Coordicide（去协调器）白皮书

IOTA基金会 Coordicide团队

2019.05

IOTA 中国社区（IOTAChina.COM）

2019 年 9 月翻译

目录

致谢

参与本白皮书中文版翻译、校对和讨论交流的各位社区成员，他们是：Vivian，Proof.K，黄俊毓，Stanley，Jodel，WanSheng，Erica，刘国威（tigermumu），大熊，Ajwechat，DAG007，JIMMY（熊志敏）。

IOTA 中国社区

IOTAChina.COM

2019.09

词汇表

拜占庭节点

分布式系统中试图故意破坏系统运行的参与者，例如，通过将消息不转发给其他参与者来破坏系统。

共识

指如何在分布式多代理系统中对存在分歧的特定数据或值达成一致的问题。

协调器

发布里程碑的可信实体，以保证最终结果并保护缠结(Tangle)免受攻击。

字典攻击

一种暴力攻击技术，通过不断尝试无数种可能性（类似于查字典），来找出密码以破坏认证机制。

日食攻击

一种网络攻击方式，旨在隔离并攻击特定节点，而不是整个网络。

创世交易

在缠结(Tangle)中生成的第一笔交易。

心跳

由硬件或软件生成的周期性信号，用于指示正常操作或同步计算机系统的其他部分。

历史

被给定交易直接或间接验证的交易记录。

里程碑

协调器发布的特殊里程碑交易。里程碑的属性将在 2.1 节中详细讨论。

邻居节点

在网络中相连的节点。

节点

作为IOTA 网络中的机器，它的作用是发布交易并验证现有交易。

对等连接

发现并连接其他网络节点的过程。

工作量证明

一种通过高成本，耗时久的方式来产生数据以满足某些要求，而其他节点可以快捷方便的验证。

彩虹表攻击

预先计算生成的表，用于反推攻击加密哈希函数。

无规行走

一种数学方法，描述由某些数学空间上的一系列随机步骤所组成的路径。

盐

一种随机数，被用于单向哈希数据的附加输入。

社交工程

以欺诈为目的，并使用不正当手段操纵个体泄露机密或个人信息。

女巫攻击

女巫攻击是指试图通过伪造多个虚假身份来控制对等网络。

Tip

尚未被验证的交易。

交易

在两个节点之间传输价值或数据的信息。如果某交易的完整历史为已知状态，则该交易是可靠的。

缩写

ASIC	Application-Specific Integrated Circuit. 特定用途的集成电路
CA	Cellular Automata. 元胞自动机
CLIRI	Coo-Less IOTA Reference Implementation. 无协调器的 IRI（IOTA 参考实现）
DAG	Directed Acyclic Graph. 有向无环图
DLT	Distributed Ledger Technology. 分布式账本技术
IF	IOTA Foundation. IOTA 基金会
IRI	IOTA Reference Implementation. IOTA 的参考实现
PoW	Proof-of-Work. 工作量证明
TSA	Tip Selection Algorithm. Tip 选择算法
VDF	Verifiable Delay Function 可验证延迟函数

1 前言

IOTA 的愿景旨在通过安全的零交易费和数据传输系统，为物联网和未来的互联网建立一个实时经济体系。实现这一愿景，是为了解决当前各种分布式账本技术(DLT)普遍存在的一系列问题：首先，IOTA 需要有比区块链明显更高的吞吐量，而区块链的内在瓶颈使得交易记录聚合成链式数据；其次，交易手续费在小额交易中是一种障碍，但这种手续费在基于 PoW，同时包含用户和矿工的 DLT 中却是必要的。与这些现存的 DLT 不同的是，IOTA 使用一种叫做有向无环图（DAG）的结构，如同 IOTA 白皮书[30]中所述，它在理论上可以达到无上限的数据吞吐量[1]。此外，每个网络节点都能够发布和批准交易，这种模型使得 IOTA 可以去除区块链架构中的交易手续费，从而促进小额支付网络系统的构建[31]。

早期 DLT 项目的一个常见问题是，网络不够健壮，不足以支撑它依赖的安全机制按照设计的方式运行，因为它们的安全机制需要假定网络已经足够成熟。因此，DLT 通常在一开始就在外部采取各种“引导”措施，确保网络安全成长到成熟阶段[2]。在 IOTA 的当前实现中，同样也是依靠一个中心化的协调器(COO)来提供安全性，以消除不诚实节点破坏尚未成熟网络的风险。在这个阶段，IOTA 对共识的定义是，要求每一笔交易都被协调器签署的交易(直接或间接)所确认。换而言之，协调器可以被视为“终极裁决设备”。

我们认为基于中本聪共识机制的加密货币体系的愿景，可以通过改变网络中关于多数算力的基本假设(即最长链机制)来进行改进。在 IOTA 当前的实现中，由于协调器的存在，则无需诚实节点来构成网络的主要算力。但这只是一个临时的举措，IOTA 的愿景是超越传统区块链的网络共识机制。本文的目的即在于，论述 Coordicide 项目是如何通过一组相互配合的网络组件，来实现协调器的移除工作。这些组件的添加，不会对缠结(Tangle)网络现存的基础功能特性造成影响。

根据 IOTA 基金会作为一个非营利组织的章程，我们研究部门的目标包含：透明度，协作和社区参与。我们计划开放式地展现研究工作，以期获取学术界和社区志愿者的反馈意见。有一点需要注意的是：由于我们的研究对象是高度动态

[1] 实际吞吐量受硬件和物理定律的限制
[2] 例如，比特币过去曾使用检查点。链接：https://en.bitcoin.it/ wiki/Checkpoint_Lockin

的，因此需要对各种提案进行模拟和测试，然后才能在主网上对这些组件进行部署。我们强调，这里展现的一些提案，依然还在研究当中，没有完全定型。当我们的研究取得进展，进行模拟的时候，这部分提案是有可能被修改的。

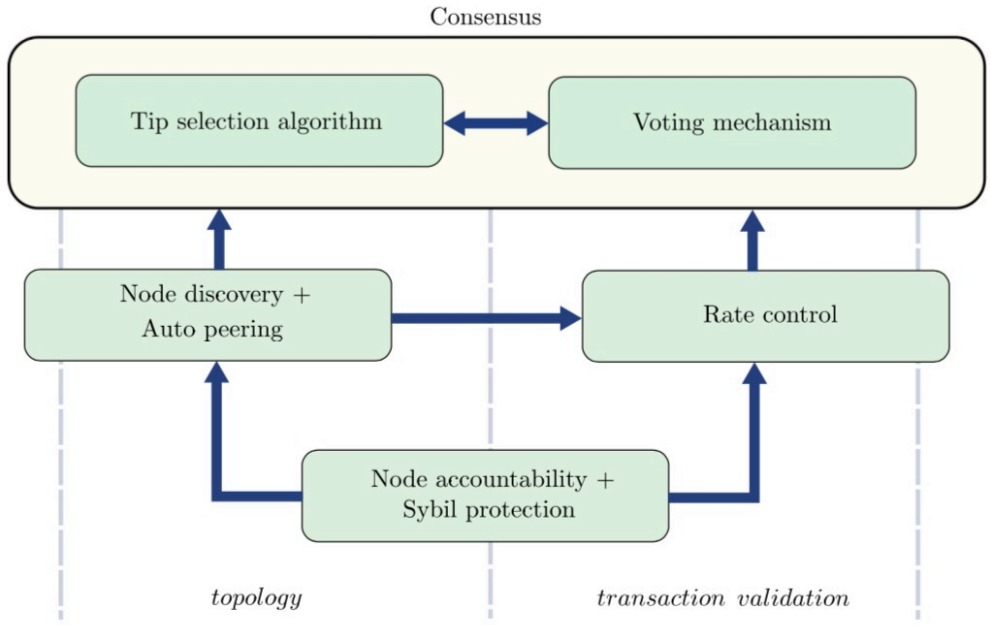


图 1：Coordicide 模块之间的连接图

为了去除协调器，需要解决一系列的挑战性问题。本文中涵盖的这些问题，可以用图 1 中的模块来表示。接下来，我们对 Coordicide 的当前状态和未来的研究方向，给出一个简要的概述：

- 节点责任。在第 3 节中，我们提出全局节点 ID 的概念，并且描述了一种在不需要节点拥有者去冒险或者展示资产的情况下，能够防范女巫攻击的保护机制。对发布消息的节点进行识别，是执行特定网络拓扑(通过节点自动互联)或者惩罚不良行为(通过速率控制)的基础。
- 节点自动互联。每个分布式系统都需要一个能够发现，并且可靠地连接到邻居的自动化过程。在第 4 节中，我们将会讨论在缠结(Tangle)网络中的节点自动互联机制。

- 速率控制。为了确保网络不超过其容量，在第 5 节中，我们引入一种机制来控制通过网络传播交易的速率。这种方法可以根据发布节点的统计信息，选择性地过滤掉一些交易。

- 共识机制。基于以上的模块，我们最终得到了第 6 节所描述的扩展网络共识框架。这个框架由两个部分组成：首先，我们描述了 tip 选择算法的当前研究成果(第 6.1 节)；然后，为了主动解决某些冲突，我们描述了两种不同投票机制，使得节点之间可以找出他们对于网络当前状态的意见。

2 最新技术情况

在介绍中，我们提到当前的 IOTA 的主网采用协调器（Coordinator）来达到共识，更通俗的来说，是为了保证网络的安全。然而，该中心化的协调器应仅仅被认作是一个必要的引导（bootstrapping ）机制，而不是一个长期的解决方案。在这个章节中，我们首先讨论通过 IRI（IOTA reference implementation）软件实现 IOTA 当前的主网状态[3]，然后介绍当我们按照 IOTA 白皮书构建一个 Coo-less 网络所面临的挑战（即一个没有协调者（Coordinator）的网络）[30]。由于在当前协调器（Coordinator）与其里程碑已深深内置 IRI，因此去除这些依赖，不仅仅意味着对软件进行全面地更改，而且还会导致新的研究问题。

2.1 当前的 IOTA 实现

根据 IRI 软件实现当前的 IOTA 主网，其中协调器（Coordinator）起到重要的作用。在下文中，我们介绍在当前主网中主要实现的任务，并不是所有的任务都与共识密切相关：

- 手动添加节点（Manual peering）。为了加入缠结(Tangle)，需要一个节点连接到一些现有的节点（peering）。当前 IRI 软件仅允许手动添加节点（peering），即节点需要手动寻找其他 Tangle 节点的地址。节点互连（peering）是传播/广播交易与同步当前账本状态的基础。至于后者，里程碑是判定两个节点的

[3] https://github.com/iotaledger/iri

是否已经同步的有用锚点：如果一个节点的最新可靠里程碑低于它的节点中的里程碑，那么这个节点可能是落后了。

- 速率控制机制（Rate control mechanism）。为了发布交易，节点必须解决加密难题（工作量证明机制）。必须保证该节点不会在在网络中任意地发送垃圾交易（spam），或者避免它们注入超过网络所能处理的交易。

- Tip 选择策略 (Tip selection strategy)。验证交易是建构 Tangle 的 DAG 结构的基本过程。要验证交易，节点必须验证以确保不会向账本引入不一致性。尽管不可能强制对哪笔交易进行验证，但 IOTA 的白皮书提出了一种基于无规行走（random walk）的 tip 选择算法：(i)阻止懒惰行为，而鼓励验证新的 tip；(ii)持续将小分支合并为一个单一的大分支，从而提高确认速率；(iii)如果发生冲突，仅会保留其中一个分支，其他所有分支都会被抛弃。

- 共识 (Consensus)。里程碑的主要作用是确定共识。Tangle 运用一个简单的规则：当且仅当一笔交易被里程碑所引用，则该交易得到确认。在 IRI 中，它体现在 getBalances 与 getInclusionStates 这两个 API 中，它们分别显示账户有多少代币以及交易是否被确认。

此外，我们还想强调里程碑用来优化 IRI 代码：例如，不是从创世交易开始计算全账本状态，而是为每个里程碑都保存了一个中间状态；类似地，里程碑用于进行本地快照，即 IRI 修剪机制，其允许节点避免存储 Tangle 中较早期的数据。

2.2 无协调器的(Coo-less)IOTA 网络

作为无协调器网络的初步实施，我们正在构建 CLIRI，它代表了无协调器的 IRI。它的核心是 IRI 的一个分支，删除了所有与协调器相关的组件。CLIRI 的主要目的是提供一个工作测试平台，运行第一个无协调器的 IOTA 网络，在该平台上我们可以模拟各种 Coordicide 提案。这是理解无协调器的主网必将面临挑战的必要第一步。

如上所讨论，协调器在目前的 IOTA 实现中起到至关重要的作用。因此，构建 CLIRI 会带来许多挑战。对于它的第一次迭代，我们在可能的情况下，遵循最初的 IOTA 白皮书[30]，我们选择启发式算法及简化模型：

- 账本确认。由于重写无里程碑账本的计算逻辑是一个重要的事情，CLIRI 第一个迭代仅支持零值交易。
- 本地快照。我们在 CLIRI 上取消了本地快照模块，并且我们在每周自动删除整个数据库。
- 无规行走起点（random walk starting point）。CLIRI 随机选择一个 tip，然后回溯到过去达到“足够远”的交易。

目前 CLIRI 处于早期的开发阶段，在 2019 年 5 月 5 日推出了第一个测试网络。

2.3 新的研究挑战

CLIRI 除了上述的"工程"选择之外，它的逻辑是基于 IOTA 白皮书的，因此它具有相同的模型假设。其中最重要的是（勤勉的）诚实交易占多数的条件[8]：具体来说，网络可行的话，白皮书上的共识算法要求大多数交易总是来自诚实的网络参与者，即诚实的行为者需要拥有绝大多数的哈希算力并持续产生交易。这意味着诚实的节点需要不断地发送交易，无论实际上他们是否使用网络。此外，实现大多数哈希是昂贵的，否则作恶代理人很容易去购买足够的哈希算力并超越网络。这里存在一个激励问题，此外发布交易还受工作量证明机制（PoW）的限制。由于它的复杂性，慢速的节点将被排除在参与网络之外。

以上问题直接导致以下的研究问题，将在整篇文章中进行研究：

- 速率控制。需要一个更有效的速率控制算法来解决以下问题：如果 PoW 的难度太高，那么小型设备（例如电话或感应器）将会花费过长时间来计算它，因此这将无法发送交易。另一方面，低难度可能会导致网络拥堵，并/或进行垃圾交易攻击。
- 共识。在没有协调器的支持下，在诚实交易是多数的假设中，我们需要一个可靠的共识机制。

在下个章节中，我们将介绍节点身份的概念，它是解决上述研究课题的先决条件。

3 节点责任

在没有协调器的网络中，各种应用程序需要可靠地将交易或其他消息与发布这些交易或消息的节点进行关联。这些应用包括：

- 速率控制：在超负荷的场景中，节点尝试发布超过整个网络能够处理的交易，这时尤其应阻止或惩罚那些交易量贡献最大的节点。
- 基于投票的共识机制：为了防止双重投票，需要将投票与节点权重相关联，实际投票必须与节点 ID 进行关联。

在 3.1 节中，我们提议一种将全局身份与节点相关联的方法。由于这个可能将网络暴露于潜在的女巫攻击，因此在 3.2 节中，我们引入了 *mana*，一种新型的防女巫攻击机制。

3.1 全局节点身份

为了识别节点，有必要引入全局节点身份。为此，我们设想使用普通公钥加密对确定的数据进行签名，并通过防篡改的方式将其关联到其发布节点。此外，我们要求发布节点添加它的公钥到每个签名信息中。这样，节点无需某种形式的全球数据库的 ID 和密钥，就可以校验发布节点的真实性。重要的是要注意，实施这些机制仅仅用来保护通信层及密钥，并且一旦节点开始处理，ID 和签名不需要存储在 Tangle 中。这允许了更好的灵活性，因为可以交换实际签名方案，而对存储的数据没有任何的影响。与之相反的是，相对于存储在 Tangle 中的任何数据，通信层现在无需使用后量子加密算法，但是当量子攻击在未来变得更加流行时可以更换它。

当节点与身份相关联的话，分布式系统容易受到女巫攻击[16]，其中恶意实体伪装成多个伪造身份。这将克服依赖于有限数量的此类身份的任何机制，网络就易于受到协调攻击。 在接下来的章节中将介绍解决这些问题的可行方法。

3.2 女巫攻击保护

让女巫攻击变得更加困难的一种常见方法称为资源测试，每个身份必须证明拥有某些难以获取资源的所有权。因为在加密货币世界中，用户拥有一定数量的某些代币，我们提出基于这些代币所有权的女巫保护机制。然而，无需要求自身所有权的证明，我们允许网络中的每个用户发布交易时，将代币分配给他选择的任何节点。我们称这些代币为 mana；它们既是难以获得的资源，也是某种形式的“名誉 ”，它可以分配给值得信任的节点。实际机制的基本原理如下：

- 当发布交易时，它会产生双重流程：（i）它将数据或代币从一个地址发送到另外一个地址，（ii）并给某些节点分配虚拟代币（称为 *mana*）。*mana* 的数量对应转移的代币数量。
- 必须在交易的签名部分指定应接收 *mana* 的节点 ID。节点在一定的时间后得到 mana。有必要防止节点为它们发出的每条信息生成新 ID。
- 只要实际代币再次转移，从先前引用的节点扣除相应的 *mana*，并可能将其重新分配给新节点。

我们再次强调，该过程不会以任何形式影响到实际余额，但它仅用来为“受信任”节点赋予更高的权重。

人们可以委托的 *mana* 数量取决于他们拥有的多少代币，这意味着在该过程中，持有更多代币的人有更大的影响力。特别的是，节点不需要在它们自己的网络中质押很多代币的，它就能积累许多 *mana*。在传统权益证明共识的女巫保护机制中，每个节点需要证明它有一定数量的抵押品。相反，委托 *mana* 带来一些关键优势：因为 *mana* 作为常规交易的一部分，节点不需要经常使用它们账户的私钥进行签名，而那就会带来严重的安全风险；此外，该方法不需要激励节点运营商拥有或申报大量代币；最后，用户可以向节点发放额外的 *mana*，以便为社区提供好的服务。

因为我们现在已经建立了可信的节点身份，我们可以使用这些身份发现和连接到网络中其他的节点。

4 节点自动互连

在 IOTA 中，一个节点是一个存有缠结(Tangle)所有信息的设备。为了让网络更高效地工作，节点之间互相交换信息并且实时更新账本状态。目前，节点需要一个手动配对过程，使之和其它节点成为邻居。然而，手动配对可能容易受到攻击（比如社会工程学攻击）从而改变网络的拓扑结构。为了阻止这些攻击，简化新节点的配置过程，我们引入了一个允许这些节点自动配对的机制。节点运营人员在选择邻居的整个过程中无需手动干涉，这种方法叫自动配对互连。

具体而言，在这个章节中我们提出的自动互连机制可以实现两个重要目标：第一，它为新创立的节点提供基础设施，使其更容易加入网络；第二，我们能够确保在互连过程中，攻击者不能瞄准一些特定的节点，比如，我们能够确保网络能对抗日蚀攻击。

4.1 节点发现

每个节点从一个潜在的互连节点列表中选择它的邻居。在无许可开放式的环境中，这个列表会随着时间变化，因为每时每刻都有节点加入或者离开网络。为了保持列表实时更新，我们假定节点会定期对已知节点列表中的一部分节点与其他节点保持联络。这个机制简单高效，因为它允许每个节点去了解其它的网络参与者。有一点需要指出，这个机制仅需要节点与一个足够大的网络子集相连接，比如潜在互连列表包括“足够”的节点[4]。

4.2 邻居节点选择

节点自己选择一半的邻居，节点的另外一半邻居是让邻居来选择它们。因此这两种不同的邻居分别称作：

- 主动选择的邻居。节点主动从它的列表中选出的邻居。
- 被动接受的邻居。邻居选择这个节点作为互连节点。

[4] 潜在配对节点的数量取决于 Gossip 协议和全网系统参数，比如邻居的数量。

为了从潜在互连节点列表中选出主动选择的邻居，我们通过如下距离方程来测量两个节点的距离 *d*：

$$d(nodeId_1, nodeId_2, \varsigma) = hash(nodeId_1 + \varsigma) \oplus hash(nodeId_2)$$

其中 ς 是公共盐[5]。

为了连接到新的邻居，每个具有 ID *ownId* 和公共盐的节点保存着根据距离 *d*（*ownId,·*, ς）分类的一个潜在互连列表。随后，节点按照这个列表的升序排序方式，发给潜在的互连节点配对请求，请求中含有节点自己的 ID，目前的公共盐和节点的地址（比如，IP+端口）。在这之后，收到请求的节点能够用如下的方式决定接受或者拒绝连接请求。需要配对的节点反覆重复这个过程直到它和足够多的邻居建立连接。这些邻居建立它的主动选择的邻居列表。算法 1 也对这个过程进行了阐述。

算法 1 邻居节点选择

Algorithm 1: Select *chosen neighbors*

Input: desired amount of neighbors k, current list of chosen neighbors $\mathcal{C}$, list of potential peers $\mathcal{P}$

```
P_sorted ← sortByDistanceAsc(P, ownId, ζ)
foreach p ∈ P_sorted do
    peerRequest ← sendPeerRequest(p)
    if peerRequest.accepted then
        append(C, p)
        if |C| > k/2 then
            return
    else
        append(P, peerRequest.proposedCandidates)
        P_sorted ← sortByDistanceAsc(P, ownId, ζ)
```

和之前的情况类似，为了接受邻居，每个带有 ID *ownId* 的节点必须生成一个私有盐 ς*，当它接收到来自一个带有 ID *remoteId* 节点的配对请求时，它测量 *d*（*ownId, remoteId, ς**），仅接受满足至少以下一个要求的请求：

[5] 密码学中，盐被定义为可用于防御字典攻击，或抵抗它们等效的哈希，即预编译的彩虹表攻击。

- 连接的节点比目前已存在的被动邻居节点更近。
- 连接的节点没有足够的邻居。

当不满足上述要求时，这个节点拒绝配对请求，同时它能够使用公共盐得到一个新的潜在节点，并加入到新的潜在互连列表中。算法 2 用形式化语言描述了这个过程。

算法 2 可接受邻居的过滤选择

Algorithm 2: Filter *accepted neighbors*

Input: incoming peering request r, desired amount of neighbors k, current list of accepted neighbors $\mathcal{A}$

```
if |A| < k/2 then
    accept(r)
else
    distance_r ← distance(ownId, r.nodeId, ζ*)
    foreach a ∈ A do
        distance_a ← distance(ownId, a.nodeId, ζ*)
        if distance_r < distance_a then
            accept(r)
            drop(a)
            return
    reject(r)
```

4.3 网络重组

我们设想通过公共和私有盐帮助产生不对称的网络视角来阻止攻击者破坏系统。事实上，瞄准一个节点的唯一方式是在自动互连过程中是通过暴力破解不同的节点身份和期望获得比已有邻居更近的距离（距离 *d*）。为了阻止暴力破解成功，我们让盐仅在特定时间范围内有效，在这之后节点更新主动选择的邻居和被动接受的邻居[6]。

另外一个降低攻击者控制网络拓扑能力的方法是，在产生盐时加入一个“全局随机源”。

[6] 降低攻击者控制网络拓扑的能力的另一种方法是在生成盐时包含“全局随机源”。

这种频繁的重组带来双重好处：首先，它可以防止攻击者影响网络拓扑; 第二，它倾向于想要加入网络的新节点，因为他们的互连请求将被更大的概率接受。

4.4 节点初始化

一个想加入网络的新节点最初对网络是一无所知的。它既不知道账本的状态，也不知道目前谁在网络中。为了让所有新的节点获取第一份含有“其他节点”的列表，我们使用一种可信的“入口节点”硬编码列表，这个列表由 IOTA 基金会或者可信的社区成员运行，从而能够响应新加入节点的互连请求。

这是常见的做法，几乎在所有分布式网络中都以这种方式来处理。

5. 速率控制

每个通信网络的基本目标是要通过限制加入网络的交易速率来控制其节点注入的流量。 事实上，由于恶意行为者的原因，这种流量可能会导致不愉快的情况，例如由于资源限制导致的网络阻塞或垃圾信息：

- 阻塞控制。在大多数网络中，传入的流量负载在有些情况下大于网络可以处理的能力。如果不限制流量的涌入，就会造成网络瓶颈，从而降低整个网络的速度。可以对分布式账本进行相关分析，其中输入的流量（即由网络中节点发布的交易）可以滥用诸如带宽，计算能力或磁盘空间等有限的资源。此外，节点可能会失去彼此之间的同步，有时甚至不会意识到这一点。
- 垃圾信息检测。Gossip 协议（目前用于在 IOTA 网络中进行交易转发）是一种有效且可靠的传播信息的方式。然而，这些协议有一个缺点：它们无法限制垃圾信息的传播。实际上，消息在网络中冗余地分布，并且一小撮节点转发垃圾消息，以致使大多数节点都接收到这些垃圾信息。

  通信网络的速率控制策略在阻塞控制[24]和垃圾信息检测[17]领域都有较为充分的研究。 对于分布式账本技术，PoW 是一种内置的速率控制机制，而不仅仅用于达成共识。 然而，PoW 会带来不良的副作用，例如挖矿竞赛：

较小的通用设备与优化的硬件之间，在 PoW 性能方面存在几个数量级的差异。因此，任何基于 PoW 的速率控制最终都会使得算力较小的设备处于劣势。因此，需要全新的交易速率控制机制来处理 Tangle 网络的全局和每个节点的控制。

5.1 速率控制算法

在纯粹基于 PoW 的体系中，高难度值会阻碍低算力节点发布交易，这不是令人满意的，尤其是在物联网环境中更不适合; 另一方面，低难度值就会很快导致网络阻塞。我们提出了一种自适应 PoW 算法，允许每个节点发布交易，同时惩罚垃圾信息攻击行为。

在我们的算法中，当一个节点决定发布一笔交易时，它必须解决一个加密难题，其中难度是所拥有的 $mana$ 和最近发布的交易数量的函数。假设节点 i 在之前的 $T \gg h$ 个时间单位时生成了 n_i^T 笔交易，其中 h 是网络延迟时间，这意味着，如果在时间 t 时发送消息，则所有在线节点将在时间 $t+h$ 内接收到相同的消息。节点 i 的 PoW 的难度设置为 d_i，其定义为：

$$d_i = d_0 + w\left(s_i, n_i^T\right),$$

其中d_0是基本难度，w 是 $mana$ 值 s_i 和交易数量 n_i^T 的函数。我们基于所期望的全网均衡水平，来设置时间窗口 T，基本难度 d_0 以及函数 w 等参数。

作为一项额外的安全措施，我们要求用户发出的交易总数限制为

$$n_i^T \leq z(s_i), \ \forall_i \ ,（1）$$

其中 $z: \mathcal{R}^+ \to \mathcal{R}^+$ 是一个依赖于 $mana$ 值 s_i 的函数，这样使得节点的 $mana$ 值 s_i 越大，同一节点可以发出的交易数量越多。方程（1）确保即使具有无限计算能力的用户也不能随意通过垃圾信息进行网络攻击。

5.2 实施细节

为简单起见，我们假设传入交易的顺序与节点发出的顺序相同。 由于执行 PoW 所需的预期时间通常远大于网络延迟时间 h，因此这是一个合理的假设。

当第一次看到交易时，节点存储发出交易的节点 id，接收它的时间 t_0 和 PoW 难度。 可以使用第 3 节中描述的方法来确定发布节点的身份标识 id 以及 $mana$ 的 s_{id} 。基于此信息，可以检查由同一节点在最近的 T 个时间单位内发布的交易数量不超过根据其 $mana$ 数量所得到的最大数量 $z(s_{id})$,并且最近这些交易的难度值确实足够大。 这些想法在算法 3 中有更正式地描述。

算法 3　速率控制算法

Algorithm 3: Rate control algorithm

Input: incoming transaction t, set of known transactions $\mathcal{X}$, time window T, basic difficulty d_0, weight function w.

Output: forward or ignore t.

$t_0 \leftarrow \texttt{time}(t)$
$id \leftarrow \texttt{nodeId}(t)$
$\mathcal{T} \leftarrow t' \in \mathcal{X}$ such that $\texttt{time}(t') \in (t_0 - T, t_0]$ and $\texttt{nodeId}(t') = id$
if $|\mathcal{T}| < z(s_{id})$ **then**
    if $\texttt{difficulty}(t) \geq d_0 + w(s_{id}, |\mathcal{T}|)$ **then**
        return forward t
return discard t

5.3 可验证的延迟函数

虽然本节中描述的自适应速率控制算法解决了 PoW 的一些缺点，但我们相信，当前分布式账本生态系统对更高效算法的需求是显而易见的。在下文中，我们提出了一种可持续性更好的机制，可代替 PoW 算法：*可验证的延迟函数*（VDFs）。

通俗地来说，VDFs 是特殊的函数：(i) 即使假定能够拥有无限并行资源（即使用无限数量的 CPU）[6]，也难以预测计算，（ii）易于验证。大量研究人员根据特定的数论函数提出了不同的 VDF（例如模态指数[17,27]，椭圆曲线上的超奇异同构[15]，椭圆曲线上的配对，有限域扩张之间的内射有理图[5]）。与 PoW 相比，这些函数具有以下优势：

- VDF 可以被认为更环保，因为它们可以避免挖矿竞争。
- 由于它们不可并行化，因此无法有效地使用专用硬件（例如 ASIC），从而解决了高算力节点和低算力节点之间的不公平问题。

抵抗并行化的条件使得对这些函数的追求成为一个有趣且非常重要的问题。从函数实现的角度来看，主要问题是函数计算求解（预测计算）所需时间与验证其正确性（验证）所需时间之间的比率。表 1 提供了不同 VDF 关于此性能指标比较的良好见解。

VDF	ratio
Exponentiation-based (RSW)	8000:1
Supersingular isogenies	400:1
Pairings over elliptic curves	300:1
Injective rational maps	n/a

表 1： 不同 VDFs 的计算与验证时间比率

关于可验证延迟函数的第一个想法可以追溯到 Dwork 和 Naor 在抵抗垃圾信息领域的开创性论文[17]，但只有在 Boneh 等人的论文[5]之后，对 VDF 的开发和实现的兴趣才大大增加。实际上 VDF 已经成为某些 DLT 设计中的基本要素（例如，Chia Network[7]）。此外文献[5]显示了去中心化随机性的潜在应用。我们相信 VDF 可以在替换基于 PoW 算法方面提供很大帮助，因为它们能够限制具有强大哈希运算能力的节点的计算能力。

6 共识

由于网络中交易的传播有延迟，在同一时刻每个节点未必拥有相同的视野。这可能使得在验证的时候，会有多个互相有冲突的交易加入到 Tangle 中。IOTA 白皮书的一个基本假设是，Tangle 本身确实可以包含有冲突的交易。然而，在发生冲突时，节点需要决定哪些交易最终应被视为有效，即，他们需要就那些冲突的交易达成*共识*。

[7] www.chia.net

IOTA 白皮书中，仅通过使用 tip 选择算法（TSA）来达成目的。即诚实节点目前基本上用无规行走（random walk）的方法来选择有效交易。如果交易有冲突，基本上会留下一个分支。但是正如本文中已经说明的那样，该方法只有在诚实交易占多数的假设下有效。此外，这个解决冲突的办法速度缓慢，这会导致选择“错误”分支的交易，从而需要进行重新添加（reattachments）。

在本文中，我们提出了一种新的共识机制，它含有着一个投票机制，有助于处理上述问题。虽然投票模型有其局限性，但它们已成功广泛用于工程和经济应用之中[2,26,33]，并导致了新兴的社会物理科学产生[10]。为了便于说明，我们将共识分为两个主要组成部分（另请参见图 2）：

- *tip 选择算法*。在 6.1 节中，我们对 IOTA 白皮书中提出的基于无规行走（random walk）的算法进行了一些重要的改进，目的是提高整体吞吐量以及对寄生链和分裂攻击的防范。
- *投票机制*。在 6.2 节中，我们描述了两种投票机制，其中节点相互通信，以便在有冲突交易时 Tangle 决定接受哪些交易。

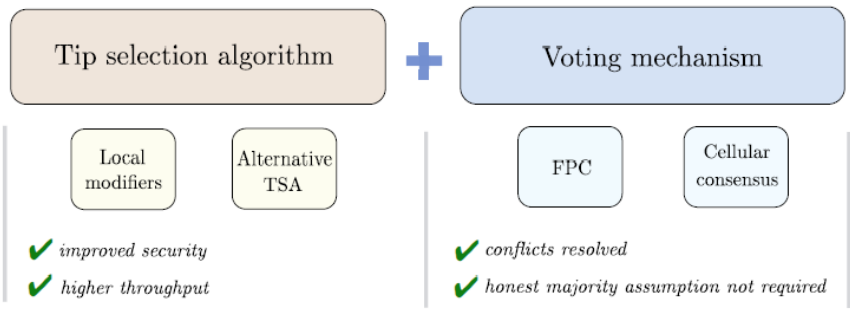


图 2：我们的创新共识机制增加了一个投票层，以提高安全性和多数诚实节点之假设的缺陷。

6.1 Tip 选择

Tangle 是根据以下规则构建而成的数据结构：

为了加入 Tangle，每笔交易必须验证已有的两笔交易。

交易的验证是验证一个地址是否拥有代币的过程[8]。如果交易 y 验证了交易 x，我们说 y 直接验证 x。相反，如果交易 x 和 y 之间没有直接的验证关系，但它们之间存在有向的验证路径，那么我们说 y 间接地验证 x。

IOTA 白皮书最初仅使用基于偏无规行走的 Tip 选择算法（TSA）来确定要验证的 tip 交易。这种 TSA 具有双重好处：首先，它可以有效地对抗自私节点进行简单的选择（例如，纯随机选择或选择创世交易），从而避免自私节点进行这些简单的 tip 选择；第二，它使 Tangle 对恶意节点执行例如寄生链攻击更具抵抗力（见图 3）。但是，白皮书中的方法如第 2 节所述也有其自身的局限性；特别是多数交易为诚实交易这一假设是算法安全性的必要条件。

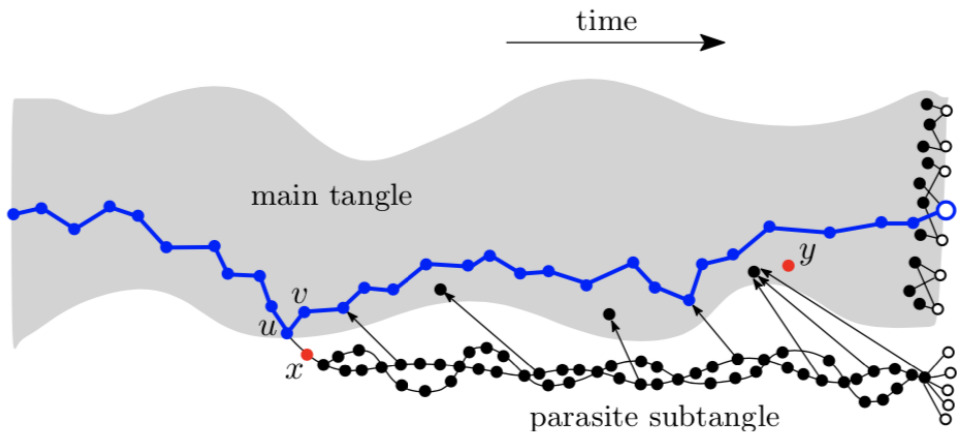


图 3：寄生链攻击的一个例子：交易 x 花费了攻击者想要在交易 y 中再次花费的资金，然后显式包含了交易 x 的子网络。为清楚起见，灰色区域包含许多没有显式表示出来的交易。粗蓝线对应于典型的无规行走轨迹。

在本节的其余部分，我们概述了 TSA 的研究现状：具体而言，在 6.1.1 节中，我们介绍了一种基于无规行走的 TSA，旨在改进原始白皮书中的算法；然后，在 6.1.2 节中，我们研究了一种替代方法，其中使用两种不同的算法来选择要验证的两个 tips 中的每一个。

[8] IOTA 中的实际验证过程更加复杂，我们邀请感兴趣的读者可以访问 https://docs.iota.org 以获取更多信息。

值得重点强调的是，由于 TSA 没有（并且不能）强制实施，特定 TSA 的选择最终取决于节点的所有者。因此，任何参与者将以合理的方式选择他们自己的 TSA，如[31]中所述。由于这种内在的选择自由，并且因为所有可能的 TSA 的“空间”是巨大的，所以可供选择的合理算法是至关重要的。我们绝对没有义务满足特定版本的 TSA; 相反，我们的愿景是，与人类社会本身类似，系统将持续发展。因此，虽然现有的基于无规行走的 TSA 已经做得很好，但我们将在下面描述几种有用的方法和研究方向，希望能够邀请学术界提供更多有价值的信息。

6.1.1 基于无规行走的 TSA

白皮书中的 TSA 是一个基于偏无规行走的算法，从创世交易开始，走向 tips。 一旦行走到达前沿 tips，该 tip 被选为第一笔直接验证的交易。然后进行第二次行走以选择第二笔直接验证交易。为了防止选择验证旧交易的“懒惰”tips 的无规行走，从交易 x 到交易 y 的转移概率是有概率取向的，并且正比例于：

$$\mathrm{P}[x \to y] \propto exp\{\alpha \cdot w_y\} \qquad (2)$$

其中 $\alpha > 0$ 是权重参数，w_y 是累积权重[9]，即直接或间接验证 y 的交易数量。为了获得实际概率，可以简单地将方程（2）进行归一化。由于每个节点看到的是一组不同的 tips，因此不可能强加特定的 TSA。

在本节中，我们提供另一种基于无规行走的算法，以提高算法性能和安全性。正如前文已经指出的那样，虽然 TSA 无法强制实施，但我们期望节点跟随可用的“最佳”算法。 主要思想是引入 Tangle 的本地（即每节点）视图，使得一些交易是基于节点本地可用的各种信息而优选的。 例如，固化时间[10]可用于降低寄生链攻击的有效性：由于攻击者需要一些时间来构建一个 subtangle，一个诚实的节点可能会决定惩罚一笔出现得比应该出现时间要晚的交易。这种局部视图通常称为局部修饰符[29]。

[9] 白皮书中对累积权重的定义略微更为一般化，并且还考虑了发布交易所需要的工作量。
[10] 交易 x 的固化时间不仅包括交易本身，而且还包括给定节点接收交易 x 所引用的所有交易的时间。

一般来说，局部修饰符可以实现多种特征：它们可以增强 Tangle 的安全性，特别是对抗寄生链和分裂攻击; 它们减少了对 PoW 和累积权重计算的依赖; 它们使网络能够在暂时失去诚实交易的情况下生存下来。

设 t_x（相应的 t_y）是给定节点处交易 x（相应的交易 y）的固化时间。 根据以上讨论，从交易 x 到交易 y 的转移概率正比例于：

$$\mathbf{P}[x \to y] \propto \exp\{\alpha \cdot w_y - \beta \cdot (t_y - t_x)\}, \quad \text{if } t_y - t_x < D, \tag{3}$$

其中 $\beta > 0$ 是权重参数，D 是时间差截止参数。如果 $t_y - t_x \geq D$，则转移概率变为 0，即从无规行走和累积权重计算中排除相应的引用边。基本上，我们可以设想一种系统，其中 TSA 在现有交易的子集上执行，具体取决于节点的本地视图。这个想法将在 6.2 节中进行扩展。

了解固化时间的引入是如何增加缠结对抗寄生链和分裂攻击的稳健性是极为有意思的。为了执行寄生链攻击，恶意行为者在包含资金的原始交易已被接受之后，附加隐藏的 subtangle（批准双重支付交易），并试图使其增长。分裂攻击则是这样的一种情况，恶意节点代理将缠结分裂成两部分：当其中一个分支增长时，恶意节点同时也在另一个分支上发布交易以保持两者都存活。分裂攻击的目的是进行双重支付并破坏网络。这两类攻击情况的共同点都是把那些交易隐藏了一段时间。从节点的角度来看，这种情况看起来象是新的交易验证旧的交易。但通过交易的固化时间可以发现这种非典型异常的时间差异，并且这些攻击可以有效地降低 TSA 选择此类交易的概率。除上述情况之外，任何通过隐藏交易攻击，或延长交易固化时间到超长时间，都可以通过使用局部修饰符来增强其稳健性。

同样重要的是要提到方程（3）可以容易地进行扩展，以便模拟了解节点已知的任何本地信息，例如发布节点信誉。作为这种扩展的另一个自然案例，考虑两笔交易 x, y，其中 y 验证 x。现在我们定义*兄弟数* $s(y, x) \in N$：如果交易 $y_0, \ldots, y_k$ 是直接验证交易 x，并且节点以此顺序连续看到它们，然后我们定义 $s(y_i, u) = i$。下面我们考虑对方程（3）进行如下修改：

$$\mathbf{P}[x \to y] \propto \gamma^{s(y,x)} \exp\{\alpha \cdot w_y - \beta \cdot (t_y - t_x)\}, \quad \text{if } t_y - t_x < D, \qquad (4)$$

其中 γ∈（0,1）是一个重要参数，兄弟数。方程（4）中，当前位于 u 的行走者因此将更偏好对应节点之前看到的那些“后继者”。为了解释为什么采用这种规则是合理的，请首先注意，平均而言每笔交易将（直接）被两笔交易验证，因此过多的直接验证可被视为是可疑的。这确实是一个恶意行动者的策略，旨在“过滤”一笔正常交易（即阻止它被 TSA 选中）：通过发布许多交易并将它们附加到同一笔交易，攻击者希望 TSA 通常会选择“他发布的”那些交易之一，而不是被他攻击的那笔良性交易。现在观察上述修改，确实可以有效地防止这种行为的发展。

6.1.2 替代 TSA

无规行走的 TSA 的随意性使得并非所有交易最终都能被验证。 一般来说，这可以被视为一个特征，因为这样系统中有些交易被“留下”来了，并对缠结具有潜在的损害。 但是，如果合法交易未在一定延迟内得到验证，则可以假设它不再会被验证，因此这笔交易应该重新附加到较新的 tips。由于较低的确认率，这会给系统引入额外开销，所以我们也在探索一种新的算法，旨在确保所有交易在有限时间内得到验证[21]。

关键的直觉是白皮书 TSA 中的高 α 值（参见前面小节）有利于从创世交易到 tips 具有更长路径，因此选择较旧的 tips 的概率随着时间增加会减少。相比之下，较小的 α 值允许选择较旧的 tips，但它们使得缠结容易受到双重支付攻击。这里所提出的方法旨在通过使用两种不同的算法来选择每个 tip 以组合两种情景（大和小 α 值）的最佳属性：通过使用具有高 α 值的白皮书算法来选择第一个 tip, 从而确保优先选择诚实的 tip 并保证安全; 至于第二个 tip，我们使用随机选择来确保不会有任何 tips 被第一个 tip 所留下。

仍需要对某些方向进行更仔细的研究。例如，无法阻止一个节点只进行 tips 的随机选择，从而减少节点自己的验证开销。在这种情况下，可能需要额外的措施来防止这种懒惰行为。例如，一种可能的措施可以使用第 3 节中讲到

的节点责任特性：网络的参与者可能以某种方式惩罚发布验证那些“不应该验证的内容”的交易节点。

6.2 投票表决机制

在本节中，我们将讨论一种投票机制，这个附加层能够确保在被攻击的情况下共识的安全性。其做法是节点向其他节点查询他们当前账本的内容，根据他们得到的其他节点意见之比然后调整自己，在经过几轮之后，决定自己账本的内容。在本节中，我们仅考虑对单个冲突取得共识的算法，其结果是将一个交易标记为“喜欢”或“不喜欢”。

基本思路是让节点相互通信，以便主动解决冲突。当有冲突时，从分布式账本的初始状态开始执如下：考虑一个交易 v，如果在给定的时间内节点没有看到有其他交易从同一地址花费，我们说该节点喜欢交易 v; 否则，节点不喜欢交易 v[11]。并且，必须考虑 tip 是怎样挑选的。最直接地的方法是，简单地删除不喜欢的交易，并且排除这个交易之后锥体形中的所有交易。

把这种机制应用到任何随机行走 Tip 选择算法的最简单的方法，是直接将这笔被否决的交易在缠结生成的边缘分支删除掉，包括这笔交易本身和他之后关联的任何交易形成的锥形分支。

之后，我们会定期对缠结中的每个交易应用一种投票方案，每个节点都会询问一些邻居的状态。在投票后，一个交易要么被节点所喜欢，要么被一个节点肯定不喜欢，直到这种状态不会改变。我们希望保持单调性，如果一个节点喜欢交易 *u*，那么它就喜欢交易 *u* 赞同的任何交易，如果节点不喜欢交易 *v* 那么它不喜欢赞同交易 *v* 的任何交易，参见图 4。实现这一点，我们可以放心地假设我们只喜欢交易 *v* 当我们喜欢它的所有前锥体形，如果我们不喜欢 *v* 那么我们不喜欢它的所有后锥体。

在以下两个小节中，我们将描述我们正在考虑的两种投票机制。第一个称为 “快速概率共识”，它已通过严格的数学证明; 但是，这个解决方案需要节点

[11] 重要的是要注意，此规则不包括重新附加：如果$v_1, \ldots, v_k$都是同一交易的重新附加，我们要么全部喜欢这些交易，要么全部不喜欢这些交易。

接受来自不是邻居节点之连接，并使用非中心化的随机数，这就需要另一个附加层。另一方面，第 6.2.2 节的 ”元胞自动机共识“ 方法没有那么多的要求，且快过第一种机制。然而，这种方案缺乏严格的证明，也需要更严格的节点自动互连解决方案，以避免日蚀攻击，不然就可能被孤立成可疑节点。这两种解决方案可以被认为是相互不排斥的投票机制的实现，它们可以组合使用，以建立一个坚固的框架。

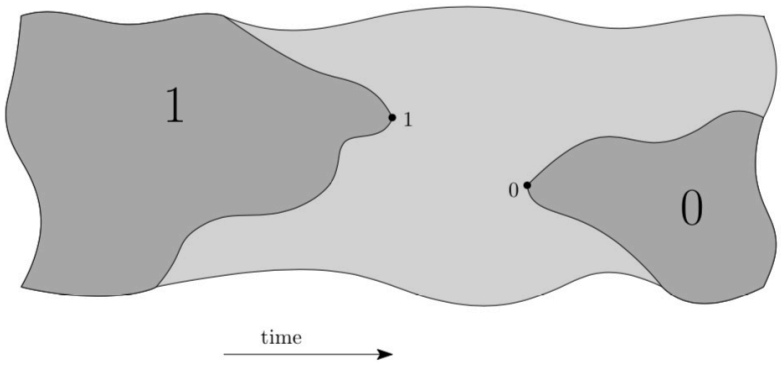


图 4. 投票必须一致

6.2.1 快速概率共识

论文[32]引入了一种通信复杂度低的协议，它允许一组节点通过（可以是随机的）投票的手段对一位变量的值达成共识。（参见文献例如[3,13,14,18,19,25]及其中有关该主题的大量文献; 我们也提到了关于（概率）拜占庭共识协议的经典工作，参见文献例如[1,4,7,11,20,22,34]，但通信复杂度要大得多）。文献[32]中描述的机制的特点是允许更多数量的可疑（或拜占庭）节点，其数量可以占节点总数的（固定）比例。那些可疑节点意图是要么推迟达成共识，要么打破共识（即至少让一些诚实节点得出不同的结论）。尽管如此，该协议在适当选择其参数时，它的成功的概率很大。在给定概率下明确估算产生共识的时间，参见定理 2.1 和 2.4。

与这一领域的经典作法不同，它并不要求在一开始，持有相同意见的比例很高来达到共识。相反：

- 如果最初时，没有绝大部分的节点[12]更喜欢 1，那么最终共识是 0 的概率就很大；
- 如果最初时，绝对多数的节点[13]更喜欢 1，那么最终的共识是 1 的概率就很大。

为解释为什么这与加密货币之使用有关，请考虑存在两个相互矛盾的交易的情况; 例如，其中一个将地址 A_1 的所有余额转移到地址 A_2，而另一个交易将地址 A_1 的所有余额转移到地址$A_3 \neq A_2$。如果两笔交易在网络中存在冲突，那么安全的做法是声明两者都是无效的。另一方面，总是声明它们无效的想法，不是一件好事。实际上，如果我们这样做，那么恶意行为者将能够通过下列方式利用它：首先，他发布了合法的交易，例如，从商家购买一些商品。当他收到货物时，他发布了如上所述的双重支付交易，希望两者都被系统取消，所以他会成功地收回他的退款（或者至少从商家那里取回钱）。为了避免这种情况，如果，第一笔交易（付款给供营商 是可以信任的）的那个时候，节点可能已经获得了一些信任，所以这笔交易将被确认，那么，随后的双重支付就可以被取消了。

这个协议[32]的一个特殊之处在于它利用了一系列随机数，这些随机数由可信来源提供或生成。节点本身使用一些分散的随机数生成协议，参见文献例如[9,28,35,36]。重要的是要注意，即使可疑对手有时设法（全部或部分）控制了随机数，这只能导致延迟达成共识，但他无法使不同诚实节点得到不同的结果，也就是不会违反安全性。而且，无需所有的诚实节点都具有相同的意见; 如果它们中大多数意见相同，这已经足够达成共识（也就是说，生成的随机数不需要任何形式的“强烈共识“）。我们没有在这里描述所述协议的所有细节，感兴趣的读者可阅读文献[32]。

[12] 松散地说，一个显著大多数是指统计上不同于 50/50 的情况; 例如，意见为 1 的比例大于某些 ϕ, 其中 $\phi > 1/2$。

[13] 这是一个松散的概念; 超级大多数是已经接近于共识，例如，超过 90%的节点具有相同的意见。

6.2.2 元胞自动机共识

本文讨论的第二种实施是元胞自动机共识（CA）。CA 方法最初是作为 Ising 模型而开发的[23]。与其他共识算法相比，CA 的最大优势之一是它能实现非常高水平的并行性。仅这一优势就能驱动更深入的研究。CA 带来了以下新颖的关键属性

- 每个节点都按照元胞自动机机制[12]运行，即在存在冲突的情况下，仅根据其直接邻居的状态改变其观点，并始终采用多数意见。
- 在每一回合的运行期间，节点的邻居集合不会改变。在这种情况下，第 4 节中提到的重组只能在不同回合的运行中发生，即不一样的冲突。
- 在评估邻居的意见时，节点将邻居的邻居意见作为“证据”。这将允许节点监视彼此的行为并防止节点独立于其邻居。
- 行为异常的邻居，即持有与证据不一致的邻居，将立即被删除掉。这个信息还会传播到网络，以便其他节点验证并将该节点标记为“恶意节点”，并防止将来的连接尝试。

在每一回合开始时，每个节点发送一个当前状态的心跳。这包括其签了名的自己的当前状态，以及上一轮中每个邻居当时的状态，每个状态均由发布节点各自签名。这使邻居们的以前状态不可能被伪造，每个节点都会接收到这个心跳，可以验证当前的状态是否正确，并遵循共识机制的规则。

我们将在下一小节中将上述想法规范化为共识协议。

算法 4: 发送心跳

Algorithm 4: Send heartbeat

Function `heartbeat`(*Node* i, *Round* m)**:**
  foreach *neighbor* $j \in N_i$ **do**
    send opinion $X_m(i)$ to neighbor j
    foreach $j' \in N_i \setminus \{j\}$ **do**
      send opinion $X_{m-1}(j')$ to neighbor j

模型

假设存在一个由 n 个节点组成的网络，并且这些节点需要对一个一位变量的值达成共识。为清楚起见，我们在下文中假设每个节点通过第 4 节中描述的节点自动互连机制直接连接到 k 个邻居。节点 i 的邻居集由 Ni 表示。节点自动互连机制是这种方法安全性的关键因素：实际上，恶意节点不应该被选择，也不能影响它的邻居。

在算法的每个阶段期间，每个节点对这个一位变量的值保持意见。意见可以是 0,1 或空，具体取决于节点是选 0，1，还是空（即不是 0，不是 1）。第 m 个回合中,节点 i 的观点由 $X_m(i) \in \{0,1,\perp\}$ 来表达。我们进一步假设，每个节点 i 具有初始意见 $X_0(i) \in \{0,1\}$。

该协议取决于以下参数 ：

- $k \in \mathbb{N}$，每个节点的（初始）邻居数量。
- $M \in \mathbb{N}$，最大回合数。
- $l \in \mathbb{N}$，最终达成一致意见所需的回合数。
- $p: \{0, \dots, k\} \to \mathbb{R}_{\geq 0}$ 为单调递增权重函数，即将邻居数量映射到权重。这样可惩罚邻居数少于 k 的节点。

算法 5　元胞自动机共识

Algorithm 5: Cellular consensus

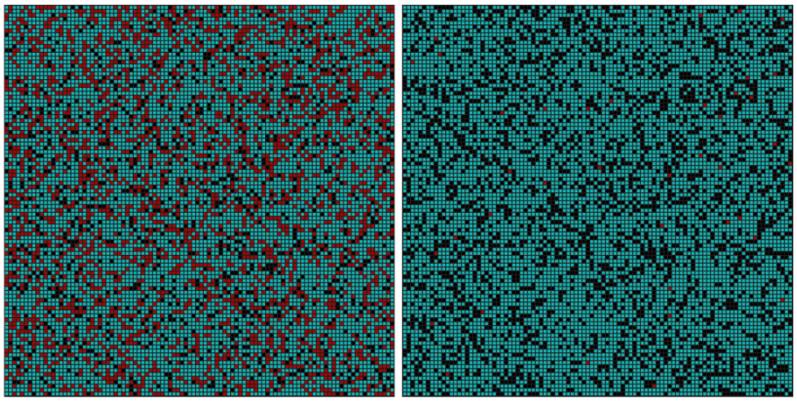
foreach *node* i **do**
  Send initial opinion $X_0(i)$ to neighbors N_i

for $m \leftarrow 1$ **to** M **do**
  foreach *node* i **do**
    foreach *neighbor* $j \in N_i$ **do**
      if $X_{m-1}(j)$ *is inconsistent wrt* $X_{m'}(j')$ *for* $j' \in N_j, m' < m-1$ **then**
        `// drop neighbor` j
        $N_i \leftarrow N_i \setminus \{j\}$
    if *node* i *finalized* **then**
      $X_m(i) \leftarrow X_{m-1}(i)$
    else
      `// adopt majority opinion`
      $total \leftarrow \sum_{\{j \in N_i\}} p(|N_j|)$
      if $\sum_{\{j \in N_i | X_{m-1}(j)=0\}} p(|N_j|) > \frac{total}{2}$ **then**
        $X_m(i) \leftarrow 0$
      else if $\sum_{\{j \in N_i | X_{m-1}(j)=1\}} p(|N_j|) > \frac{total}{2}$ **then**
        $X_m(i) \leftarrow 1$
      else
        $X_m(i) \leftarrow \perp$ `// cancel all`
    `heartbeat(`i, m`)`
    if *opinion* $X(i)$ *did not change in the last* ℓ *rounds* **then**
      mark node i finalized

算法

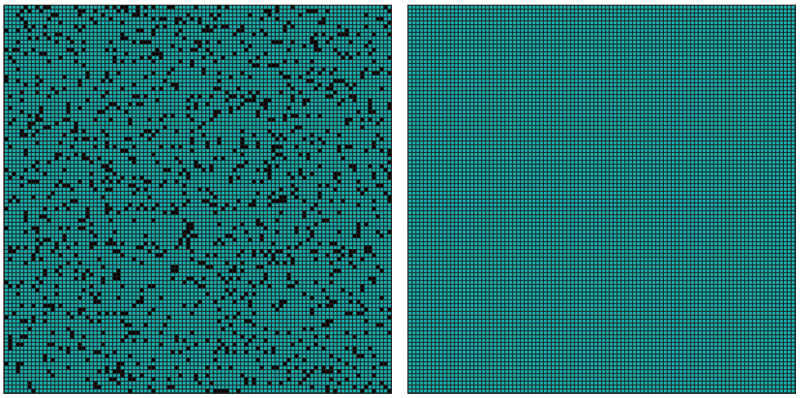
每个节点 i 知道其邻居 $j \in N_i$ 的意见以及他们邻居的邻居 N_j 的意见。这通过广播步骤来确保所有意见都以这样的方式签名，即意见始发节点以及广播节点是没法伪造的。这已在算法 4 中表现出来。

CA（元胞自动机共识）是这样一种共识机制，其中每个节点都使用其邻居的意见来更新自己的状态。当大多数邻居支持 0 或 1 时，节点采用该意见。如果不存在被大多数邻居支持的意见，则采用"空"，即没有。如果集合 N_i 至少对于 i 所有的邻居都是已知的，任何节点都可以使用这些简单规则来验证所报告的邻居 i 的意见与 N_i 中所有节点的意见是否一致。在算法 5 中更形象地解释了元胞自动机共识机制，并且在图 5 中可以找到示例场景的 CA 过程。



(a) opinions “collide” (first nodes update opinion)

(b) opinions “compete” (nodes update their opinions)



(c) opinions “compete” (nodes update their opinions)

(d) a single opinion “survives” (network reaches consensus)

图 5：CA 共识的可视化过程：每个方块对应一个连接到随机邻居的节点。最初，两笔互相冲突的交易在网络上传播。然后，节点在最终达成共识之前不停地调整他们的意见（0：红色，1：青色，空：黑色）。

(a)意见“碰撞”(第一个节点更新意见)　　(b)意见竞争(节点更新他们的意见)

(c)意见竞争(节点更新他们的意见)　　(d)唯一一个意见”幸存”下来 (网络达成共识)

7 结论

在本文中，我们概述了 Coordicide 项目的整体方案。其中，我们重点围绕着共识机制，安全性和防御攻击，垃圾交易管控和节点自动互连等方面阐述了我们的想法； 所有这些都为实现 Coordicide 项目提供了重要的基础。尽管我们已经明确定义了实现 Coordicide 的步骤，但我们仍然在评估各种选项和设定，以便在整体框架中实施和部署这些组件。

参考资料

[1] Marcos K. Aguilera and Sam Toueg. The Correctness Proof of Ben-Or's Randomized Consensus Algorithm. Distributed Computing, pages 371–381, 2012.

[2] Sven Banisch, Tanya Ara'ujo, and Jorge Lou¸c˜a. Opinion dynamics and communication networks. Advances in Complex Systems, pages 95–111, 2010.

[3] Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, and Luca Trevisan. Stabilizing Consensus with Many Opinions. In Symposium on Discrete algorithms, pages 620–635. SIAM, 2016.

[4] Michael Ben-Or. Another Advantage of Free Choice (Extended Abstract): Completely Asynchronous Agreement Protocols. In ACM Symposium on Principles of Distributed Computing, pages 27–30, 1983.

[5] Dan Boneh, Joseph Bonneau, Benedikt B¨unz, and Ben Fisch. Verifiable Delay Functions. In Annual International Cryptology Conference, pages 757–788. Springer, 2018.

[6] Allan Borodin and Ian Munro. The Computational Complexity of Algebraic and Numeric Problems. North-Holland, 1975.

[7] Gabriel Bracha. Asynchronous Byzantine Agreement Protocols. Information and Computation, pages 130 – 143, 1987.

[8] Quentin Bramas. The Stability and the Security of the Tangle. Preprint, 2018.

[9] Ignacio Cascudo and Bernardo David. SCRAPE: Scalable Randomness Attested by Public Entities. In International Conference on Applied Cryptography and Network Security, pages 537–556. Springer, 2017.

[10] Claudio Castellano, Santo Fortunato, and Vittorio Loreto. Statistical physics of social dynamics. Reviews of Modern Physics, page 591, 2009.

[11] Miguel Castro and Barbara Liskov. Practical Byzantine Fault Tolerance and Proactive Recovery. ACM Transactions on Computer Systems, pages 398–461, 2002.

[12] Edgar F. Codd. Cellular Automata. Academic Press, 1968.

[13] Colin Cooper, Robert Els¨asser, and Tomasz Radzik. The Power of Two Choices in Distributed Voting. In International Colloquium on Automata, Languages, and Programming, pages 435–446. Springer, 2014.

[14] Colin Cooper, Robert Els¨asser, Tomasz Radzik, Nicolas Rivera, and Takeharu Shiraga. Fast Consensus for Voting on General Expander Graphs. In International Symposium on Distributed Computing, pages 248–262. Springer, 2015. 28

[15] Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable Delay Functions from Supersingular Isogenies and Pairings. Cryptology ePrint Archive, Report 2019/166, 2019., 2019.

[16] John R. Douceur. The Sybil Attack. In International Workshop on Peerto-peer Systems, pages 251–260. Springer, 2002.

[17] Cynthia Dwork and Moni Naor. Pricing via Processing or Combatting Junk Mail. In Advances in Cryptology, pages 139–147, 1993.

[18] Robert Els¨asser, Tom Friedetzky, Dominik Kaaser, Frederik MallmannTrenn, and Horst Trinker. Rapid Asynchronous Plurality Consensus. arXiv preprint arXiv:1602.04667, 2016.

[19] Giulia Fanti, Nina Holden, Yuval Peres, and Gireeja Ranade. Communication Cost of Consensus for Nodes with Limited Memory. arXiv preprint arXiv:1901.01665, 2019.

[20] Paul Feldman and Silvio Micali. An Optimal Probabilistic Algorithm for Synchronous Byzantine Agreement. In Automata, Languages and Programming, pages 341–378. Springer, 1989.

[21] Pietro Ferraro, Christopher K. King, and Robert Shorten. IOTA-Based Directed Acyclic Graphs without Orphans. arXiv preprint arXiv:1901.07302, 2019.

[22] Roy Friedman, Achour Most'efaoui, and Michel Raynal. Simple and Efficient Oracle-Based Consensus Protocols for Asynchronous Byzantine Systems. IEEE International Symposium on Reliable Distributed Systems, pages 228–237, 2004.

[23] Ernst Ising. Beitrag zur Theorie des Ferromagnetismus. Zeitschrift f¨ur Physik A Hadrons and Nuclei, pages 253–258, 1925.

[24] Frank P. Kelly, Akhil K. Maulloo, and David K. H. Tan. Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability. Journal of the Operational Research Society, pages 237–252, 1998.

[25] Frederik Mallmann-Trenn. Probabilistic Analysis of Distributed Processes with Focus on Consensus. PhD thesis, PSL Research University, 2017.

[26] Hong-Li Niu and Jun Wang. Entropy and recurrence measures of a financial dynamic system by an interacting voter system. Entropy, pages 2590–2605, 2015.

[27] Krzysztof Pietrzak. Simple Verifiable Delay Functions. In Innovations in Theoretical Computer Science Conference. Schloss Dagstuhl-LeibnizZentrum fuer Informatik, 2018.

[28] Serguei Popov. On a Decentralized Trustless Pseudo-Random Number Generation Algorithm. Journal of Mathematical Cryptology, pages 37–43, 2017. 29

[29] Serguei Popov. Local Modifiers in the Tangle. 2018.

[30] Serguei Popov. The Tangle, 2018.

[31] Serguei Popov. IOTA: Feeless and Free. IEEE Blockchain Technical Briefs, 2019.

[32] Serguei Popov. On Fast Probabilistic Consensus in the Byzantine Setting. Preprint, 2019.

[33] Piotr Przyby la, Katarzyna Sznajd-Weron, and Maciej Tabiszewski. Exit probability in a one-dimensional nonlinear q-voter model. Phys. Rev. E, 2011.

[34] Michael O. Rabin. Randomized Byzantine Generals. In Annual Symposium on Foundations of Computer Science, pages 403–409. IEEE, 1983.

[35] Philipp Schindler, Nicholas Stifter, Aljosha Judmayer, and Edgar Weippl. HydRand: Practical Continuous Distributed Randomness. Cryptology ePrint Archive, Report 2018/319, 2018.

[36] Ewa Syta, Philipp Jovanovic, Eleftherios Kokoris Kogias, Nicolas Gailly, Linus Gasser, Ismail Khoffi, Michael J. Fischer, and Bryan Ford. Scalable Bias-Resistant Distributed Randomness. In IEEE Symposium on Security and Privacy, pages 444–460, 2017.