Ælf

一个基于多级侧链的并行化区块链框架

社区翻译版本

摘要

区块链技术在过去几年中得到了快速的发展。中本聪的比特币起初是作为一个安全的去中心化的P2P交易机制出现，进而证明了去中心化的数字货币是可行的。接着以太坊又成功实现了多功能的“智能合约”，为区块链技术发展做出了重要的贡献。以太坊极大地释放了区块链跨行业、跨场景的应用潜力，并引出了许多基于以太坊发行的代币。然而，这仅仅只是区块链技术突破商业应用开始，还有更多令人振奋的“区块链技术革命”待我们去开拓。现今我们正处于区块链和现实商业世界结合的转折点上，我们必将会带来更多可靠的数字资产。

为了跨入区块链技术的新时代，我们需要一个能满足各种商业需求的多功能系统。这个系统必须解决以下几个主要问题：

1. 现在的区块链扩展性差，单个节点/矿机的性能往往制约着整个系统的性能；
2. 现在的区块链并没有为不同的智能合约划分不同的资源，这导致了智能合约间执行的干扰；
3. 现在的区块链并没有可以接纳更新和新技术的共识机制。

在这篇白皮书里，我们将介绍一个真正达到了商用标准的、采用了IT行业最前沿设计和技术的高效区块链系统。我们希冀它创造一个区块链世界的“LINUX生态”。我们专注于定义和提供最基础的、关键的、以及耗费开发者精力的系统组件，并且在现有区块链技术的基础上对它们做出极大改进。这个系统允许开发者自定义它以满足商业场景的需求，特别是对于跨行业的不同商业场景。该系统包含以下主要特点：

1. 引入了主链和多级侧链的设计理念用于处理不同的商业场景模型， 每条链针对一个业务目标，将不同的业务分配到多条侧链上来提高处理效率；
2. 可以通过消息和外界的区块链通信，比如比特币、以太坊；
3. 对非竞争交易和云服务允许并行计算；
4. 为每一条链定义最小可行区块和创世智能合约集，从而降低数据复杂度以及实现高度可定制化；
5. 允许利益相关方对协议作出修改，包括重新定义共识机制；
6. 允许侧链根据共识机制动态加入或者退出主链，引入侧链间的竞争和完善的动力；

目录

摘要 **2**

目录 **3**

1. 现有区块链系统 **5**

1.1 通用区块链 vs 复杂业务场景 5

1.2 串行处理的局限 5

1.3 数据复杂度和冗余 6

1.4 协议升级困难 6

1.5 区块膨胀 6

1.6 低效的点对点通信 6

1.7 亟需突破的跨链通信 6

2. ÆIf的主要目标 **7**

2.1 针对商业应用的高度可定制的OS 7

2.2 跨链交互 7

2.3 性能提升 7

2.4 协议升级 7

2.5 联盟链模板 7

3. 实现ÆIf系统的主要途径 **8**

3.1 性能提升 8

3.2 资源隔离 8

3.3 治理结构 8

3.3.1 和代议民主的相似性 9

3.3.1 由委任节点行使权力 9

4. ÆIf系统 **10**

4.1 ÆIf架构 10

4.1.1 一链一合约 10

4.1.2 侧链动态索引 11

4.1.3 树形侧链延展 11

4.2 ÆIf主链 12

4.2.1 侧链索引系统 12

4.2.2 ÆIf代币系统 15

4.2.3 共识机制 15

4.2.4 DPoS 16

4.2.5 交易确认 18

4.3 ÆIf侧链 18

4.4 ÆIf的经济系统 19

4.5 ÆIf系统内置侧链 21
4.5.1 信息登记认证侧链 21
4.5.2 数字资产侧链 21
4.5.3 数字资产发售侧链 21
4.5.4 去中心化交易侧链 21
4.6 ÆIf跨链优化 22
5. ÆIf操作系统 23
5.1 最小化区块链系统的定义 23
5.2 ÆIf内核 23
5.2.1 内置最小区块链系统 23
5.2.2 统一账户系统 23
5.2.3 区块内并行处理交易 23
5.2.4 区块标记 25
5.2.5 智能合约集 26
5.2.6 智能合约更新 26
5.2.7 可定制共识机制 26
5.2.8 可定制区块头 26
5.3 ÆIf操作系统用户接口 27
5.3.1 智能合约执行 27
5.3.2 微服务 28
5.3.3 基于云 28
5.3.4 轻节点 28
5.3.5 可选模块 28
5.3.5.1 数据清洗机制 28
5.3.5.2 数据隧道 29
5.3.5.3 快速确认模型 29
5.3.5.4 代币模块 29
5.3.5.5 自定义 29
6.1 技术 30
6.2 商业应用 30
6.3 资本 31

1. 现有区块链系统

如今，区块链技术及其应用正处于飞速发展之中。许多行业都在尝试从传统的IT架构向区块链架构迁移。然而，当前的区块链技术还不足以支撑一个完备的系统来搭载多种应用。比特币作为区块链的最早设计更接近于一个应用而非系统。以太坊展示了一些操作系统的特征——开发者可以用智能合约编写应用，它还提供了Solidity和其他程序语言。不过，以太坊也有其他一些缺点，例如系统组件之间深度耦合、绝大多数组件不能定制、数据接口不够完善等等。
这些项目都缺少系统级的总体设计，而且也无法在跨行业的应用场景中发挥潜能，极大限制了区块链的商业运用。

1.1 通用区块链 vs 复杂业务场景

当前限制区块链支撑大规模商业应用的主要原因是它还不能满足纷繁复杂的业务场景需要。这些场景一般在处理和逻辑执行上都有不同的特点，从而需要不同的解决方案。因此一个“通用”的区块链系统就面临着平衡不同业务情景需求的难题。比如，票务系统需要高TPS（Transaction per Second，每秒交易量），而数字法律合规系统却更注重强调安全性和可靠性。
目前有两种通用的解决方案来满足这些需求：

1. 把区块链仅仅作为数据库来使用，不处理业务逻辑。这个方案旨在对任何业务场景保持兼容。许多类比特币的区块链底层设计就采用这种方案。他们把业务相关的数据和哈希写到一笔交易输出的“OP_RETURN[1]”。
2. 在一条链上记录不同的复杂智能合约。这些智能合约是为了服务不同业务场景下预先定义好的商业模型的。以太坊代表了这类区块链。由于所有的智能合约都是写在一条链上的，所以区块链会变得十分复杂，维护成本高昂，也缺少有效的智能合约执行架构。

1.2 串行处理的局限

随着区块链技术更广泛地被应用，其线性处理压力日趋面临超过其设计容量的风险，进而导致了现在的网络性能瓶颈。目前的区块链系统面临着亟需提高吞吐能力的多重考验，有时甚至必须付出牺牲交易效率的代价。例如，比特币手续费随着交易量的增加正在急剧升高，并且有大量的交易需要等待很久才能被确认；以太坊在代币发售时也时常经历大面积的拥堵。在传统的IT架构里，现代优化技术例如分库、分表或改用分布式架构等策略都被证明可以极大提高系统性能。
另一方面，并行任务处理的概念还没有被应用于提高区块链效率。当一个区块包含大量交易数据和复杂智能合约的时候，串行处理压力已逼近区块铸造和验证的效率极限。

1.3 数据复杂度和冗余

正如在1.1里所提到的，目前的状况是一个通用区块链系统需要被用来处理不同的业务情景。通用区块链系统的缺点在于过度复杂的智能合约和共识机制、缺少对于特定业务情景的定制解决方案、以及数据冗余。

---

[1] OP_RETURN是比特币的一个脚本指令，由于该指令能够返回任何数据并且比特币的交易会被记录在区块链中，因而被广泛用于数据存储。

1.4 协议升级困难

尽管区块链技术发展十分迅速，但它目前还是处于初级阶段。重大的改进和更新还会在未来陆续出现。这些更新对于区块链的进化和保持与时俱进是至关重要的。区块链生态系统里利益相关方的多样性导致在缺乏有效治理机制的情况下通常很难达成一致，这也造成了现在许多协议升级被搁置的情况。以比特币系统为例，近几年许多新特性的引入都经历了社区中长期的争执。

1.5 区块膨胀

一个区块链系统越成功，其维护成本也就越高。现在运行一个比特币全节点需要超过130G的硬盘空间，以太坊更是超过了180G，且这种情况远期也不会得到改善。随着更多的人接纳了区块链并且开始进行更多的交易，区块的存储空间会加速膨胀，维护成本也会日渐高昂。对此，我们必须采取行动来缓解这个恶性循环。

1.6 低效的点对点通信

现有的区块链主要是通过广播网络来进行通信的，且对于P2P网络的支持既低效又不安全。举例来说，如果某种数据只针对一个用户群，那么这些数据就应该只在有限的节点里传播，而不是被广播到所有节点。

1.7 亟需突破的跨链通信

现有区块链技术领域内已有一些在区块链系统间处理相关业务逻辑的尝试，然而数据的跨链交互一直是业界棘手的技术难题。现有的跨链通信包括中心化的实现方案和基于HTLC的方案。中心化的方案背离区块链的本质，而且面临授信困难、单点故障、单点性能瓶颈等问题，应用场景较为局限 。基于HTLC的方案则只能在处理资产交换等一些特定应用模式时发挥作用，同时，该方案对参与通信的两条链的协议和共识机制有严格的要求，实现方案复杂。无论哪一种实现方案，在区块链系统间协议的差异适配及数据标准交互格式定义这两个核心议题上都有待突破性的技术进展。

2. ÆIf的主要目标

2.1 针对商业应用的高度可定制的OS

我们对ÆIf的愿景是一个高效的、高度可定制的OS，我们期待它能成为区块链里的“Linux”。就以Linux为例，Linux内核和各种Linux版本共同组成了庞大且成功的Linux家族。Linux内核解决最基本的、关键的、和耗费开发者精力的部分，从而让其他开发者根据应用场景和客户需求来开发定制的系统。这使得Linux成为最受欢迎的服务器操作系统，并支撑着各行各业的发展。

同样的理念也被运用到了ÆIf的设计之中。首先，我们定义并实现ÆIf内核，它包含一个区块链系统需要的最基础的功能，也就是最小化区块链系统。其次，我们会开发一个“shell”作为和内核交互的最基础的接口。用户既可以使用完整的区块链OS，也可以通过该接口在内核基础上快速开发定制的OS。

2.2 跨链交互

ÆIf会和比特币、以太坊以及其他一些区块链系统产生交互。和这些主流链的跨链交互会通过消息实现。并且基于跨链交互还会形成一种天然的多级跨链结构从而可以共享数字资产、用户和信息。

2.3 性能提升

在传统的IT架构里，分布式架构是一种非常流行的用来解决系统瓶颈的架构。区块链系统也应该支持并行处理。例如，并行处理没有数据竞争的多个交易来提高交易效率。另外，当一条链变得太复杂而不能被有效处理之后，那么就应该被分成几条并行的链来分流。

高性能区块链的初始设计应该专注于具体的业务场景，而非将所有的智能合约都在同一条链上做普适处理。为了给业务需求提供极致的性能，该链必须能提供高效且能定制的数据结构、智能合约逻辑以及专门为特定业务目标设计的共识机制。

另外，ÆIf能够定义并触发系统快照的机制。到达特定的周期后，系统将当前的数据进行一次快照，然后丢弃掉快照前的详细交易数据。一个新的创世区块会包含所有的交易。这个想法已经被用在传统IT数据库系统里，来减轻系统的膨胀问题。

2.4 协议升级

在区块链创始的时候，我们必须清晰地定义好投票和更新机制。这样在未来更新共识机制的时候，就能避免围绕协议更新产生的分歧。

2.5 联盟链模板

许多商业机构都对联盟链有着浓厚的兴趣，想要利用区块链技术带来的技术优势。这些联盟链一般都与外界的系统与业务保持隔离，没有任何依赖关系。我们提供了一个类似于亚马逊云服

务“AMI市场 ”的模式 ，用户能基于联盟链模板，快速创建一条独立的链，独占地处理自己的业务。

3. 实现ÆIf系统的主要途径

3.1 性能提升

ÆIf的核心原则是使用成熟高效的技术来解决实际的技术问题。我们更关注如何提供一个可以稳定运行商业应用的可靠配置，而不是“优化”区块链的概念。

以下是一些如今已经被探索过的想法：
大部分的区块链分片（sharding）方案都是通过把一个单独的共识划分成许多子共识。基本上来说 ，就是作为整体的共识被分开，变成一些更容易被攻击的子共识群。人们也可以通过增加随机性来复杂化路由路径，但是这也限制了挖矿节点的专门化。

随着更多的矿池使用专门的记账系统取代PoW的挖矿节点，这些节点的数量急剧减少。这些矿池能保证挖矿效率和交易的即时广播、减缓区块链的分叉，也同时保证系统的稳定。借鉴IT行业的经验，矿池已经抛弃了官方的软件，转而通过负载均衡和并行运行智能合约来聚集算力，并且在全球部署节点来提高广播的效率。但是，矿池的效率还是受到矿池里用到的技术差别的限制，也受到每个节点等同的设计以及协议本身的限制。所以升级一个节点并不能带来整个网络的提升。

ÆIf的逻辑是这样的：
ÆIf里的节点根据不同的职能分类，使运行在集群上提供标准服务的节点开源，并通过DPoS来达成主链上的共识。被委托的挖矿节点能够在最大程度上保护侧链，并且还能分享主链的强共识。这个方法增加了每个节点的压力，但是效率会随着更多的侧链的加入而得到提高，因为被委托的挖矿节点能够在集群上运行。侧链之间是相互独立的，所以每多一条侧链都会增加整个系统的效率，每一条侧链的效率也因为并行处理而得到提高。

3.2 资源隔离

为保护智能合约免受不必要的干扰，维持其在区块链上的稳定运行，ÆIf抛弃了“一链治所有”的方案，自己设计了一个可以保证每个合约都正常运行的公链。

ÆIf的愿景是实现像AWS一样的计算平台，任何业务都不会被其他业务干扰，如期货市场的交易不会被黑色星期五的流量所干扰。然而，这种看起来不可能的干扰在区块链世界里却很常见。因此我们认为，正是区块链的初始设计妨碍了区块链技术在现实场景中的运用。

3.3 治理结构

由于历史原因，现有的区块链治理结构通常在初始时都没有被很好地定义，导致当有重大功能更新或者Bug导致系统错误时问题更加突出。例如，比特币在扩展性问题上停滞了两年，最终还是选择了分叉；以太坊社区和基金会之间关于Dao事故的分歧，导致了ETC的诞生。我们在用户开始使用ÆIf之前就澄清达成决议的方法：

我们认为ÆIf代币持有者拥有ÆIf未来的最大话语权，而且代币持有者的利益与ÆIf的命运息息相关，特别是那些持有长期锁定代币的投资人。

3.3.1 和代议民主的相似性

ÆIf的特点之一是指定一些特殊节点来执行特定的任务。在ÆIf里，重要的决定是由一种类似于代议民主制度的机制来做出的。被委派的节点必须拥有来自其他利益相关方的投票才能参与到ÆIf的治理里来。挖矿节点在一定程度上促进了ÆIf系统运行的健康，这些节点要负责记账、分发奖励并反馈给那些通过智能合约构建信任的利益相关方。

3.3.1 由委任节点行使权力

基金会通过提交代码然后委任挖矿节点审阅和投票来行使它的治理责任。过程如下：基金会成员提供开源代码并且提交新的功能。然后委任节点根据他们的需要整合特定的功能。如果一个功能被足够多的节点采用，它就会被整个系统接受。

4. ÆIf系统

4.1 ÆIf架构

ÆIf包含一条主链和多条附加在主链上的侧链（图4.1）。ÆIf和传统的单链系统最大的不同是它是一个“分支生态系统”，主链作为整个系统的主干索引多条侧链（甚至可能包含多级侧链）。

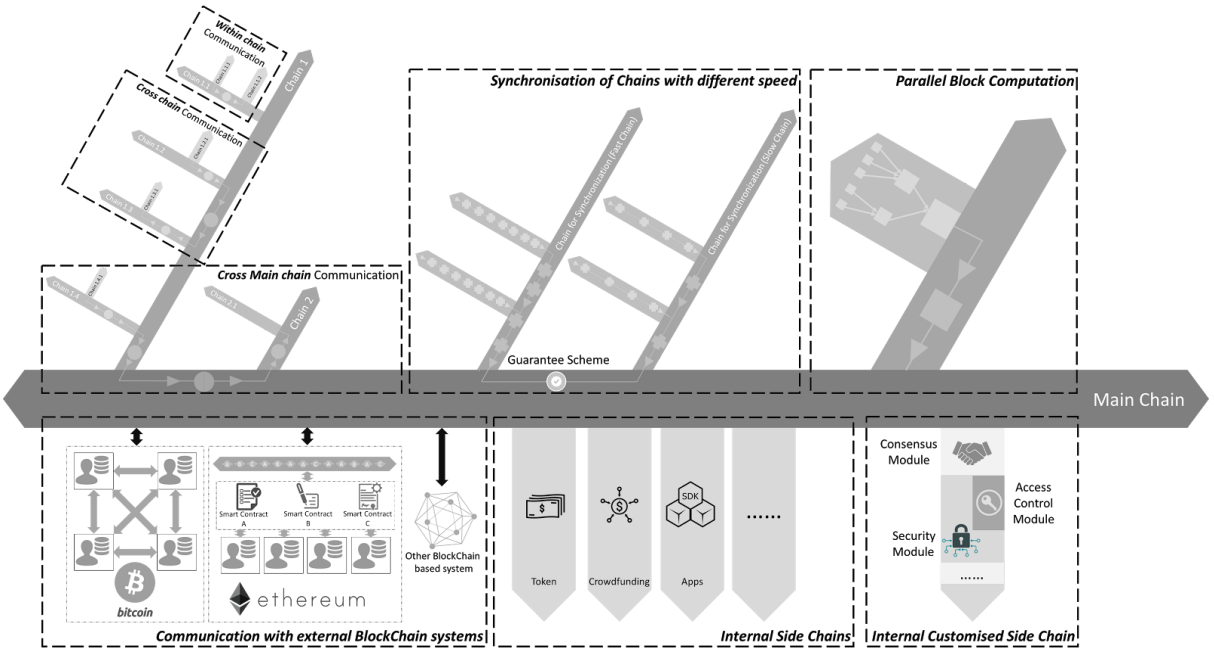
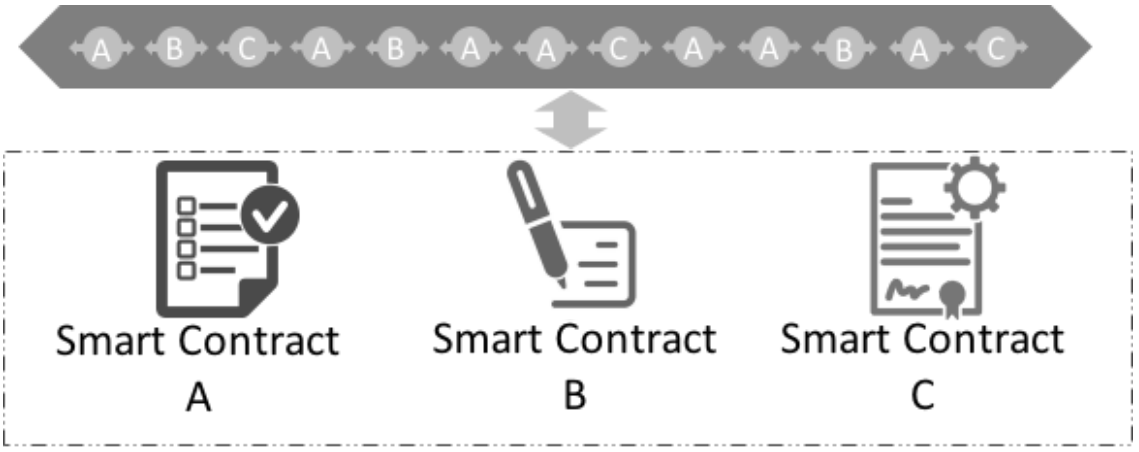


图4.1：ÆIf架构总览

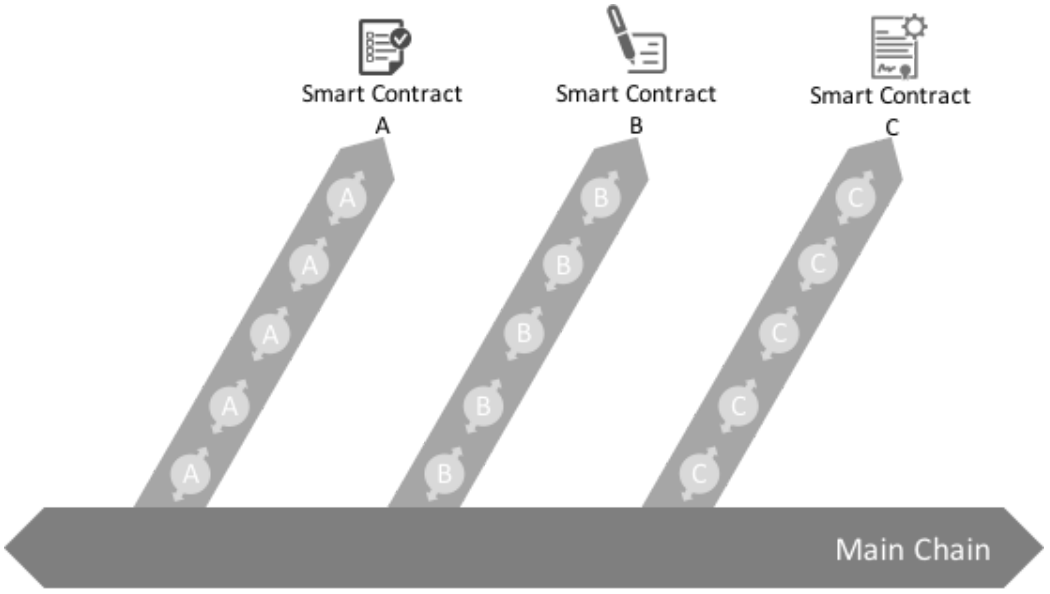
ÆIf可通过适配器和比特币、以太坊以及其他一些区块链系统对接，从而兼容已有的主流生态系统。ÆIf的侧链包含系统内置的侧链和其他基于ÆIf OS内核创建的侧链。主链通过动态索引实现与侧链的交互。

4.1.1 一链一合约

不同于传统的“所有合约一条链”，ÆIf提出“一链一合约”架构。如图4.2（b）所示，每条链都专门处理一种类型的交易，解决一种类型的业务问题。这就让整个架构和数据变得更简单，更加契合业务需求。通过向ÆIf增加侧链，ÆIf拓展了更多的功能，同时还能保持易于维护的架构。



(a) 多个合约一条链



(b) 一链一合约

图4.2：区块链的复杂数据结构

4.1.2 侧链动态索引

ÆIf是一个动态的系统，所有的侧链都依附于主链。主链包含了系统边界的索引（记录挂载的侧链）。链与链之间通过主链的Merkle Tree以及外部消息的输入验证来交互，而不是直接交互，这样就可以很容易的在ÆIf系统里添加或者删除侧链。

4.1.3 树形侧链延展

如图4.3所示，ÆIf定义了一个“主侧链架构”。理论上来说，任何侧链都可以与其下的一些子链连接，成为那个部分的“主链”。这造就了系统里的分级结构，使得ÆIf同时具有了横向与纵向延展的能力。这个想法和数据库架构里的分库和分表类似。每个分表能执行专门的功能，并且

当单表太大到难以管理时，它就会被进一步被分成多个表。在Ælf里，这对应的是侧链。

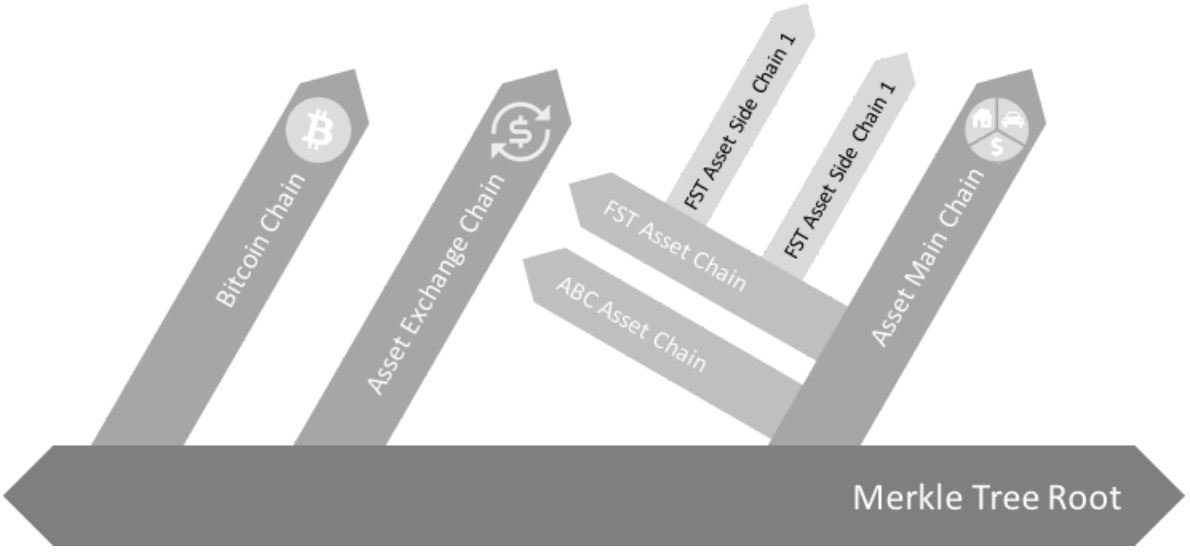


图4.3：多级侧链结构

4.2 Ælf主链

Ælf主链是基于Ælf OS 的区块链，它是整个系统的基石。主链包括侧链动态索引系统、代币系统、以及DPoS共识机制。

4.2.1 侧链索引系统

侧链索引系统连接Ælf生态系统里所有的链。Ælf索引两类链：

- 重要的外部链，用来扩充Ælf的边界，如比特币、以太坊；
- 在Ælf OS 下运行的内部侧链，使用Ælf代币为整个Ælf系统做出贡献。

侧链索引系统通过以下步骤运行：

- 主链节点从侧链读取信息并生成一个Merkle Tree；
- 新区块的区块头记录Merkle Tree Root，如图4.4所示，如果我们想要验证BTC第1000个区块的交易TX1，我们只需要通过主链的Merkle Tree Root，证明BTC第1000个块的Merkle Tree Root的存在，并基于这个（BTC第1000个块的）Merkle Tree Root和附加的消息证明TX1的存在。这个方法也同样被用于其他的链比如以太坊，只要区块是基于Merkle Tree的形式组织的。

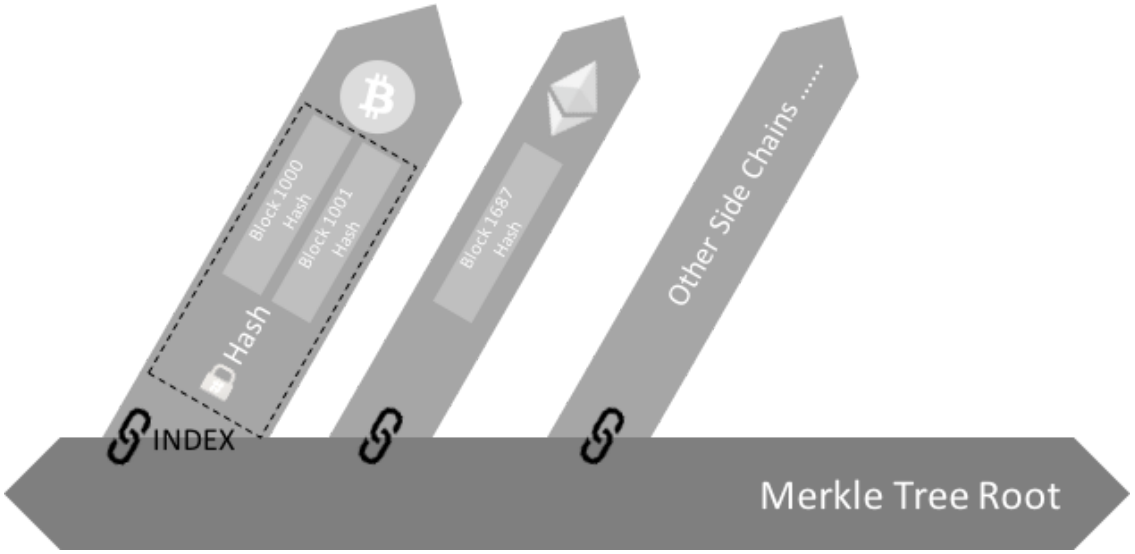


图4.4：侧链索引

为了提高验证效率，我们建议扩展Merkle Tree的结构，除了包含区块哈希还要包含图4.5里交易的Merkle Tree Root，以及图4.6里的状态的Merkle Tree Root。

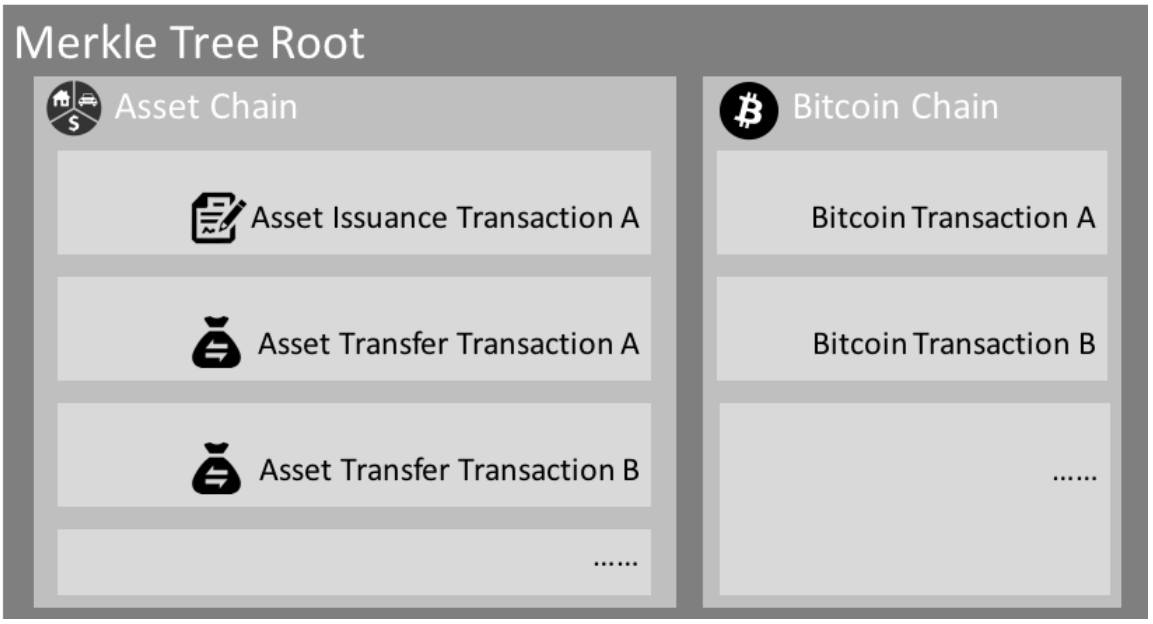


图4.5：交易索引

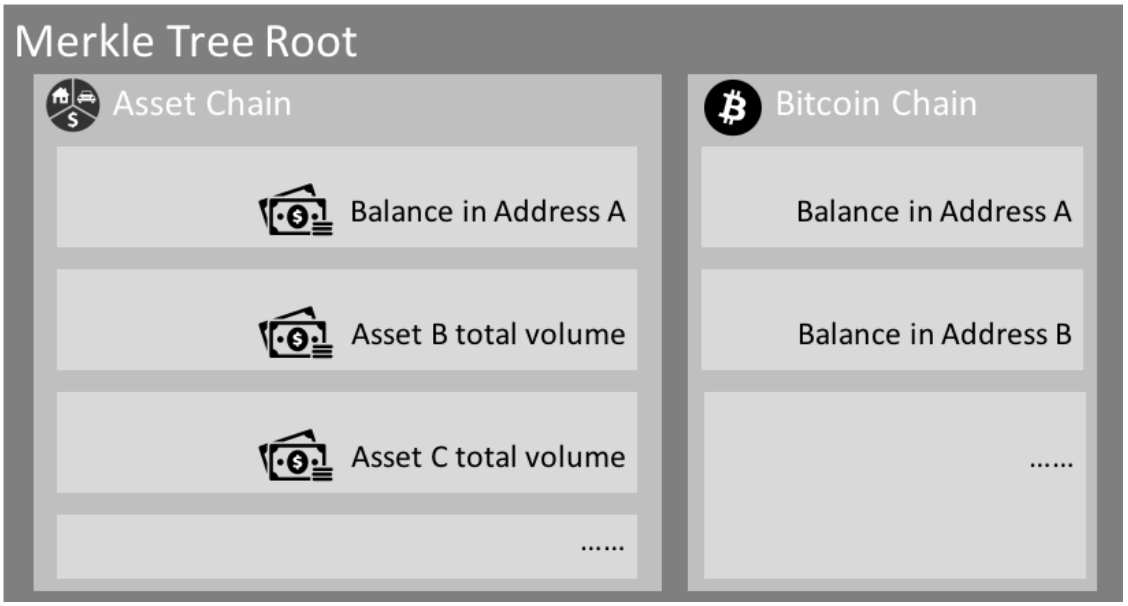


图4.6：状态索引

还有一个关键的问题是侧链被主链索引的时机。如果主链频繁地索引一条很可能分叉的侧链，就是在索引孤块上白费力气。所以我们为每一条链基于它自己的特点提出不同的索引策略，并且这些可以在系统里预先定义好。对类似比特币的系统进行索引可以在区块形成一分钟之后进行。因为数据证明区块形成一分钟之后基本可以被确认不是孤块。在ÆIf里，如果一条侧链选择与主链一起联合挖矿，则该侧链可被实时索引，因为主侧链被同样一群矿工维护。

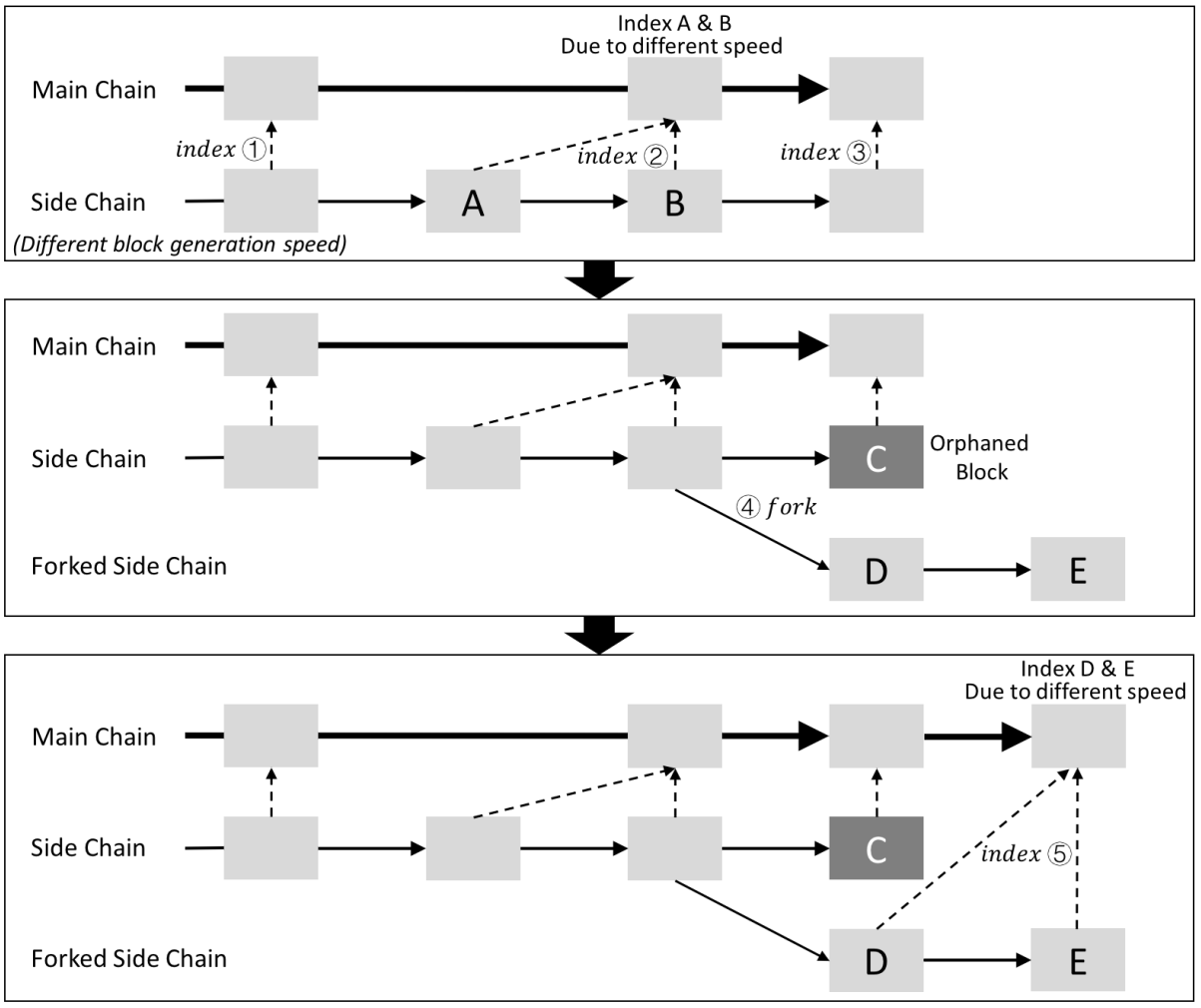


图4.7：索引时机

4.2.2 ÆIf代币系统

ÆIf代币激励了系统里的诚实行为。所有的ÆIf侧链接受ÆIf代币作为价值的存储媒介和转换方式。代币可以在接受ÆIf代币的侧链之间转移。

当一个侧链被主链索引的时候，它从主链获得一些锁定的代币。当侧链收到交易手续费时，它也会和主链的矿工一起分享。当主链发现索引一条侧链在经济上不能获利的时候，主链有权终止索引，或者说允许两条提供相同服务的侧链之间互相竞争。

4.2.3 共识机制

一个稳定有效的区块生成机制是ÆIf系统的基石。ÆIf系统的运行和维护比比特币和以太坊更复杂。这主要是因为ÆIf的区块生成需要主链记录侧链的信息，而且ÆIf也被设计成提供更复杂的基于云的企业级服务。另外，矿工需要并行地在多条链中更新信息。主链会采用DPoS来保证高频、稳定地生成区块，以提升用户体验。

4.2.4 DPoS

ÆIf委托2N+1个挖矿节点。N从8开始，并且每年加1。这些节点确保ÆIf里的所有共识机制的执行。

这些委托节点的功能有交易转发、交易确认、打包区块和数据传输。因为ÆIf采用多级侧链架构，挖矿节点也必须为一些侧链挖矿。

2N+2个节点每周会进行一次随机顺序计算。随机过程如下：

ÆIf在我们称为“轮”的执行单元上按时间运行（图4.8和4.9里的横箭头）。每一轮里，一个挖矿节点每次产生一个区块，同时该节点会在每轮结束时产生另外一笔交易（图4.9里的竖箭头）。

每一个挖矿节点(node)在特定的一轮(t)里有3个主要特性：(1)私钥，$in_{node(t)}$，这是一个从挖矿节点输入的值，并且在第t轮被挖矿节点私密地保存起来。 它会在第t轮的所有区块都产生后被公布；(2)公钥，$out_{node(t)}$，它是私钥的哈希值。ÆIf网络里的每个节点都能在任何时候查询这个值；(3)签名，$sig_{node(t)}$，这是由挖矿节点在第一轮自己产生的。第一轮之后，只能在前一轮结束之后重新计算该值。它被用来作为本轮该节点的签名，而且它也和公钥一样任何时候都对公众开放。详情请看图4.9。

在DPoS里有两个主要的过程 ：(1)预验证；(2)每轮里的按序计算

预验证（图4.8）：（t+1）轮里一个节点开始产生区块之前，它在t轮里的状态需要先被验证。在t+1轮里，$in_{node(t)}$ 已经被公布了，$out_{node(t)}$ 也能被随时查询。所以要验证节点在t轮的状态，其他的节点可以验证 $hash(in_{node(t)}) == out_{node(t)}$ 。

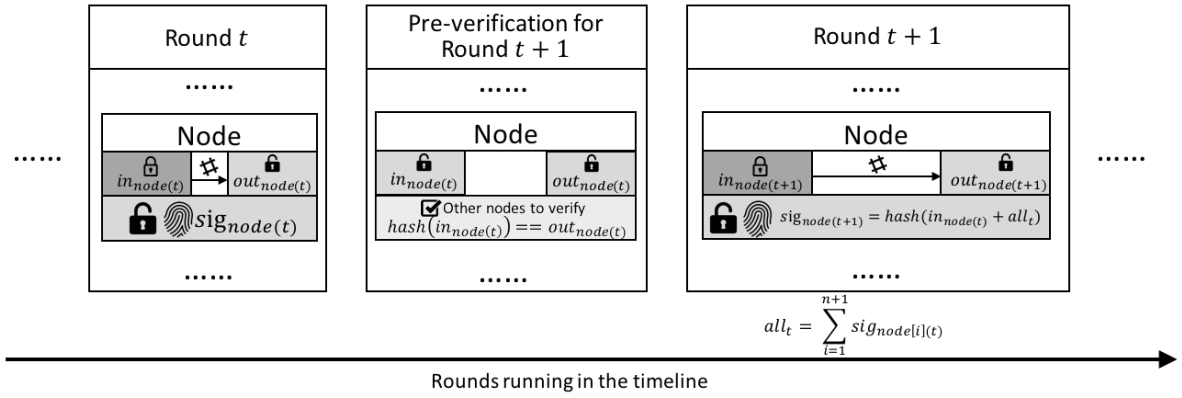


图4.8：预验证

顺序计算（图4.9）：在图4.9里，我们使用了4个挖矿节点来举例说明我们的顺序计算策略。在每一轮里，N个挖矿节点产生N+1个区块。在第一轮里（图4.9里的第1轮），每个节点的签名和区块产生顺序是完全随机的。在第二轮里，区块生成也是随机的。但是，从第二轮开始，签名是由

$sig_{node(t+1)} = hash(in_{node(t)} + all_t$ 其中

$$all_t = \sum_{i=1}^{n+1} sig_{node[i](t)}$$

计算的。这里，$node[i](t)$ 代表这个节点处理t轮里的第i个交易。
从第3轮开始，每一轮里的顺序是从前一轮的顺序和签名生成的。在第t+1轮里，我们按照顺序遍历t轮里的签名。t+1轮里的顺序是由

$$sig_{node(t)} mod(N) = \begin{cases} 0, & first\ place \\ 1, & second\ place \\ 2, & third\ place \\ \cdots & \\ n-1, & n^{th}\ place \end{cases}$$

计算得到的。

考虑冲突的情况，即结果指向非空的位置，我们就把节点指向下一个空缺的位置。如果该节点和第n个位置冲突了，我们就从从头开始寻找可用的地址。

用来执行额外的一笔交易的节点是从前一轮的第一个节点的签名计算得到的。

$$sig_{node[0](t)} mod(N) = \begin{cases} 0, & A \\ 1, & B \\ 2, & C \\ \cdots & \end{cases}$$

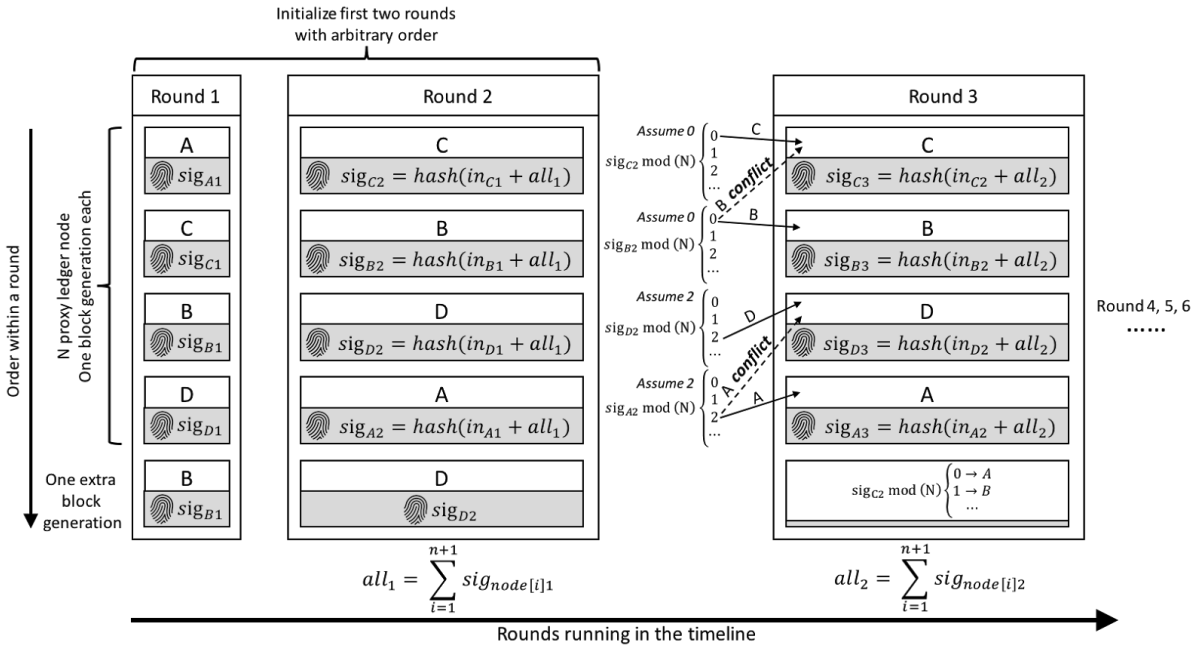


图4.9：前3轮计算顺序细节

$sig_{node[0](t)}$ 是由以下决定的：(1)前一个(t-1)轮的所有签名；(2)它自己在(t-1)轮的in值；(3)哪一个节点生成多余的区块。所以它只能在(t-1)轮结束后才能得到。另外，因为它需要前一轮所有

的签名，而且in值也是有每个节点单独输入的，所以不可能控制这个顺序。总的来说，我们创造了一个依靠外界输入的随机系统。基于每一轮里任何一个节点都不知道其他所有节点的输入的假设，可以认定没有人能控制排序。

如果一个节点在t轮不能产生区块，它也不能在这轮输入in值。这种情况下，前一轮的in值将会被使用。既然所有的挖矿节点是被选举出来的可靠节点，这种情况不太会常发生。即使这种情况发生了，上面提到的策略也足够很好地应付这种情况。

每一个节点只有特定的T秒来处理交易。在当前网络环境下，T=4是合理的值，意味着每个节点只有4秒来处理交易以及将结果提交到网络上。任何没能在4秒内提交结果的委托节点，都被认为放弃了这个区块。如果一个节点连续两次失败，就会给这个节点一个W小时的窗口期（W= 2^N ,N是失败的次数）。

在系统设计里，ÆIf定义在一段时间里只有一个节点产生区块。所以在挖矿节点保持连接良好的情况下不太可能产生分叉。如果由于网络问题产生许多网络孤岛，系统会采用最长的链，因为它最有可能来自于有最多挖矿节点的孤节点群。如果一个恶意的节点同时在两条分叉的链上挖矿来攻击网络，这个节点会被投票踢出整个网络。

DPoS挖矿节点是被类似于代议民主制度的方式选举出来的。被选举出来的节点决定如何分配收益给相关利益人。在之后的章节里我们会更深入地讨论这个话题。

4.2.5 交易确认

与现有区块链系统相比，ÆIf有更快更稳定的交易确认。不同于PoW，DPoS不需要反复生成随机数尝试打包。所以节点打包区块的时间是稳定的，而且可以被控制在T(4 秒)内。

ÆIf建议：对于一般的交易，只需要5个区块就可以快速确认；15个区块确认来针对大额交易。所以，一般的交易会在20秒内确认，大额交易会在60秒之内。

值得注意的是，这只是保守建议。比特币建议6个区块的确认，但是许多用户只使用一到两个区块就确认了。对于进阶用户，通过观察和统计过去一段时间的区块接收情况，根据记账节点平均可用时间、网络平均可用时间等参数，可以通过概率模型选择最适合自己业务的确认数。

4.3 ÆIf侧链

那些被ÆIf主链索引过的链都是侧链。如之前所述，建议每条侧链都被设计成只处理一种特殊类型的交易。

当通过ÆIf OS创建一个新的侧链的时候，我们建议和主链联合挖矿，并且建立自己的共识机制。要为ÆIf 生态系统出力，侧链应该预留一些ÆIf代币，并且和主链分享一部分手续费。

当一个侧链需要验证从另一条侧链来的信息的时候，它必须要包含ÆIf主链的区块头信息。验证是通过主链提供的Merkle Tree Root完成的。

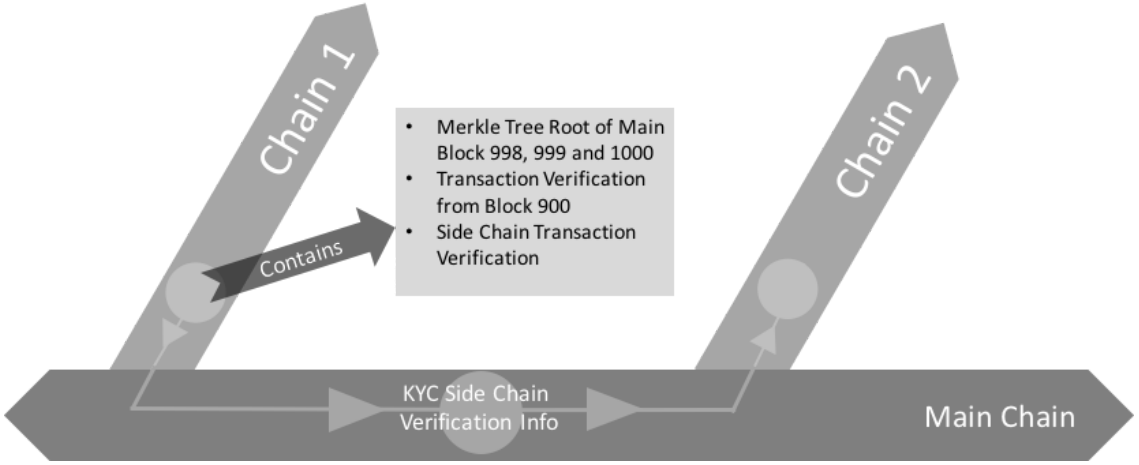


图4.10：两条侧链之间的交互机制

从例如比特币这样的UTXO系统是很难取得像某个地址的余额这样的状态信息的。跨链通信可以通过区块链适配器来实现。适配器会创建一个兼容的区块头，其中包含比特币的Merkle Tree。ÆIf利用这样的适配器来建立完全兼容的比特币侧链，从而能和比特币配合并调用它的资产。

4.4 ÆIf的经济系统

一个良性的经济系统是ÆIf生态可持续发展的基础。

对于PoS和DPoS来说，任何持币人都可以在短时间内卖掉他们的代币并且退出整个生态系统（PoS有一个的锁定期）。PoS和DPoS都面临的一个问题，就是交易所持有大量代币，因此他们几乎能无成本地获得利息。

对PoW来说，矿工退出前要考虑的更多。受到外界因素，例如电费、矿机折旧、场地以及人力资源的限制，矿工的退出更加困难。

ÆIf会在主链上使用DPoS机制来激励持币大户去维护一个稳定的系统，同时在侧链上使用PoW来产生ÆIf代币。在ÆIf系统里，每个链上的共识机制都可以根据实际需求定制。

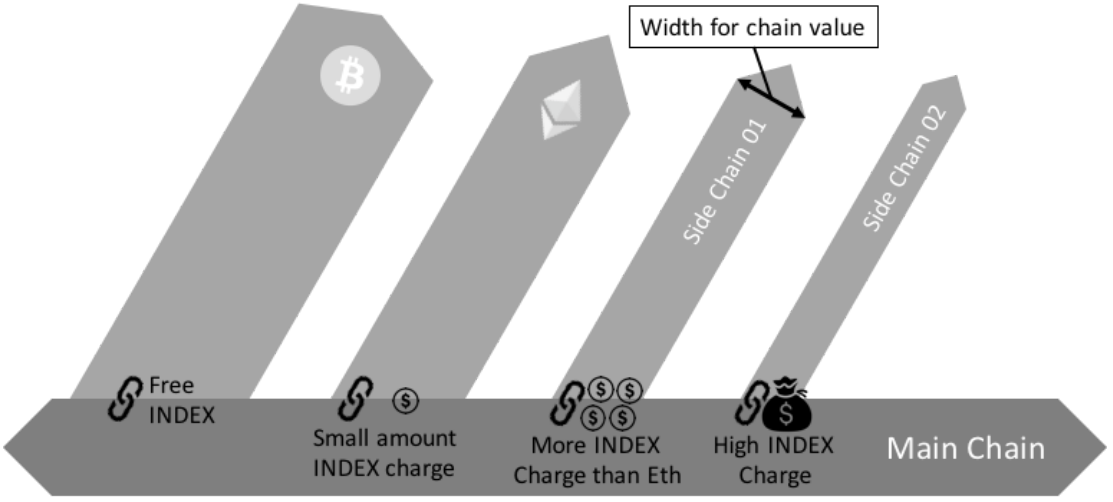


图4.11： 索引侧链的收费机制

在一条侧链被Ælf生态系统接纳后，它需要给主链支付一定的费用才能被索引。Ælf采用动态收费策略来反映每条侧链对Ælf生态系统的贡献度。例如，Ælf会向贡献多的链收取更少的手续费（比如索引比特币不会收取任何费用，因为它的流行度和其上的资产。但是对系统没什么贡献还从其他链获取资源的侧链会被收取高额手续费）。

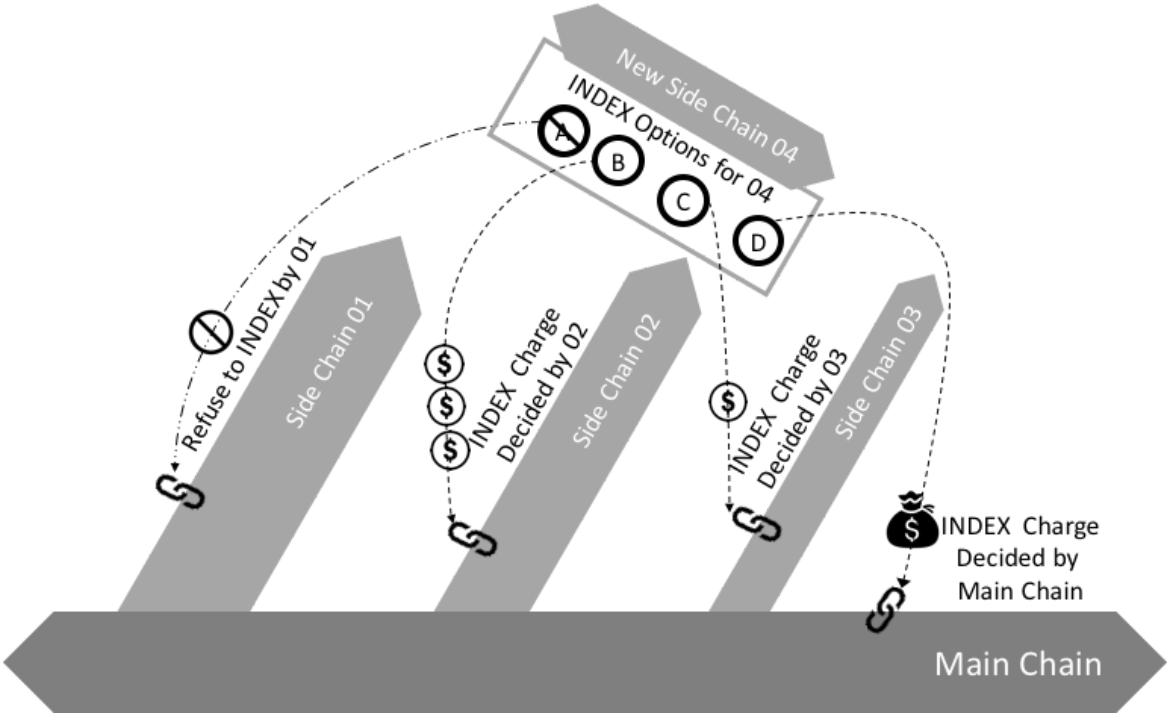


图 4.12： 子链索引

每条侧链的生态系统投票决定它独立子链的索引策略。它自己的策略包含但不限于商业范畴（例如，一个保险链只会包含同处于保险行业里的子链）或子链的收费策略。任何链也可以决定不包含哪个子链，或者主动邀请一个链成为子链来丰富它自己的生态系统。在Ælf生态系统里面，任何链都可以申请成为另外一条或者几条链的子链。

4.5 ÆIf系统内置侧链

ÆIf 节点的拓扑结构包含主链节点之间的内联P2P网络、轻节点以及侧链节点。非矿工节点通常是轻节点，账本节点是全节点。侧链节点根据它与主链的索引关系按ÆIf拓扑结构分布。侧链会在基金会的指引下开发。我们相信打造这样的一个系统是必要的。ÆIf不是要自己造侧链，而是提供一个开发侧链的模板和基础设施，让侧链之间能够相互通信。

比如，有一个内容网络，用户可以在里面使用这个网络的代币购买内容。当一个去中心化的“twitter”加入了ÆIf，ÆIf会帮助用户通过这个网络分享内容，并且在一个去中心化的交易所交换”twitter”代币和这个内容分发网络的代币。

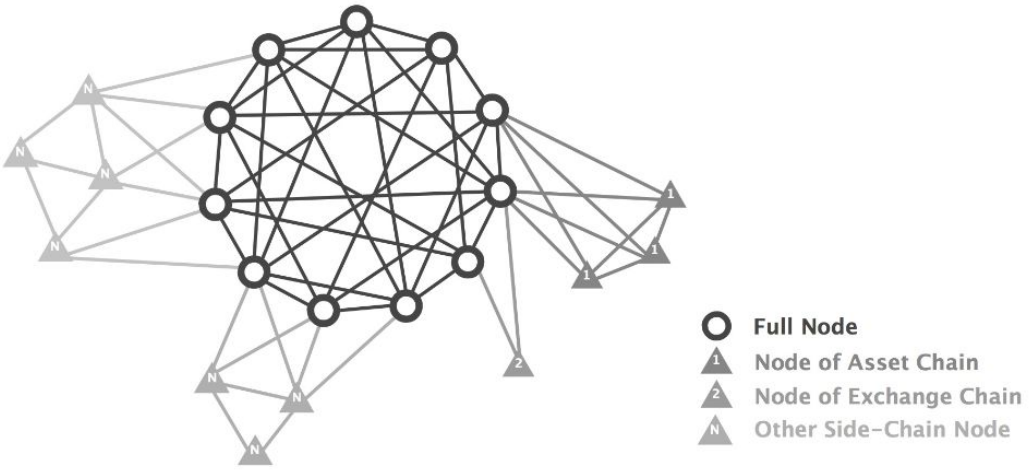


图4.13： ÆIf节点拓扑结构

4.5.1 信息登记认证侧链

登记认证侧链是线上线下行业亟需的价值链。在互联网行业，电商、外卖、网约车等线上线下应用交互已经普及。供应链金融、物流、存证、征信等拥有巨大潜力的业务等待我们开发，将来，他们海量的信息资产都能转移到侧链上来。

4.5.2 数字资产侧链

该链的基本功能是存储数字资产并保存所有权信息。

4.5.3 数字资产发售侧链

这个侧链的基本功能就是ICO(代币首发）。一旦分发完成，资产就会被转移到数字资产侧链。这么做的好处就是一般的交易不会受到大规模代币发售的影响。

4.5.4 去中心化交易侧链

去中心化交易侧链可以作为一个去中心化交易所。它实现了 KYC认证、资产充值提现、挂单、撤单以及成交撮合的逻辑。

4.6 ÆIf跨链优化

跨链交易需要进行优化才能匹配不同链之间的区块生成速度。我们设计了两种机制来解决这个问题。首先是层级侧链机制。我们根据链的区块生成速度把链划分到不同的层，然后为每一层提供一个专有的适配链或者适配模块来带动同层的跨链交易。其次，跨层担保机制。对于不同层的跨链交易，主链为较慢的链作担保。不过这只是一种可选的机制。这两个机制是加强ÆIf跨链交易速度的有效解决方案。

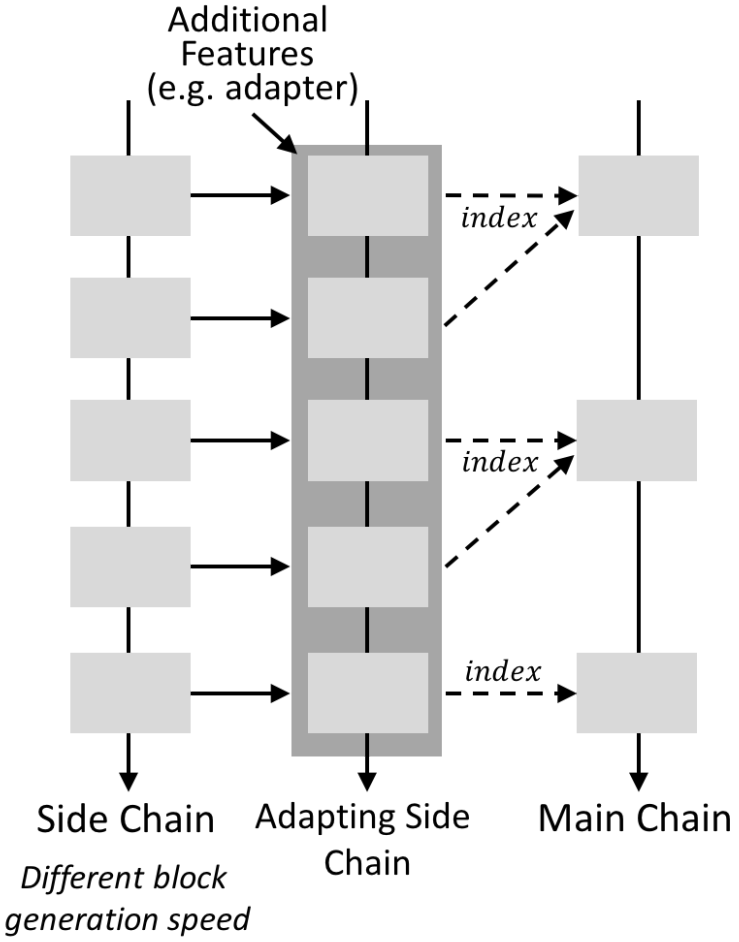


图4.12 ÆIf节点拓扑

5. ÆIf操作系统

5.1 最小化区块链系统的定义

ÆIf系统创建高度专业且有效的链结构来处理各种各样的商业情景。它还使用“分链”来解决需求增长带来的性能问题。为了进一步加强它的商业潜力，给开发者和社区划定系统最基本的模块和组件是很有必要的。下面这一章节将讨论最小可行区块链系统以及ÆIf操作系统是如何作为系统的基石以达到高可定制型和高性能的。

区块：区块是用来记录系统状态的。从上一个区块到当前区块的变化是定义在当前区块的交易里的。

交易：交易逻辑是被智能合约定义的。智能合约本质上就是一种协议。对于同样的输入，它总是给出同样的输出。

账户：账户使用来区分数据存储的边界的。它包含公钥私钥系统。

P2P网络通信：节点间的数据传输是通过底下的P2P网络实现的。

共识机制：共识机制定义了更新区块链状态的规则和权力。

5.2 ÆIf内核

5.2.1 内置最小区块链系统

这些是运行在ÆIf内核里最基本的区块链组件。他们通过相关的接口连接来定义智能合约、共识机制、以及区块链头部可以定制的部分。

5.2.2 统一账户系统

比特币系统把公钥私钥带到了账户的概念里。支付脚本的哈希（Pay to Script Hash）把交易权力给了智能合约。以太坊定义了外部拥有的账户和合约账户。ÆIf内核把两种账户都定义成合约。

5.2.3 区块内并行处理交易

ÆIf分析交易的静态状态并评估每个交易的数据影响域 。如图5.1所示，多个不产生读写冲突的交易可以被并行处理，不影响任何一笔交易的输出。在区块产生的过程中，节点根据交易的互斥锁把交易分配给不同的组。同一组里的交易会被串行处理，所有的组会被同时处理。

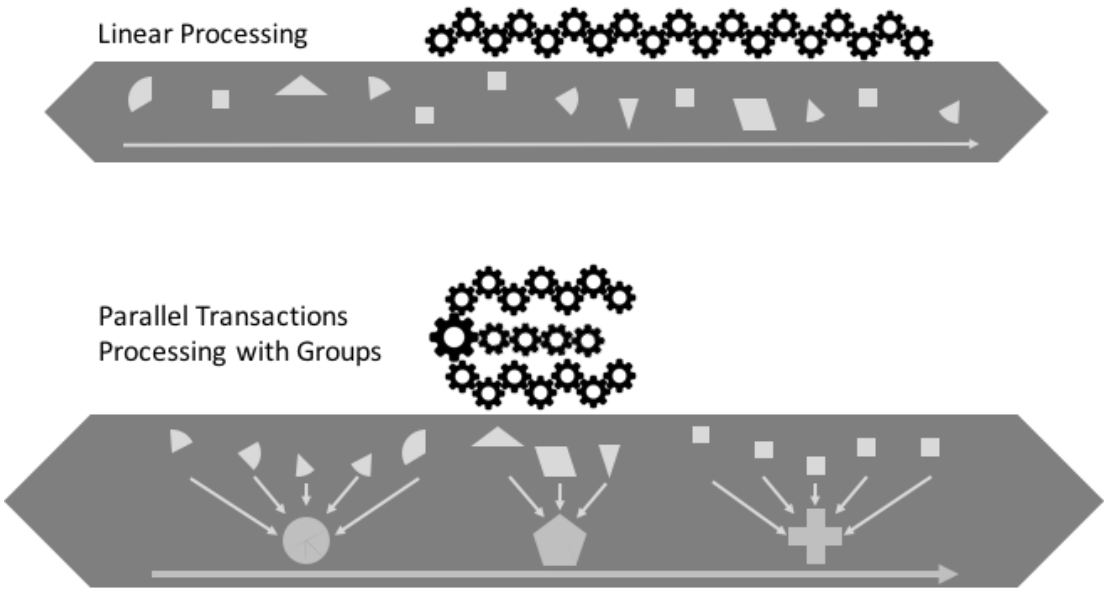


图5.1：区块内并行处理交易

有一些特殊的交易是无法被并行处理的，因为它们的数据影响域会在处理其他交易的时候变化。在这种情况下，节点会优先处理能被并行处理的交易。只要有足够的交易费，这些位于非并行组的特殊交易会被串行处理。否则节点可以拒绝处理这些交易。需要指出的是，当一个不良的节点接受耗时且不能被并行处理的节点的时候，其他节点拒绝该区块的概率会增加。

阿姆达尔定律是计算机界的一个经验法则。是以计算机科学家吉恩·阿姆达尔来命名的。它给出了使用并行处理理论上能带来的效率提高。

考虑运行在单核处理器上的一个程序。在执行时间上来说，“f”是并行计算部分执行时间所占比例，所以（1-f）是串行处理部分所占比例。如果有“m”个处理器并行运行，那理论上的优化空间计算如下：

$$SpeedUp_{Amdahl} = \frac{1}{(1-f) + \frac{f}{m}}$$

我们可以推导出两个重要结论：

1. 当f最小的时候几乎没什么提升
2. 随着m增大到极限，加速被限制在1/(1-f)

阿姆达尔定律是一个固定大小的模型，意味着它可以解决固定大小的带有固定并行处理比例的问题。

大多数的区块链交易是不相关的。从阿姆达尔定律出发，数据处理可以被大大加速。但是，绝大多数现有的区块链系统采用串行处理，并且所有节点都进行相同的计算。这既浪费资源又影响了交易速度。拿EVM举例，它不仅是串行处理交易的，还要求Gas费，所以它的性能十分低下。

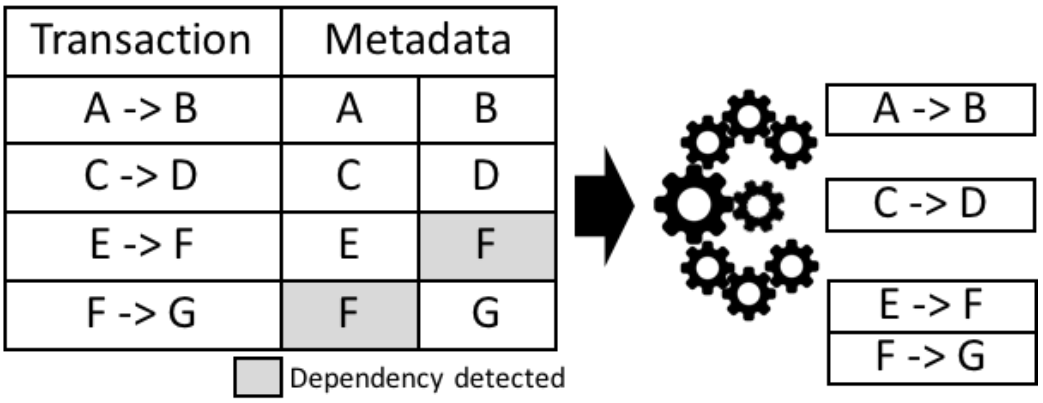


图5.2

为了解决区块链的问题，低速的交易绝对不是一个选择。ÆIf试图通过并行计算构建一个拥有高TPS的区块链系统。这个方案的关键在于分离交易数据和计算依赖来解决数据风险。我们可以引用Intel 微处理器的架构，通过驻留站分离电路依赖以及其他一些诸如寄存器重命名的技术来处理在RAW、WAW和WAR里经常发生的大数据风险，并行执行ALU（算术逻辑单元）。

ÆIf并行执行调度器（GPES）采用了相似的方法。在定期的内部测试里，ÆIf分离在区块链内存池里的计算依赖，数据依赖。GPES也有一系列的预处理，即计算时间跨度预测，预索引能够被并行处理的代码块，初始化管道以及在多个维度上执行并行计算。

这一系列的索引原语能被用来解决更复杂的并行逻辑问题。

ÆIf的流水线是另一个重要的能提速的方法。它被广泛应用于CPU、元函数（映射、聚合、包含）处理。这一系列非图灵完备语言可以完美处理数据流（或者简单的说交易流）。并行处理函数和上下文无关/不可变计算能够完全利用处理器和节点。

总的来说，并行处理是一个综合的策略。

5.2.4 区块标记

一个处于广播和确认之间的有效交易被认为是“未确认的”状态。通常情况下，交易会被迅速的打包和确认。但是，也有一些情况交易会被长期搁置，例如，比特币网络拥堵的时候，或者当大部分矿工对交易费不满意的时候。当一笔交易长时间既没有被确认也不能撤销的时候，我们认为它处于“混乱”状态。

ÆIf要求每一笔交易在广播时均带有一个“区块标记”，也就是当交易发生时最近的一个区块头哈希。然后矿工只需要处理带有最近的64个区块头哈希的交易。如果一笔交易在64个出块之后还没有被确认，那么这个交易就会被认为是过期了。换句话说，如果一笔交易没有在5分钟内被确认，那么代币持有人就可以重新构建这个交易。

另一个标记交易的作用是有效的防止区块链分叉。对于一个节点，如果一笔交易的区块标记包含在最近的64个区块中，则该标记是有效的，反之则是一个无效的区块标记。对于工作在分叉链上的记账节点，它接收到的大部分主链交易均带有无效的区块标记，因此不会被打包。对

于交易节点，如果接受到大量具有无效区块标记的交易，则意味着此时区块链可能发生了分叉。这时，节点就可以暂停交易来避免风险。

5.2.5 智能合约集

ÆIf链合约有一个在创世块里定义的智能合约集合。这个集合命名为创世智能合约集合，以向中本聪致敬。智能合约集的本质就是一个定义主要函数、链的共识机制以及合约集更新机制的类。

5.2.6 智能合约更新

ÆIf的函数是由智能合约集来定义的。因此更新集合会影响到这个链的功能。集合的更新机制是由前一个集合来定义的。例如，我们定义如果最近的第100个区块里80%的投票通过一个新的智能合约集，然后被接下来的2000个区块确认，新的集合就会取代原来的。不更新合约集的节点就会被停止工作。

5.2.7 可定制共识机制

对于特定的业务情景，共识机制对参与者的决定有重大影响。在具有很高信任等级的私有链环境中，PBFT是一个流行的共识机制。它通过预分配矿工来达到很高的效率。在一个低信用等级的环境，区块链的稳定性是由例如PoW、PoS和DPoS这样的共识机制来维护的。

ÆIf把共识机制定义成智能合约集的一部分，并且可以根据业务情景实现任何种类的共识机制。我们使用比特币和点点币来举例说明如何选择共识机制。

比特币使用的PoW只使用没有任何输入的区块头来验证。而点点币使用的PoS需要区块内的PoS交易数据，以及除区块头之外的自我验证。我们建议未来的用户优先考虑只需要区块头的共识机制，这样来达到及时验证。另外，对于特定场景，应该实现定制的共识机制。

5.2.8 可定制区块头

为了弘扬共识机制应该只使用区块头信息的建议，我们引入了可定制区块头。点点币的区块头不包含验证区块的合法性的信息，所以PoS区块不能自己验证区块的合法性。ÆIf内核允许在创建链时定制区块头结构。通过在区块头中添加Merkle树证明的相关信息(TxID + N + Value)，可以只凭借区块头完成对PoS区块的验证。

5.3 ÆIf操作系统用户接口

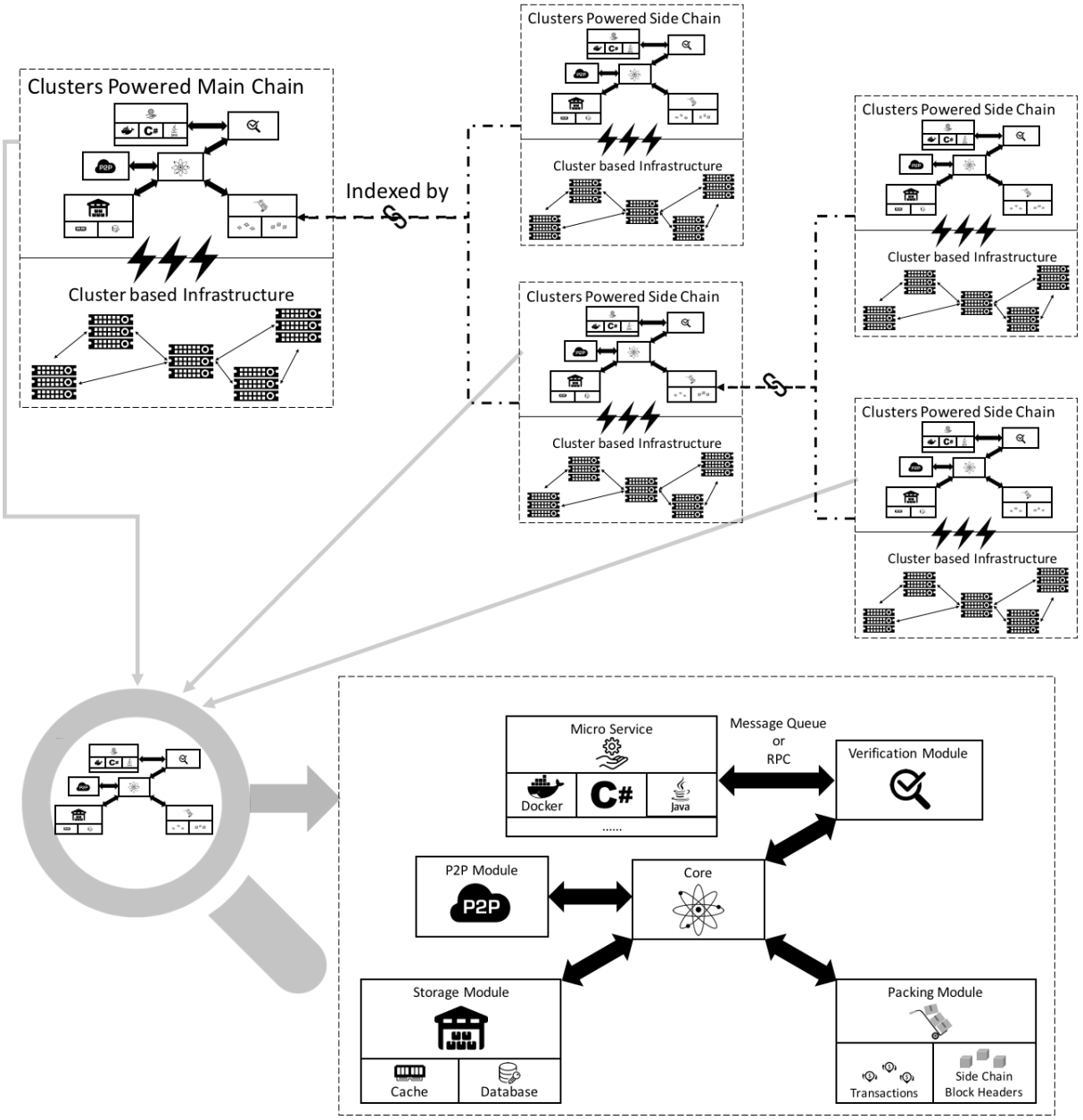


图5.3： ÆIf操作系统接口

5.3.1 智能合约执行

ÆIf操作系统把智能合约作为协议来定义。它可以以任何服务实现的形式来执行。

ÆIf操作系统倾向于Docker并且还支持原生程序语言例如Java、C#、Go、Javascript、Lua。

对于Docker，ÆIf在智能合约实现期间提供内部RPC服务来批准读取变量和用户账户的权限。对于原生程序语言，ÆIf提供相对应的SDK来执行函数。

5.3.2 微服务

在ÆIf里智能合约被定义成微服务。这使得智能合约与具体的程序语言分离开来。共识机制也基本成为定义在智能合约里的一个服务。

5.3.3 基于云

通过微服务的方法，ÆIf内核把并行计算扩展到云上，从而实现了基于云的合约执行。

ÆIf内核定义了数据结构和标准，因此热数据存储在内存里。通过使用成熟的分布式的数据库服务，可以有效地提高系统的IO性能。

5.3.4 轻节点

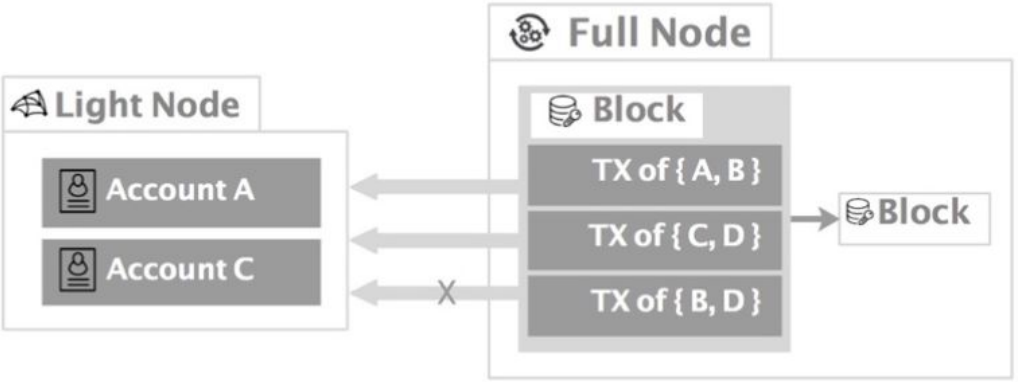


图5.4： 轻节点数据结构图示

通过定制和内部Merkle Tree验证机制，ÆIf里的每个节点只处理系统里相关的信息。这使得节点变得更轻并且极大提高了桌面和移动客户端的兼容性。

5.3.5 可选模块

5.3.5.1 数据清洗机制

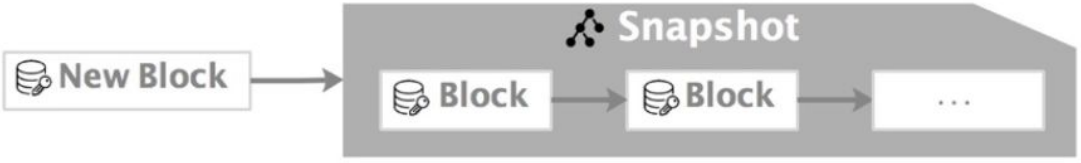


图5.5： 数据清洗图示

ÆIf系统采用快照机制复位区块信息，把原始的数据加到新的创世区块里。但是ÆIf系统不会依赖于历史数据，而是只专注于处理新的数据。人类历史丢失了许多细节，但这并不影响人们做出当下的决定。类似的，如果数据太大而难以记录，ÆIf系统要能够丢弃一些历史数据。

5.3.5.2 数据隧道

数据隧道是执行P2P传输的一个机制。这些数据不会被存储在区块里。数据隧道只能被应用于加密的P2P传输。比如，如果A从B那儿购买数据，B把数据传输给A的同时A把资产传输给B，都通过数据隧道。这个设计是为了让两个节点之间能够直接传输数据。在现有的区块链系统里，唯一的策略就是广播交易并且所有的节点都得处理这个交易。这是资源的浪费，而且也会限制处理的交易量。

数据隧道可以通过插件协议来实现。但是这需要所有节点的同意。如果这行不通的话，事情会变得很棘手（例如当IE不支持Chrome里的一些功能的时候开发者经常遇到麻烦）。有了这个协议，ÆIf会支持更多的应用，比如，一个数据购买合约（参考下文）。

5.3.5.3 快速确认模型

ÆIf在接受者被发送者授权的情况下允许快速交易确认。这个授权只在被委派的地址之间的特定时间内针对特定类型的交易有效。例如A在资产转移的时候希望和B发起一个快速确认模型。A需要预留一定量的资产来发起交易，然后指定B是另一方。在实际的交易过程种，A会通过数据隧道发送签名过的交易给B。B接受到这个交易的时候会立刻确认。相关的资产会在B使用它的地址签名这个交易之后被转移。A会收到剩下的资产。数据隧道会在交易完成后关闭。

5.3.5.4 代币模块

代币模块定义代币所有的逻辑和算法。它专门服务于例如支付这样的情景来做资源分配，或者是维护ÆIf稳定的奖励。

在大多数的公链里，代币机制是不可或缺的。它是用来激励整个网络健康发展并且决定不同角色贡献的。所以ÆIf设计了代币模块，每个ÆIf承认的侧链都被允许接受ÆIf代币。

5.3.5.5 自定义

ÆIf通过重新定义每个模块里的参数来让开发者快速自定义系统，并且为了实现ÆIf操作系统的侧链，ÆIf遵照“一链一个特定业务场景”的原则，创建了高度抽象和模块化的架构。对于企业用户和创业者，这加速了实现他们自己商业想法的过程。对于高级的用户，它允许高度定制他们自己的链，并且还下放了对区块链的所有操作。

6. ÆIf生态系统建设

任何新的技术没有商业应用以及可持续的生态系统都不会成功。ÆIf提出了一份有商业应用的技术蓝图，贯穿它的整个设计。建立ÆIf生态系统是至关重要的，包括内部和外部的资源。我们会通过同时追求三个维度的发展：技术、商业和资本来达成我们的目标。

6.1 技术

前一个章节描述了ÆIf的主要技术特点。ÆIf团队有多年区块链开发经验，特别是参与了一些商业导向的企业级项目。我们提出的ÆIf技术解决方案意在解决区块链商业化最紧迫的障碍，例如可扩展性、安全性、可定制性以及可互用性。它提供了一个高效的基础设施来采用新的协议和支持未来各种各样的商业情景。

6.2 商业应用

ÆIf试图最终成为新的“因特网基础设施”来支持下一代的“数字商业”。团队和顾问在过去已经指导了许多区块链项目，我们认为一些行业会成为ÆIf 上的“早期采用者”和“区块链之星”。

图6.1 ÆIf商业应用图示

1)金融服务
区块链吸引了金融服务业的大量注意，因为它大量减少了中间人并且保证了交易的安全。极有可能ÆIf上会有许多链专门为金融服务而开发，例如跨境支付、贸易金融、供应链等等。并行计算的特点使得它有能力处理全球化的商业交易，内部链通信又使得十分适合资产注册、账户管理、实时交易。

2)保险

保险是另一个被区块链改变的十分吸引人的领域。一个针对保险的专门的ÆIf侧链会整合各种各样的DAPPs，改变从用户身份，到保险合约执行，再到索赔受理的整个行业的价值链。

3)数字身份和IP
ÆIf的多链结构里有一个内置的数字身份链。这保证了如果另一条链很繁忙的情况下这样的侧链的性能，比如，一个新的代币在另一条链上发行了。

在ÆIf里，数字身份能被其他链通过“消息”来使用。通过适配器，ÆIf还能从其他的链上获取信息和数据，比如比特币和以太坊。

4) 智能城市
政府也会对ÆIf感兴趣因为它让他们在ÆIf上可以安全便捷地实行特定的管理任务。政府或者组织能够定制共识机制来满足国家安全要求。一些行为，例如设施记录、公民身份、政府机构信息曝光以及选举都能在ÆIf上以极高的透明度和效率来实现。爱沙尼亚共和国、新加坡、中国等国家正在在这些领域做试验。

5)物联网
ÆIf支持轻节点和云服务，这降低了连接它的设备的算力要求，同时还能保持高的性能。对于管理成千上万的设备以及实现它们之间的微支付来连接物联网是关键的。

ÆIf为以上的行业描绘了坚实的基础。我们会积极地发现新的商业机会和DAPPs来加入ÆIf生态系统。

1) 和现有链上的DAPPs共同合作
在现有链上目前已经有很多成功的DAPPs ，比如比特币和以太坊。ÆIf会权衡它的交互功能来连接这些DAPPs来允许更多的资产交换并获得交易数据。

2) 孕育新的创业想法
我们的开发团队和顾问团队在全球的区块链社区里都深度参与到了新想法的产生和商业化。新的创业想法找到我们来寻求技术和商业的建议。我们会借助这种连接来孕育新的创业想法并把他们加入到ÆIf生态系统里。和风投一起，我们有信心能识别并把最有潜力的项目带入到ÆIf里来。

3) 对现有企业进行区块链转型
现有企业呈现了成为ÆIf生态系统的另一种机会。他们已经拥有大量的用户基础，并且证明了现有业务的价值。ÆIf能够帮助他们转型成更加强大的拥有更强激励和客户回报的，解决上述行业特定痛点的模型。ÆIf团队已经在和一些互联网公司以及传统企业讨论在ÆIf上尝试革命性的商业模式。我们能遇见不久的将来会有一些激动的宣告。另外，团队有意向和全球化的策略咨询顾问公司来共同推动ÆIf生态系统上下一代的商业模型。

6.3 资本

打造一个生态系统无疑需要大量的资本。除了利用好代币发售筹集的资金，团队和顾问还和全球领先的加密基金建立了很紧密的联系。团队和它的顾问们已经指导了许多的国际的代币发售项目。从长远来看，国际资本网络和我们的声誉保证了我们有强大的经济实力来支持项目未来的发展。

引用

1. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.
2. Vitalik Buterin. Ethereum White Paper: A Next Generation Smart Contract and Decentralized Application Platform. 2013.
3. Melanie Swan. Blockchain: Blueprint for a new economy. ” O’Reilly Media, Inc.”,2015.
4. Frederick P. Brooks. The Design of Design: Essays from a Computer Scientist. “Addison-Wesley”, 2010.
5. Andrew S. Tanenbaum. Modern Operating Systems “Pearson”, 2007
6. Joseph Poon and Thaddeus Dryja, The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments. 2016
7. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. 2014.
8. Hyperledger Whitepaper. 2016
9. Muhammad Saqib Niaz and Gunter Saake. Merkle Hash Tree based Techniques for Data Integrity of Outsourced Data. 2015
10. Robert McMillan. The inside story of mt. gox, Bitcoin’s 460 dollar million disaster. 2014.
11. Sunny King, Scott Nadal. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. 2012
12. David Schwartz, Noah Youngs, and Arthur Britto. The ripple protocol consensus algorithm. Ripple Labs Inc White Paper, 5, 2014.
13. Leslie Lamport. The Part-Time Parliament. ACM Transactions on Computer Systems, 21(2):133–169, May 1998.
14. Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. ACM Transactions on Programming Languages and Systems (TOPLAS), 4(3):382–401, 1982.
15. Leslie Lamport. Time, Clocks, and the Ordering of Events in a Distributed System. Communications of the ACM, 21(7):558–565, Jul 1978.
16. Paul Tak Shing Liu. Medical record system using Blockchain, big data and tokenization. Information and Communications Security, pages 254–261. Springer, 2016.
17. Robert Love. Linux Kernel Development. “Addison-Wesley”, 2010
18. Shawn Wilkinson and Tome Boshevski, Storj: A Peer-to-Peer Cloud Storage Network. 2016.
19. Contract. URL https://en.Bitcoin.it/wiki/Contract, 2014.
20. Mandatory activation of segwit deployment, UASF, BIP 0148. URL https://github.com/Bitcoin/bips/blob/master/bip-0148.mediawiki, 2017
21. Smart Property. URL https://en.Bitcoin.it/wiki/Smart_Property, 2016.