JURA

一种超高速、免费的自主调节去中心化网络

contact@jura.network

版本：1.1

2018 年 6 月 17 日

摘要

JURA 在过去几年见证了区块链技术的飞速发展，并致力于构建下一代底层公链技术。恶意的网络攻击日益猖獗，这对分布式网络在可扩展性、最终性和可靠性等方面提出了更高的要求。尽管目前已经出现了各种各样的解决方案，但受制于每秒处理交易数、网络安全性和可拓展性匮乏等，区块链技术大规模应用依旧无从谈起。有鉴于此，我们将在本文中为你详细介绍由 JURA 团队创立的项目——JURA。它包括 Fusus（一种基于区块链技术和 DAG 结合的数据结构）、PoU（自主调节的分布式网络的共识机制）、PoVRT 模块（可随机时间反欺诈验证）、可扩展的分片技术、以及人工智能的安全滤层。我们相信，JURA 将为去中心化应用的普及铺平道路，并打破当前的市场格局开辟新航路。

1 背景

2008 年，中本聪创建了世界上第一个加密货币——比特币[1]，这也是区块链技术的第一个应用。自那以后，许许多多的区块链应用（主要集中在金融行业）相继诞生。Ethereum[2]是一个去中心化的全球计算系统，它允许任何人创建、存储和运行基于“智能合约”的去中心化的应用，这极大地增强了构建区块链技术的可能性。IOTA[3]设计了 tangle（缠结），将区块链技术和物联网行业有效结合，tangle（缠结）是一种有向无环图（DAG），该有向无环图具有代表交易处理的顶点和通过加密连接边。Filecoin 是一种以区块链技术为基础的储存网络和加密货币，但它仅仅在达成共识方面做出了一些改良。因而，JURA 项目对区块链技术的关键部分进行重大改进，并致力于构建下一代的区块链技术。

尽管为数众多的区块链技术仍在开发之中，但实际上目前区块链技术依旧处于探索阶段，区块链技术的主要目的是满足稳健网络对“高度可扩展性、快速完成交易和强可靠性”的要求。但遗憾的是，目前还没有一项区块链技术能够同时满足这三项要求。目前，比特币平均每秒可处理 7 笔交易，而 Ethereum 每秒可处理 15 笔交易。尽管许多项目声称可以达到无限的 TPS，但事实是只实现了少量的 TPS。例如，Raiblocks （Nano）[4]平均可达到 100 TPS，峰值性能[6]时可达到 300 TPS 。Ripple 可以达到 1500 TPS 左右，但它是一项私有链技术，而不是公有链技术。这在参与者匿名的真正去中心化网络中至关重要。目前为止效率最快公链技术是 Steem，其目前记录约为 3500~7000 TPS，比 BTC 大约快 500~1000 倍。实现这一显著的增长主要得益于容易受到恶意攻击的委任权益证明（DPoS）共识协议。

尤其值得注意的是，目前这些技术都不能满足未来拥有数百万 TPS 的可扩展性要求。

就完成交易而言，比特币大约需要 6 个区块或 1 个小时来确认交易。而 Ethereum 在 3 分钟内大约确认 12 笔交易。许多其他区块链项目声称可以即时完成交易，但实际上他们根本没有实现这一目标。通过审视他们的各自的技术设计，我们发现，这些区块链项目仍有很大的改进空间。稳健网络的可靠性也是关键之一；过去几年，有关使用加密货币和 P2P 网络过程中的安全问题频频见诸报端，更加验证了可靠性在稳健网络中的重要性。在去中心化网络环境中，为了牟取巨大的经济利益，恶意用户以谋取巨大的经济利益为动机的同时，往往造成整个系统瘫痪。因此要想设计一个安全可靠的去中心化网络，就必须从多角度全方位思考。

鉴于当前区块链技术的局限性，为了更好地满足未来的需求，我们设计了 JURA，去中心化网络解决方案的集大成者，通过设计以下环节来大大增加可扩展性：一种作为分布式分类账依据的新型基础数据结构——Fusus，；确保网络安全的效用证明共识机制；可验证随机函数（VRF）[7]的应用（而这种可验证随机函数（VRF 经证实可提高安全性）；可过滤掉无效交易并对潜在恶意节点生成警报的 AI 安全层，；分片技术（这种分片技术可以通过减少每个节点在 P2P 网络中必须处理的信息和通信量来大幅提升可扩展性）。

本文组织结构如下：第 2 节将讨论 JURA 协议的总体设计；第 3 节讨论协议的实现；第 4 节详细介绍网络中的分片技术；第 5 节介绍 AI 提供的两个功能，并说明对我们技术的重要性；第 6 节对此内容进行扩展，介绍多种攻击场景，以及在 JURA 协议下如果解决这些场景的方法；第 7 节总结全文。

2 JURA 协议

JURA 协议与其他协议的不同之处在于，提出了一整套完善的解决方案。JURA 的核心是 Fusus 数据结构。PoVRT 取代了 PoW，它是一种反垃圾邮件工具。每个账户都有自己的 Fusus 链。然后，效用证明共识机制（PoU）起到了整个系统监督者（守夜人）的作用。

2.1 Fusus

（JURA 项目的）有关区块链技术的革命性设计之一，它以区块链作为数据结构的基础，（在交易过程中）进行加密并将后一段交易附加到前一段交易。这些交易在 Merkle 树中归类整理，可以把 Merkle 树的根放入到区块的头部。这种设计为验证和检查交易完整性提供了便利。而且这种设计还便于简单支付验证（SPV），简单支付验证使用布隆过滤器（BF）和 HyperLogLog（HLL），以实现快速交易。这一论述表明，数据结构在塑造系统中所有其他部分起着基础性作用。实际上，我们竭力希望找到新的数据结构。特别是，有向无环图（DAG），作为强有力的替代者，从传统区块链技术中脱颖而出。
的一项强有力的技术。

2.1.1 DAG

有向无环图（DAG）作为一种通用图形结构已被广泛应用于许多领域，例如调度、数据处理网络、因果结构、数据压缩等。这种有向无环图（DAG）也作为基础被应用于加密货币领域。

例如，塞尔吉奥勒纳于 2012 年首次尝试在 DAG 的基础上创建一种区块链免费的加密货币——DagCoin，不过仅仅停留在概念层面。随后，IOTA、Byteball 和 NANO 都提出了他们自己的 DAG 变体，以在各自协议中发挥作用。特别是，由 IOTA 设计的 DAG——Tangle（缠结）把额外的概念引入到了 DAG 中，例如深度、重量、累积重量和提示，这些概念一起应用于以马尔可夫链蒙特卡罗（MCMC）随机采样算法为基础的提示选择算法中。遗憾的是这些算法过于复杂，并且单个 DAG 的使用并不能很好地随系统一起扩展，使得脆弱点容易受到恶意攻击。字节雪球引入了见证者模式以便构建主链。这实质上引入了一种类似于现实世界信誉的机制以及一个统一管理机构来管理们们的去中心化系统。分布式系统中任何形式的集权化都有可能使其受到攻击和控制。在理想环境中，这个系统当然会正常运行，但从长远来看，由于自身存在弱点，系统崩溃的可能性很大。NANO 通过使用区块点阵 DAG 数据结构而作为一个特殊的框架脱颖而出。在这种结构中，每个账户（或用户）都有自己的区块链，并且在发送和接收交易之间进行了区分，使得账户维护简单，完成交易速度加快。可以在每个节点上跟踪每个账户中的余额。这种方式在搜索所有 UTXO（未使用的交易输出）时可以节约大量时间以便在每次交易前计算出当前余额，这就像在当前的比特币区块链系统中一样。当交易量较大、交易历史较长且对快速检索的要求较高时，这种特殊框架的优势更为明显。此外，区块链在 NANO 公平处理、发送和接收交易方面的线性结构，以及把 PoW 用作为主要的反垃圾邮件策略，这都会导致交易速度变慢，且安全性不足。

通过这些观察研究，我们发现目前需要一种新型数据结构来克服当前区块链技术固有的局限性。在下一节中，我们将为你介绍 JURA 区块链技术的基础——Fusus。

2.1.2 Fusus

Fusus 作为一种灵活实用且有效的数据结构，是区块链、区块点阵和 DAG 的多重继承，可同时实现可扩展性、快速交易、轻量化和即时确认。

账户

账户是加密签名密钥对的公钥部分。从公钥获得的地址可以和整个 P2P 网络共享，而私钥只有账户持有人掌握。传统上，一个账户可以一次创建交易并可以把交易发送到目标地址。该账户并不会加密存储它过往的交易记录。在 Fusus 中，允许每个账户持有、存储和跟踪自己的交易。无论是比特币的区块链，还是 IOTA 的 Tangle（缠结），每次交易时，系统都需要搜索整个历史来计算账户的余额。当历史记录被快照（一种删除早期部分历史记录的技术）精简时，由于随机密钥生成和余额搜索机制，许多账户余额将变为 0。因此，跟踪每个账户中的余额不仅可以实现快速离线检索，还可以降低余额出错的可能性。

每个账户都使用具有非零余额的创始区块来启动。这是为了免受女巫攻击（有关女巫攻击的内容我们将在后续章节中为你详细介绍）所采用的重要手段。对每个账户上的 Fusus 进行任何更改都需要该账户的私钥，从而满足速度和便利性的同时再加上一层“安全膜”。

交易

交易是把价值从一个地址转移到另一个地址。JURA 与传统区块链技术的区别在于 JURA 将发送和接收交易进行了分离。在传统的区块链技术中，交易是从整个系统层面来看的，并被简单地视为从 A 到 B 的单一动作。但从账户的角度来看，每个账户都有发送交易（ST）或接收交易（RT）。考虑到不同的处理时间要求，两者应该分开处理：发送交易确实存在固有的时间紧迫性，核心前提是账户中必须有足够的余额来实现发送交易。另一方面，接收交易（RT）不存在时间紧迫性，因此相对被动。同时，让许多接收交易传入是可能的，而传入的发送交易过多可能会对系统造成压力。交易对账户余额没有要求（除非有特别需要），可以缓慢地更新账户。这一观点可以通过对现有加密货币的使用信息进行数据分析来提供支持。

```
发送{
    以往账号：29DF39JD...F9dfd，
    余额：010a8033s..1dfd9d，
    目标账号：mft_df90dfafd，
    消逝时间： 0.23214289304，
    时间戳： 1521161583，
    类型：发送
    签名：83BE...EFND89F
}
```

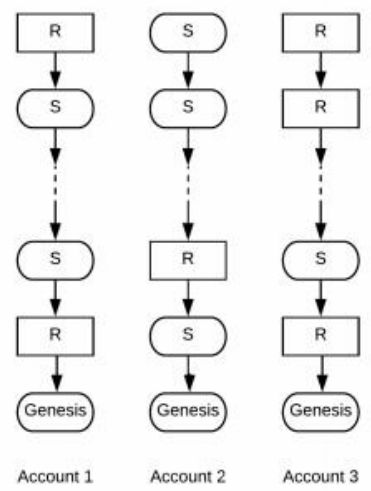
清单 1 ：发送交易中的字段

Fusus 数据结构

Fusus 数据结构本质上是一个 DAG 点阵。也就是说，账户的交易历史记录构成点阵，而每个账户的历史记录由 DAG 构建。交易较低时，DAG 可有效缩减到一个区块链结构。对于相关示例，请参见图 1。业务量较高时，接收交易会构成一个 DAG，每个发送交易起到了新创始区块的作用。

对于相关说明，请参见图 2。Fusus 始于创始区块，而接收交易则会构成 DAG。新交易涉及到 $k \geq 1$，以前的接收交易取决于业务情况。交易量较高时，新交易可以充分利用 DAG 的潜力尽快地并尽可能多地吸纳交易。每当出现发送交易时，Fusus 就会成为创始区块或先前发送交易之间所有接收交易的验证器，如果尚未确认发送交易的话，Fusus 则会用作当前发送交易的验证器。

1. 记录初始余额的创始区块。
2. 接收区块高低交易量模式出现，并产生窄/宽的 Fusus。
3. 每个发送区块参考所有确认的接收区块，将 Fusus 缩小到单个节点，并在对接收区块中的余额做出结算后确认净余额。
4. 发送区块将作为新的创始区块插入接收 DAG 中，并且现有接收区块的确认数量保持不变。
5. 未确认的接收区块将继续增加 DAG。
6. 允许精简以减少历史记录。



Genesis	创始区块
Account 3	账户 3
Account 1	账户 1
Account 2	账户 2

图 1 ：低流量场景中的 Fusus 数据结构。

交易时间

通过对交易过程的分析可以了解完成交易所需的时间。

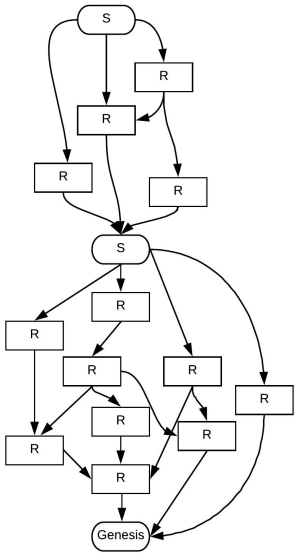
- 将交易从发送地址发送到接收地址的通信时间：t_1
- 从接收地址至发送地址广播的接收交易通信时间：t_1
- 用于发出发送交易、PoVRT 和散列时间等的处理时间：t_2
- 用于确认接收交易、添加到 DAG、广播到网络等的处理时间：t_3
- 总时间约为：$t_0 = 2 * t_1 + t_2 + t_3$

在该公式中，t_1 通常是固定的，t_2 是随机的，它具体取决于 PoVRT 共识机制（这种 PoVRT 共识机制将在下一节中解释）。Fusus 可以显著减少 t_2 和 t_3，这是我们认为 Fusus 在处理速度方面将优于单个 DAG 或区块点阵网络。考虑到 IOTA 和 NANO 的零交易费，乃至反垃圾邮件 PoW，两者都因无法处理垃圾邮件攻击而饱受诟病。拥有大多数算力就可以轻松发起数百万次的交易，突破 PoW 并发起攻击。JURA 也免除交易费用，但 Fusus 通过引入以下机制来从而在设计上避免上述攻击。

PoVRT

通过观察比特币生成区块所需的时间，我们可以得出一个重要的结论，工作量证明本质上是一种分配随机强制等待时间的机制。由于现代超级计算机的处理速度快，净效应不一定要通过解决复杂数学难题来实现（这种难题就是留有一个正在受攻击的漏洞）。相反，我们提出了一种可随机时间反垃圾信息验证（PoVRT）机制的验证方法，从而通过可验证随机函数（VRF）模块[7]生成可验证随机时间。该模块可以创建一个认证，任意节点都可以使用该认证来验证发送节点是否以正确方式以等待合适的随机时间。

需要注意的是，PoVRT 不仅仅是 PoW 的技术替代方案，而且实际上可以防止上述恶意攻击。无论掌握算力有多强大，系统自始至终都会强制在一个随机等待时间内。



Genesis	创世区块
Account 1	账户 1

图 2：高流量场景中的 Fusus 数据结构。

以下设计能够让正常账户的交易速度更快，并让恶意攻击的难度变得越来越大。

1. 在时长为 t_{cap} 的时间窗口中，第一个发送交易只须等待几毫秒的时间。
2. 随后交易的等待时间将不得不逐渐延长，直到达到上限 t_{cap}。
3. 确切的计划是，第 1 个 Tx：0 ~ t_{lim1}s；第 2 个 Tx：t_{lim1} ~ t_{lim2}s；第 3 个 Tx：t_{lim2} ~ t_{lim3}s；第 4 个 Tx：t_{lim4} ~ t_{lim5}s；第 5 个或更后的 Tx：t_{lim5} ~ t_{lim6}s。t_{cap} 秒是限定的等待时间。

因此，普通用户不会感觉到存在强加的等待时间，但是恶意用户会感觉在一定时间段内进行的交易会越来越困难。

广播和验证算法

每个用户都有自己的 Fusus 交易历史记录。如前所述，发送交易起到新创世区块的作用。每当创建新的发送交易时，就会向整个网络广播前一个发送交易、新发送交易和所有中间接收交易进行确认。出于工程设计目的，接收交易被添加在一个 DAG 结构中，以实现更大的可扩展性。然而，就验证和计算而言，精确的图解结构并不重要。我们将广播数据简化成了一个以下结构。对于固定账户 A，D^k 是第 k 个广播数据：

$$D^k \equiv S_1^k \rightarrow \{R_1^k, R_2^k, \ldots, R_{N_k}^k\} \rightarrow S_2^k,$$

其中 S_1 是前一个发送交易，S_2 是当前发送交易，R_i^k 是第 k 个广播中的第 i 个接收交易，Nk 是接收交易的数量。从更高形态上来看，这三个组成部分存在一个连续顺序，即 S_1 是第一个，最初以图解形式组织在 DAG 中的数据块 $\{R_1^k, R_2^k, \ldots, R_{N_k}^k\}$ 是第二个，S_2 是第三个。

Dk 被广播到整个网络，并由所有完整节点接收以进行验证。这里我们假设 S_1^k 已在网络上达成共识。更具体而言，S_i^k 是来自账户 A 的发送交易，其向账户 B_i^k 发送 m_i^k 的金额，留下余额 n_i^k，而 R_i^k 是来自账户 C_i^k 的接收交易，其接收 u_i^k 的金额。

将 L 表示为一个给定的完整节点所保留的账本，是从完整节点的公钥到相应 Fusus 历史记录的映射。对于第 j 个完整节点 FN_j，将其当前账本表示为 L_j，并将 $L_j(A)$作为 L_j 上账户 A 的 Fusus。

定义：F（账本, 接收交易）$\mapsto\{0，1\}$ 如下：

$$F（L, RT）= \begin{cases} 1 \text{ 如果我们能在 L 中找到 RT，} \\ 0 \text{ 不能在 L 中找到 RT} \end{cases}$$

该函数用于检查接收交易是否处于特定的账本中。

对于账户 A，2 个发送交易之间的等待时间至少为 δt_A。将 t_j^k 表示为 FN_j 从网络上听到 D^k 的时间，$t_j^k - t_j^{k-1}$ 为 FN_j 上账户 A 两个接收交易之间的差异。可使用一个简单的条件来检查两个发送交易是否在时间上过于接近。如果 $t_j^k - t_j^{k-1} < \delta t_A$，则拒绝新的发送交易。如果 $t_j^k - t_j^{k-1} >= \delta t_A$，则认为新的发送交易通过了时间测试。

然后将 $B = n_i^k$ 设置为逐笔结算余额，D_{New}^k 为空 Fusus，那么检查一个完整节点上新发送交易的有效性的算法如下：

对于 *range(1, Nk)的 i，执行以下程序*
    如果 $F(L_j, R_i^k) = 1$，则 $B = B + u_i^k$ ； 将 R_i^k 加入到 D_{New}^k
    或者
        通过
    结束
结束
如果 $B < m_2^k$，则拒绝新 ST S_2；
或者
    更新 $\bar{S_2^k}$ 越= $B - m_2^k$ ；
    将 $\bar{S_2^k}$ 加入到 D_{New}^k
    $L(A) += D_{New}^k$
结束

算法 1：检验一个完整节点上新发送交易有效性的算法。

使用该算法时，每个发送交易将由每个单独的完整节点来验证。

2.2 效用证明

2.2.1 背景

目前普遍采用的共识机制是比特币和以太坊网络中所使用的工作量证明（PoW）。众所周知，工作量证明在将算力转化为有价值的输出方，这既费精力，又不可扩展。因此，为了克服这些弊端，人们又提出了权益证明（PoS）机制。

在 PoS 共识机制中，验证者随机轮流地在下一个区块进行提议和表决，权重取决于权益大小以及为获得表决权而持有的加密货币数量。

然后，在该系统内会以权重大小来决定在某个间隔内（例如每 10 秒）选择某一节点验证下一个区块的概率。这种机制在一定程度上克服了 POS 的弊端，但由于其以个人权益为基础，因此仍不可避免地遭受恶意攻击，并且很难惩罚不诚实的行为。

委托权益证明（DPoS）在 PoS 机制中引入了委托人的概念，以克服 PoS 机制的缺点。在该机制中，使用委托实质上是在节点顶部安排一层具有可信实体的集中化管理，以实现某种形式上的技术民主。BitShares、NANO 和 Steem 都是目前使用 DPoS 的共识机制，然而，人们普遍抱怨 DPoS 带来了的中心化的问题，引起了普遍担忧。按照设计，委托人应随机挑选，以确保实现网络的安全性，然而在实践中，在一个去中心化的网络中找到可信赖的非官方委托人是一项巨大的挑战。因而，设计者们必须亲自挑选值得信赖的委托人，而他们则往往成为共识中的主导力量。这就意味着中心化的出现。例如，在 NANO 网络中，（截至白皮书成稿前），52 %的权益牢牢掌握在以 NANO 核心开发者为代表的官方团队手中。此外，人们发现用户钱包中的所有的默认设置都直指该项目的某位官方代表。这种中心化赋予了开发团队至高无上的权力，同时这也会导致他们的地位岌岌可危，因被恶意活动的所控制甚至取而代之。

我们认为，将中心化和现实世界的信任体系引入到一个分布式系统中只是实现全面去中心化的特定阶段的中间步骤，换句话说，一种鼓励人人参与的设计才是真正意义上实现去中心化共识的基础。

2.2.2 共识机制

在 P2P 网络中的理想情况是，共识机制既能为按照协议正常运行的诚实节点给予奖励，又可防止不诚实节点的恶意攻击。从经济角度来看，该系统的激励机制足够完善，恶意攻击的机会成本足够高。

因此，在受到由 Peercoin [8]提出的货币制度概念的启发下，JURA 的效用证明（PoU）机制主要有两大基本内容：第一种是通过效用来管理表决，第二种是惩罚不诚实行为。效用是通过一组指标的函数而获得的，该指标用于测量对系统正常运行的贡献，而惩罚机制现阶段正在研究中，以用于下一个 PoS 协议，即通过以太坊提出的 Cashere 协议 [13]。

时间是 PoS 系统的一个重要信息。自此之后，时间是指可以从交易和权益中记录的时间戳中直接获得的任何信息。这些基于时间的指标可以与物理世界中的指标相关联，例如对调节人们行为具有重大影响的信用评分。然而，在一个参与者匿名且侧重于隐私保护的分布式系统中，并不存在像信用评分之类的指标。JURA 的希望通过时间信息来深入了解各个节点的“信誉”，并将规则创建到相关协议之中从而使系统以应激的方式来运行。

从理论上讲，权益的大小可以被视为表决的效用，效用也被广泛应用于研究人们的经济行为。而在这里效用指维持系统正常的影响力、贡献度和善意。效用较高的权益应当具有更强的主观能动性来促进系统的稳健运行，反之亦然。

在数学上，将权益大小设定为一个随机变量 S∈（0，∞），让效用成为可以表示为 U（S）：$R^+ \rightarrow R^+$ 的 S 函数，，那么在当前的 PoS 系统中，权益的效用就是它的大小，即 U（S） = S。权益大小和表决权之间的线性关系很容易导致出现垄断问题，该问题将在第 6 节中予以讨论。

在仔细研究现有的分布式系统内容后，我们认为，通过将时间信息考虑在内，效用函数能够防止出现 PoS 系统存在的弊端，并使权益变得更加智能化从而维持系统的稳健性。效用证明(PoU)**共识机制可根据这一原理来构建。详情请见下一节内容。**

2.2.3 PoU

至少早在 2010 年，中本聪便已知晓使用币龄的这种概念，并将其用于比特币中来帮助确定交易的优先级，不过它在比特币的安全模式中并未发挥出关键作用。币龄是指加密货币金额和持有期间的乘积。例如，如果 Bob 从 Alice 那里收到了 10 个货币，并持有该货币 90 天时间，那么 Bob 所积累的币龄为 900 天。Peercoin、Vericoin [9]和 Reddcoin 中也存在类似的概念。年龄与权益大小的倍增可能会导致“史前巨鲸”在表决时会突然产生巨大的冲击力。Vericoin 提出了一个非线性加权函数来限制年龄，但这显然只是一种“权宜之计”，而不是“长久之道”。

我们的将该概念置于效用框架中，货币年龄的效用便是 U（S，T） = S * T，其中 T 为持有期的随机变量。但币龄并不是使用时间信息的唯一方式。

在 JURA 中，时间信息被转换成三个指标指标：一个是可表示为随机变量 $T_c \in$（0，∞）的累积权益时间（CST）或权益年龄，；第二个是平均投注间时间 （AIST）$T_a \in$（0，∞）；第三个是最后投注间时间（LIST）$T_l \in$（0，∞）。

这三个组成部分发挥着三种不同的作用：CST 表示资历，AIST 表示积极参与率，LIST 表示最近的积极参与率。这三个变量可以很容易地从交易和系统信息中导出，并且是效用函数的额外输入变量。新的效用函数将采取 U (S, Tc, Ta, Ti)的形式。请注意，Tc 与上文提到的持有期相同，但 Tc 和 S 之间的交互将会不同，效用函数中引入了两个额外指标。确切形式将在进一步讨论后给出。

我们通过相对性来考虑效用的确切形式。任何描述队列特征的数字量只有在放入其分布时才有意义。例如，在相当活跃的系统中，一天可能很长，而在不活跃的系统中，一天可能很短。这意味着需要将基础随机变量的系统级视图放入效用函数中。

JURA 协议通过一个具有四个指标的系统级视图来实施效用函数。首先检查每个指标，并将它们放在一起，以获得效用函数的形式。

CST

假设单个权益的创建时间是 T_0，当前时间是 Tnow，在这种情况下，CST 为 $T_c = T_{now} - T_0$。对于特定的权益 i，CST 满足 $t_{ci} = t_{now} - t_{0i}$，其中 i = 1，2，.....，n，其中 n 是表决系统中的权益总数。

CST，无论是一天、一个月，还是一年，都不会泄露任何有关权益的信息，除非与其他 CST 比较。根据累积密度函数（CDF）算得的分位数是累积密度排名值的指标，介于 0 到 1 之间。我们使用$\varphi: R^+ \rightarrow (0, 1)$来表示 T_c 的 CDF，于是φ可以用来评估 T_c 取小于或等于 t_c 值的概率，即$\varphi(t_c) = P(T_c < t_c)$。图 3 示出了模拟 CST 的直方图，图 4 是相应的 CDF 图。很明显，这是一个非线性标准化过程，从正实数到其百分位数$\varphi(t_c)$，此时可用作指标 T_c 分配给权益的边际权重。

一方面，CST 的标准化可确保任何一个单独权益均无法仅因为存在时间长来支配整个系统。另一方面，直接跳过系统的权益并不会产生影响，因为 CST 将接近于零，权重几乎忽略不计，从而防止突然的恶意攻击。

随着权益越来越大，权重会逐渐增加，从而产生更高水平的影响。权益从零上升到任何重要权重所需的时间可视为发起攻击的机会成本。

权益与币龄交互的概念也与民主选举制度（虽然每个人都有一张票）产生了很好的共鸣：更专业的资深成员和更强大的网络往往具有更大的影响力。

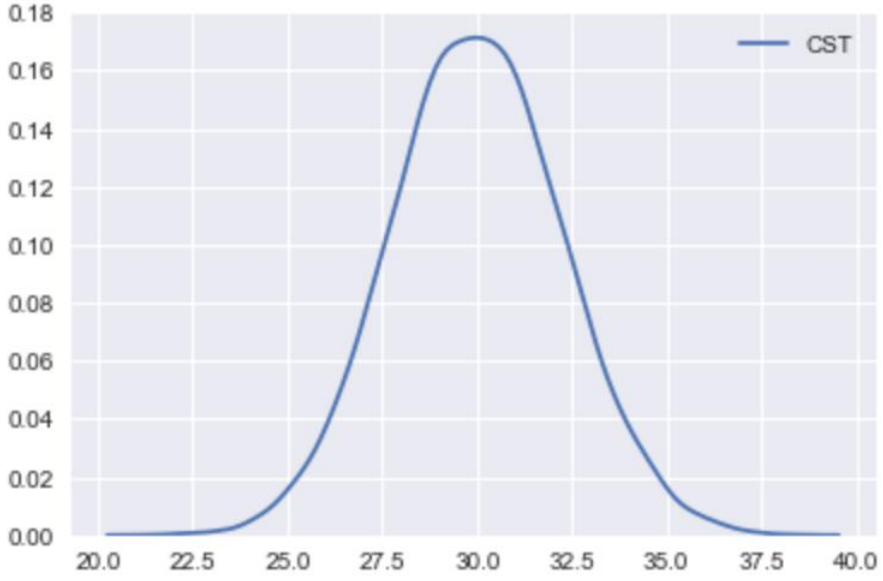


图 3 ：CST 的模拟概率密度函数。

AIST

假设自创建以来，第 i 个权益在进入和退出时间点 $t_{i1}^{En} < t_{i2}^{Ex} < \ldots, < t_{in_i}^{En} < t_{in_i}^{Ex}$ 的序列上已经参与了 ni 次投注，那么平均投注间时间是 $t_{ai} = \frac{1}{n_i - 1} \sum_{k=2}^{n_i} \left[t_{ik}^{En} - t_{i(k-1)}^{Ex} \right]$。这一数量可用于衡量投注参与率。由于 AIST 是一个平均值，因此根据中心极限定理（CLT）并假设均值和方差均为有限值，即 $0 < \mu_{ai} < \infty$ 和 $\sigma^2_m < \infty$，随着参与数额越来越高，预计 AIST 会呈正态分布，即随着时间的推移，每个账户的习惯性行为就会出现并集中于一个具有某些参数的分布。形式上，当 $n_i \to \infty$ 时，

$$\sqrt{n_i}\left(t_{ai} - \mu_{ai}\right) \xrightarrow{d} N(0, \sigma_{ai}^2).$$

在总体水平上，t_{ai} 的百分位数为Ψ（t_{ai}），其中Ψ（T_a）：R +→（0,1）是 T_a 的边际 CDF。在边际权重为 1 - Ψ（T_a）的情况下，取较小值显然更胜于取较大值。如果 AIST 很小，那就意味着参与率很高，如果参与者无任何不诚实行为，理应得到更高的权重。相反，不活跃的参与者则无法给予很高的权重，除非其通过一段时间内的积极参与来加以弥补。这也避免了无任何历史记录的权益对共识机制产生重大影响。此外，缺乏参与也可视为是一种机会成本。如果权益不经常用于参与交易，则其效用就会降低。这自然会激励更多利益相关者参与进来。

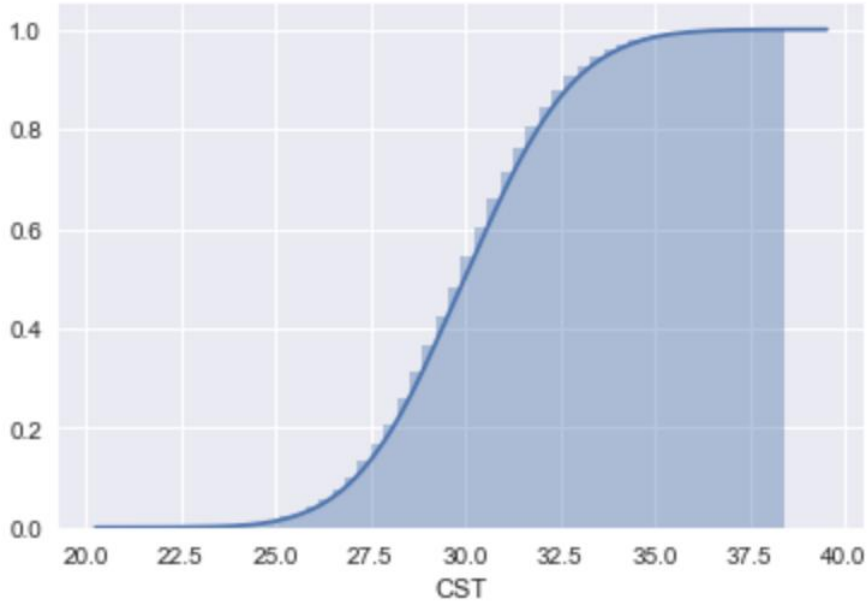


图 4：CST 的模拟（经验）累积密度函数。

LIST

使用与上面相同的符号，最后一次互测时间的计算公式为 $t_{li} = t^{En}_{in_i} - t^{Ex}_{i(n_i-1)}$。这一数据衡量的是参与者最近一段时间内的权益，因此更关注最近的短期行为，而非长期的习惯性行为，如上文 AIST 分析部分所述。这也与市场条件和其他宏观因素有关。

CDF 为Γ（T_l）：R +→（0,1）。在边际权重为 1 - Γ（t_1）情况下，取较小值显然更胜于取较大值。CDF 可以作为对具有积极参与历史但最近存在巨大投注缺口的权益起到缓冲作用。

在实施阶段，我们以离散方式对叠后时间进行追踪。我们并不需要确保（网络）时间戳的准确性。我们的方案是，通过对使用了叠后的投票轮数进行了跟踪，然后该轮数会显示模拟时间和易于保持网络的一致性。

数值计算

尽管我们系统的精确 CDF 并非是现成的，但我们预计系统会生成大量数据，我们能够获得经验累积密度函数（ECDF）。即使在系统的早期阶段，来自其他 PoS 系统的数据也可用来为我们的网络提供与 ECDF 有关的粗略估算。

需要注意的是，分布模型其实并未被参数化。鉴于当前的算力，通过黑客攻击或使用人工智能的方式来推断在这种情况下的参数其实并不可取。此外，由于时间戳不可变，即使了解这些分布模型也无法增加权益的权重。而且，效用函数将使任何试图钻空子的行为变得徒劳无力。

双变量联合 CDF

上面关于 CST 和 AIST 的讨论揭示了一个观点，即这两个指标描述出了利益相关者的长期特征。二者不可分割，并且它们可能会以一种通过联合 CDF 完美诠释的方式进行共变。单独使用联合 CDF 而非边际 CDF 可以提高权益真实效用估计量的一致性，即 U（S，T_c，T_a，T_i）的估计量 f [S，Λ（T_c，T_a），T_i] 具有比另一估计量 g [S，Φ（T_c），Ψ（T_a），Ti]更好的一致性，其中Λ：$R^+ \times R^+ \rightarrow$（0,1）是（$T_c$，$T_a$）的联合 CDF。对于联合 CDF 的图示，请参见图 5，图 5 中的一个维度用于选择各边际，另一个维度仍在不断变化。因此可以通过将联合 CDF 转化为单变量来进一步说明现行 PoS 系统的局限性。

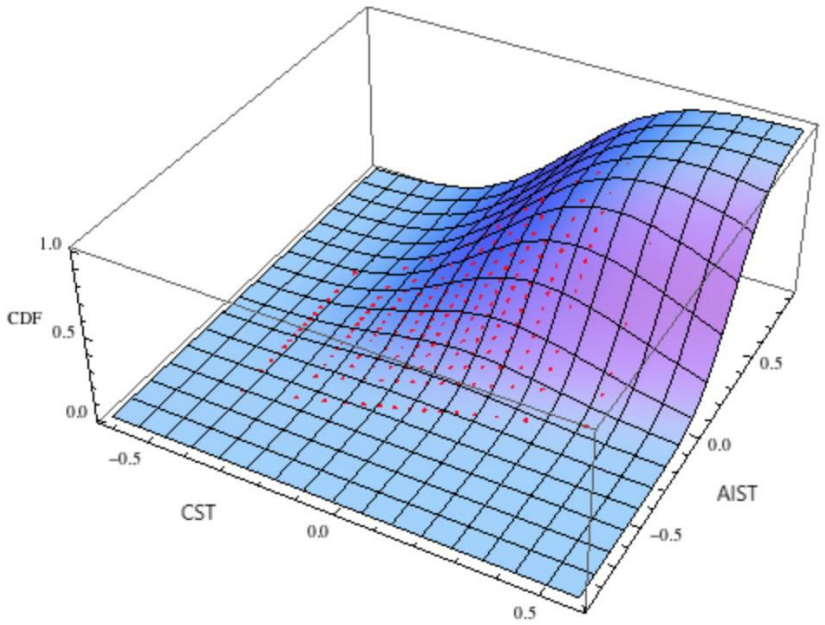


图 5：CST 和 AIST 的模拟双变量联合 CDF。

四个指标的效用

如果在效用函数中对所有的四个指标做了考虑，则可以进一步改善这种一致性。假设 V =（S，T_c，T_a，T_l）表示与三个时间相关指标和权益大小有关的四维随机向量， vi =（s_i，t_{ci}，t_{ai}，t_{li}）表示观察到的第 i 个权益， 则其效用为：

$U(v_i) = P(S < s_i, T_c < t_{ci}, T_a > t_{ai}, T_l > t_{li})$

$$= \int_0^{s_i} \int_0^{t_{ci}} \int_{t_{ai}}^{\infty} \int_{t_{li}}^{\infty} \theta(x_1, x_2, x_3, x_4) dx_1 dx_2 dx_3 dx_4$$

$$\approx \frac{1}{n} \sum_{k=1}^{n} \delta_k(s_i, t_{ci}, t_{ai}, t_{li})$$

其中 $\theta(x_1, x_2, x_3, x_4) = \frac{\partial^4 \Theta(x_1, x_2, x_3, x_4)}{\partial x_1 \partial x_2 \partial x_3 \partial x_4}$ 是 V 的概率密度函数，Θ（x_1，x_2，x_3，x_4）是相应的 CDF，而δ_k（s_i，t_{ci}，t_{ai}，t_{li}）是指任意第 k 个权益是否满足条件的狄拉克δ函数，

即第 i 个权益的权重为 pi = $\frac{U(v_i)}{\sum_{i=1}^{n} U(v_i)}$，其中 n 是参与投票者的 总数。

实际问题

实际上，快速计算效用值以及计算经验联合 CDF 通常面临着诸多挑战且非常耗时，使得 O（nlog（n））的排序变得相当复杂。当权益很大时，计算效用值任务所消耗的时间也会日益剧增。

Sklar's 定理指出，任何多变量联合累积分布函数均可以分解为多个边缘分布函数和一个 coupula 函数（该函数描述变量之间相关性）。

特别是，根据极大似然估计（MLE）方法，对权益样本中的参数分布来获得作为 f_1（tc）、f_2（ta）、f_3（t_i）和 f_4（s）的 v 的边缘分布。然后，所得到的参数边缘分布可用于计算像 $\hat{F}_1$（tc）、$\hat{S}_2$（ta）、$\hat{S}_3$（ti）和 $\hat{F}_4$（ts）之类的边缘 CDF（或生存函数，即 1 - CDF）如。定期计算所估计的关联矩阵Σ，然后通过高斯连接函数 $C_{\Sigma}^{\text{Gauss}}(v) = \Phi_{\Sigma}\left(\hat{F}_1^{-1}(t_c), \hat{S}_2^{-1}(t_a), \hat{S}_3^{-1}(t_i), \hat{F}_4^{-1}(t_s)\right)$来计算效用值。

参数和分布的估计可以每天或每周进行一次。使用这套行之有效方法可以减少权益信息在 O（1）时间内转换为效用值，这是分布式网络中亟需的功能。

潜在的问题包括某种程度上的精准失真，即边缘分布时存在的误差，关联矩阵会使特定权益的效用变得不精准。应设法把误差控制在最小范围内，其原因描述在图 4 所述的标准 CDF 中，如果把误差值控制在正常范围内，即 CDF 的线性部分，则 MLE 估计的误差应该非常小，因为数据点在分布高密度范围内。如果数据点分布在坐标的最远端，不过由于两尾部均为非线性，因而即使误差很大，分位数的最终值依旧偏差很小。此外，由于效用作为概率抽样程序的输入函数，因此该影响将进一步降低，即效用值中 0.2 的误差不会改变被显著选择的概率，即随着 n→∞，$0.2/\sum_{i=1}^{n}$U（v_i）→0，但如果效用值偏离其真实值 0.2 个单位，则会产生较大的误差。

抽样问题 计算权重的时间与 PoS 的数量级相同，但是，使用 PoU 系统的一个巨大优势是可以获得通过设置效用值阈值来过滤掉不合格权益的标准和方法，例如，只有高于 0.6 的效用才有资格进行下一轮抽样，这大大减小了抽样空间。100 万权益的样本可以减少到仅仅需 5 万权益用于抽样，抽样速度会显著提高。在 PoS 系统中，根本不存在这种优势，因为权益大小不受抽样周期的限制且与其不一致。计算效用和被选中的概率的时间需忽略不计。一旦获得每个权益的被选概率，随机抽样步骤就会变得很简单。

更新频率 根据稳定性和必要性，可以通过累积法或窗口运行的方式，按天、按周或按月使用 JURA 系统中的流量来更新 ECDF。可以连续不断地通过系统级多变量 Kolmogorov - Smirnov 拟合优度检验[11]来监测 ECDF 的稳定性。更新选在低交易量时进行以免对系统产生影响，这非常类似于比特币网络定期难度调整的方式。

场景分析

对于我们的基本场景，我们使用 0 - 10 来表示每个指标的可能取值，并且可以通过效用函数 U（S，T_c，T_a，T_l）中这些数值的组合来表示对投注场景进行定性分析。默认选项下，对于新权益，将 AIST 和 LIST 设置为 10。

1. U（10，0，10，10）：系统中大量的新购权益。虽然权益大小有可能达到最高，但其他三个指标均处于不利的价值。该权益的效用实际上为 0，因此，该权益不具有任何表决权。
2. U（5，5，5，5）：具有正常历史记录长度的正常权益、正常 AIST 和正常 LIST。该权益没有任何可疑之处，且效用应具有很高的价值，尽管这些价值都只是平均值。
3. U（5，10，10，10）：系统中存在时间最长的正常权益，但几乎没有任何参与的历史记录。如果立即给予很高的权重，则该权益可能具有风险。因此，效用的价值也应该很低。

4. U（5，10，2，10）：存在时间长、历史上正常活跃的权益，但长期不活跃。为安全起见，为该权益设置了缓冲效应。也就是说，权重相当小，一旦权益有了一定程度的参与，效用将迅速上升到一个高价值。

5. U（5，10，8，2）：存在时间长的正常权益，这些权益在历史上是不活跃的，但最近却很活跃。该权益有可能产生影响，但建立起历史记录需要假以时日。效用最初只发生在后半部分，然后迅速发展到上半部分。

6. U（1，5，1，1）：少量平均币龄权益，历史和近期参与率都非常高。尽管权益的规模很小，但高参与率可以弥补规模小的不足。因此，虽然效用不会很高，但不会低到按权益的规模计算的边际位置。

2.3 共识模型

本节通过结合 Fusus 数据结构和效用证明共识机制（PoU）展现在网络上达成共识的细节。步骤和关键项将依次给出。

1. 通过 PoU 随机选择 n 个完整节点。为这些节点做出正确决策的概率应远高于通过 PoS 系统选择的概率。用ΔL_i来表示从全节点 i 更改账本。

2. 对于每个候选ΔL_i，网络需要检查ΔL_i的一致性，如果不一致，则拒绝ΔL_i。具体的检查过程如下：

   (a) 检查ΔL_i是否与广播数据一致，即是否存在任何交易冲突。

   (b) 检查每个 RT 是否可以追溯到每个ΔL_i中的 ST。

   (c) 检查ΔL_i中的每个 ST 是否有足够的余额，以保持其有效。

3. 对于任何ΔL_i，典型的发送数据形式为

$$D_A^k(i) \equiv S_1^k(i) \rightarrow \{R_1^k(i), R_2^k(i), \ldots, R_{N_k}^k(i)\} \rightarrow S_2^k(i),$$

   其中D_A^k（i）是由账户 A 广播的第 k 个数据，记录在账本 i 中。

   备注：

   - 记录在账本 i 中的D_A^k（i）可能不会记录在账本 j 中。
   - 对于账本 L_i 和 L_j，ΔL_i和ΔL_j可以在 S1 和 S2 之间选择不同的 RT 组，因此D_A^k（i）和D_A^k（j）可以不相同。

4. 用 Rset = {Rt |Rt∈D_A^k（i）for any i }表示来自D_A^k（i）i = 1……n 的 R 的并集，并将 $\widetilde{D_A^k} = S_1^k \prod_{R \in R_{set}} R^k S_2^k$定义为“超级”广播数据。注意上面的 2（a），$\widetilde{D_A^k}$仍然是广播数据的子集。此外，$\widetilde{\Delta L}$表示为所有$\widetilde{D_A^k}$形成的账本的变化。因为 2（b），我们知道$\widetilde{\Delta L}$也是一致的。此外，$\widetilde{\Delta L}$包含网络可能批准的所有潜在交易。而$\widetilde{\Delta L}$形成要验证的交易的候选池。

5. 我们声明$\widetilde{\Delta L}$形成一个账户间 DAG，其中每个节点都是一个交易。对于任何一对发送和接收交易，我们将 Node（S）和 Node（R）都表示代表交易的节点。对于来自 D≡S_1→{R_1，R_2……R_N}→S_2的任何 S_T S_2，我们在节点（S_2）到节点（S_1）和每个节点（R_i）放置一个直接边。对于任何节点（R_i），我们可以识别其为发送交易的节点，并放置更远的直接边。

[这里将插入一个账户间 DAG 的图表。]

注意，这个账户间 DAG 可能没有连接，我们称之为 DAG 林，每个连接的组成部分都是一个 DAG 树。

6. 对于每个 DAG 树，网络将从叶子到根对其进行验证，验证采用贪婪算法。对于任何验证节点（交易），网络都会检查两件事：一是当前余额是否支持该交易，这是通过检查输出直接边（RTs）来完成的；二是该节点是否获得了所有ΔLi 的 70 %投票。如果一个节点被认为无效，我们通过删除该节点和所有连接边来精简 DAG 树。

7. 此时，我们必须有一个精简过的有向无环图（DAG）森林，所有交易将在所有完整的节点和账户中形成$\widetilde{\Delta L^*}$。所有完整节点将附加$\widetilde{\Delta L^*}$到其当前账本中，所有账户都可以检查$\widetilde{\Delta L^*}$，看看其交易是否被批准。此时，该网络已经达成了新的共识。

3 执行

3.1 可验证的随机函数

为了替换节点中的反垃圾邮件的 PoW 模块，我们选择了一个可验证的随机函数来生成随机等待时间的 PoVRT 机制。等待指定随机时间的证明后续将由分布式系统中的其他节点来加以验证。这种具有随机分散性、唯一确定性、非交互性的友好型“分布式密钥生成（DKG）”签名方案源自 BLS [10]。

叶甫根尼·杜迪斯和阿勒克桑德·亚姆波尔斯基提出了一种有效且实用的可验证随机函数[5]。VRF（Virtual Routing and Forwarding，虚拟路由转发）的机制如下所示：

在输入 x 和密钥 SK 后，我们的 VRF 会计算（FSK（x），π（x）），其中 FSK（x） = e（g，g）$^{1/(x+SK)}$ 是 VRF 值，而 π（ x） = $g^{1/(x+SK)}$ 是其正确性证明。可以通过抗冲突的哈希函数先将不限制精度的随机等待时间转换为字符串。

$$F_{SK}(x) = e(g, g)^{1/(x+SK)} \text{ and } p_{SK}(x) = g^{1/(x+SK)},$$

其中 e (·, ·)是双线性映射。如要验证 FSK (x)是否计算正确，可以检查以下等式是否成立

$$e[g^x P K, p_{SK}(x)] = e(g, g) \text{ and } e[g, P_{SK}(x)] = F_{SK}(x).$$

对安全性的证明依赖于一个新的决定性双线性“迪菲-赫尔曼”反演假设，其将给定的(g, g^x, ..., $g^{(xq)}$, R)作为输入值来区分 R = e（g，g）$^{1/x}$ 与随机值。

尽管相互发送指令的时间间隔是随机的，但都可以进行跨网验证。它提供了一个可控、可验证和行之有效的签名方案，使得发送指令都可以通过系统的自主调节。

3.2 开户费

对群体攻击的另一个防护是要求一定数量的开户费用，换言之，只有当余额触及到阈值时，地址才能被有效激活并进行正常交易。对于普通用户来说，这种费用几乎可以忽略不计。但要想创建数百万个账户并发送交易冲击网络，代价将会非常昂贵。

3.3 分布式系统中的时间

网络时间的准确性和一致性使得评估分布式系统中的时间这一重要的概念成为了可能。网络时间协议（NTP）是在数据网络潜伏时间可变的计算机系统之间通过分组交换进行时钟同步的一个网络协议。自 1985 年前投入使用至今，NTP 是目前仍在使用的最古老的互联网协议之一。NTP 由特拉华大学的 David L. Mills 设计，提供了一个评估系统成分的时间相关性框架。

3.4 智能合约

智能合约是一种规范合约的特殊协议，也是构架、验证或促进协商以及执行合约条款的特殊协议。智能合约允许在没有第三方（担保）的情况下进行可信交易，这些交易是可追踪和不可逆转的。智能合约包含有关合约条款的所有信息，并自动执行所有设想的操作。计算机科学家和密码学家尼克 萨博早在 1994 年就提出了这一个概念[12]。

其主要原理可以与自动售货机的工作类似，它们都只自动执行所接收到的指令。首先，资产和合约条款被编码并放入区块链，智能合约在平台的节点之间被同步传送。合约生效后，将按照合约条款执行智能合约，而程序将自动检查这些合约的执行情况。

智能合约通常有以下部分组成：

- 合约主体：程序必须能够访问合约项下的产品或服务，才能自动锁定和解锁它们。
- 数字签名：所有参与方必须通过私钥验证才能启动合约。
- 合约条款：智能合约的条款采取了一系列精确操作的形式。所有参与方必须签署这些条款。
- 去中心化平台：智能合约被部署至该区块链平台上，并分布在各个节点之间。

在 JURA 中，智能合约的执行是简单自然的。与其他 DAG 结构不同，可在 Fusus 数据中轻松地整理相关交易的顺序。每次账户创建智能合约时，Fusus 都会创建一个自治账户，让其充当智能合约。我们将来自外部账户的合约呼叫或消息呼叫视为来自这些账户的“发送交易”。智能合约将这些交易识别为“接收交易”。同样，智能合约将该合约响应视为发送交易，并形成了发送链。如果智能合约对接收交易的顺序无要求，接收交易将简单地附加到其发送链。否则，对于带有顺序的“接收交易”，每次智能合约监测到“接收交易”时，就会立即发送给自己。因此，该顺序属性被保存在具有总体顺序的发送链中。

WebAssembly（WASM）虚拟机

我们将使用 WebAssembly（WASM）虚拟机来启用 JURA。WASM 被广泛认为是一种比 EVM（以太坊虚拟机）更快、更全面的解决方案，甚至连以太坊也在致力于 WASM 的实施，以太坊目前使用的是一款被称为“以太坊虚拟机（EVM）”的专有虚拟机。以太坊的其他竞争对手（诸如 EOS 和 Dfinity）也将会使用 WASM 来启用 JURA。WASM 具有以下优点：

- 速度和性能提升
- 支持 C、C ++和 Rust，而用于其他语言的编译器也正在开发中

这意味着已经具有这些语言经验的开发人员可以快速开始构建 JURA，而不必学习像 Solidity 这样的新语言来创建 dApp 和智能合约。此外，这也意味着开发人员在 JURA 上构建时可以利用已经为这些语言构建出的各种工具和软件库。最后，使用 WASM 将提供优越的优化和调试工具。所有这些特性将有助于加速和简化开发流程。

4 动态监控与分布式分片(DMDS)

DMDS 是一种通用型数据库分区，它将大型数据库分成更小、更快、更便于管理的数据碎片。Fusus 分片的概念也与之类似。由于空间存储问题，我们希望将单个 Fusus 分成多个子部分，而它们被称为 Fusus 分片。每个分片包含不同的子 Fusus，它们通常并行运转，主要原因有二：

1. 随着系统的完善，将会有越来越多的客户参与进来。随着信息量的日益庞大，Fusus 图中的每个节点将很难存储所有交易历史记录。

2. 随着流量的增加，压缩频率和新源代码的产生将显著增加，这将导致高昂的时间成本和维护成本。

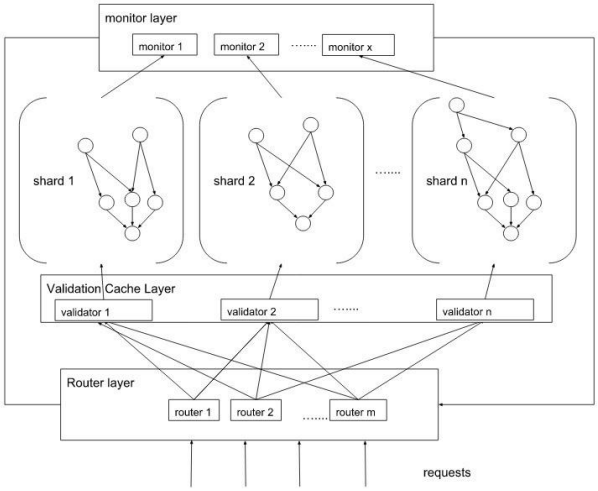


图 6：DMDS 的结构

monitor layer	监控层
monitor 1	监控器 1
Validation Cache Layer	验证缓存层
validator 1	验证器 1
Router layer	路由器层
router 1	路由器 1
shard 2	分片 2
requests	请求

4.1 路由器层

我们构建了一个包含 m 台路由器的路由器层。当收到请求时，我们将进行循环调度，以选择一台路由器将此请求重定向到相应分片。每个路由器包含两个相同的哈希图，如下图所示：

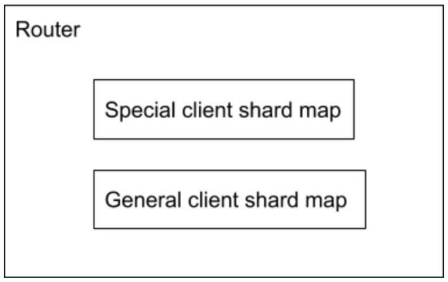


图 7：路由器层的图示

Special client shard map	特殊客户端分片映射
General client shard map	一般客户端分片映射
Router	路由器

特殊客户端分片映射的关键是特殊客户端分片密钥，其值为真实分片地址。当收到请求时，我们检查此请求的客户端分片密钥是否在此映射中。如果它在此映射中，我们将该请求直接路由至分片。否则，该请求将被传递给一般客户端分片映射。一般客户端分片映射的关键是分片索引，其值为真实分片地址。

shardIndex = hash（分片密钥） mod（分片数）*

分片密钥可以识别交易请求所属。分片密钥有多种选择，例如：“交易 ID”或“地址”。

当我们使用交易 ID 进行分片时，我们将交易 ID 的散列空间划分为多个范围，并将每个范围映射到某个分片。这样，每个分片中的节点将只包含 n_{tx} / n_{shd}，其中 n_{tx} 是交易数量，n_{shd} 是分片数量。然而，本例的主要问题是双花。随着流量的增长，如果冲突发生在不同的分片中，该分片中的节点必须接触所有分片方可确认过于高昂的交易。

为了确保双花检测能够行之有效，我们可以将自己的地址作为分片密钥。这样，来自同一地址散列范围的所有交易将落入同一分片中。因此，我们无需接触其他分片。所有双花检测将在分片内部正常进行。

4.2 验证缓存层和交易泛滥防护器

一个请求通过路由器层后，将进入验证缓存层。每个分片都有一个验证器来校验请求中是否包含正确的信息，以避免错误的请求。

然而，偶尔会发生交易泛滥冲击，即攻击者可能会发送尽可能多的有效交易以使网络饱和。通常，攻击者会将交易发送到他们控制的其他账户，这样交易就可以无限期持续下去。普通验证器无法识别有效交易的意图。因此，我们向每个验证器添加一个缓存模块。当请求中的地址散列位于缓存中时，我们将拒绝这个请求，否则我们将把它定向到分片。

我们使用一个 LRU（稍早使用）算法来设计每个缓存。在这个过程中，我们会保持一个队列，该队列将包含时间窗口 T（最近 T 秒）中的所有交易请求。该队列中的每个节点都是成对的（地址散列，使用相同地址散列发出请求的最后一个时间戳访问该验证服务器）。然后，我们每秒检查一次该队列的头指针，并删除最近使用次数最少的节点（如果其已经过期）（即使用相同地址的最后一个时间戳请求已经访问了这个验证器，且位于当前时间戳 - 时间窗口 T 之前）。

当新请求传入时，我们将检查该地址散列是否已经在队列中。如果不是，我们将创建一个仅包含该“对”的新节点，并将其添加到队列尾部。如果地址散列已经在队列中，我们将把它移到尾部，并更新该对中的时间戳。例如，如果我们的时间窗 T 是 3 秒：

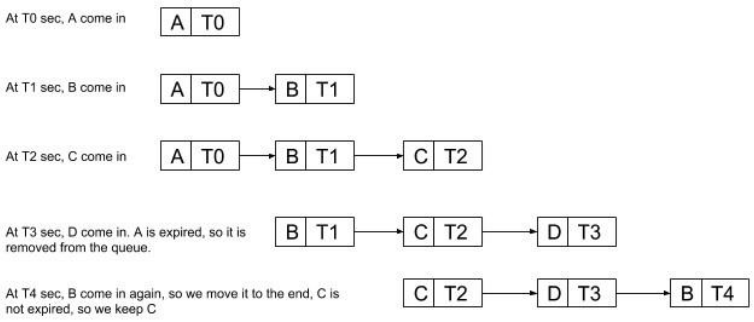


图 8：LRU（最近最少使用）算法

At T0 sec, A come in	在 T0 秒时，A 传入
At T1 sec, B come in	在 T1 秒时，B 传入
At T2 sec, C come in	在 T2 秒时，C 传入
At T3 sec, D come in, A is expired, so it is removed from the queue.	在 T3 秒时，D 传入，A 已过期，所以将它从队列中删除。
At T4 sec, B come in again, so we move it to the end, C is not expired, so we keep C	在 T4 秒时，B 再次传入，所以我们把它移到末尾，C 没有过期，所以我们保留 C

很多请求可能会同时到达验证器，但是由于我们只保留非常小的地址散列和时间戳，该队列仍可存储于缓存或系统空间。

这个 LRU 缓存将能够处理交易泛滥冲击的问题。当这种攻击发生时，只有第一个请求会被接受，其他请求会被拒绝并等待 T 秒。

4.3 监控层

监控层包含多个监控器，每个监控器被定义为同一分片中的一组协调器节点。每个监控器将监督该分片的一般信息。有时，即使我们将地址用作分片密钥来处理分片交易请求，但单个节点使用的存储空间和 CPU 仍然不足，因为这些请求很难被完全分离。有些地址由于其固有属性而比其他地址更加活跃。对此，我们将添加监控层来解决这个问题。需要压缩常规节点以将其选为协调器，而协调器根据整个分片组的买入价获得奖励。每个监控服务器分为三种型号：

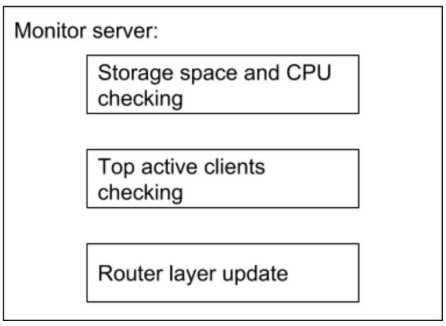


图 9：监控服务器的图示

Monitor server:	监控服务器：
Storage space and CPU checking	存储空间和 CPU 检查
Top active clients checking	最活跃客户端检查
Router layer update	路由器层更新

存储空间和 CPU 检测模型将检查当前用于节点的分片存储空间和 CPU 资源。如果其超过阈值，我们将触发压缩过程并构建新的源代码，并找到最活跃客户端的地址。然后，系统会将他们的地址添加到路由器层中的特殊客户端分片映射中，用于路由器更新。更新完路由器层后，这些特殊客户端将被路由到流量更少、账户地址更少的新备用分片中。由于这个分片中的流量和用户远远少于正常分片，我们可以通过每次定期压缩来节省存储空间，以便承受那些备用分片中的 Penny - spend 攻击。

4.4 实际问题

在网络中使用分片的实践中可能会出现几个关键问题：跨分片通信、完整节点选择和分片数量不平衡。我们将在下面的小节中逐一讨论。

跨分片通信

即使来自发送地址的交易总是被定向到同一分片，接收端的目的地也不一定在同一分片中。这需要路由器层找到接收者账户的分片，向接收者账户发送接收请求，并在另一个分片上更新接收者账户。如果更新成功，然后同步到发送人账户。否则，就将错误返回给用户，并要求用户稍后重试。

全节点选择

全节点选择一开始就要固定，由监控层控制。当监控层发现有一些全节点关闭时，就会自动将新的全节点添加到该分片中。

分片数量

分片数量一开始就要固定，并由监控层控制。当逾半的全节点发现流量超过了他们能承受的限制时，我们可以创建一个新分片，通过把原分片的散列范围对半分，将一半的请求重定向到新分片中。

5 人工智能

人工智能不是一个新概念，早在 20 世纪 90 年代就已被广泛讨论，最近由于新的互联网设备（如手机和特定网站）产生的海量数据而再度成为热门话题。人工智能在 JURA 中有主要有两方面应用：

1. 交易过滤：通过离线学习的方式研究和学习交易特征，并在收到交易后将交易特征将作为第一层发布。基于交易数据特征的模式识别技术，如时间戳、余额等，可以用来推断交易的异常性。如果交易可疑，将会发布该地址的软锁，并向发送人地址发出通知。如果通知被确认，软锁就会打开，否则，就会变成硬锁，并采取更严格的调查行动。
2. 恶意节点检测。系统级节点级信息可以被记录下来，并用离线方式对其进行再次研究。这可以进一步研究看似为异常值或极端值的节点，以防止其恶意攻击。

6 攻击途径

JURA 与所有去中心化的加密货币一样，可能会受到牟取经济利益而导致系统崩溃的恶意用户攻击。在本节中，我们概述了一些可能的攻击场景和后果以及 JURA 如何采取预防措施。

6.1 垄断问题

如果单个实体（以下称垄断者）控制诸如传统 PoS 系统中投入大量的资源，它就可以利用这些资源对其余网络施加影响。垄断者可能会通过恶意的方式，比如双花或者拒绝服务。如果垄断者选择恶意策略，并长期保持这种控制，那么人们对加密货币的信心将会被削弱，货币购买力就会暴跌。即使垄断者“高风亮节”，但系统始终是中心化的的，由单方面掌控。

PoU 系统则使垄断形成变得极其困难（除了 PoS 系统对此问题作出的争论外），如收购 50 %以上权益的高昂成本，若存在恶意行为，则不会有经济利益等。PoU 让垄断的界定完全模糊，没有人知道，甚至连核心开发者也不知道如何拥有超过 50 %的表决权。

每个单一实体的表决权都是在群体中博弈论方法中许多因素共同作用的结果，通过 CDF 从权益规模到边际效用的非线性转换消除了在任何方面占据主导地位的可能性。这类似异常值对平均值和中值的影响。异常值很大程度上可以改变平均值，但无论异常值有多大，它都不会改变中间值。

6.2 块隙同步

交易无法正确传送，可能导致网络忽略后续的区块。如果一个节点观察到一个没有引用前一个区块的区块，那么它有两个选项：

1. 忽略该区块，因为它可能是恶意垃圾块。
2. 请求与另一个节点重新同步。

在重新同步的情况下，必须要引导节点形成 TCP 连接，以增加重新同步所需的通信量。然而，如果该区块实际是一个不良区块，则无法重新同步，也不能网络通信量。因为不良块是一个网络放大攻击，并可能导致拒绝服务。

为了避免不必要的重新同步，节点将会等待直到观察到潜在恶意块的某个投票阈值后才启动连接引导节点进行同步。如果一个区块没有得到足够的票数，就有可能被认为是垃圾数据。

6.3 双花

双花可以通过多种方式产生：一种方式是由用户故意向两名接收者发送相同的余额。在这种情况下，由于 Fusus 的设计，双花攻击基本上得以幸免。第二种方式是发送操作。一旦进行了第一次发送操作，余额将被搁置，新的发送操作将无法两次支付相同的余额。不同于防止双花的区块链验证确认模式。每当发送块被激活时，它必须基于前一发送块的结束余额或所有未决的前一接收块的总数来发送。考虑到上述发送块的时间限制，第二发送块必须具有账户的更新余额，该余额必须能够表明余额是否足以支付付款。因此，Fusus 可以防止双花的发生，换言之，一旦用户向另一个地址发送了一定数量的余额，就不能同时向另一个地址发送完全相同的余额，除非第一笔余额没有通过，或者余额随着接收块的增加而增加，这样就有了足够的余额。

由于编程错误和网络环境差，双花也有可能发生。如果不阻止这种情况，验证者会进一步加剧网络冲突。自表决权对每个人隐藏以来，并伴随随着群体的动态变化，否则，没有任何一方会有信心和动力以其他方式去投票。此外，如果进行了不诚实投票，将会受到惩罚。

6.4 女巫攻击

在出现**女巫**攻击的这种情况下，一个实体可以创建多个账户，其目的是获得不对称的表决权。但是，非零开户费用需求使得这种攻击在起初成本就会很高，此除之外，PoU 共识机制以 CST、AIST 和 LIST 指标作为权重来贴现新的权益。由于经济成本很高，而且系统规则阻止这种行为的发生，因此**女巫**攻击成功的机会可能会很小。

6.5 Penny - spend 攻击

Nano 和 IOTA 网络都在一个节点上遭受了大量零价值的垃圾邮件攻击，这类攻击也属于 Penny - spend 攻击的范畴。在 JURA 网络中，由于 VRF 机制方法（而不是 PoW 反垃圾邮件），最初不可能会发生垃圾邮件攻击。即使发生这种攻击，对于接收方而言，Fusus 也有能力快速吸收和消耗大量发送的区块，因此不会发生网络拥塞。

具体而言，发送块所需的 PoVRT 只会使得发布不合理数量的交易变得越来越难，即自上次发送操作后第一次发送操作或者阈值有零时间消逝要求，然而，立即发送动作将被迫由 VRF 分开。同样，发送区块和 PoVRT 的基本机制可以防止这类攻击。

7 通证经济

对于普通用户来说，像 JURA 这样的免费系统应尽量避免交易摩擦的同时能够显著提高交易效率。为了更好地向更多人普及这项应用，它还通过降低 3 % - 5 %的交易费用的方式为小商户分享由网络带来的收益。

对于提供资源维持系统运行的节点，有两层激励：一层是隐性的和隐藏的，另一层是明确的，要求节点向系统提供某些功能。

含蓄地说，基于该技术的业务运营就是运行全节点支持网络并满足商业需求。运行全节点的成本低，每月大约 3 美元，但使用传统付费方式（Visa，Mastercard，Amex）的小型企业将要按照所有销售额的 3% - 5%进行支付，这一费用远高于运行一个 JURA 的全节点。

明确地说，奖励的直接来源可以从投注系统的罚款中获得，也就是说如果投注过程中的参与者没有做好投票工作，那么权益将被收回并作为奖励分配给全节点。此外，更多奖励只对全节点的运营者开放，他们至少可以担任三个主要角色中的其中一个：审判者，存储器和协调器。我们将在以下小节中介绍每个角色的详细信息。

7.1 审判者

审判者是 PoU 系统为了验证交易和解决冲突所充当的一个角色。定期选择少数利益相关者组成委员会，来解决像交易冲突、协议变更等诸如此类的问题。他们会因为做出正确的决策而获得奖励，也会因为以权谋私而受到惩罚。

PoU 系统每隔 1 小时就会选择一些高效权益者充当审判者，委员会的规模应该是一个相对较大的奇数，比如 101。其基本原理是保护委员会免受拜占庭式错误的影响，使投票结果更加稳定。

鉴于 PoU 机制的性质，我们希望挑选出来的 101 名审判者首先是善意的。像 101 这样的数字应该足以维持系统的一致性。

每隔 1 小时，这些审判者就会得到相应报酬。PoU 机制确保了本身不是一个让“让富人更富”的系统，而是一个“因高效而更富”的系统。

这种机制为参与投注的所有用户创造了强大的动力。

7.2 存储器

存储器是指可以存储大型非结构化数据（图像、音频、视频）的存储实体。存储器可以帮助降低网页和物联网应用程序存储内容的成本和复杂性。与 FileCoin 不同的是，JURA 可以在没有潜在诱导因素的情况下快速存储和检索数据，这是 Filecoin 为实现复制完整性而做出的努力。此外，我们不并会为检索市场服务，我们希望执行像函数之类的 CDN （内容传递网络）。而与家庭存储设备相比，我们的存储器组通常由一组选定的具有高网络带宽连通性的全节点组成。我们网络上的存储器会根据每一周期添加的数据量获得相应通证奖励。数据均匀分布在存储器中。

7.3 协调器

在之前介绍的分片架构监控层中，全节点将通过定期计算、通信和偶尔警报的方式来持续监控碎片信息。提供这种功能的节点称之为协调器。他们还将定期根据对协调器所做的全碎片贡献来评估应获奖励。

7.4 奖励池

奖励池将随着时间的推移而逐渐减少，因为通胀协议规定了一段时间内发放的通证数量。随着应用程序数量的增加，奖励池将更多地应用于审判者、存储器、协调器。这种方式可能会提升奖励的通胀率。为了网络和矿工的共同利益，通胀协议将根据激励实体的出价率来设定回报大小，且该比率可以每天随通证的变化而变化。

8 结论

本文提出了一种具有超高 TPS 和免费的去中心化的对等网络系统。JURA 协议围绕一种新的数据结构 Fusus 而展开，从而奠定了坚实的处理基础设施，并随 PoU 系统一起获得了进一步的增强，这种 PoU 系统使可靠的自调整共识成为可能。此外，人工智能增加了一层额外的防护，分片提供了范围更大的可伸缩性。因此，该网络在结构清晰性和统计管理模式方面极其稳健。我们也深信，JURA 协议的技术先进性将使其在各个领域里得到广泛的应用。

参考文献

[1] 中本聪（2008）“比特币：一种点对点电子现金系统”。

[2] V. Buterin.（2013）““与维塔利克 • 布特林共建 在线合约”。*Eyerys.com*.

[3]S. Popov（2016）“The tangle（缠结）”

[4]C. LeMahieu.（2017）“Nano ：一个免费的分布式加密货币网络”。

[5]Y. Dodis； A. Yampolskiy（2005）“具有更短密钥与证据的可验证随机函数”。*第八届国际公钥密码学理论和实践研讨会*。第 416 - 431 页。

[6]B. Pugh.（2018）“RaiBlocks 网络影响因素试验：第 II 部分”。*https://medium.com/@bnp117/stress - testing - the - raiblocks - network - part - ii - def83653b21f*.

[7]S. Micali； M. Rabin； S. Vadhan.（1999）“可验证随机函数”。*第四十届 IEEE 计算机科学基础研讨会纪要*。第 120 - 130 页。

[8] S. King；S. Nadal. (2012 年）。"点点币：基于权益证明的点对点加密货币"。

[9]D. Pike；P. Nosker；D. Boehm；D. Grisham；S. Woods；J. Marston.（2015）“一个在非线性分布式共识中的可接受时间的周期证明因子”。

[10]D. Boneh； B. Lynn； H. Shacham.（2001）“韦伊配对的短签名”。第七届国际密码学和信息安全理论与应用会议纪要：密码学进展，ASIACRYPT'01，第 514 - 532 页，英国，伦敦。施普林格（科技出版社）.

[11]A. Justel；D. Pea；R. Zamar.（1997）“一个多变量拟合优度 Kolmogorov - Smirnov 检验”。统计学与概率论杂志，**35**（3）:251 - 259.

[12]N. 萨博.“智能合约：数字市场的基石”。于 2017 年 7 月 29 日检索 www.fon.hum.uva.nl.。

[13] https://github.com/ethereum/casper