户以后，该智能合约将自动给用户分配等量的 IBTC 代币，该 IBTC 代币的交易代表了真正 BTC 的交易，当用户想取回 BTC 的时候，把 IBTC 返回给智能合约，智能合约将自动把账户的 BTC 打给用户指定的账户。

这种内置的跨链机制，可以帮助链上很方便的实现去中心化交易所等对跨链有强烈需求的场景。

2.14.2 INB 公链内部的跨链：委托执行和公证人

INB 公链由于有主链和子链两种链，就涉及到两种跨链方式：主链和子链之间的跨链以及子链之间的跨链。

由于用户可能同时在主链和子链上同时进行交易，所以涉及到主链和子链之间的跨链问题，INB 公链提供了两种方式进行交易数据的跨链：

- 一种类似于 INB 公链和其它公链的跨链方式，提供一个智能合约来解决跨链问题，此种方式相当于子链是主链的一个状态通道；
- 一种是委托执行的跨链方式，当用户在子链上执行交易的时候，该交易将通过子链的智能合约转到主链去执行，当主链执行完成以后再将结果返回给子链，以确保交易的正确执行。

对于子链之间的跨链，将使用主链作为公证人，在两个子链的智能合约中对相关的数据和资产进行监控和锁定，在两个子链的交易或者数据操作都完成以后，才进行解锁，以保证整个跨链操作的原子性。

2.15 基于 ECDSA 算法的加密和验证机制

INB 公链将使用基于公钥和私钥的账户体系，每个账户对公链发起的交易、数据上链、数据交易等请求，都将由非对称加密：椭圆曲线数字签名(Elliptic Curve Digital Signature Algorithm，缩写为 ECDSA)[15]算法进行加密，超级节点将对加密的数据进行验证是否为该账户加密，来确认请求的合法性，只有合法的请求才可

以被执行同时，区块的验证也将基于 ECDSA 算法，对每个区块内的请求进行验证，防止请求的伪造和篡改。

2.16 基于 MPT 的数据结构

MPT 是 Merkle Patricia Trie 的缩写，源自于 Trie 结构，又分别继承了 Patricia Trie[20]和 Merkle Tree[23]的优点，并基于内部数据的特性，设计了全新的节点体系和插入、载入机制。INB 公链在交易数据、用户状态、业务数据和数据状态的存储以及子链数据的验证中都用到了 MPT 数据结构，以及基于 MPT 的快速查询和验证。

INB 公链将使用 RLP(Recursive Length Prefix)[40]编码将数据进行序列化，并存储到 LevelDB[41]中。

2.17 链上数据的查询

由于业务数据的数据量十分庞大，INB 公链将利用缓存机制、NoSQL 等提供更快速的链上数据查询方式，真正满足亿级互联网应用的需求。

3. Insight Chain 操作系统

基于以上的技术架构，INB 公链提供了 9 大系统组成了完整的操作系统，供 DApp 等公链用户使用。

3.1 智能合约管理系统

INB 公链提供了智能合约管理系统，用户可以对自己创建的 Token 和智能合约进行管理。

Token 管理包括了 Token 的创建、查询、交易等功能。

智能合约管理包括创建智能合约、调用智能合约、付费设置、收费查询等功能。

3.2 可信数据源系统

可信数据源系统用于对 INB 公链中的可信数据源进行管理，可信数据源产生的数据可以保存到公链上，并进行数据交易，功能包括：发起可信数据源投票、举报可信数据源、可信数据源查询等。

3.3 子链管理系统

子链管理系统用于对 INB 公链上的子链进行管理，包括启动子链投票、举报子链、查看子链数据、子链服务商申请等。

3.4 账户管理系统

INB 公链设计了一种账户的分层管理模型，包括了账户之间的从属关系、账户权限的分级、账户的授权等，该系统功能包括：从账户管理、权限管理、多账户授权、授权管理、多重签名、费用代付(INBC、INBM、INB 都可以支持代付)等，

为企业账户管理账户等场景提供便利，并充分保证了账户安全。

公链将提供统一的账户体系，公链上的 DApp 都可以使用公链账户进行登录，登录以后用户可以直接在第三方 DApp 中授权使用钱包的功能，并且用户可以对已经授权的 DApp 进行管理。

3.5 数据交易系统

INB 公链提供了内置的数据交易系统，包括：数据的托管、购买、查询等功能。数据托管以后其他用户可以直接通过委托的托管方购买数据。

3.6 投票系统

INB 公链提供的投票系统包括：超级节点的选举投票、子链的投票、投票投诉节点和数据、可信数据源投票等功能。

投票可以分为匿名投票和实名投票两种，匿名投票将使用环签名机制进行加密，实名投票将在链上记录用户的每一次投票过程。

3.7 资源管理系统

INB 公链提供的资源管理系统包括了两个重要功能：INBC 的管理和 INBM 的管理。其中 INBC 为使用 INB 兑换获得，INBM 为抵押 INB 自动获得，两种代币的管理都通过智能合约自动完成，分别用于使用子链和主链的资源付费。

3.8 任务系统

INB 公链内置了任务系统，为 INB 公链在业务层的一个深入探索，将任务的发起、接受、定价、付费等流程固化为智能合约，并且提供机制进行统一管理，供调研、分包等有和任务相关的场景使用。

3.9 RPC API 系统

INB 公链的各种类型的节点都会提供 RPC API，以对外提供链上资源的使用和链上数据的查看功能。

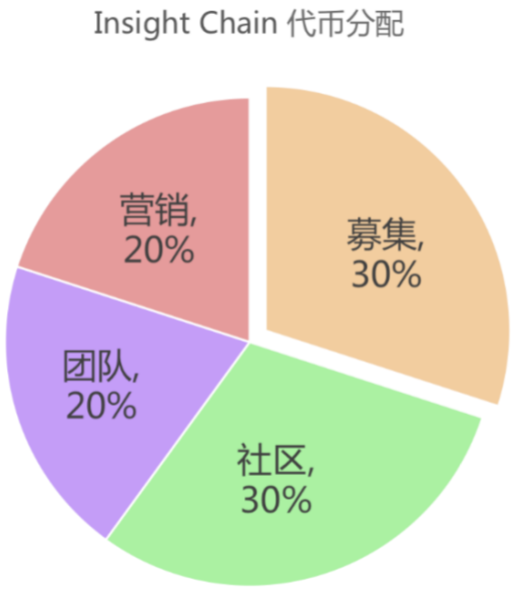
4. Insight Chain 生态系统

作为 3.0 时代的公链，Insight Chain 认识到除了高 TPS，能真正的支持互联网和传统商业的场景落地也同样重要，同时我们相信 2019 将成为应用落地元年，在区块链行业、互联网、商业的共同探索下，将会有大批不同于之前大家耳熟能详的落地应用场景出现，将把区块链行业推到一个新的高度，是整个区块链行业进入 2.0 阶段的标志。所以我们把 Insight Chain 的核心目标定义为两个：在满足去中心化和安全性的前提下提高可扩展性、满足更多应用业务数据上链需求。

在 INB 公链中，各方参与者构成了一个完整的生态系统，以保证 INB 公链的健康运行和发展，包括：超级节点、验证节点、监管节点、普通节点、子链服务商、子链节点、智能合约服务商、INB 社区、INB 委员会和 INB 团队。

4.1 经济模型

Insight Chain 在初始设计时已经设计好币的使用场景，引用 Insight Chain 白皮书中[1]内容如下：



- 募集：30%，以私募的方式募集资金，限机构和大型个人投资者认购，锁定期一年；
- 社区：30%，将用于补贴公链中的合作方，例如合作的 DApp、节点等；
- 团队：20%，将用于奖励团队、顾问等参与者，以及之后的基金会运营，锁定期三年；
- 营销：20%，用于产品、募资等营销活动。

INB 公链中并不会对 INB 分配方案做出改动，只是为了防止 INB 的滥用和增发，将 61%的币改为挖矿产出机制(实际上在 INB 公链并没有真正的挖矿，在此

我们把节点获得公链的激励过程称为挖矿)，具体策略如下：

募集：30%，由于在一年的锁定期内要发放完毕，需要在挖矿之前预留；

社区：30%，分为三部分，一部分 20%挖矿产生，分配给挖矿节点和验证节点；一部分 9%，跟随挖矿逐步产生，分配给社区；剩下的 1%，预留出来，分配给社区的早期参与者。

团队：20%，分为两个部分，一部分 1%预留出来分配给 INB 的开发和运营团队(依然遵守白皮书书中所承诺的 3 年解锁期)，剩余的 19%将都随着挖矿逐步产生。

营销：20%，分为两个部分，一部分 1%预留出来做 INB 公链相关的营销，剩余的 19%随着挖矿逐步产生。

从总数量来看，INB 将预留 33%分配给募集、营销和团队，剩下的 67%都将随着挖矿产生，逐步释放，其中的 20%分配给超级节点、验证节点、监管节点等作为奖励，47%分配给团队、社区和营销，详情如下表所示。

分配	总量	比例	说明	预留	挖矿
募集	30%	30%	预留给私募参与者，一年释放	30%	
社区	30%	20%	矿工费，给参与节点的奖励		20%
		9%	随挖矿等比例产生，社区激励		9%
		1%	早期社区激励	1%	
团队	20%	1%	预留给早期团队激励，分三年释放	1%	
		19%	随挖矿等比例产生		19%
营销	20%	1%	预留给 INB 的早期营销	1%	
		19%	随挖矿等比例产生，用于后期营销		19%
合计	100%	100%		33%	67%

4.2 公链治理

INB 公链的治理分为两种：一种是链上治理，一种是链下治理。链上治理主要由超级节点执行，包括子链的启动、有问题的节点和业务数据的处理等；链下的治理由 INB 决策委员会负责，决策委员会由 INB 的持有者进行选举，一年一届。

4.2.1 超级节点的选举

超级节点将由 INB 的持有者进行选举，采取一币一票制，对参选的实体进行选举，投票可以实时进行，INB 公链每隔一定时间(1 小时内)将根据实时的投票结果更新超级节点。

4.2.2 提案

INB 公链采取提案制收集社区对 INB 公链各种发展的建议，被委员会采纳的天将会获得一定的 INB 激励，该奖励来自于社区的 INB 分配部分。

4.2.3 监管节点

INB 公链中引入监管节点角色，对公链的运行情况进行监督，当发现问题将提交给超级节点做投票处理，发现的问题包括但不限于：数据的质量问题、超级节点的作弊、子链节点的作弊、验证节点的作弊等。如果超级节点投票通过将对监管节点进行奖励，并且对作弊行为作出相应的处罚：包括但不限于扣除抵押的 INB、列入黑名单、封禁数据等。

在 DApp 中该角色类似于运营审核的角色，执行数据上链以后的审核职责。发现有问题的数据如果超级节点投票同意，则会封禁起来，其它任何对该数据的请求节点都不再处理。

4.3 服务商生态

INB 公链将引入子链服务商和智能合约服务商。前者负责启动子链为子链上的 DApp 提供数据上链服务，并且通过收取 INBC 盈利。后者将编写大数据处理、AI 等相关的高质量智能合约，并将该智能合约以服务的形式开放给其它 DApp 使用，收取 INB 使用费。

4.4 数据交易生态

INB 公链将内置对数据交易业务的支持，交易的数据可以是链上原生的数据，也可以是经过验证的第三方的数据。

4.5 生态应用场景

INB 公链有两种重要的应用场景：对业务数据有上链需求的应用和去中心化交易所，前者例如调研行业、电商行业、社区等。

4.6 INB 公链在调研行业中的应用

目前的调研行业中，调研用户作假、调研数据作假、数据链条长的问题十分突出，Insight Chain 首提调研数据上链的概念，将调研的场景、数据链上化，去除数据中介，将调研的逻辑、数据的处理等智能合约化，实现调研数据价值回归

个人和调研数据的点对点交易，真正做到调研行业的区块链改造。

5. Insight Chain 路线图

INB 公链目前已经处于开发阶段，路线图如下：

2018 Q3-Q4: Insight DApp 的开发和运营，INB 公链的设计;

2019 Q1：技术白皮书的撰写、方案可行性验证和原型系统的搭建;

2019 Q2: 测试网和主网上线, 实现基于 DPoS+BFT 共识的多主链运行机制;

2019 Q3：实现多子链运行机制、业务数据上链、INB 交易引擎;

2019 Q4：实现基于 IVM 的智能合约，引入智能合约服务商;

2020 Q1：实现公链利用率模型、主链的自适应分片;

2020 Q2：实现基于 VRF 的随机验证机制，完整实现 VDPoS 共识算法;

2020 Q3：实现公链之间以及主链、子链之间的跨链机制;

2020 Q4：实现公链上数据交易和业务数据的高速查询机制。

6. 参考文献

[1] Insight Chain Founding Team, Insight Chain White Paper V 1.0, https://github.com/insight-chain/documentation/blob/master/en_US/insight_chain_whitepaper_v1.0_en_US.pdf, 2018

[2] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system[J]. 2008.

[3] Buterin V. A Next-Generation Smart Contract and Decentralized Application Platform[J]. white paper, 2014.

[4] EOS.IO Technical White Paper v2[EB/OL]. https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md, July, 3, 2018.

[5] The ZILLIQA Team, The ZILLIQA Technical Whitepaper, 8, 2017

[6] Joseph Poon, Vitalik Buterin, Plasma: Scalable Autonomous Smart Contracts, 11, 2017

[7] Wood T, Singh R, Venkataramani A, Shenoy P, Ceeehet E. ZZ and the art of practical BFT execution. In: Proc. of the 6th Conf. on Computer Systems. New York: ACM Press[J], 2011.

[8] Micali, Silvio; Rabin, Michael O.; Vadhan, Salil P. (1999). "Verifiable random functions". Proceedings of the 40th IEEE Symposium on Foundations of Computer Science[J]. pp. 120–130.

[9] Castro M, Liskov B. Practical Byzantine Fault Tolerance And Proactive

Recovery. ACM Trans. on Computer Systems[J]. 2002.

[10] Serafini M, Nokor P, Dobre D, Majuntke M, Suri M. Scrooge: Reducing the Costs of Fast Byzantine Replication in Presence of Unresponsive Replicas. In: Proc. of the 2010 IEEE/IFIP Int'l Conf. on Dependable Systems and Networks[J]. 2010.

[11] Dmitry Khovratovich, Christian Rechberger & Alexandra Savelieva (2011). "Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 family" (PDF). IACR Cryptology ePrint Archive. 2011:286.

[12] Mario Lamberger & Florian Mendel (2011). "Higher-Order Differential Attack on Reduced SHA-256" (PDF). IACR Cryptology ePrint Archive. 2011:37.

[13] Ji Li, Takanori Isobe and Kyoji Shibutani, Sony China Research Laboratory and Sony Corporation, Converting Meet-in-the-Middle Preimage Attack into Pseudo Collision Attack: Application to SHA-2

[14] Dodis, Yevgeniy; Yampolskiy, Aleksandr. (2005). "A Verifiable Random Function With Short Proofs and Keys". 8th International Workshop on Theory and Practice in Public Key Cryptography. pp. 416–431.

[15] Koblitz, N. (1987). "Elliptic curve cryptosystems". Mathematics of Computation. 48 (177): 203–209. doi:10.2307/2007884. JSTOR 2007884.

[16] Miller, V. (1985). Use of elliptic curves in cryptography. CRYPTO. Lecture Notes in Computer Science. 85. pp. 417–426. doi:10.1007/3-540-39799-X_31. ISBN 978-3-540-16463-0.

[17] Bernstein, Daniel J.; Lange, Tanja. "SafeCurves: choosing safe curves for

elliptic-curve cryptography". Retrieved October 1, 2016.

[18] Perlroth, Nicole; Larson, Jeff; Shane, Scott (2013-09-05). "N.S.A. Able to Foil Basic Safeguards of Privacy on Web". New York Times. Retrieved 28 October 2018.

[19] Morin, Patrick. "Data Structures for Strings" (PDF). Retrieved 15 April 2012.

[20] Knizhnik, Konstantin. "Patricia Tries: A Better Index For Prefix Searches", Dr. Dobb's Journal, June, 2008.

[21] Morrison, Donald R. Practical Algorithm to Retrieve Information Coded in Alphanumeric

[22] Merkle, R. C. (1988). "A Digital Signature Based on a Conventional Encryption Function". Advances in Cryptology — CRYPTO '87. Lecture Notes in Computer Science. 293. pp. 369–378. doi:10.1007/3-540-48184-2_32. ISBN 978-3-540-18796-7.

[23] Becker, Georg (2008-07-18). "Merkle Signature Schemes, Merkle Trees and Their Cryptanalysis" (PDF). Ruhr-Universität Bochum. p. 16. Retrieved 2013-11-20.

[24] Koblitz, N. (1987). "Elliptic curve cryptosystems". Mathematics of Computation. 48 (177): 203–209. doi:10.2307/2007884. JSTOR 2007884.

[25] Miller, V. (1985). Use of elliptic curves in cryptography. CRYPTO. Lecture Notes in Computer Science. 85. pp. 417–426. doi:10.1007/3-540-39799-X_31. ISBN 978-3-540-16463-0.

[26] Bernstein, Daniel J.; Lange, Tanja. "SafeCurves: choosing safe curves for

elliptic-curve cryptography". Retrieved October 1, 2016.

[27] Perlroth, Nicole; Larson, Jeff; Shane, Scott (2013-09-05). "N.S.A. Able to Foil Basic Safeguards of Privacy on Web". New York Times. Retrieved 28 October 2018.

[28] Adamsky, Florian (2015). "P2P File-Sharing in Hell: Exploiting BitTorrent Vulnerabilities to Launch Distributed Reflective DoS Attacks".

[29] Denardo, E.V. (2003), Dynamic Programming: Models and Applications, Mineola, NY: Dover Publications, ISBN 978-0-486-42810-9

[30] Sniedovich, M. (2010), Dynamic Programming: Foundations and Principles, Taylor & Francis, ISBN 978-0-8247-4099-3

[31] Moshe Sniedovich (2002), "OR/MS Games: 2. The Towers of Hanoi Problem", INFORMS Transactions on Education, 3 (1): 34–51.

[32] Floyd, Robert W. (June 1962). "Algorithm 97: Shortest Path". Communications of the ACM. 5 (6): 345. doi:10.1145/367766.368168.

[33] Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L. (1990). Introduction to Algorithms (1st ed.). MIT Press and McGraw-Hill. ISBN 0-262-03141-8. See in particular Section 26.2, "The Floyd–Warshall algorithm", pp. 558–565 and Section 26.4, "A general framework for solving path problems in directed graphs", pp. 570–576.

[34] Johnson, D. S.; McGeoch, L. A. (1997). "The Traveling Salesman Problem: A Case Study in Local Optimization" (PDF). In Aarts, E. H. L.; Lenstra, J. K. Local Search in Combinatorial Optimisation. London: John Wiley and Sons Ltd. pp.

215–310.

[35] "'Travelling Salesman' movie considers the repercussions if P equals NP". Wired UK. April 26, 2012. Retrieved April 26, 2012.

[36] "ERC-20 Token Standard - The Ethereum Wiki". Theethereum.wiki. Retrieved 30 August 2017.

[37] Bancor Protocol - Bancor Network, https://www.bancor.network/

[38] "WebAssembly High-Level Goals". GitHub / WebAssembly / design. 11 December 2015.

[39] Bright, Peter. "The Web is getting its bytecode: WebAssembly". Ars Technica. Condé Nast. 6, 2018.

[40] Recursive Length Prefix, Web3j, https://docs.web3j.io/rlp.html

[41] "Google Open-Sources NoSQL Database Called LevelDB". ReadWriteWeb. July 30, 2011. Retrieved July 30, 2011.

[42] Joseph Poon and Tadge Dryja. Lightning Network. https://lightning.network/lightning-network-paper.pdf, Mar 2015.

[43] Kademlia: A Peer-to-peer information system based on the XOR Metric.