Tangle 2.0

无领导中本聪共识：最重 DAG

Sebastian Müller*†, Andreas Penzkofer† , Nikita Polyanskii† , Jonas Theis† , William Sanders† , Hans Moog† *Aix Marseille Université, CNRS, Centrale Marseille, I2M - UMR 7373, 13453 Marseille, France † IOTA Foundation, Berlin, Germany

摘要-我们介绍 Tangle 2.0 的理论基础，这是一种基于有向无环图（DAG）的概率性无领导共识协议，称为 Tangle。Tangle 自然而然地继区块链之后出现，是其下一个进化阶段，因为它所提供的功能适合创建更高效和更具扩展性分布式账本解决方案。

共识不再出现在最长链上，而是见于最重 DAG 上，其中工作量证明（PoW）被基于权益或声誉的权重函数所取代。DAG 结构和底层基于现实的未花费交易输出（UTXO）账本允许并行验证交易，而无需全排序。同时，它支持删除矿机和验证器这些中间环节，实现了纯两步式流程，该流程在节点层面，而非验证器层面遵循「提案-投票」范式。

我们提出一个框架来分析不同通信和对手模型下的活性和安全性。这样可以在一些边界案例和异步通信模型中提供不可能的结果。我们提供协议安全性形式证明，该证明假定公共随机币种。

关键词：共识协议；无领导；异步；容错；有向无环图；安全

1. 引言

在分布式系统中，不同事件可能同时发生，但参与者感知它们的顺序有所不同。相比之下，像比特币[1]这样的分布式账本技术（DLT）常常用完全有序的数据结构——区块链来记录交易，这些交易阐明了账本状态。这种设计会造成挖矿机或验证器等瓶颈，而每笔交易都必须通过它们进行。区块创建也可能同时发生在网络不同部分，导致区块链分叉，需要解决。这是往往通过最长链规则[1]或某个最重子树变体[2]来完成的。为了确保系统的安全性，我们人为抑制系统吞吐量，这样在下个区块创建前，每个区块都能充分传播，而极少数「孤块」会自发分裂区块链。限制可扩展性另一个的影响是交易是分批处理的。挖矿机创建这些交易的批次或区块，区块链可视为三步式过程。在第一步，客户端向区块生产者发送某项交易，然后某个区块生产者提出包含一批交易的区块，而在最后一步，验证器验证该区块。

IOTA 采用一种更新颖方法来处理分布式系统的异步设置[3]。该方法不需要集群交易，而采用有向无环图（DAG）（作为底层数据结构）来表示同时发生的事件。在该模型中，单次交易被添加到账本中，而每次交易都引用至少两次先前交易。这种特性将账本更新简化成两个步骤：一个节点向账本提出交易，并等待其他节点验证该交易。去除矿机或验证器中间环节来解决（或至少减轻）与之有关的几个问题，如挖矿比赛[4]、集中化[5]、矿机可提取价值[6]和负外部效应[7]，从而实现免费用架构。但是，向账本添加新交易的并行性意味着，共识必须见于「更广」子图，而非仅仅最长链或最重子树。

通常而言，共识协议（特别是 DLT）是非常大的研究领域，我们必须参照一些综述文章来获得更详细介绍，如文献[8]-[10]。虽然共识协议取决于许多不同方面，但在引言其他部分，我们将侧重于与所提出协议的设计选型关系最密切的方面。

账本模型。分布式账本（DL）通常以两种方式实现收支平衡：一种是基于账户的模型，其中资金直接和用户账户关联，如以太坊[11]；而另一种则是未花费交易输出（UTXO）模型，在该模型中，令牌链接到所谓输出，而用户拥有输出密钥，如比特币[1]及其诸多衍生品，以及艾达币[12]、AVAX 币[13]和 IOTA 币[14]。后一种情况的重要观察结果是，UTXO 本身形成 DAG。对于许多用例和情况来说，对交易进行全排序是非必要的，因为它们中大多数是可并行处理的。但如果有冲突交易，UTXO 账本的只添加属性会妨碍这种并行处理优势。在文献[15]中，我们提出乐观更新账本的增强 UTXO 账本模型，来跟踪可能有冲突的依赖关系。我们构建共识协议，并利用此账本模型快速并行地解决冲突。

Tangle 与局部秩序。Tangle 是储存分布式账本（DL）所有交易的 DAG。每个 DAG 归纳了顶点集（顶点集是我们设定中的交易集合）的偏序。这种特性与建立交易全序的区块链形成对比。因为在故障崩溃系统中，原子广播和共识是等价问题，见文献[16]，DAG 的偏序在共识协议中造成其他「困难」。更确切地说，基于 DAG 的 DLT 在安全性方面存在严重限制。在 Tangle 最初提案中，文献[3]用「最重子图」替换最长链规则，即包含最多交易的子 DAG。但事实证明，这该设计容易受到各种类型攻击，而且过多依赖发出交易所需的工作量证明（工作量），如文献[17]。与许多其他基于 DAG 提案的共同之处是，该设计的另一个关键元素是有存活性问题。如果诚实交易涉及某个将来被证明是恶意的交易，无法添加到账本状态。本文提出的协议依靠权重函数和基于现实的账本状态来解决安全性问题[15]。通过将交易与其容器（即区块[1]）分隔开来，利用新区块引用方案来处理存活性问题，特别是这种无批处理架构支持面向流过程的 DLT 设计。

Sybil 保护。在人人都能参与的「无需许可环境」中，Sybil 保护起到至关重要的作用。比特币中本聪共识是第一个在如此开放环境中利用工作量证明（PoW），实现共识的共识。由于 PoW 会导致巨大能源浪费和许多负面外部效应，许多人努力提出更可持续性的替代方案。其中最突出的是权益证明（PoS），其中验证器投票权与其在系统中的权益（按照基础加密货币计算）成正比。本文所用 Sybil 保护是以节点身份为基础的。我们一般将其描述为稀缺资源函数或抽象声誉函数。该函数称为权重，为每个节点身份分配一个正数。比如，此权重可对应于赌注令牌、授权令牌或文献[14]所述的「MANA 币」金额。我们需要注意的是，权重不一定要连接到底层令牌，但可被任何其他「权重」所取代，从而很好实现 Sybil 保护。尤其是我们框架也可用于需许可环境，在该环境中只有预定义验证器有正权重，而且可用于具有动态委员会筛选的情形。

中本聪共识。分布式共识使参与者可以就不断增长的交易日志达成一致。数十年来，这一直是重要研究课题，在计算机科学中的重要性从未有过争议。对共识协议进行分的方法有许多种。比如，在 PAXOS 和 BFT 方面有经典里程碑式结果和新颖的中本聪式共识机制。我们将中本聪共识理解为选择最长子链的原则，比如见文献[10]，或权重最重子链（作为一种变体）。我们将此概念扩展到最重子 DAG。更确切地说，

我们认为，中本聪区块链共识遵循的是提议-投票范式，该范式概述如下。

时间被划分成时期（epoch），每个时期都有 「民选」领导。该领导将交易批处理到某个新的区块，然后提出该区块。接着，其他参与者对提出的区块进行投票，比如通过将区块链扩展到所提区块所在区块链。一旦投票数达到一定阈值，所提出区块就被认为构成账本一部分。上述各要素具体定义可能有所不同，相应地「中本聪共识」有不同变体。在某种程度上，上述范式简化为每个时期必须约定一位唯一领导。一旦参与者就领导达成共识，区块链的线性意味着就账本状态达成共识，而只有领导才能推进账本状态，这一事实明显造成瓶颈和公认性能限制。我们提案完全删除了「领导」的角色，允许参与者同时提出区

[1]与许多区块链协议不同，我们要求每个区块都刚好包含一项交易。但原则上，该协议可改编为区块包含不止一项交易。

块和它包含的交易。一旦区块提出，所有参与者都可以投票并参与共识发现。投票权重和上文所述节点的权重成正比，这样协议就能适应不同权重分布。该协议也被归类为非二元共识协议，因为可同时决定多项交易，是一个不断进行的投票过程，形成递进成长史[2]。同时，它也涉及一种概率共识，即某笔交易积累的辅助节点越多，该笔交易最终被确认并添加到账本的可能性就越大。

投票。在我们无区块架构中，每个新区块均引用至少两个现有区块，从而形成上述 DAG 结构。和区块链一样，新区块不仅就直接引用 进行投票，而且就其过去锥体进行投票。虽然这是一种有效投票方案，但也存在孤块或存活性问题。如果一个区块在其过去锥体中包含一个无效区块，那么它将不再是投票对象，所包含的交易也不能被纳入账本。我们通过引入两种不同参照来解决该问题。第一个参照是 Tangle 结构，而第二个参照是源自 UTXO 账本的 DAG 结构。最后一个参照允许我们对最初孤立交易进行投票，改变先前发布的投票。最终，这两种投票都被累积到一个投票权重，我们将其称为赞成权重（AW）。该 AW 越高，交易最终纳入账本的概率越高。我们参照图 1 来了解投票机制例子。

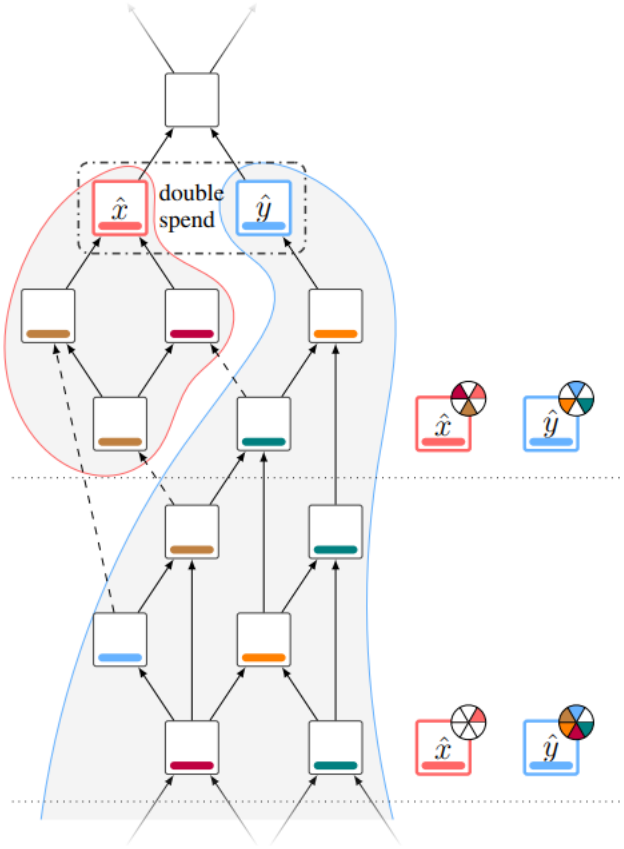


图 1： Tangle 被用作节点投票层，可就冲突结果达成共识。节点利用无领导协议在冲突交易 x 和 y 之间约定获胜者。不同颜色代表不同节点特征。对于交易 y，辅助节点数量（如右侧所示）随时间推移而增加。虚线引用是所谓的交易引用，可以「挽救」投给「失败方」的交易。

通常，投票机制可用于任何基于 DAG 的数据结构，该结构带有能够引用前一个区块的添加程序。它需要三个主要组成部分：第一个基本成分是有效投票和传播选票的参考方案。

[2]账本 DAG 结构和 Tangle 在技术上都是瞬态型数据结构，从理论上看二者都可以被修剪。为此，投票也是一种短暂知识。但为了简便起见，我们假设两个数据结构均没有修剪。

第二个必要成分是构建允许冲突共存的广义数据不变式结构，见文献[15]。该功能使我们能够「乐观」地对待交易；每笔新交易都被认为是「诚实的」，除非与其他交易冲突。因此，节点可能开始构建于每笔新交易之上，即使该笔交易被证明存在冲突。第三个成分是被称为 On Tangle voting （OTV）的投票机制它，可以同时有效地对无限交易进行投票。该效率是通过保持较低区块开销来实现的，因为其他节点投票均可通过隐式投票机制绑定。此外，与经典拜占庭容错相比，我们无需监视节点活动，因为交易的发出（将选票投出）是功能正常的明显标志。

安全性。自共识协议研究开始以来，安全性概念一直是业界关注的焦点。任何共识协议都是为了就数据达成共识。有些参与者可能出错，甚至积极阻止达成共识，而有人则关心达成共识所需要的条件。

提议-投票共识协议的安全性通常分为两点：存活性和安全性。存活性指的是任何正确交易最终被所有诚实参与者接受，而安全性指的是所有参与者最终就同一组交易达成一致。某个共识协议是否满足这些属性在很大程度上取决于模型假设。模型大致可分为通信模型和攻击者模型。

在最具限制性通信模型（同步模型）中，自里程碑式结果出现以来[18]，人们已知晓许多不同解决方案[18]。异步模型对区块传输延迟不设置任何限制，通常以 A 表示。但在最通用通信模型——异步模型下，情况并非如此，关于共识协议最著名结果之一是 FLP 不可能性结果[19]，它表明在异步通信模型中，单个出错参与者可能阻碍共识的发现。

由于 FLP 不可能性结果依赖区块延迟的特定配置，许多从业者认为其不适合在现实世界中实现，因为这些特殊情况不太可能发生。人们在这两个极端通信模型之间，提出了几个中介模型，并在较强网络延时假设下取得许多积极结果，比如部分同步模型[20]、定时异步模型[21]和带故障检测异步模型[16]。

除通信模型外，对手模型在中本聪协议中起重要作用，特别是安全性分析。协议安全性通常以稀缺资源数量来表示，如 PoW，这是攻击协议，是恢复已确认交易所必需的。中本聪[1]通过考虑某种特定攻击，即所谓私有双花攻击，分析了该特性。注意，这里经典通信模型属于部分同步模型。在过去十年里，一个相关研究问题是搜索最坏情况下攻击策略，识别在对手权重方面的安全阈值。近几年文献[22]和[23]给出几种最长链协议的严格一致性限制，但这些安全阈值在部分同步情况下有效，在异步情况（如[24]）下无效。

还有一项研究调查如何通过对通信层面施加更大影响来弥补其较低权重。这种攻击中最突出特点是均衡攻击[25]，包括具有均衡挖矿能力的多个节点子组之间的网络通信。

我们对这一讨论特别感兴趣，因为我们提出了同时在通信层面和对手层面建模的框架。我们在异步通信模型中获得了不可能性结果，这点不足为奇。尽管如此，在进一步同步性假设下，我们证实如果对手权重不超出特定阈值的话，那么协议可以保证存活性和安全性（以非常高概率）。所得安全限制是为任何可能攻击策略设定的，可根据协议进行配置。

在异步模型中导致不可能性结果的情形通常被认为与实际目的无关，如文献[26]和[27]。这么做的理由是，在实际应用程序中，区块延迟随机性如此之大，以至于不可能发生特定情况。虽然我们在一定程度上同意这种 OTV 推理，但我们在核心投票协议中添加了第二个共时性级别，以此获得严谨安全阈值。我们将共识协议视为两层解决方案。第一层按异步设置工作，在正常网络条件下可快速安全地确认。第二层以节点的可选同步性为基础，可在最坏情况下找到共识。同步水平取决于分散随机信标或公共货币，使协议对类似上述均衡攻击的攻击具有鲁棒性。自文献[28]发表以来，人们就知道利用共识协议的随机化来规避不可能性结果，这样便引入了局部随机性。该文献介绍一种常见币种[29]，并将其运用

于几种方法。

性能。为共识协议效率界定指标并非易事，因为它依赖许多不同方面。自然选择是参与者之间发送的区块数量，对于同步模型则是通信步骤数量。在 DLT 中，常用指标是每秒交易数量和确认时间。由于我们协议采用的是隐式投票，节点之间不交换直接区块，其区块复杂度达到最佳（如果选票是通过无论如何都会发送的区块投出）。我们估算了确认时间，并显示其对权重分布的依赖关系。在本研究中，我们并未评估定量性能指标，如吞吐量和能源消耗。此类研究将在后续研究中进行。

一个常见误解是，异步共识协议并不适合时间紧迫的应用程序[26]，其谬论是同步协议提出很强的同步性假设，而一旦这些假设得不到满足的话，安全性就会受损。我们认为恰好相反，异步协议可能更适合时间紧迫的应用程序。在通信良好情况下，以具有基本安全裕度的网络延时估算的同步模型为基础，交易审批速度要快得多。

基于领导的区块链架构的一个主要缺点是缺乏可拓展性。为了更加精确，我们设Δ为网络延时，λ为区块发布率，q 为对手权重，那么在文献[2]和[22]之后，协议安全性条件表示为：

$$q < \frac{1-q}{1+(1-q)\lambda\Delta}. \qquad (1)$$

为了设计能够应对最大对手权重 q 的弹性系统，此方程给出安全发布区块的最大速率限制。比如，如果在近期领导方面有分歧，可能出现安全违规。这些分歧可能是并行[30]生成区块或某些攻击场景[31][32]造成的。为此，区块链可能会重组，特别对于区块生产率很高的 DLT[33]。

在我们的案例中，本文在协议吞吐量方面无理论上限；但我们协议在可拓展性方面局限性仍需在未来工作中进行研究。

1.1. 基于 DAG 协议的相关工作

我们在总论中提到许多相关工作。在本节中，我们将重点介绍通用架构，提到以往使用 DAG 作为底层数据结构提案。

基于区块链协议是以区块链条化或「线性化」为基础，这种链条化或「线性化」构建起交易的全序。这通常是通过选择领导来实现的。其中一个主要问题是，如果区块创建速度高于其传播时间，可能会造成许多竞争甚至冲突区块，从而严重影响性能。因此，为了保证系统在安全参数范围内，低区块创建率限制了交易确认率和系统吞吐量。

性能。业界已经提出了几种方法规避上文提出的线性化并提高性能。它们大多都有一个共同点，即区块不仅可以引用前一个区块，而且可以引用多个区块，将底层数据结构从链条变为有向无环图。的确，使用 DAG 的想法在近十年来变得非常流行，如文献[3]、[34] -[39]和总结性论文[10]。

DAG 结构的应用使对账本并行写入访问成为可能，可以有效通过多个领导实现更快区块时间。事实上，上述研究证实 DAG 结构（而非链条）能够减少确认时间，并提高协议总体安全性。

排序。不同 DAG 协议在达成共识方式上可能有所不同。尤其 DAG 数据结构的使用可以（但不要求）规避交易完全排序。例如在文献[3]中，节点遵循并附加到最重 DAG，而在多数其他提

出的协议（如文献[39]和[37]）中，共识仍然是通过创建全序实现。另一种通过交易排序来实现共识的方法是借助原子广播协议。此类协议使网络参与者能够就接收的交易（总）排序达成共识，而这种线性化输出形成账本。最可靠协议在异步环境中实现拜占庭容错，并达到最佳通信复杂度，见文献[40]和[41]。如文献[42]和[43]以「通信史」编码为基础，以 DAG 为形式，提出了改进措施。我们协议与上述方法有所不同，因为在我们设置中只需要偏序。

无领导。DAG 也能促进无领导共识发现。比如，DAG 数据结构可以用来记录直接投票机制结果，该机制在参与者之间进行直接查询，如文献[13]和[44]。但我们注意到，文献[13]作者并未正确分析其提出的协议，该协议是否具有所需属性尚不明确，如文献[45，第 2.3 节]。直接投票法可以视为与我们的 ansatz 正交。在 ansatz 中，我们跟踪现有依赖关系，并借助 DAG 隐式投票方案来解决冲突。文献[43]也采用隐式投票方案，但最终需要对 DAG 进行线性化。

1.2. 结果

中本聪「最长链规则」的两个主要问题是可扩展性严重受限，缺乏并行性。缺乏并行性导致底层通信网络需要强同步性假设。我们所提出的共识协议，在异步模型中高效快速运行，并具有高度并行性，这点是通过以「最重 DAG 规则」代替「最长链规则」实现的。由于所产生共识并非基于交易全序，这样可以对交易进行流处理。对于智能合约更新和可选共享解决方案验证而言，优化变得越来越重要。

区块链另一个不太为人所知的缺点是需要矿机或验证器这类这些中间媒介。我们通过启用账本无领导写入权限，将系统简化为资金拥有者和节点二分结构，来消除这种依赖性。节点额外承担类似验证器的角色。节点提出新的区块，而这些区块包含来自资金拥有者的交易，并将其添加到 Tangle，节点利用添加程序对前面区块进行验证和投票，从而形成一种高效的隐式投票方案。

我们提出了一种以广义权重函数为形式的节点投票权的泛化。这种泛化使我们协议有较高可配置性，适应需要实现该投票权的系统的需求和安全要求，比如无需许可或需许可。我们引入一种异步无领导协议，该协议在 Tangle 中采用基于权重投票方案。通过隐式投票跟踪交易支持者（即节点）。交易确认状态可通过阈值标准加以确定。我们提供各种核心组件算法。更准确地说，阐述如何通过隐式投票方案来更新支持者列表，以及节点如何将其区块附加到 Tangle。我们提供了系统收敛性、存活性和安全性定理。首先，给定一个随机的、不可预测区块流，定理 7.1 保证，如果对手权重小于 50%，系统将最终收敛至共识状态，但在这种情况下，系统不给出安全保证。其次，我们通过扩展协议，借助某个常见币种，引入以特定时间间隔同步节点的能力，并提供安全性和存活性保证。定理 10.1 给出该扩展协议安全保证。

1.3. 论文结构

本论文结构如下。第 1.4 节概述所用符号、首字母缩略语和术语。第 2 节概述本文所用一些图表理论的初步研究。第 3 节提供基本网络环境，所提出的 Sybil 保护机制在该环境中运行。第 4 节描述 Tangle 数据结构功能，以及如何用其确认区块。第 5 节概述基于现实的 UTXO 账本，该账本构成我们方法的中心组件，有助于跟踪诚实节点对冲突交易的看法。第 6 节介绍投票协议和交易的确认。第 7 节定义通信模型和对手模型，并在第 8 节和第 9 节中探讨系统存活性和安全性，特别演示了某些试图创造「亚稳态」的攻击，在特定情况和较强对手假设下，可能会造成问题。第 10 节提供解决方案，引入以更大时间间隔同步节点的能力。最后，第 11 节对未来研究方向做出展望。

我们以若干图表结构作为共识协议基础。表 1 概述了所用图表。

DAG	顶点	边
Tangle	区块	引用
账本 DAG	交易	花费关系
投票 DAG	区块，交易	投票引用

表 1：DAG 概述

1.4. 首字母缩略语和符号列表

为方便读者，在本节中，我们总结了贯穿全文的重要符号和首字母缩略语。附录 B 提供本文使用的术语表。

首字母缩略语

AW 赞成权重

dRNG 分布式随机数生成器

DAG 有向无环图

DLT 分布式账本技术

OTV Tangle 线上投票

P2P 点对点

PoW 工作量证明

PoVP 投票权证明

TSA 末梢选择算法

TTC 确认时间

UTXO 未花费交易输出

WW 见证权重

符号

集合符号

B	分支集合
C	冲突集合
N	网络节点集合
L	账本或交易集合
T	区块集合

时间符号

$\tau_f(\cdot)$	按 T 定义的确认时间
$\tau_{cf}(\cdot)$	按 T 定义的汇合时间
$\tau_s(\cdot)$	按 T 定义的凝固时间

权重函数

w（·）	按 N 定义的权重函数
AW（·）	按 L 定义的赞成权重
WW（·）	按 T 定义的见证权重

DAGs

$D_{\mathcal{L}}$	账本 DAG
$D_{\mathcal{T}}$	Tangle DAG
$D_V V$	投票 DAG
child_V（x）	DAG D=（V， E）中顶点 x 的子节点集
$\text{cone}_V^{(f)}(x)$	DAG D=（V， E）中顶点 x 的未来锥
$\text{cone}_V^{(p)}(x)$	DAG D=（V， E）中顶点 x 的过去锥
D=（V， E）	带有顶点集 V 和边集 E 的有向无环图（DAG）

ρ 出度为零的原点或顶点

$\max_V$ （S） 集合 S 的最大元素集（根据 DAG D=（V， E）为最大）

$\min_V$ （S） 集合 S 的最小元素集（根据 DAG D=（V， E）为最小）

N_V （x） 图表 G=（V， E）顶点 x 的相邻点集合

$\leq_V$ 集合 V 上的偏序（通常用给定 DAG D=（V， E）诱导）

par_V （x） DAG D=（V， E）中顶点 x 的父节点集

$sprt_V$ （x） DAG D=（V， E）中 x 的支持者

2. 图表理论初步研究

在本节中，我们将概况本文其他部分使用的基本图表理论符号。

1 和 m 之间整数集用[m]表示。图 G 是一对（V， E），其中 V 代表顶点集，E 代表边集。

如果每条边都有自己的方向，则该图表称为有向图，比如对于边（u， v），其方向未从 u 到 v。

定义 2.1 （DAG）。有向无环图 （DAG）是一种不形成有向环的有向图，即沿着边的方向，永远不会形成闭环。

若（u， v）∈E，则图 G=（V E）中顶点 v 称为顶点 u 的相邻点。

若 e 包含 v，则边 e∈E 称为与顶点 v∈V 相邻。

有向图 G=（V， E）顶点 v 的出度和入度分别是以（v， u）和（u， v）为形式的相邻边的数量。若图中顶点没有相邻边，则称为孤立顶点。

定义 2.2（图表中的相邻点）。令 G=（V， E）为图表。对于顶点 v∈V，将其相邻点集合（或 G-相邻点，记为 N_v（v）[3]）定义为与 v 相邻的顶点集。

定义 2.3 （DAG 中父节点、子节点和叶子节点）。令 D=（V， E）为 DAG。对于顶点 v∈V，父节点集（记为 parV （v））被定义为顶点 u∈V 的集合，使得（v， u）∈E。

同样，我们也定义子节点集合，将其记为 $child_V$ （v），它是顶点 u∈V 的集合，使得（u，v）∈E。入度为零的顶点 v∈V 称为叶子节点。

定义 2.4（由 DAG 诱导的偏序）。令 D=（V， E）为 DAG。对于某个 u，我们记为 u ≤V

[3] 在本文其它部分，我们常常用顶点集来确定图表，因为对于给定顶点集 V，我们将只有一个 DAG D = (V, E)。因此，相邻点集合 N_V (v)和其他以 V 作为下标的概念将在语境中清晰明了。

v，当且仅当

有从 u 到 v 的有向路径，即存在一些顶点 w_0=u， w_1， . . . ， w_{s-1} 时，v∈V，w_s=v。而对于所有 i∈[s]，(w_{i-1}, w_i) ∈E。此外，如果 u≤V v，x≠v，我们记为 u<V v。

注意：不同 DAG 可能产生相同的偏序。偏序≤V，边数最少的 DAG 通常称为 D 的传递约简或≤V 的哈斯图

定义 2.5 （由一组顶点诱导的最小子 DAG）。令 D=（V， E）为 DAG。对于顶点$S \subseteq V$的子集，我们将 S 所诱导的 D 的最小子 DAG 定义为 DAG D'=（V'，E'），其顶点集为 V'= S，当且仅当 u， v∈S， v <V u 时，存在边（v， u）∈E，而且不存在 w∈S\{u， v}，使得 v <v w <v u。

定义 2.6（最大和最小元素）。令 D=（V， E）为 DAG，设≤v 为 D 所诱导偏序，对于顶点 S⊆V，若不存在 v∈S\{u}，使得 u≤v v （v≤v u），则元素 u∈S 称为 S 中 D-最大值（D-minimal），$\max_V$ （S）和 $\min_V$ （S）分别被定义为 S 中 D-最大值和 D-最小值元素集合。

定义 2.7（未来锥和过去锥）。令 D=（V， E）为 DAG。对于 x∈V，D 中 x 的过去锥（记为 $\mathrm{cone}_V^{(p)}(x)$）被定义为使得 x≤v y 的所有顶点的集合 y∈V。同样，D 中 x 的未来锥（记为 $\mathrm{cone}_V^{(f)}(x)$）被定义为使得 y≤v x 的所有顶点的集合 y∈V。

定义 2.8 （过去闭合集）。令 D=（V， E）为 DAG。当且仅当对于每个 u∈S，过去锥 $\mathrm{cone}_V^{(p)}(u)$包含在 S 中时，子集 S⊂V 被称为 D-过去-闭合集。

3.节点与参与

在较高层面，DLT 可分为需许可和无需许可网络。在需许可环境中，只有特定参与者可以参与，而在无需许可环境中，任何人均随时加入网络。在需许可网络中，参与者要么有读取权限，要么有写入（验证）权限。 「完全」需许可（或私有）DLT 提前选择参与者，并将网络所有活动局限于这些参与者中。这与无需许可网络形成对比，任何人均可参与网络并验证账本。我们协议可以在这两种环境中工作，在参与节点上采用通用权重函数。在无需许可环境中，此权重函数作为 Sybil 保护，而在需许可环境中，此权重函数负责调节参与者影响力。

在 3.1 节中，我们将介绍被称为节点的网络参与者。在第 3.2 节中，我们将描述一种为节点分配特定权重的 Sybil 保护机制。最后，在 3.3 节中，我们将讨论如何通过权重控制节点写入能力。

3.1. 网络

DLT 网络参与者称为节点，我们用 N：= {1， . ., N}来表示所有节点集合，其中 N 是节点总数。从先验角度看，不同节点可能对节点集有不同感知。比如，在无需许可环境中，要将一个节点加入网络，只需知道单个节点入口就足够了。为了更好进行表示，我们假设每个节点均知晓所有其他节点。节点通过双向通道直接与其他节点子集（即相邻点）进行通信。所有节点共同建立点对点（P2P）覆盖网。节点以公钥密码进行识别，其唯一节点 ID 来自公钥，而其所有区块都使用私钥进行签名。

与其他 DLT（其中节点可以划分为不同函数类别）不同的是，我们假设所有节点行为均相同。具体地说，所有节点都有两个主要角色。第一个是通过网络传播特定区块，与相邻点来回传输这些区块。第二个是节点通过建立新区块并将它们添加到数据结构中，隐性地前面区块及其包含的交易状态进行投票。该过程称为 Tangle 线上投票（OTV），见第 6 节。对于投票部分，我

们假设有一个稀缺资源，见第 3.2 节。该资源为每个节点赋予一定权重，并将此权重用于隐式投票过程。

3.2. Sybil 保护

在无需许可分布式系统中，一个常见问题是容易产生大量节点，这点也称为 Sybil 攻击。为此，任何关键组件都必须确保节点动作是有限的，否则攻击者获得太大影响力并破坏协议将变得不费吹灰之力。

为了限制或防止 Sybil 攻击，我们假设每个节点都可以与特定信誉或权重关联，并在所用投票机制中赋予其同等比例投票权。

定义 3.1（权重）。对于给定节点 i∈N，我们有相关权值 w（i），该权值是通过函数$\mathcal{N} \to [0,1]$获得的。假设权重通过归一化处理，即

$$\sum_{i \in \mathcal{N}} \mathbf{w}(i) = 1.$$

上述权重函数在验证过程中起到关键作用，见 4.4-6.4 节。

备注 3.1。我们在 3.3 节中以相同权值控制写入权限。但是请注意，写入和验证权重可能有所不同。

实现该权重的一种常见方法是所谓的资源测试，在该测试中，每个 ID 均需证明自己对特定难获取资源的所有权。由于在加密货币中，用户拥有一定数量稀缺资源（即代币），实用 Sybil 保护机制可基于证明对代币和一定数量抵押品的所有权。

实现权重的另一种方法是通过委托。接着，生成权重的源代币所有者可以将这些权重授权给其选择的任何节点，这么做有几个关键好处。例如，资金拥有者可以将权重授权给提供良好服务的节点，或者在节点行为不符合预期时撤销其权重，以实现有「信誉」系统。在极端情况下，甚至将权重和代币分配「去耦合」，纳入真实信任模型。

通常，由于权重变化或不可避免变更（节点加入和离开），系统权重分布可能随时间发生变化。由于协议异步性，感知权重可能因节点而异。协议设计考虑到了这种影响，允许权向量有一定发散性。这种对不同感知的容许使得协议具有一些附加特性。但更详细谈论节点在权向量看法上的分歧超出了本文的范围。为了简便起见，我们做如下假设。

假设 3.1（关于权重稳定性的约定）。网络所有节点均认为节点 i 权值刚好为 w（i）。假设该权值随时间保持不变。

3.3. 写入权限

协议的分布属性及其所在拜占庭环境对写入权限造成一些约束。这些约束对我们协议而言甚至更至关重要，因为它既不基于领导，也不依赖于矿机和区块创建者这些中间环节。与文献[46]类似，我们需要以下条件：

1) **一致性**：如果一个诚实节点发出的区块被一个诚实节点写入（分布式）数据库，其最终应当被所有诚实节点写入。

2) **公平性**：给定权重函数和最大带宽，节点可以以按其权重成正比的速率发出区块。

3） **安全性**：上述约束在拜占庭环境中是有保证的。

因此协议应该确保在拥堵场景中，只有有限区块被传播，即区块速率以一定吞吐量为限。同时这种传播应当均衡。这些要求可防止节点过载，以及所创建的账本不一致。原则上，这点可通过收费和 PoW 实现，也可通过文献[46]提出的访问控制算法（作为更新颖方案）来实现。

为了保证共识机制的安全运行，我们假设该机制可用。所需工具应当为上述约束提供保证。我们做出如下假设。

假设 3.2（写入权限）。对写入权限进行控制，以保障给定权重函数 w 写入权限的一致性、安全性和公平性。

4. 区块结构和见证权重

在本节中，我们将介绍我们协议数据结构的概念。为在分布式网络复制一定内容，节点必须将该内容封装于区块中[4]。然而，当内容只是简单交易时，我们要求区块在其有效负载中只包含一笔交易。该假设是为了更好表示，可以放宽要求，使区块包含多笔交易。此外，每个区块必须引用至少两个过去发出的区块。后一项要求是通过我们协议无领导架构驱动的，在该架构中，每个节点均可独立发出区块。此外，我们还讨论区块上某个尺度，该尺度被称为见证权重，该权重允许节点可靠了解网络重要部分在何时看到给定区块。

第 4.1 节正式定义区块。第 4.2 节讨论 Tangle，一种由区块及其引用组成的 DAG。第 4.3 节介绍特定节点所见本地版 Tangle。我们利用第 3.2 节介绍的节点权重函数，第 4.4 节局部 Tangle 正式界定给定区块的见证权重，4.5 节展示如何使用该尺度作为区块的确认规则。关于见证权重的增长分析，见第 4.6 节。

4.1. 区块

协议目标是在网络节点之间可靠地复制某些内容。比如，这些内容可以是资金拥有者余额的原子更新结果。这些内容被封装至被我们称为区块的对象中。有意在整个网络向 Tangle 添加某些内容的节点负责组装该区块，该区块包括内容、k 个前面区块的引用和节点签名（见图 2）。我们将组装和初始广播过程称为区块发出。每个接收新区块的节点都将其转发给相邻点。

定义 4.1（区块）。设 ref为(r_y, v)对，其中 r 是对前一个区块的引用[5]，v 是标签值。我们将区块 x 定义为有内容的对象。

$$x = (\{\mathrm{ref}_1(x), .., \mathrm{ref}_k(x)\}, \hat{x}, \mathrm{nodeID(x)})$$

其中$\hat{x}$为一笔交易，节点 ID（x）识别发出的节点。

[4] 在以往研究中，我们将该对象称为消息。

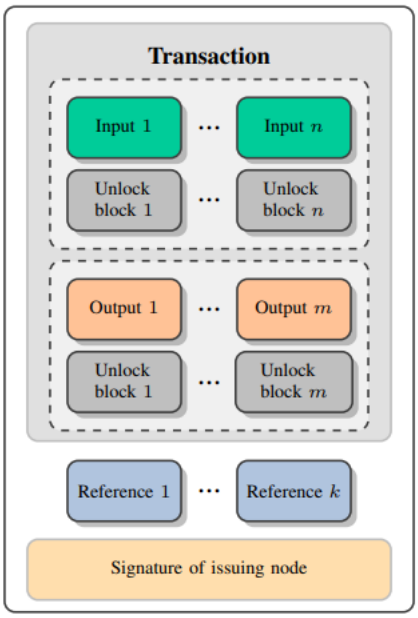


图 2：以交易为内容的简化区块布局。资金拥有者向节点提供交易。节点将交易封装于区块中，并在区块上签名。

备注 4.1：标签 v 表示引用或投票类型，我们将在后面第 6.3 节中看到。

发出的节点通过与内容发布者的服务端-客户端关系获取内容，这点可通过调用应用程序编程接口（API）实现。或者，节点本身也可以是内容发布者。内容基本应用是资金转移，即输出结果的消耗和创建。我们将此类内容称为交易。在本文中，为了便于表示，我们假设所有区块在其有效负载中均包含交易。但通常来说，区块并不仅限于此用例。

区块也将用于传播选票，所以跟踪发出的节点至关重要。

定义 4.2（区块发布者）。对于区块 x，发出 x 的节点表示为 issue（x），其中 issue（x）∈N。

4.2. Tangle

正如 Tangle 原始论文[3]所述，Tangle 是按照以下规则构建的数据结构的：「为了发布[区块][6]，节点选择另外两个[区块]进行审批。」

更笼统地说，我们通过允许区块引用最多 k 个现有区块来修改它。数据结构采用 DAG 的形式，其中区块对应顶点，而引用形成边。

[6] 最初白皮书中所使用的术语是交易，但在本文中，我们将区块及其包含的交易区分开。

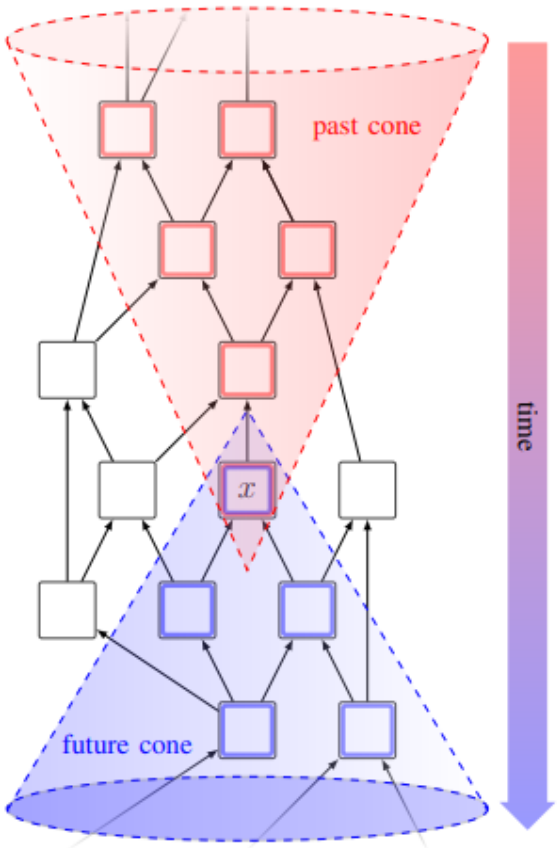


图 3：Tangle 中区块 x 的未来锥和过去锥

让我们更正式地定义该数据结构。我们用 T 表示一组区块。有个特殊区块叫做原点，用 p 表示。该区块不包含任何引用。其他区块都必须直接引用至少两个（不一定不同）区块。因而，引用关系可编码至 DAG 中。

定义 4.3（Tangle）。Tangle $D_{\mathcal{T}}$ 是一个 DAG，其顶点集是区块 T 的集合。当且仅当 y 直接指向 x 时，y 和 x 之间有一条有向边。

我们利用第 2 节符号，以≤r 表示由 $D_{\mathcal{T}}$ 诱导的区块集的偏序。对于区块 x∈T， x 的 Tangle 过去锥和未来锥分别用 $\mathrm{cone}_{\mathcal{T}}^{(p)}(x)$ 和 $\mathrm{cone}_{\mathcal{T}}^{(f)}(x)$ 表示。x 的父节点和子节点分别记为 $\mathrm{par}_{\mathcal{T}}(x)$ 和 $\mathrm{child}_{\mathcal{T}}(x)$。如果 $x <_{\mathcal{T}} y$，我们说区块 x 赞成或引用区块 y。确切地说，如果 $x \in \mathrm{child}_{\mathcal{T}}(y)$，则 x 直接引用 y；如果 $x \notin \mathrm{child}_{\mathcal{T}}(y)$，则 x 直接引用 y，如果 $x \notin \mathrm{child}_{\mathcal{T}}(y)$，$x <_{\mathcal{T}}$，则 x 间接引用 y。Tangle DAG 的叶子节点称为末梢。

例 4.1。我们参考图 3 解释区块 x 的 Tangle 和 Tangle 未来锥和过去锥。

4.3. 局部 Tangle

由于网络的分布性，节点可在不同时间点接收区块，甚至不按顺序。节点第一次接收区块的时间称为到达时间。

在广播期间区块可能会丢失。虽然这通常可能造成问题，但 Tangle DAG 可获得得体的解决方案，通过被称为凝固的过程来弥补损失。如果某一节点收到某个父节点未知的区块，它会向对等节点索取缺失区块。一旦收到缺失的父区块，过去锥便宣告完成（除非其父区块缺失——这

种情况下，节点必须递归地重复此过程）。一旦区块过去锥完成，节点就会将该区块标记为实心区块。区块 x 在节点 i 上凝固调度示为 $\tau_{s,i}(x)$。我们只考虑在标记为实心后，Tangle所包含的区块。

为此，我们可以认为网络中不存在单一 Tangle 这类东西，因为每个节点可能对它有不同感知。为此，在 t 时刻，节点 i 只知道满足 $\tau_{s,i}(x) \leq t$ 的区块 x。我们用 $\mathcal{T}_{i,t}$ 和 $D_{\mathcal{T}_{i,t}}$ 分别表示区块集的局部感知和节点 i 在（局部）t 时刻感知到的 Tangle DAG。接着，用局部形式 $\text{cone}^{(f)}_{\mathcal{T}_{i,t}}(x)$ 和 $\text{cone}^{(p)}_{\mathcal{T}_{i,t}}(x)$ 给出过去锥和未来锥。如果从语境可以清楚地看出对 i 和 t 的依赖性，我们就省略下标，简单地写 $D_{\mathcal{T}} = D_{\mathcal{T}_{i,t}}$。

4.4. 见证权重和加权局部 Tangle

在最初 Tangle 白皮书[3]中，区块累积权重在共识发现中发挥作用。累积权重指的是引用特定区块的区块数量。在出现冲突的情况下，节点跟随的是 Tangle 中包含最大累积权重的部分。

我们将该基本理念运用到每个节点都具有一定权重的环境中。这样，节点权重便取代区块创建中的 PoW，成为 Sybil 保护机制。每个区块节点签名将发出区块的节点链接至该区块（见第 4.1 节）。为此，一个节点可以关联到该节点发出的 Tangle 上的区块集，而节点的权重可映射到这些区块。

定义 4.4（区块支持者和见证权重）。令 $x \in \mathcal{T}_{i,t}$ 为一个区块，用 $\text{sprt}_{\mathcal{T}_{i,t}}(x)$ 表示，它是在 x 未来锥中发出区块的节点集：

$$\text{sprt}_{\mathcal{T}_{i,t}}(x) = \left\{ j \in \mathcal{N} : \exists y \in \text{cone}^{(f)}_{\mathcal{T}_{i,t}}(x),\ j = \textbf{issue}(y) \right\}$$

我们把来自 $\text{sprt}_{\mathcal{T}_{i,t}}(x)$ 的节点称为 x 的支持者。定义函数 $\mathbf{WW}_{i,t} : \mathcal{T}_{i,t} \to [0,1]$，该函数被称为节点 i 在 t 时刻看到的区块的见证权重（WW），如下所示：

$$\mathbf{WW}_{i,t}(x) := \sum_{j \in \text{sprt}_{\mathcal{T}_{i,t}}(x)} \mathbf{w}(j). \qquad (2)$$

当总权重被归一化为 1 时，WW 表示赞成给定区块的权重百分比。只要语境清晰明了，我们将省略索引 i 和 t。

例 4.2。在图 4 中，我们给出赞成给定区块 x、y 和 z 的节点集的实例。在区块底部用唯一颜色表示发出的不同节点的签名。很容易看出 $\text{sprt}_{\mathcal{T}}(x)$ 由棕色、青色和灰色所对应的节点组成。

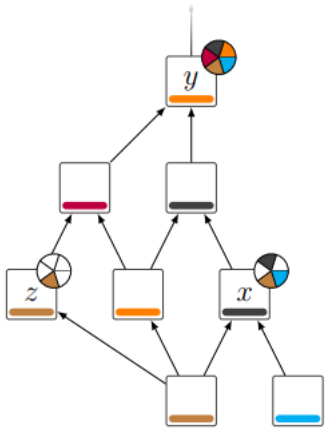


图 4：Tangle DAG，区块发出的节点可通过区块底部显示的唯一颜色来鉴别。区块 x、y 和 z 支持者颜色绘于右上角。

引理 4.1（WW 单调性）。对于使 x ≤T y 的任意两个区块 x， y∈T， $\mathrm{sprt}_{\mathcal{T}}(x) \subseteq \mathrm{sprt}_{\mathcal{T}}(y)$ 成立，为此 **WW**（x） ≤**WW**（y）。

引理 4.2（WW 的增长）。对于使 $t_1 < t_2$ 的任意区块 x∈T，节点 i∈N，以及瞬间时刻 t_1 和 t_2， $\mathrm{sprt}_{\mathcal{T}_{i,t_1}}(x) \subseteq \mathrm{sprt}_{\mathcal{T}_{i,t_2}}(x)$ 成立，为此 $\mathbf{WW}_{i,\,t1}$（x） ≤ $\mathbf{WW}_{i,\,t2}$（x）。

第 4.6 节提供关于 WW 在某些假设下增长的更细致分析。

4.5. 区块确认规则

区块流受到写入权限控制，见 3.3 节。从先验角度看，该控制本身可能不足以保证所有节点都看到网络中所有区块。但为确保系统安全性，节点必须对数据集 T 中哪些区块应当被永久接受达成共识，否则节点之间将出现矛盾。如果达成这种共识，我们就认为区块得到了确认。此外，为保持数据结构 $D_{\mathcal{T}}$ 一致性，区块 x 只有在 $\mathrm{cone}_{\mathcal{T}}^{(p)}(x)$ 所有区块都被确认的情况下才能得到确认。

提供区块确认状态信息，同时考虑到特定安全性和存活性的工具，通常称为确认规则。我们根据区块 WW 概念设计该工具。WW 允许节点和用户创建自己的主观确认标准。一个区块 WW 越大，该区块永远保存在账本中的概率越高。该理念和区块链交易的「深度」相似。为此，实际确认标准可能取决于协议环境和底层用例。

定义 4.5（经确认区块）。令 θ∈（0.5，1]为固定阈值。我们说对于某个 s≤T，若 $\mathbf{WW}_{i,s}(x) \geq \theta$，则在 t 时刻，对于节点 i∈N，区块 x∈T 是确定的。

一旦某个区块被确认为节点，它将永远保持确认状态。该确认状态的不可逆性对该状态的收敛性提出很高要求。更具体地说，一旦某个节点达到给定区块阈值，所有节点最终都应以非常高概率达到该阈值。

在真实场景中，该假设很容易得到满足，因为较高 WW 也意味着大量节点将「看到」给定区

块，发出赞成该区块的区块。如果选对默认末梢选择算法，且有足够多节点跟随，那么所有节点都将最终以非常高概率将区块附加到该区块未来锥上（详见 4.6 节）。在第 8 节中，我们将详细讨论协议的存活性和安全性。

4.6. 见证权重的增长

在本节中，我们将建模区块的发布，并讨论 WW 的增长及其对协议环境的依赖性。

我们考虑以下假设。

假设 4.1（发出速率）。每个节点 i∈N 以泊松速率 Ai（每秒）发出区块。速率 Ai 与对应权重 w（i）成正比（见定义 3.1），即对于某个常数$\lambda > 0$，$\lambda_i = \lambda \mathbf{w}(i)$。我们假设每个节点都独立于其他节点发出区块。所有节点发出速率为

$$\lambda = \sum_{i \in \mathcal{N}} \lambda_i.$$

备注 4.2 根据假设 4.1，从节点 i∈N 出发的两个相邻区块之间时间是独立的，且与参数 λ_i 呈指数分布。

为了给 WW 开发启发式，我们运用以下方法。假设存在一个「无所不知的观察者」，它可以立即意识到所有节点发出的所有区块。观察者对状态的感知可能不同于对给定节点的感知，但这些差异对启发式结果并没有实质影响。我们参考文献[47]和[48]，其中此方法已被证明能够得到很好启发式。这个观点通过省略索引 i 体现在符号表达中。例如，Tt 表示在 t 时刻，这个无所不知观察者所感知到的区块集，而 $\mathbf{WW}_{t\,(x)}$ 表示在 t 时刻区块 x 对应的 WW。

设 x 为 t_0 时刻发出的区块，用 $E_i(\delta, x)$ 表示，即节点 i 在 x 未来锥时间间隔 $[t_0, t_0 + \delta]$ 内发出区块的事件。对于 $t = t_0 + \delta$，无所不知的观察者所感知到的区块 x 的 WW 满足

$$\mathbf{WW}_t(x) = \sum_{i=1}^{N} \mathbf{w}(i) \mathbf{1}\{E_i(\delta, x)\}. \qquad (3)$$

节点 i 以 $\lambda \mathbf{w}(i)$ 速率发出区块，于是有

$$\mathbb{P}(E_i(\delta, x)) \leq 1 - \exp(-\delta \lambda \mathbf{w}(i)). \qquad (4)$$

注意等式不一定成立，因为并非所有新入区块都必须见证区块 x。取式（3）期望值，运用不等式（4），我们得到

$$\mathbb{E}[\mathbf{WW}_t(x)] \leq \sum_{i=1}^{N} \mathbf{w}(i) \left(1 - \exp(-\delta \lambda \mathbf{w}(i))\right). \qquad (5)$$

式（3）给出的公式适用于非常普遍的情况。但对于协议分析，重要的是考虑具体权重分布。或许最适合权重分布建模取决于普遍现象。关于这一普遍现象最著名的例子是中心极限定理。虽然中心极限定理适用于描述数值具有相同量级的统计量，但并不适用于建模更异质的情形，即数值相差好几个量级。这些异质情形往往用齐波夫定律来描述，且出现在许多领域，如城市人口、互联网流量数据、P2P 社区形成、公司规模和科学引用。我们参照文献[49]的简介和更多参

考，并参照文献[50]-[52]了解因特网、计算机网络和 DLT 中的齐波夫定律。

我们考虑存在 N 个元素或节点的情形。齐波夫定律预测秩为 r 的节点的（归一化）权重可通过下式求出：

$$\mathbf{w}(r) = \frac{r^{-s}}{\sum_{j=1}^{N} j^{-s}}, \qquad (6)$$

式中，$s \in [0, \infty)$为齐波夫参数。由于式（6）中的、权重 w（.）只依赖于、两个参数 s 和 N，这为在广泛网络情境中研究协议性能提供方便的模型。比如，具有相等权重的 N 个节点同构网络可通过选择 s=0 来建模。随着 s 值的增加，网络变得越来越集中。

例 4.3 我们参考图 5。WW 增长取决于几个因素，尤其是发出率

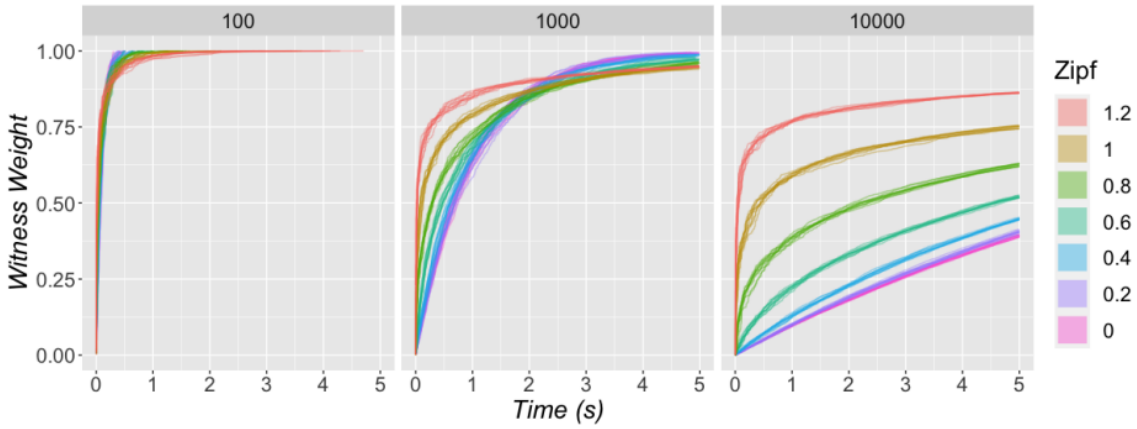


图 5:例 4.3 中以每秒 1000 区块发出的 WW 的增长率。我们看到 100 个节点（左）、1000 个节点（中）和 10000 个节点（右）不同行为。增长主要取决于所选齐波夫参数 s（以颜色表示）和节点数量。

λ 和节点权重分布。对于齐波夫分布，权重取决于两个参数，即节点数 N 和齐波夫参数 s。上限（5）为 δ 和 λ 的单调凸函数，对参数 N 和 s 的依赖性不是很明显。为此，我们对 N∈{100, 1000, 10000}，s∈{0, 0.2, 0.4, 0.6, 0.8, 1, 1.2}和 λ=1000[7]进行一些蒙特卡罗模拟。对给定的 t_0，我们将 t_0 时刻 WW 近似为在时间间隔$[t_0; t_0 + t]$内发出区块的所有节点权重之和。

这就为 t_0 发出的区块的 WW 提供下限估计值。在图 5 中，每行均对应已发出 WW 增长的一个实现（对于 t_0=0 而言）。

4.7. 确认时间估计

正如第 4.5 节讨论的，确认规则对于许多用例而言是必不可少的，确认时间（TTC）无疑是每个共识协议的重要性能指标。由于深入分析 TTC 超出本文的范围，我们在本节中给出第一个「启发式」上限。

[7] 在前 100 个地址的流行加密货币中发现的齐波夫参数典型值在 s=0.7 到 s=1.2 之间[53]。

定义 4.6 （确认时间）。我们将节点 i 对于区块 x（在 θ 层面）的确认时间定义为：

$$\tau_{f,i} = \tau_{f,i}(x) := \inf\{t > 0 : \mathbf{WW}_{i,t}(x) \geq \theta\} - \tau_{s,i}(x), \tag{7}$$

式中，$\tau_{s,i}(x)$为 x 的凝固时间（见 4.3 节）。

在本节其它部分，我们省略索引 i∈N，因为所提供的分析和所有节点有关。我们将 TTC 分成两个阶段。在第一个阶段，我们等待汇合时间 $\tau_c = \tau_c(x)$，直到给定区块 x 被包含在（几乎）所有当前末梢过去锥中。在第二阶段，发出时间 τ_{iss}，我们令 WW 增长，直到达到阈值 θ，则 TTC T_f 上界为

$$\tau_f \leq \tau_c + \tau_{iss}. \tag{8}$$

τ_{iss} 的估计值是通过式（3）得到的，此公式可简化为权重特定选择方法（见例 4.5）。

例 4.4 我们借助图 6，显示汇合时间和发出时间。带实心框架的区块位于区块 x 未来锥内。在汇合时间之后，所有区块均赞成 x。黄色、绿色和紫色以权重较大的节点表示区块，即节点对确认影响比较大。一旦 x 未来锥中发出的节点累积权重达到阈值 6，区块就会被确认。

我们可以通过一些附加假设得到类似于文献[3]的汇合时间估计值 τ_c。我们的第一个假设是，从区块的创建到网络其他节点接收到此区块之间的延时是恒定的。

假设 4.2（网络延迟恒定）。我们假设从区块的创建到其他节点接收到此区块之间的时间等于某个常数 h。

定义 4.7（末梢数量）。令 L（t）为 t 时刻 Tangle 末梢总数。

如第 4.3 节所述，并不存在「客观的 Tangle」，每个节点都有自己的感知。虽然，既往研究[47]表明，本节中所做近似导致对 Tangle 某些定量属性的合理近似，如末梢数量和汇合时间。为此，我们省略下标「i」，并在本节中采用唯一目标 Tangle。与文献[3]相似，我们假设末梢数量处于固定状态。

假设 4.3（Tangle 宽度恒定）。我们假设 Tangle 末梢数量 L（t）固定，其均值为 L_0。

我们使用假设 4.1、4.2 和 4.3，遵循[3，第 3 节]所述启发式。第一个观察结果是，在任何给定时间 t，平均有 λh 个隐藏末梢，这些区块在 t-h 之后发出，但对于网络而言还不可见。与文献[3]一样，我们假设通常存在 r 个露出末梢，这些末梢在 t-h 之前已经附加，但现在仍然是末梢。因此，末梢总（平均）数可以写为 as $L_0 = r + \lambda h$。根据假设 4.3，我们认为末梢数量 L（t）大致是固定的。这意味着，自 λh 末梢加入末梢池后，在同一时间内，约 λh 个在 t-h 时刻属于末梢的区块被引用，现在已不再是末梢。为此，大小为 L_0 的末梢池可分成 r 个露出的末梢和 λh 个不再是末梢的区块。这种划分可产生重要的观察结果：一个新区块（有 k 个父节点）平均赞成$kr/(r + \lambda h)$ 个（露出的）末梢。此外，在末梢池大小 L_0 近似恒定的平稳状态下，所选末梢平均数量应当为 1;否则，末梢数将会改变。

求解$kr/(r+\lambda h)=1$，得到

$$L_0 = L_0^{(k)} = \frac{k\lambda h}{k-1}. \qquad (9)$$

正如文献[3]预测的，此结果已通过文献[47]和[48]模拟研究和文献[54]理论结果得到证实。

文献（9）第一个结果是，如果 L_0 很大，则区块第一次被赞同的预计时间近似于：

$$h + L_0/(k\lambda) = h + \frac{h}{(k-1)}. \qquad (10)$$

自然地，末梢池大小与给定区块的 WW 增长有关：末梢池越大，WW 增长越慢。

备注 4.3。对于任意给定 λ 和 h，我们可选择足够大的 k，使得$k > L_0^{(k)}$。在这种情况下，区块基本在可见后立即被引用，但代价是区块的尺寸更大。

我们可通过类似于文献[3]的操作来求解

$$\tau_c \approx \frac{h}{W\left(\frac{(k-1)^2}{k}\right)}\left(\log L_0 + \log \varepsilon\right), \qquad (11)$$

式中，log 表示自然对数函数，W 是朗伯 W 函数的主要分支，被定义为$z = we^w$的逆函数，即 w=W（z）。对于较的 k，我们可以采用以下近似：

$$W\left(\frac{(k-1)^2}{k}\right) \approx 2\log(k-1) - \log k \approx \log k$$

由此求出

$$\tau_c \approx \frac{h}{\log k}\log(L_0) \approx \frac{1}{\log k} h \log(\lambda h). \qquad (12)$$

在第 A 节，我们将给出更多关于汇合时间推导的细节。

例 4.5 发出时间τ_{iss}的行为在很大程度上取决于实际权重分布，如图 5 所示。但所有节点具有相同权值这种极端情况可以更有条理地进行处理。这里极端指的是 WW 增长在某种程度上是最小的。因此对于所有节点，$i \in \mathcal{N}$，设 w（i）=1/N，并假设我们想要得到确认时间的界限，即给定区块 x 第一次达到 $\mathbf{WW}_t$（x） >θ 的时间。

$$X_{(i)} \sim \frac{1}{\gamma}\sum_{j=1}^{i}\frac{Z_j}{N-j+1}, \qquad (13)$$

式中 Z_j 是参数为 1 的 i.i.d.指数分布随机变量。最后，[θN]节点发出一个区块所需时间分布为 $X_{(\lceil\theta\cdot N\rceil)}$。期望值公式如下：

$$\mathbb{E}[X_{(i)}] = \frac{1}{\gamma}\sum_{j=1}^{i}\frac{1}{N-j+1},$$

其中，i =[θ·N]。通过使用上述和的标准积分近似，当 N 较大时，我们得到

$$\mathbb{E}[X_{(i)}] \approx \frac{N}{\lambda}\left(\log(N) - \log(N-i)\right)$$

因此，对于 i=θN，

$$\tau_{iss} \approx \mathbb{E}[X_{(i)}] \approx \frac{N}{\lambda}\left(1 - \log(1-\theta)\right).$$

将该结果与文献（8）中汇合时间的界限（12）相结合，我们可得到对于较大 k （其他参数固定），TTC 的渐进上限如下：

$$\tau_f \lesssim \frac{1}{\log k}h\log(\lambda h) + \frac{N}{\lambda}\left(1 - \log(1-\theta)\right)$$

5. 账本

本节将介绍几个表示交易及其相互关系的新概念。回顾在标准 UTXO 无冲突模型中，交易将之前交易输出指定为输入，并通过花费（或消耗）输入来创建新输出。没有两笔交易消耗相同输入。这种无冲突数据结构可在由共识机制过滤交易的网络中实现。后者通常是通过在参与者中选择「领导」来完成的，领导将交易区块添加到无冲突账本中。为了绕过这一「集中」瓶颈，我们提出基于现实的 UTXO 账本概念，这是标准无冲突 UTXO 账本的增强版，可以有不止一笔输出花费。我们建议读者参阅平行研究[15]，在这些研究中，我们已详细讨论所有概念。

在第 5.1 节中，我们回顾 UTXO 模型和账本中交易的定义，账本是所有交易的集合。在第 5.2 节中，我们介绍冲突交易、冲突和分支的定义，它们代表着「非冲突冲突」的适当子集。现实是最大可能分支，它将账本限制为无冲突 UTXO 账本中的现实结果。

5.1. UTXO 模型和交易

在未花费交易输出（UTXO）模型中，交易将之前交易的输出指定为输入，并通过创建新输出来花费它们。

因此，交易由输入列表和输出列表组成，见图 2。注意输出必须是唯一的。唯一性通常是通过创建涉及哈希函数的输出 ID 来实现的。笔如，输出 ID 可以是输出索引和交易内容哈希值的拼接。每个输出都代表特定数量底层加密货币。所有输入数值（即花费的输出）都必须等于一次交易所有输出的数值。每个输出都有一个声明，即在什么条件下由谁来使用它。解锁区块包含证明，比如证明某个输入地址所有权的签名，显示交易发出者被允许使用输入。我们参考了图 2 的一般交易布局。如第 4.1 节所述，区块在其有效负载中包含交易。接下来，我们用 X 表示区块 x 有效负载中包含的交易。

让我们更正式地定义交易和账本模型。我们运用了文献[55]和[15]的方法。

定义 5.1（输出和输入）。输出是一个数值对$v \in \mathbb{R}^+$和一个解锁条件 cond。我们用$o = (v, \text{cond})$表示输出。输入 i 是对输出的引用。我们说输入消耗了输出。

定义 5.2 （交易）。交易 X 是输入 in$(\hat{x})$、输出 out$(\hat{x})$和解锁数据的集合。

解锁$(\hat{x})$，式中

1） in$(\hat{x}) = (i_1, \ldots, i_n)$是输入列表，即对未消耗输出的引用。我们说，这些输出由是交易 x 花费或消耗的；

2） out$(\hat{x}) = (o_1, \ldots, o_m)$是由交易 X 产生的新输出列表；

3） unlock$(\hat{x})$是对输入进行解锁的数据，往往通过加密授权证明来完成，该证明保证交易发出者满足被消耗输出的条件 cond。

定义 5.3（账本）。账本是一组交易，记为 L。

UTXO 账本从所谓原点开始，包含输出但不包含输入。我们强调，对所有区块和所有交易最终前身使用相同术语。经回顾，原点-区块记为 p，而原点-交易用$\hat{\rho}$表示。

通常，每个输出最多可被一笔交易所消耗，因此，所有未花费输出数值总体上是保守的。具体来说，在标准无冲突 UTXO 模型中，账本不能包含所谓双花，即两笔交易消耗一笔交易的相同输出。

在下一节，我们将放宽这种无冲突限制，同时允许账本包含冲突交易。

5.2. 基于现实账本

在本节中，我们提出标准无冲突 UTXO 账本模型的增强版，该模型允许包含双花。我们建议用不同结构来跟踪冲突交易，而不需要达成共识。关于更详细描述，建议读者参见[15]。

首先，我们解释交易及其输入和输出如何形成 DAG 结构。纳入账本 DAG 的信息被分成冲突图，该冲突图只跟踪冲突交易。接着，我们引入分支的概念。分支形成账本的一种可能的非冲突状态。然后获得名为现实的概念，它使我们能够将 L 简化为交易最大子集，产生无冲突（基于现实）账本。

定义 5.4（账本 DAG）。我们将账本 DAG$^{D_{\mathcal{L}}}$定义为顶点集为账本 L 的 DAG。当且仅当$\hat{x}$的输入引用输出$\hat{y}$时，$D_{\mathcal{L}}$边集存在有向边 $(\hat{x}, \hat{y})$。

通过第 2 节中的符号，我们用$\leq_{\mathcal{L}}$表示$D_{\mathcal{L}}$诱导的交易集偏序。用$\text{cone}^{(p)}_{\mathcal{L}}(\hat{x})$表示交易$\hat{x}$的过去锥，即交易$\hat{x}$直接或间接从$\text{cone}^{(p)}_{\mathcal{L}}(\hat{x}) \setminus \{\hat{x}\}$交易中花费价值。

备注 5.1 在文献[15]中，我们讨论 UTXO DAG 的概念，它将所有输入和输出以及所有交易 ID 作为顶点集。UTXO DAG 的边集则是由这些顶点之间关系组成的。在本文其它部分，我们将重点关注账本 DAG，因为它更简单。

通常来说，在向此类数据结构添加交易时，只能添加与之前记录交易不产生冲突的交易，即账

本 DAG 是无冲突的，但需要一个预选交易共识机制。

文献[15]提出一种新账本设计。在此设计中，这种约束被更宽松的约束取代——即如果$\mathrm{in}(x)$是$\mathrm{cone}_{\mathcal{L}}^{(p)}(\hat{x})\backslash\{\hat{x}\}$中已经消耗输出的引用，则新交易$\hat{x}$可以被添加到账本中。下面我们将概述允许冲突交易共存的拟议解决方案的一些最重要概念，

定义 5.5（冲突）。如果两笔不同交易$\hat{x},\hat{y}\in$至少有一个共同输入，它们就会直接产生冲突。当且仅当存在交易 $\hat{y}\in\mathcal{L}\setminus\{\hat{x}\}$，使$\hat{x}$ 和$\hat{y}$直接冲突时，交易 $\hat{x}\in\mathcal{L}$被称为冲突。

定义 5.6（冲突交易）。如果存在不同$\hat{x}_2,\hat{y}_2\in\mathcal{L}$，其中$\hat{x}_1\preceq_{\mathcal{L}}\hat{x}_2$， $\hat{y}_1\preceq_{\mathcal{L}}\hat{y}_2$，使得 x2 和 y2 直接冲突，将两笔不同交易$\hat{x}_1,\hat{y}_1\in\mathcal{L}$可称为有冲突。

在冲突 DAG 和冲突图帮助下，我们可编码冲突之间相互关系。

定义 5.7（冲突 DAG 和冲突图）。所有冲突集合用 C 表示，称为账本$\mathcal{L}$冲突集。我们将冲突 DAG Dc 定义为$\mathcal{C}\cup\hat{\rho}$所诱导的账本 DAG 最小子 DAG（见定义 2.5）。我们将冲突图 Gc 定义为顶点集为 C 的图，当且仅当这些冲突产生冲突时（作为交易），两个冲突由一条边连接。

我们可根据交易之间是否存在冲突对交易进行分组。

定义 5.8（无冲突集和冲突集）。如果交易子集$S\subseteq\mathcal{L}$不包含任何两笔冲突交易，则称为无冲突交易。如果没有$\hat{x}_1\in S_1$和$\hat{x}_2\in S_2$，使得$\hat{x}_1$和$\hat{x}_2$冲突，我们也称$S_1\subseteq\mathcal{L}$对于$S_2\subseteq\mathcal{L}$是无冲突的。或如果S_1对于S_2不是无冲突的，则S_1与S_2有冲突。

我们进一步专门研究无冲突集，并引入分支的概念。

定义 5.9（分支和分支集）。当且仅当以下两个属性成立时，一组冲突 B⊆C 称为分支：

1) B 无冲突（见定义 5.8）；

2) B 为 Dc–过去-闭合集（参见定义 2.8）。

将 β 定义为所有分支集合。代表空集的分支称为主分支。

现在，我们引入「现实」这一概念，这一概念可定义为冲突图中最大可能分支或等效的最大独立集。换言之，在保留非冲突性质的同时，现实聚集了最大数量冲突。

定义 5.10（最大分支和现实）。如果不存在其他分支 A∈B 使 B⊂A，则分支 B∈β 是最大的。最大分支称为现实。

下面我们将描述给定交易最大内含分支的概念，该分支由给定交易过去锥中冲突交易的集合组成。

定义 5.11（最大内含分支）。设 β 为所有分支集合，分支$\iota_{\mathcal{L}}^{(p)}:\mathcal{L}\to\mathcal{B}$是一个函数，对于给定交易$\hat{x}\in\tilde{\mathcal{L}}$，它可返回$\mathrm{cone}_{\mathcal{L}}^{(p)}(\hat{x})$所包含的最大分支。

在文献[15]中，我们证明此定义是正确的，只有一个最大内含分支。

定义 5.12（现实账本）。设 R∈β 为现实。将 R 账本（记为$\mathcal{L}(R)$）定义成使得分支$\mathcal{L}^{(p)}(\hat{x}) \subseteq R$的所有交易集合$\hat{x} \in \mathcal{L}$。

在文献[15]中，我们证明$\mathcal{L}(R)$是无冲突的且代表统一 UTXO 账本。此外，文献[15]还讨论分支和现实的不同属性以及基于现实账本的操作，如添加和修剪冲突。

备注 5.2（本地账本）。如第 4.3 节所述，Tangle DAG 可能有主观版本。同样，每个节点对账本都有自己的看法。为此，当我们讨论节点 i∈N 在 t 时刻的观点时，我们在$\mathcal{L}$、$D_{\mathcal{L}}$ 和其他相关概念使用下标 i 和 t。

为了发布新区块并验证交易，网络中每个节点必须选择自己喜爱的账本部分。一旦选中现实 R，节点就可对 R 账本进行不同操作。

定义 5.13（偏好现实）。节点 i∈N 在 t 时刻选择某个现实$R = R_{i,t} \in \mathcal{B}$，该现实称为节点 i 的偏好现实。

备注 5.3（现实选择算法）。在文献[15]中，我们针对一类通用权重函数提出一种现实选择算法。在本文中，我们给出该权重函数在赞成权重方面的具体例子（见 6.4 节）。

在第 10 节中，我们提出一种基于赞成权重和常见随机数的权重函数变体。

6. Tangle 线上投票

在本节，我们提出基于 Tangle 和账本 DAG 的投票机制。该机制能够在基于现实账本中选择现实。

在第 6.2 节中，我们概述两种适用的 DAG 结构，这两种结构可用于对现实进行投票。第 6.3 节将这两种结构整合为一个投票 DAG，并介绍由此产生的基本概念。我们也讨论对两种 DAG 进行投票如何提高协议存活性。第 6.4 节定义了被称为赞成权重的指标，该指标在第 6.5 节中用于识别偏好现实，并用适当末梢选择算法投票给它。

6.1. 见证权重和存活性问题的延伸

在第 4 节中，我们介绍见证权重，这种权重是用于确认区块的指标。本节将寻找用于确认交易的类似工具。

见证权重具有单调递增特性，因为它代表见证区块存在的权重之百分比。对于想要通过节点权重在冲突交易之间做出决定的交易，则情况有所不同。为了保证存活性，节点必须有改变其投票的可能，并从给定交易的赞成权重中撤回其权重[8]。然而，改变看法可能意味着对带被拒交易的区块进行引用（和投赞成票）的区块可能永远不会被确认。

这种情形造成引用新末梢的消极动机。更确切地说，可能刺激节点只引用来自可信实体的区块或一定年龄的末梢，或最糟糕的是古老且已确认的区块。后者可能最终导致不再有新区块被确认。目前，上述问题一直是基于 DAG 共识协议（如文献[3]）所关注的重要问题。我们提出通过使用基于现实账本和扩展引用方案来解决这些问题。

6.2. 不可变 DAG

[8] 相反，如果权重是增加而不是撤回，可能两笔冲突交易获得完全相同权重，从而造成僵局。

区块是网络主要信息载体，即其包含交易并表达发出节点的看法。区块中的引用，再加上节点签名和输入解锁区块，形成两个不可变数据结构，类似于区块链。

首先，在区块集 T 上构建 Tangle D_T。用区块中包含的引用定义相互关系，由发出节点进行选择和签名（详见第 4 节）。其次，账本 DAG $D_{\mathcal{L}}$ 是建立在交易集 $\mathcal{L}$ 上的，二者之间相互关系是根据输入消耗定义的，而这些输入是之前交易的输出。输出的消耗和创建由资金拥有者签名进行加密验证（详见第 5 节）。

为了使节点客观地就事件偏序达成一致，我们需要做出如下假设。

假设 6.1（过去锥完整性）。若交易 x 花费交易 y 所创建的输出，则区块 x 包含在 y 的 Tangle 未来锥中成立，即 $x \in \text{cone}_T^{(f)}(y)$。

换言之，我们有合理假设，即输出的花费应当发生在「创建」这些输出区块的未来锥中。

引理 6.1。根据假设 6.1，D_L 所诱导的偏序 $\preceq_L$ 与 D_T 所诱导的偏序 $\preceq_T$ 一致。更具体地说，如果对于某些区块 x，y∈T，我们有相应的交易满足 $\hat{x} \preceq_{\mathcal{L}} \hat{y}$，则 $x \preceq_T y$ 成立。

证明。此命题可通过 D_L 中 $\hat{x}$ 和 $\hat{y}$ 之间最短路径长度归纳并简单表示。当路径长度为 1 时，假设 6.1 隐含了基本情况。

6.3. 投票和投票 DAG

引理 6.1 的结果是，Tangle 和账本 DAG 均适合让节点表达它们对冲突交易中更喜欢哪笔交易的看法。更具体地说，通过创建和附加新区块，节点有隐式方式来投票选中「正确」分支。让我们更确切定义它。

我们使用包含在区块中的引用。这些引用构成了 Tangle 的边（见第 4 节），以表达节点[5]的看法。如定义 4.1a 所示，引用 ref 包含两个字段：一个是对区块 x 的引用 r_x，另一个是标签值 v。我们称标签 v 是可以接受 $\{v_T, v_{\mathcal{L}}\}$ 数值的投票类型。该标签为 Tangle 中 x 的引用赋予了额外含义，并定义以下两个专门引用。

定义 6.1（区块引用）。我们说如果 y 引用 x，则从区块 y 到区块 x 的引用 ref = (r_x, v) 是区块的引用。在这种情况下，我们设置标签 $v = v_T$。

为了克服第 6.1 节所说的存活性问题，我们额外添加引用，该引用绕过区块，直接处理其包含的交易。

定义 6.2（交易引用）。我们说如果 y 引用 $\hat{x}$，则从区块 y 到区块 x 的引用 $\text{ref} = (r_x, v)$ 是交易的引用。在这种情况下，我们设置标签 $v = v_{\mathcal{L}}$。

备注 6.1 区块自然引用作为区块内容的交易。因此，诚实节点不会发出带有不在其偏好现实内的交易的区块（见第 6.5 节）。

例 6.1。考虑图 7。区块 y 和区块 y '均包含相同交易 $\hat{y}$，但 y 指的是区块 x 中的交易 $\hat{x}$，发出的是交易引用，而区块 y '指的是区块 x，发出的是区块引用。

备注 6.2 子类别的划分（交易引用和区块引用）只是为了投票。见证权重的定义（见第 4.4 节）不受影响。

我们定义数据结构，将第 6.2 节中两个不可变数据结构整合为一个 DAG，用于传播选票。

定义 6.3（投票 DAG）。投票 DAG D_V 是顶点集 V 为区块集 T 和交易集 L 并集（即$\mathcal{V}=\mathcal{T}\cup\mathcal{L}$.）的 DAG。设 v 和 u 是 V 中的两个顶点。当且仅当以下属性之一成立时，D_V 中存在从 u 到 v 的有向边：

1） u,v∈T, u 包含对 v 的区块引用;

2） u∈T, v∈L, u 包含对交易 v 的交易引用;

3） u∈T，$v=\hat{u}\in\mathcal{L}$,，即 v 是区块 u 的一笔交易;

4） u,v∈L，交易 u 花费交易 v 的输出，即 v∈$\mathrm{par}_{\mathcal{L}}(u)$。

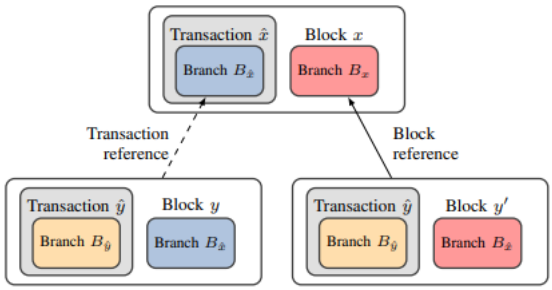


图 7：分支的继承：我们考虑两个潜在区块 y 和 y '，它们包含相同交易$\hat{y}$，但要么有一个交易，要么有对区块 x 的消息引用。为此，一个节点可以以两种方式投票。具体来说，区块可以通过交易引用或区块引用赞成前一个区块，同时分别继承被引用交易或被引用区块的分支。

目前，我们描述了如何为区块之间引用赋予额外含义，以此构建投票 DAG。该 DAG 允许节点递归地表达自己看法。根据定义 2.7，我们将$\mathrm{cone}_{\mathcal{V}}^{(p)}(x)$定义为投票 DAG 中的区块或交易 x 负责投票的过去锥。

定义 6.4（投票）。节点 i 通过引用区块 y∈T 中的 x 来表示对投票 DAG D_V 中某个顶点$x\in\mathcal{V}$的直接投票，式中发出（y）=i。我们说节点 i 间接投票给 $\mathrm{cone}_{\mathcal{V}}^{(p)}(x)$中任何顶点。

例 6.2 我们在图 8 中给出投票 DAG 概念的示意图。投票 DAG 从 Tangle 和账本 DAG 收集信息。我们假设发出区块 x 的节点不赞成交易 y，因此既不能投票给区块 y，也不能投票给区块 z，但它可以通过交易投票来投票给交易$\hat{z}$。更确切地说，区块 x 投票通过创建对区块 z 的交易引用，避开以灰色显示的顶点 $z,y,\hat{y}$ 。我们也可以用递归方程来描述投票过去锥。

命题 6.1 假设给定区块$x\in\mathcal{T}$对$y_1,\ldots,y_s$有区块引用，对区块 $z_1,\ldots,z_r$ 有交易引用（其中 s+r =k），那么 x 的投票过去锥可递归地写作

$$\mathrm{cone}_{\mathcal{V}}^{(p)}(x)=x\cup\mathrm{cone}_{\mathcal{L}}^{(p)}(\hat{x})\cup C_{\mathcal{L}}(x)\cup C_{\mathcal{V}}(x)$$

式中

$$C_{\mathcal{L}}(x) := \mathrm{cone}_{\mathcal{L}}^{(p)}(\hat{z}_1) \cup \ldots \cup \mathrm{cone}_{\mathcal{L}}^{(p)}(\hat{z}_r)$$
$$C_{\mathcal{V}}(x) := \mathrm{cone}_{\mathcal{V}}^{(p)}(y_1) \cup \ldots \cup \mathrm{cone}_{\mathcal{V}}^{(p)}(y_s)$$

基于现实账本阐述了分支的概念，见第 5 节。消耗不同分支多个输出将创建一个新分支，而该分支是被消耗输出分支的并集。在文献[15]中，此操作是为交易而定义的。我们将此概念扩展到区块，可通过投票给前面区块或交易来组合分支。更确切地说，我们可将区块中给定引用和分支相互关联。区块分支定义如下。

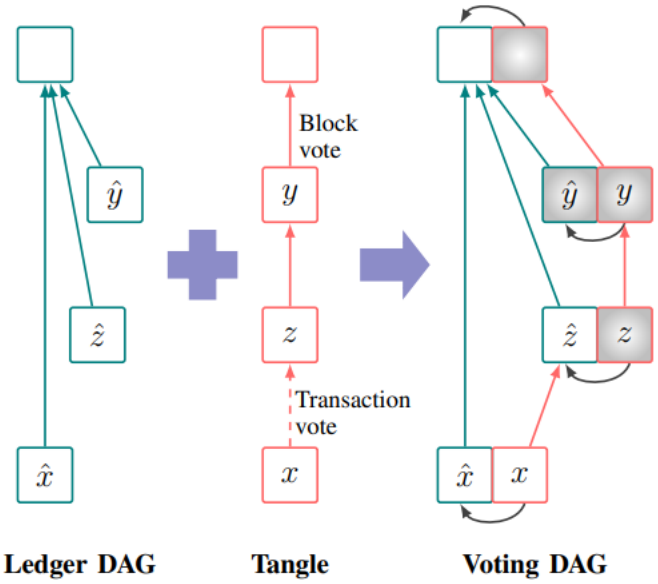


图 8：根据 Tangle 和账本 DAG 组装投票 DAG 的示意图。更确切地说，区块 x 投票通过创建对区块 z 的交易引用，避开以灰色显示的顶点$z, y, \hat{y}$。

定义 6.5（投票分支）。假设区块 x∈T，我们将 x 投票分支定义为：

$$\mathrm{branch}_{\mathcal{V}}^{(p)}(x) := \mathrm{cone}_{\mathcal{V}}^{(p)}(x) \cap \mathcal{C}.$$

备注 6.3 我们强调，为了保证协议正确性，节点在为新区块 x 创建引用时，必须确保$\mathrm{branch}_{\mathcal{V}}^{(p)}(x)$的确是定义 5.9 中定义的分支。$\mathrm{branch}_{\mathcal{V}}^{(p)}(x)$ D_C-过去-闭合集的属性简单地遵循$\mathrm{cone}_{\mathcal{V}}^{(p)}(x) \cap \mathrm{L}$ 是 D_L-过去-闭合集这一事实。但$\mathrm{branch}_{\mathcal{V}}^{(p)}(x)$的无冲突性通常不一定为真，必须进行检查。我们在第 6.5 节讨论末梢选择算法时解决这一问题。

回顾定义 5.11，它引入了交易$\hat{x}$的最大内含分支，将其记为$\text{branch}_{\mathcal{L}}^{(p)}(\hat{x})$。利用命题 6.1，我们将在以下语句，把区块 x 的投票分支和交易 $\hat{x}$ 的最大内含分支关联起来。

命题 6.2（分支的继承）。假设给定区块 X∈T 对区块 $y_1, \ldots, y_s$ 有区块引用，对区块 $z_1, \ldots, z_r$ 有交易引用（其中 s+r =k）。区块 x 继承了与这些投票有关的分支的并集，即投票分支可以分解如下：

$$\text{branch}_{\mathcal{V}}^{(p)}(x) = \text{branch}_{\mathcal{L}}^{(p)}(x) \cup B_{\mathcal{L}}(x) \cup B_{\mathcal{V}}(x)$$

式中

$$B_{\mathcal{L}}(x) := \text{branch}_{\mathcal{L}}^{(p)}(\hat{z}_1) \cup \ldots \cup \text{branch}_{\mathcal{L}}^{(p)}(\hat{z}_r),$$
$$B_{\mathcal{V}}(x) := \text{branch}_{\mathcal{V}}^{(p)}(y_1) \cup \ldots \cup \text{branch}_{\mathcal{V}}^{(p)}(y_s),$$

例 6.3 我们遵循与图 7 相同的例子。假设区块 y 中交易的最大内含分支为主要分支，即$B_{\hat{y}} = \emptyset$。区块 y 投票给区块 x 中包含的交易，从而继承分支$B_{\hat{x}}$。由于$B_{\hat{y}} = \emptyset$，区块 y 的投票分支为$B_{\hat{x}}$。同样，区块 y'投票给区块本身，并继承投票分支。我们强调区块 y（和 y'）中所含交易分支不受投票选择的影响。

我们可以把给定区块 x 和分支$B_{\mathcal{T}} = \text{branch}_{\mathcal{V}}^{(p)}(x)$关联起来。此外，x 的内容（即交易 x）也可以和分支关联起来。根据引理 6.1，我们有$B_{\mathcal{L}} \subseteq B_{\mathcal{T}}$。由于节点可能会改变自己对冲突的看法，投给 x 的冲突交易，只有从引用区块 y 的角度来看选票是有效的。后续改变节点选票是有可能的，我们可通过发出另一个区块，由该区块投给与 x 有冲突的交易。

定义 6.6（投票和当前选票变更）。设$\hat{x}$为节点 i∈N 投给的交易。设交易 $\hat{y}$ 与$\hat{x}$有冲突。如果节点 i 在投给$\hat{x}$之后，投给$\hat{y}$，则节点 i 不再赞成$\hat{x}$。我们说 i 撤销对 $\hat{x}$的投票。如果 i 最新投票赞成$\hat{x}$，即投票不会被撤销，我们说 i 当前投票赞成$\hat{x}$。

备注 6.4 定义 6.6 中「时间」的概念及其对「之后」涵义的影响至关重要。自然选择是交易内的时间戳或包含给定交易的区块的凝固时间。

例 6.4 定义 6.6 原理见图 1。具体地说，交易$\hat{x}$和$\hat{y}$是冲突的；区块引用用实线表示，而交易引用用虚线表示。最初，棕色和紫色节点投票给$\hat{x}$。但过一会儿，绿色节点撤销投给$\hat{x}$的票，最新选票赞成$\hat{y}$。$\hat{x}$和$\hat{y}$支持者分别显示于两个时间段各自区块的右上角。

6.4. 交易赞成权重和确认规则

节点必须能跟踪交易的接受进度。我们将第 4.4 节「见证权重」概念扩展到交易赞成权重（AW）。目的是为交易定义一个可参数化确认条件，该条件类似于第 4.5 节中讨论的区块条

件。

定义 6.7（交易支持者和赞成权重）。设$\hat{x} \in \mathcal{L}$为交易。$\overline{\text{sprt}}_{\mathcal{L}}(\hat{x})$表示当前投给$\hat{x}$的节点的集合。这些节点被称为$\hat{x}$的支持者。我们定义某个被称为赞成权重 （AW）的函数 $\mathbf{AW} : \mathcal{L} \to [0,1]$。

$$\mathbf{AW}(\hat{x}) := \sum_{j \in \text{sprt}_{\mathcal{L}}(\hat{x})} \mathbf{w}(j) \qquad (14)$$

当每个冲突集的总权重归一化为 1 时，AW 表示赞成给定交易的网络的百分比。

备注 6.5 区块 x 的 WW 与其包含的交易$\hat{x}$的 AW 是相关的，但不存在「单调性」。如果 x 只包含于区块 x 中，我们有$\mathbf{AW}(\hat{x}) \leq \mathbf{WW}(x)$。如果$\hat{x}$包含在多个区块[9]中，我们可以认为交易的 AW 甚至大于每个包络区块的 WW。

交易支持者可通过投票 DAG，传播支持者信息以进行更新。更确切地说，当区块 x 到达时。我们遍历$\text{cone}_{\mathcal{V}}^{(p)}(x)$。提出算法 1，在处理新区块时更新交易支持者，然后通过式（14）更新 AW。

与定义 4.5 相似，我们定义交易的确认。在我们讨论节点 i 在 t 时刻的感知时，我们将使用下标 i 和 t，比如$\mathbf{AW}_{i,t}$。

定义 6.8（经确认交易）。设 t θ∈（0.5, 1]为固定阈值。我们说对于某个 s≤t，若 $\mathbf{AW}_{i,s}(\hat{x}) \geq \theta$，则在 t 时刻对于节点 i∈N 而言，交易 $\hat{x} \in \overline{\mathcal{L}}$ 得到确认。

我们也定义了分支的 AW，这将构成下一节算法的基础。分支支持者等于分支中冲突支持者的交集。更正式地说，我们有：

定义 6.9 （分支支持者和赞成权重）。设 B∈β 为分支。我们将$\text{sprt}_{i,t}^{\mathcal{L}}(B)$定义为节点集，这些节点发出区块，赞成 B 中所有冲突。同样，我们将 B 的 AW 定义为：

$$\mathbf{AW}(B) := \sum_{j \in \text{sprt}^{\mathcal{L}}(B)} \mathbf{w}(j). \qquad (15)$$

我们将主要分支 AW，即空集∅定义为 1。

---

[9] 资金拥有者可请求多个节点广播一笔交易$\hat{x}$，或某个节点可发出几个包含$\hat{x}$的不同区块。

算法 1：在新区块到达时更新交易支持者

数据：Tangle DAG D_T, 账本 DAG D_L，节点 j 发出的新区块 x，$\{\mathrm{sprt}_{\mathcal{L}}(y)\}_{y\in\mathcal{L}}$

结果：更新后的$\{\mathrm{sprt}_{\mathcal{L}}(y)\}_{y\in\mathcal{L}}$

1 **对于**$\forall z \in \mathrm{cone}_{\mathcal{V}}^{(p)}(x) \cap \mathcal{L}$

2 $\quad \mathrm{sprt}_{\mathcal{L}}(z) \leftarrow \mathrm{sprt}_{\mathcal{L}}(z) \cup \{j\}$

3 **结束**

4 对于与$\mathrm{cone}_{\mathcal{V}}^{(p)}(x)$有冲突的$\forall z \in \mathcal{L}$，进行

5 $\quad \mathrm{sprt}_{\mathcal{L}}(z) \leftarrow \mathrm{sprt}_{\mathcal{L}}(z) \setminus \{j\}$

6 **结束**

6.5. 末梢选择算法

共识协议在很大程度上依赖于一种隐式投票机制。节点通过在新发出区块中选择引用来表达其看法并投出选票。确定引用的过程称为末梢选择算法（TSA），本节将围绕它进行讨论。

对于每个区块，节点可通过利用区块或交易引用，投票决定其喜欢 Tangle 和账本 DAG 哪个部分。Tangle 和账本 DAG 的首选部分由偏好现实来定义。节点 i 跟踪每个分支 AW，并保持其偏好现实$R = R_{i,t}$处于最新状态，比如用文献[15]定义的现实选择算法，结合第 6.4 节定义的 AW，见备注 5.3。更普遍的是，我们可使用其他指标，如第 10 节中以共享随机性作为额外偏差指标。

现在，我们描述一种同时考虑区块投票和交易投票的末梢选择机制。注意，根据引理 6.1，账本 DAG 诱导偏序与 Tangle 诱导偏序一致，出于这个原因，账本 DAG 投票允许对 Tangle 发起更具选择性但效率更低的投票。

让我们定义 Tangle DAG 和账本 DAG 上一些依赖于现实的末梢集。

用$\mathbf{T}_{\mathcal{T}}(R) \subset \mathcal{T}$表示 Tangle DAG 的顶点，其中 Tangle 过去锥只包含现实 R 中的交易。更确切

地说，对于任意$x \in \mathbf{T}_{\mathcal{T}}(R)$，不存在 y∈$\mathrm{cone}_{\mathcal{T}}^{(p)}(x)$使$\hat{y} \in \mathcal{C} \setminus R$。节点必须使用以下末梢选择和引用设置。

定义 6.10（针对某个现实的均匀随机末梢选择）。为了发出一个新区块，节点 i 从 Tangle DAG 所有末梢中随机选择要均匀赞成的末梢，直至创建 k 个引用。对于随机选择的末梢，节点执行以下步骤：

1） 如果所选区块位于集合$\mathbf{T}_{\mathcal{T}}(R)$内，则创建一个区块引用。

2） 否则如果所选末梢包含集合$\mathbf{T}_{\mathcal{L}}(R)$中的一笔交易，则创建一个交易引用；

3） 如果上述两种情况均不适用，则丢弃该区块。

算法 2：仅限于现实 R 的均匀随机末梢选择

数据：Tangle DAG D_T，账本 DAG D_L，偏好现实 R∈B

结果：末梢 $L_{\mathcal{T}} \cup L_{\mathcal{L}}$

1. $L_{\mathcal{T}} \leftarrow \emptyset$
2. $L_{\mathcal{L}} \leftarrow \emptyset$
3. $cnt \leftarrow 0$
4. **当**$cnt < k$ 时，**执行**
5. 在$D_{\mathcal{T}}$中随机均匀选择末梢 x
6. 将$Q_{\mathcal{V}}$设为$\mathrm{cone}_{\mathcal{V}}^{(p)}(x)$中包含的冲突
7. 将$Q_{\mathcal{L}}$设为$\mathrm{cone}_{\mathcal{L}}^{(p)}(\hat{x})$中的冲突
8. **如果**$Q_{\mathcal{V}} \subseteq R$，**则**
9. $cnt \leftarrow cnt + 1$
10. $L_{\mathcal{T}} \leftarrow L_{\mathcal{T}} \cup \{x\}$
11. **否则**

12. **如果**$Q_{\mathcal{L}} \subseteq R$**，则**

13. $cnt \leftarrow cnt + 1$

14. $L_{\mathcal{L}} \leftarrow L_{\mathcal{L}} \cup \{x\}$

15. **结束**

16. **结束**

17. **结束**

我们将此算法称为仅限于现实 R 的均匀随机末梢选择算法（简称 R-URTS），并引用算法 2 作为伪代码。

一个节点之前可能投票给不再是其首选分支的分支。为此它不得不改变自己的投票。有了上述末梢，节点便可以投给它们以前不「喜欢」的分支（通过投票给与之有冲突的交易），而不投给它们以前投的分支。为此，每个节点都必须随时更新每个分支支持者及其 AW。一个重要结果是，某些分支 AW 可能会随时间增加，而其他分支 AW 则随时间减少。交易投票增加表明，我们可以找到末梢选择算法解决方案，以缓解或减少存活性问题，最终将交易纳入考量，做出末梢选择。下面我们将根据以下假设开展研究。

假设 6.2（区块包含）。设 R 为偏好现实。末梢选择满足：对于每个交易$\hat{x} \in \mathcal{L}(R)$（见定义 5.12），$\mathrm{cone}_{\mathcal{V}}^{(f)}(x)$中至少有一个元素可供末梢选择算法选取。尤其是$\mathbf{T}_{\mathcal{T}}(R) \cup \mathbf{T}_{\mathcal{L}}(R)$并集中至少存在一个可用末梢。

我们也参考第 11 节进行更详细讨论。

7. 通信和对手模型

在说明协议安全要求之前，我们需要对底层通信模型进行假设。通常阐述攻击者（控制区块的延迟）发起通信的相关不确定性。通信模型界定对手延迟节点之间通信所受到的限制。作为一个模型，它只是一种简化，但允许对最关键组件开展系统研究。

为了简单起见，我们也在没有 TSA 等细节的情况下分析投票机制。我们想强调的是，我们的建模也可以应用于其他共识协议，从而为比较不同 DLT 提供框架。

7.1. 通信模型

参与节点通过点对点（P2P）协议或网络进行通信。在该 P2P 协议中，节点将经过其签名的区块发给相邻节点。只有当相邻节点验证其有效性时，才会转发来自覆盖网的其他节点的区块；如果交易无效的话，则传播将会停止。区块在两个节点之间传输是通过传输包含该区块的数据包来完成的。

节点之间 P2P 通信有三种基本（或经典）模型：同步模型、异步模型和部分同步模型，如文献[56]和[57]。

在同步模型中，存在已知有限时限 A，对手可通过该时间限制延迟数据包的递送。在异步模型中，对手可通过未知有限时间推迟数据包的递送。递送区块的时间没有限制，但每个数据包最终都必须递送。在部分同步模型中，我们假设区块递送存在有限未知上界 A。这个界限是事先不知道的，可由对手进行选择。

部分同步系统可被视为最初异步，但最终会变成同步。系统变为同步的时间称为全局稳定时间（GST）。

我们也考虑某个概率同步模型，见文献[44]。在该模型中，我们假设对于每个 ε> 0 和 δ∈[0, 1]，在有界（已知）时间$\Delta = \Delta(\varepsilon, \delta)$内传送比例为 δ 的区块，这取决于 ε 和 δ，其概率至少为 1-ε。概率同步模型类似于带崩溃故障的异步模型，见[57]。

共识机制具体实现在很大程度上取决于底层同步性假设。区分在某个数据集中找到共识的共识协议和在越来越多决策中找到共识的共识协议似乎也是妥善的做法。后者允许如果数据是通过引用关联的话，「加强节点之间同步性」。

7.2 Tangle、凝固与同步

构成 Tangle 的引用对于每个节点信息一致性来说至关重要。考虑到某个数据包只传播到部分网络，例如，在通信层某些传播过程中丢失。但是，已经接收到该区块的节点开始在其基础上进行搭建，并将其区块发送给网络。这些新区块包含对部分丢失区块的引用。因为节点必须知道任何区块过去锥，以便从该区块角度获得完整 Tangle 史，

为此，我们使用一种称为凝固过程的机制。在该机制中，接收给定区块的节点只有在其过去锥完成的情况下才会处理它，否则就向其对等节点请求丢失的引用区块（详见 4.3 节）。换言之，凝固过程是一种恢复丢失区块的机制，它加强通信模型的「同步性」。我们认为这在一定程度上支持了所有区块都在限定时间 A 内以高概率递送的假设。

7.3. 对手模型

我们区分三种类型节点：诚实、错误和恶意。诚实节点遵守协议，错误节点无法正常工作（如不发送任何交易），而恶意节点试图通过不主动遵守规则来扰乱协议。在多数情况下，我们认为恶意节点是由一个抽象实体控制的，并将该实体称为攻击者。我们认为攻击者在计算能力上是有限的，无法破坏所涉及的签名方案或加密哈希函数。但我们假设攻击者是无所不知的，且「立即知道」诚实节点所有状态变化。在经典共识协议中，通信模型已覆盖对手行为，因为延迟区块本质上是攻击者影响系统的唯一途径。对于我们共识协议而言，这将不再成立。在本文中，对抗策略可分为两大类：协议层攻击和投票层攻击。

7.4. 配置图和调度

如何触发分布式系统中事件取决于一些通常称为环境的外部因素。我们遵循文献[58]，通过调度器的抽象化对该环境进行建模。为此，我们考虑节点之间所有通信赖以进行的通信网络。这些网络通常被称为 P2P 网络。我们用有向图对其进行建模，其顶点集是参与节点的集合。如果节点 i 可直接将数据包发送给节点 j，则存在从 i 到 j 的有向边。

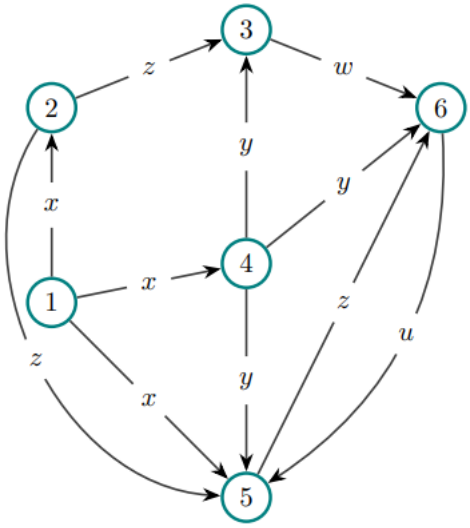


图 9：6 个节点网络示意图。数据包 u,w,x,y 和 z 沿着代表通信通道的有向边发送。

我们假设这个图是相互连接的。沿着这个图有向边，节点之间交换数据包。在我们案例中，这些数据包带有区块。

定义 7.1（数据包和通信图）。对于每个从节点 i 发送至节点 j 的区块 x，我们都添加一条从 i 到 j 的有向边（称为数据包），并用向量进行标记。

$$e(x,i,j) := (x,i,j,t(x),\delta_{i,j}(x)). \qquad (16)$$

此标记表示数据包 x 在 t(x）时刻从节点 i 发送，并在$t(x)+\delta_{i,j}(x)$时刻到达节点 j。所有数据包的状态空间用 M 表示，我们将所得到的图称为协议的通信图 G。

例 7.1 我们在图 9 中给出网络和数据包概念示意图。在该图中，网络由六个节点组成。一些节点之间存在有向边，并显示通信通道。我们指出这些通信不一定是对称的。纳入区块的数据包可沿边缘发送。

每个节点均保留本地版 Tangle $D_{\mathcal{T}_i}$，我们将其视为节点 i 的（本地）配置。为了方便表示，我们考虑以下简化版 OTV，它不跟踪 Tangle 中实际区块位置，而只跟踪分支支持者或冲突。因而（本地）状态空间由 $\mathcal{Q} = \underbrace{2^{\mathcal{N}} \times \ldots \times 2^{\mathcal{N}}}_{|\mathcal{C}|}$ 给出，式中 C 是冲突集，2^N 是 N ={1,...,N}所有可能

子集的集合。

备注 7.1 上述简化版使我们更容易分析冲突交易的投票，这给不描述非冲突交易的确认带来成本，我们将在第 8.1 节中详细讨论这些交易的「存活性」。

我们将节点 i 接收到的数据包解释为所有数据包 M 空间内输入的赋值。每个输入赋值 α 均会产生当前配置的更新结果，而每个配置ω_i都会产生输出赋值。为此，我们考虑两个函数

$$I(\alpha, \omega_i) : \mathcal{M} \times \mathcal{Q} \mapsto \mathcal{Q}, \qquad (17)$$

和

$$O(\omega_i) : \mathcal{Q} \mapsto \mathcal{M}^{|\mathcal{N}_i|-1}, \qquad (18)$$

式中，N_i是节点 i 的相邻节点。节点 i 运行算法 A=（I,O），该算法通过更新其内部状态ω_i来响应传入的数据包，并最终发出传出数据包，表明其状态更新情况。我们也考虑取值为$\hat{\omega} = \{\omega_1, .., \omega_N\} \in \mathcal{Q}^N$的整个系统配置。对应的算法用 A 表示。

基本上，节点 i 一旦接收到来自节点 j 的数据包后，就开始跟踪。它检查包含在数据包 e 中的区块 x 是否被处理过。如果是处理过，则节点状态保持不变，不会发出新的数据包。如果该区块是新的话，节点就会检查其有效性，将该区块添加到本地 Tangle 中。如果适用的话会更新分支支持者，如果区块包含的交易有冲突，则添加新冲突。在此步骤之后，节点将以新数据包的形式，该区块转发给所有该节点未接收到该区块的相邻节点。

节点也可创建区块。一旦创建区块 x，就会将 x 附加于本地 Tangle。然后创建包含区块 x 的数据包 e，并将该数据包副本发送给其所有相邻节点[10]。

通过 TSA，利用随机性（通过设计）创建区块。此外，区块发出时间可能取决于节点与我们系统环境的交互。为此，我们用随机变量对节点两个连续区块之间的时间进行建模。这种随机性将我们的协议变成了随机协议，而这种随机性是用概率度量来描述的。

定义 7.2 （配置图）。假设 e 是 G 中从 i 发送到 j 的数据包，$\hat{\omega}, \hat{\omega}' \in \mathcal{Q}^N$是两个（全局）配置。当且仅当以下情况，对于某个 i，我们写作$\hat{\omega} \xrightarrow{e} \hat{\omega}'$

$$\mathbb{P}[I(e, \omega_i)) = \omega_i'] > 0 \qquad (19)$$

对于所有$j \neq i$, $\mathbb{P}(\omega_j' = \omega_j)) = 1$。我们说 e 可从$\hat{\omega}$访问$\hat{\omega}'$。可访问性符号根据（全局）

[10]在此例子中，我们用洪泛式协议来传播区块。

配置集定义一个有向图，我们将其称为算法配置图。

定义 7.3 （有效数据包）。当且仅当以下情况，从节点 i 到节点 j 的数据包（或边）e 被称为有效给定全局配置$\hat{\omega}$。

$$\mathbb{P}(O(\omega_i) \ni e) > 0 \qquad (20)$$

换句话说，任何有效边都必须是算法 A 的结果，边序列 ei，e2，...被称为有效给定（初始）配置$\hat{\omega}(0)$，根据归纳，当$\hat{\omega}(0)$是有效给定$\hat{\omega}(\ell-1)$时，式中 $\hat{\omega}(\ell-1)$，使得

$\hat{\omega}(\ell-2) \overset{e_{\ell-1}}{\to} \hat{\omega}(\ell-1)$。

下面，我们将假设诚实节点只发出有效数据包。

定义 7.4 （配置通信）。当且仅当配置图中存在从$\hat{\omega}$到$\hat{\omega}'$的有限有效路径时，我们说从配置 U 可访问（全局）配置$\hat{\omega}'$。当且仅当两个配置$\hat{\omega}$和$\hat{\omega}'$能够互相访问时，我们说它们可以相互通信。在这种情况下，我们写作$\hat{\omega} \to \hat{\omega}'$。

关系$\leftrightarrow$在配置集中界定等价关系。

定义 7.5（通信类）。等价关系$\leftrightarrow$的等价类别被称为配置集的通信类。当且仅当某个（通信）类没有向外的边时，它才被称为闭合，否则将被称为开放。

封闭通信类在描述协议结果时起到关键作用。设 R∈B 为现实。那么对于所有 i∈N，带$\mathrm{sprt}_{\mathcal{L}_i}(R) = \mathcal{N}$的配置是一个封闭类。这里注意到，我们仍然假设所有节点都是诚实的，按协议行事。

定义 7.6（共识状态）。对于某个现实 R，当且仅当以下情况下，状态ω被称为共识状态：

$$\mathrm{sprt}_{\mathcal{L}_i}(R) = \mathcal{N}, \quad \forall i \in \mathcal{N}, \qquad (21)$$

备注 7.2.在此强调一下，「共识状态」的定义仅和偏好现实的约定有关，而不考虑确认的意义。见定义 4.6。以下章节将探讨与确认有关的存活性和安全性。

我们对通信层做出关键假设。

假设 7.1 （随机区块的发出和数据包的延迟）。区块发出和数据包延迟是随机的，它们满足：

1) 节点独立发出新区块，按某种概率分布μ_{iss}进行分布。

2) 两个节点之间数据包延迟是独立的，按一定概率分布μ_{pack}进行分布。

3) 区块发出和数据包延迟是独立的。

4) 按照正概率，数据包递送速度比新区块发出速度快。更确切地说，如果$X \sim \mu_{\text{iss}}$，$Y \sim \mu_{\text{pack}}$，则$\mathbb{P}(Y < X) > 0$。

引理 7.1. 根据假设 7.1，对于每个给定配置$\hat{\omega}$，存在共识状态$\hat{\omega}_c$使得$\hat{\omega} \xrightarrow{e} \hat{\omega}_c$。

证明。假设$\hat{\omega}$是一种配置。我们一直等到所有现有数据包和相应投票变更都被发送到所有其他节点。在此期间，没有新区块按正概率被发出。其发生概率为正。当每个节点都看过所有当前区块后，每个节点对不同现实支持者具有相同感知。换句话说，节点在不同分支 AW 上达成共识。现在，每个节点都改变其对偏好现实的看法，并发出表明其投票变更的交易，用通信图上的数据包对其进行流言式传播（gossip）。一旦所有这些数据包被所有节点看到，就会达到共识状态。

引理 7.1 有两个直接结果。

推论 7.1 根据假设 7.1，当且仅当通信类包含一个共识状态时，它是闭合的。

推论 7.2。根据假设 7.1（在没有对手的情况下），协议（$\mathbb{P}$-几乎肯定）会收敛到共识状态。

定义 7.7（调度）。通信图 G 的调度是由（有限或无限）有效边 e_1，e_2，....组成的边序列。$\hat{\mathbf{A}}$在 G 上对边序列 e_1，e_2，...的（有限或无限）执行是配置序列$\hat{\omega}(0) \xrightarrow{e_1} \hat{\omega}(1) \xrightarrow{e_2} \cdots$，式中 $\hat{\omega}(0)$ 为初始（全局）配置。

上述定义可以自然扩展到区分诚实节点和对手节点的模型。我们假设对手节点不必遵循算法 A，但可以为非偏好分支产生信息投票。在通信层面上，对手节点可能比诚实节点更为有效，即可以更频繁发出区块，延迟诚实数据包的中继转发。但我们假设 7.1 适用于所有诚实和恶意节点。如果所有诚实节点最终青睐同一现实，我们就说协议达到了共识状态。让我们用 Nh 和 Na 表示诚实节点和恶意节点集合。以此类推，得到以下结果。

定理 7.1（最终一致性-随机区块）。假设 7.1 适用于诚实和恶意节点的区块和数据包，假设 q 为对手权重。则如果 q<1/2，所有诚实节点（$\mathbb{P}$-几乎肯定）最终都会选择相同现实。

证明。由于 q<1/2，对于某个现实 R，如果所有诚实节点都有相同偏好现实，而且所有节点都知道它的话，则达到共识状态，即

$$\text{sprt}_{\mathcal{L}_i}(R) \supset \mathcal{N}_h, \quad \forall i \in \mathcal{N}_h, \qquad (22)$$

我们必须证明对于每个给定配置 ω，存在可用共识状态。这点被证明与引理 7.1 相似，以及对手既不发出区块也不能延迟诚实数据包的情况，这种情况在假设 7.1 中按正概率发生。

8. 存活性与安全性

在上一节中，我们对最终收敛感兴趣，并在假设随机区块发出和随机数据包延迟情况下证明定理 7.1 的最优结果。在本节中，我们将把交易确认状态纳入考量，将安全分为存活性和安全性，以进行更详细和定量分析。

从一般角度来看，存活性意味着最终会有好事发生，而安全性则意味着不会发生任何错误。在我们案例中，这点可以转化为以下内容。安全条件是任意两个诚实节点始终达成一致，该决定满足规定的有效性条件。此外，任何两个节点都不应当确认冲突交易。存活性指的是每个诚实节点最终都应就交易确认状态做出决定，即在我们案例中，所有节点最终都达到确认阈值 0，见定义 4.5 和 6.8。

备注 8.1 总体而言，我们还需要共识协议满足完整性。完整性要求共识协议最终结果最初至少有一个节点提出。由于在 OTV 中，诚实节点总是选择最大分支，一旦协议终止，完整性就会得到满足。

8.1. 非冲突交易

非冲突交易存活性是其最终被纳入账本状态这一属性。从最强形式来看，它指的是每个非冲突交易都会被确认，见定义 4.5 和 6.8。因此，存活性安全阈值顶多是权重的一个比例（1 - 0），因为持有该比例（1 - 0）的攻击者或错误节点可以通过不再发出任何区块来停止确认。存活性与 TSA 和孤立问题之间有着内在联系。我们对 TSA[11]做出如下假设，并参考第 6.1 节进行讨论。

命题 8.1 （非冲突交易的存活性和安全性）。我们假设在异步模型中，末梢池大小是固定的，满足假设 6.2。恶意节点的权重为 q，则当 q<1-0 时，最终所有诚实节点的非冲突交易都将被确认。

证明。假设 x 是包含非冲突交易 x 的区块，考虑任意诚实节点 i。根据假设 6.2，每当该节点发出新区块时，它引用（并投给）x 的概率为正。

这时，重要的是有第二种引用类型，它允许只投给交易$\hat{x}$，而不投给 x 的整个 Tangle 过去锥。我们用 p_j 表示第 j 个发出区块的这个最后概率。接着，由于末梢池大小的固定性假设，对于无穷多个指数 j，存在某个 $\varepsilon>0$ 使得 $p_j \geq \varepsilon$。假设 6.2 保障这些事件的独立性，而 Borel-Cantelli 引理意味着节点 i 最终会投给区块 x。由于节点数量是有限的，所有节点最终会投给 x。因此，对固定末梢池大小假设的有效性进行讨论是妥当做法。整个第 4.7 节也做出这种假设，作为假设 4.3。让我们根据通信模型和对手模型来回顾这一假设。

攻击者可以延迟诚实交易，从而增加网络延迟 h。这反过来会增加末梢池大小和最终化时间 [59]。在异步模型中，这可能导致节点内存溢出或停止某些交易的确认。虽然这种攻击在该模型中理论上是可能的，但更多是一种理论兴趣而非实际问题。这里我们还要指出的是由于凝固过程，节点确实具有有效方式来「同步」其对 Tangle 的感知；见 7.2 节。

在 Tangle 层，「最糟糕的情形」似乎如下所示。对手发出区块，引用已引用的区块，而不把任

[11]正式地讲，更多是对「末梢」定义的要求。

何末梢移出末梢池。在「节点可以按其权重比例发出区块」的假设下，我们得出每个诚实区块均发出 q/（1−q）个恶意区块。诚实节点可增加引用数量以保持 Tangle 宽度不变。更确切地说，诚实节点区块平均移除 K：=（q/（1−q））+1 个末梢就足够。换句话说，我们可以选择引用数量 k >K 来保障对这种攻击的鲁棒性。例如，q =1/2 会导致 K=2。

8.2. 冲突交易

与冲突交易存活性和安全性有关的理论结果在很大程度上依赖于底层通信和对手模型假设。此外，OTV 协议分析比较复杂：需要建模部分网络，建模权重分布，以及各种（甚至是无限数量的）对手策略。下一节，我们将展示在特定或边缘情况下，对手可能阻碍共识的发现。但我们所要强调的是，这种干扰只影响冲突交易的存活性，而适当 TSA 则可保证非冲突交易的存活性；见命题 8.1。在第 10 节中，我们在协议中添加特性，该特性使我们可以获得与冲突交易存活性有关的理论结果。

9. 不可能性结果与亚稳态

不可能性结果在共识协议理论中起至关重要的作用，因为它们强调了局限性和关键边缘情况。最著名的不可能性结果是 FLP 结果[19]，它显示对于确定性协议来说，在异步通信模型中实现共识通常是不可能的。从一般角度看，这种不可能性是由于 P2P 通信中数据包可能延迟和由此产生的妨碍共识发现的「对称」情形造成的。

我们将考虑两种或多种直接冲突交易情形。共识机制作用是约定最终应当接受哪笔交易。可以认为，将冲突交易保持在未决状态（即违反存活性）是可以接受的。但出于几个原因，这么做是有问题的。比如如果节点将交易保持在无限未决状态，将大大增加投票层所需通信，妨碍账本的修剪。而长期未决的交易也可能危害安全性。总是有某个节点证实未决「交易」。虽然该事件概率可能很小，但仍然为正，因此这个不可能事件将在某个时间点发生。我们还注意到，单纯拒绝恶意交易并不能提供解决方案，这样会使交易延迟取消，从而破坏系统安全。

在本节中，我们给出冲突交易存活性和安全性未得到满足的例子；按照相同原则，可以举出更复杂例子，它们构成不可能性结果，因为所提出协议并不能保证异步通信模型中的存活性或安全性。这些情形依赖于对攻击者的强假设。我们将通信层面攻击和投票层面攻击区分开来。通过对这两个层面提出要求，我们给出当安全性得不到保证时的理论结果，引理 9.1。

9.1. 通信层面

我们首先举个例子，在该例子中，攻击者并不直接参与投票，而只控制诚实节点区块调度。我们指出，在这种情况下，攻击者不需要控制任何权重。

第一次对手攻击被称为亚稳态攻击，因为其试图使诚实节点处于未决状态。我们借鉴文献[45]，获得关于此类攻击的更多详情和分析。在概念层面，此类攻击利用系统处于两种不兼容选项之间的大致对称状态这一事实。一旦对称场景被打破，节点可能迅速收敛至其中一个选项。

例 9.1 （亚稳态攻击 I）。我们考虑 N=4 个直接通信的参与节点{1，2，3，4}；通信图是有四

个顶点的完整图表。我们假设每个节点都具有相同权重，即对于所有 i∈N，$m_i = 1/4$。我们考虑简单双花场景。为此，冲突集为$\{\hat{x}, \hat{y}\}$。为了简便起见，我们假设如果两个冲突有50%AW，则节点更偏向自己看法。在 t_0 时刻，每个节点 i 都通过附加区块 x_1 将其投票情况传递给每个相邻节点。攻击者将这些区块（更确切地说，相应数据包）延迟 δ>0 或 γ>δ。更确切地说，如式（16）定义的，在通信图中有如下边：

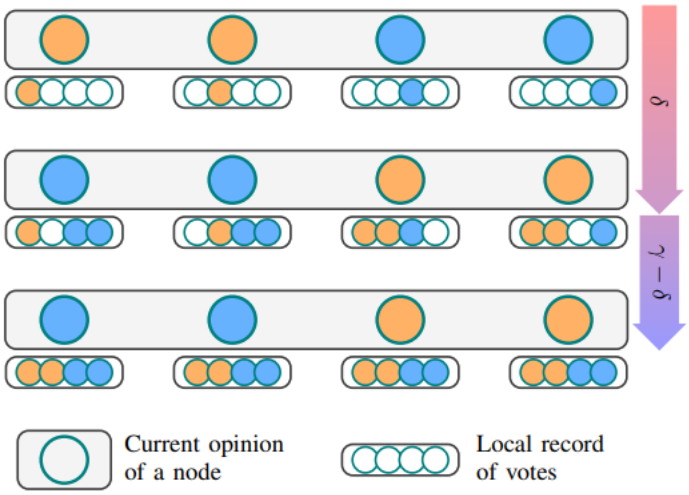


图 10：例 9.1 示意图。节点投给交易 $\hat{x}$ （蓝色）或$\hat{y}$ （橙色）。每个节点最终得到与其开始时相反的看法，从而造成僵局。

$$(x_i, i, 3, t_0, \delta), (x_1, i, 4, t_0, \delta), \quad i \in \{1, 2\}$$
$$(x_j, j, 1, t_0, \delta), (x_j, j, 2, t_0, \delta), \quad i \in \{3, 4\}$$

和

$$(x_1, 1, 2, t_0, \gamma), (x_2, 2, 1, t_0, \gamma)$$
$$(x_3, 3, 4, t_0, \gamma), (x_4, 4, 3, t_0, \gamma)$$

在 γ 时刻，此调度可造成偏好冲突的逆转，见图 10。为此，控制通信层面的攻击者可随意延迟共识发现。为了更正式描述前个一攻击者，我们必须明确对区块发出和通信模型的假设。例如，在已知网络延迟上限为 Δ 的同步模型中，如果 δ，γ < Δ，那么该攻击是成功的，诚实节点将定期发出区块。在异步环境下，攻击者可调整延迟 δ 和 γ，即使诚实节点并不同时连续发出交易。

备注 9.1 上述情形无疑是一个特例，引起我们的理论关注。但它提出以下问题：这种调度在什么条件下存在，以及其在实际应用中是否切实可行。

9.2. 投票层面

本节描述一些情形，在这些情形中攻击者可利用投票层来成功干扰共识的发现。我们不需要条

件来控制诚实节点之间通信，但需要相对强假设来判断对手是否能够发出新区块并可靠地将其转发给诚实节点。

例 9.2 （亚稳态攻击 II）。我们再次考虑一个双花的情形，即冲突集 $\{\hat{x}, \hat{y}\}$。在该攻击中，对手投票给少数，即有较小 AW 的冲突。假设攻击并不会影响通信层，我们在同步通信模型假设下进行研究，假设区块传播速度很快，即每个区块都更新所有其他节点状态。此外，我们假设对手能够以很高速率发出，这样对于每个其他诚实节点区块而言，对手都可以发出一个区块。

我们考虑偶数 N_h 个诚实节点和 3 个恶意节点，其中每个节点都有相同权重。我们说如果一个节点投给 $\hat{x}$ 或 $\hat{y}$ 的话，它将分别位于集合 X 和 Y 中。这个协议从 $1/2N_h$ 个诚实节点最初投给 $\hat{x}$，$1/2N_h$ 个诚实节点最初投给 $\hat{y}$ 开始。我们参考图 11 进行说明。接着，对手（通过所有三个节点）投给 $\hat{x}$，得到投票 $|X|/|Y| = (1/2N_h + 3)/(1/2N_h)$。X 中节点继续投票支持 $\hat{x}$。另一方面，Y 中诚实节点最终改变其投票，发出赞成 $\hat{x}$ 的交易，为此，集合 Y 变成集合 X。此时，在其他诚实节点表达自己投票前，攻击者将其投票切换为 $\hat{y}$。于是，我们共有 $|X|/|Y| = (1/2N_h+1)/(1/2N_h+2)$。此时，诚实节点将投给 $\hat{y}$。但只要来自 X 的节点改变其投票，所得情形就会与最初情形形成对称。对手可以无限重复此过程。

备注 9.2. 我们想要指出的是，在例 9.2 中，攻击者严重依赖于对手能力，即在 2 个以上诚实节点将其投票变更为大多数前，立即调整其选项。

以下例子是偷梁换柱攻击（Bait-and-SwitchAttack），它不太依赖对手发出率，但需要更高的权重。

例 9.3（偷梁换柱攻击）。我们考虑对手有最高权重节点时的评估。策略是频繁切换看法，使诚实节点始终「追求变化莫测的最重分支」。例如，考虑总权重为 W_h 和单个权重为 w_h/N_h 的 N_h 个诚实节点，以及一个权重为 w_a 的对手节点。令 n_{cr} 为使得 $n_{cr} \cdot \frac{w_h}{N_h} < w_a.$ 的最大自然数。

起初，恶意节点在交易 $\hat{x}_1$ 中等待输出。接着，在总权重小于 w_A 的 n_{cr} 个节点表达其投票前，对手在交易 $\hat{x}_2$ 中使用同样输出，即用 $\hat{x}_1$ 来创建冲突交易，并（隐式）投给新交易。由于 $\hat{x}_2$ 已成为最重分支，所有诚实节点都会投给该交易。对手通过重复创建额外双花来重复此过程。

9.3. 通信和投票层面

在前几节中，我们举例说明对手如何破坏冲突交易的存活性。攻击者策略需要对通信层进行实际控制，或将高发出率和相当大权重相结合。在本节中，我们证实安全性方面的不可能性结果，而该结果需要使用两个层面攻击策略。

定义 9.1 （遭到破坏的安全性）。我们说当且仅当存在两个节点 i 和 j，以及冲突交易 $\hat{x}$ 和 $\hat{y}$

（后者使得对于某些时刻 s 和 t 我们有$\mathbf{AW}_{i,t}(\hat{x}) > \theta$ and $\mathbf{AW}_{j,s}(\hat{y}) > \theta$）时，安全性遭到破坏。

我们得到以下「负面」结果。

引理 9.1。设 q>θ−0.5 为对手权重。假设诚实节点权重平均分布于足够多诚实节点，则存在一种破坏安全性的对手策略。

证明。我们选择足够多诚实节点 N_h，以便使某个$N_h^* < N_h$，于是我们有

$$\frac{\theta - q}{1 - q} < \frac{N_h^*}{N_h} < \frac{0.5}{1 - q}$$

攻击者开始发出两个冲突交易 $\hat{x}$和$\hat{y}$。攻击者将诚实节点分为两组 X 和 Y，这样每组都会形成底层通信层连接子图，而攻击者则与两个组相连。X 组由N_h^* 个节点组成，而 Y 组由$N_h - N_h^*$ 个节点组成。

攻击者可干扰调度，使每组节点只收到来自本组的区块。攻击者改变调度，使 X 中节点在$\hat{y}$之前接收交易$\hat{x}$，Y 中节点在$\hat{x}$之前接收$\hat{y}$。所有诚实节点均为其首选交易准备初始语句（X 组为$\hat{x}$，Y 组为$\hat{y}$），并将其发给自己相邻节点。

攻击者向 X 个区块发送状态，表明其偏好$\hat{x}$。结果，来自 X 的节点确认交易$\hat{x}$，因为 $\mathbf{AW}(\hat{x}) = (1-q)\frac{N_h}{N_h} + q > \theta.$

随后，攻击者将区块发送给 Y（和 X），此时它投给交易$\hat{y}$。在攻击者未投给交易$\hat{x}$的情况下，X 中$\hat{x}$的 AW 简化为$\mathbf{AW}(\hat{x}) = (1-q)\frac{N_h^*}{N_h} < 0.5$。

接着，攻击者让 X 知道 Y 的偏好。此时，$\mathbf{AW}(\hat{y}) > \mathbf{AW}(\hat{x})$，结果，来自 X 的节点更新其偏好现实，并投给$\hat{y}$。这最终导致对于所有节点，$\mathbf{AW}(\hat{y}) > \theta$，根据定义 9.1，安全性遭到破坏。

以上证明表明，攻击者需要对通信层具有很强控制才能进行此类攻击。但是，它为协议安全性提供合理理论安全阈值。更重要的是，我们可以根据第 10 节假设$q < \theta - 0.5$证明安全性。

9.4. 现实条件

以上例子表明两个维度（即通信和投票层面）之间的交互可能有利于攻击者，也有利于协议鲁棒性。在所有情况下，攻击者似乎都需要对协议通信层进行良好控制。通信层随机性或不确定性可能干扰对手策略，最终造成诚实节点看法趋同。

我们推测这些强假设在大多数合理现实场景中是无法得到满足的，单纯通信层面的攻击在实践中很难执行。

在数据包完全随机调度情况下下，系统最终会收敛至共识状态。攻击者控制不超过总权重的一半的权重，见定理 7.1。但对于实际应用而言，该收敛时间可能偏长，不切实际，而且可能破坏安全性（对于确认来说），如引理 9.1 所示。针对实际实现系统固有随机性的理论研究仍处于早期状态，而目前量化甚至控制它似乎还无法实现。我们参考[58]以获得一种描述与交易调度有关的熵理论方法。

下一节将提出一种更复杂变体使我们得以更直接进行理论处理，并提供「最佳」安全阈值。

10. 同步随机现实选择

在前一节中，我们证明在几种情况下，当前提出协议可能导致节点无法在几个有效选项之间达成一致。本节将提供一种利用外部随机性来克服这种状况的机制。比如在文献[45]、[60]和[61]中，共同随机性可以成功引导系统远离这种不良状况。

预共识类是网络最终据以达成共识的类别。为此，共识协议设计目的是构建协议，使其全局状态迅速达到这种预共识状态，从这个角度看，实际共识状态是不可避免的。

OTV 是一种异步协议，它既有优点也有缺点。一个缺点是节点之间缺乏同步可能性，这种可能性可用于对抗通信层面的对手攻击。上一节的论据和例子表明，理论上攻击者可以使诚实节点保持长期未决状态。为了排除这些情形并获得理论结果，我们利用分布式随机数生成（dRNG）程序来同步节点，并干扰潜在对手。

我们选择一个参数 D 来描述同步次数之间时期（时期）长度。换言之，每 D 个单位时间一次，我们在给定 dRNG 程序帮助下同步节点。此程序受到文献[60]的启发，其中 dRNG 用于在拜占庭环境中建立基于投票的共识协议。dRNG 使得共识协议可按正概率达到预共识状态。此概率在节点看法和投票上是一致的，为此，协议以长度为 D 的几何分布周期数进入预共识类。最后一步，我们证明从预共识状态过渡到共识。

我们考虑有 N_h 个诚实节点和 N_a 个对手节点的 $N = N_h + N_a$ 个节点的系统，用 $\mathcal{N}_h = \{1, \ldots, N_h\}$ 集合来表示诚实节点，而用 $\mathcal{N}_a = \{N_h + 1, \ldots, N_h + N_a\}$ 集合来表示对手节点。

我们首先介绍我们的模型假设。

假设 10.1 我们做出如下假设：

10.1.1 一个诚实节点每个区块在 $\mathbf{d} = \mathbf{d}(\varepsilon)$ 时刻，被另一个诚实节点内以至少 1-ε 的概率接收。常数 ε>0 可选择任意小。每个区块事件互相独立。

10.1.2 对手控制着权重比例 q。在 10.1.1 前提下，对手可能对区块调度产生影响。

10.1.3　冲突集 C 是固定的，不会随时间变化。所有节点均感知到相同冲突集。

10.1.4 存在一个 dRNG，每 D 个单位时间发出一个随机变量。该随机变量均匀分布于区间[0.5，θ]，其中 θ 为确认阈值；见 4.5 节。每个给定节点在（每个时期）d 时刻前，以至少 1-ε 的概率（独立）接收此数值。

10.1.5 累积权重至少 θ 的诚实节点在至少每 D/2 个单位时间发出支持其偏好现实[12]的区块，其概率至少为 ε。

让我们探讨上述假设的有效性。假设 10.1.1 本质为一个概率同步性假设。选票是 Tangle 中可被重新广播或通过请求凝固获得的区块，该事实支持概率 ε 可被选为任意小的事实；见 7.2 节。独立性假设是必要的，但相关误差研究超出了本文范围。假设 10.1.2 在概率同步模型中是合理的。假设 10.1.3 基本是为了方便而设。由于节点只考虑具有一定年龄（比 D 更老）的冲突，假设 10.1.1 保证节点中具有很高概率的冲突集有相同感知。假设 10.1.4 在以往研究文献[45]、[60]和[61]中使用。这些常见随机数序列可由外部来源提供，或系统自身节点生成。见文献[62]-[67]。我们强调，dRNG 随机性必须是不可预测的，且在每个时期由具有正概率大多数权重获得。但我们不要求所有诚实节点都在该随机数上达成一致[13]。最后一个假设（假设 10.1.5）是保证交易有机会被确认的（几乎）必要条件。起初，在 D 时刻前，每个冲突 c∈C 的 AW 都通过投票方式，根据第 6.5 节所述机制增长。在初始区间结束时，每个节点都对冲突 c 的 AW 有自己感知，记为$\mathbf{AW}_{i,\mathcal{D}}(c)$。

在（D 和 D+d）之间）第一个 dRNG 随机性到来后，每个诚实节点均选择自己的偏好现实，并在下个长度为 D 的区间内坚持选择它。

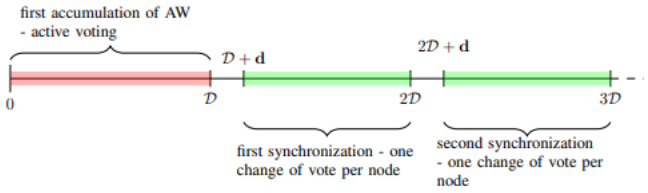


图 12：同步过程中的不同时期

算法 3：常见币种现实选择算法

数据：冲突图 GC=（C，E），共同随机性 X 均匀分布于[0.5，θ]。

结果：偏好现实 R∈B

---

[12] 换言之，对于偏好现实中的每个冲突 c，节点都至少发出一个声明该节点投给此冲突 c 的区块，而不会发出任何声明该节点投给与 c 冲突的交易的区块。

[13] 其概念是对于 dRNG 随机性的「弱共识」最终会演变为对账本状态的「强共识」

1 R←0

2 U←C

3 **当**|U|≠ 0 时，执行

4 $c^* \leftarrow \arg\max\{\mathbf{AW}(c) : c \in \max_C(U)\}$ ；/*用 max hash（c）打破关联*/

5 **如果 AW**（c*） >X，**则**

6 R ← R ∪ {c ∗}

7 U ← U \ {NC（c ∗ ） ∪ {c ∗}}

8 **否则**

9 打破 while 循环

10 **结束**

11 **结束**

12 **当**|U| 6=0 时，执行

13 $c^* \leftarrow \arg\max\{\text{hash}(c||X) : c \in \max_C(U)\}$

14 $R \leftarrow R \cup \{c^*\}$

15 $U \leftarrow U \setminus \{N_C(c^*) \cup \{c^*\}\}$

16 **结束**

在算法 3 中，受文献[68]的启发，我们阐述让节点选择偏好现实的迭代过程。首先，它将集合 R 初始化为空集，并将 U 初始化为冲突集 C。在第一个 while 循环每一步，节点均在 U 中找到具有最高 AW 的冲突 c*。如果 **AW**（c*） >X，则将 c 添加到 R，删除 U 中与 R 冲突的所有交易，并重复此步骤。我们也要求 c*须来自 $\max_C(U)$（见定义 2.6），以保障将 c*添加到 R 后，更新后集合 R 是一个分支。如果 AW（c*）≤X，则我们运行下个迭代过程（while 循环），用 c*更新 R，其中 c*是 U 中冲突 c，它可获得级联 c||X14 中的最大哈希值，并继续类似过程，直至 U 变为空集。通过构建，所得集合 R 是最大分支或现实。我们将在以下命题概括这些结果。

命题 10.1 算法 3 所得集合 R为一个现实。

备注 10.1 利用抗碰撞哈希函数将任意大小数据映射到大小固定的二进制序列，即散列：$\{0,1\}^* \to \{0,1\}^h$。此外，要求对于给定序列 x，实际上不可能找到另一个序列x'，使得 hash（x） =hash（x’）。在本文其余部分，我们假设特定哈希函数是固定的，被所有参与者使用。

用$\mathrm{sprt}^{(h)}_{\mathcal{L}_{i,t}}(\hat{x})$表示 t 时刻从节点 i 看到的诚实节点集，该集合发出投给交易 $\hat{x}$的区块（关于支持者类似定义，见定义 6.7）。节点 i 在 t 时刻看到的$\hat{x}$ 的诚实 AW 定义为

$$\mathbf{AW}^{(h)}_{i,t}(\hat{x}) := \sum_{j \in \mathrm{sprt}^{(h)}_{\mathcal{L}_{i,t}}(\hat{x})} \mathbf{w}(j)$$

、

根据假设 10.1.5，由于诚实节点最多只改变一次其投票，所有其他诚实节点都以很高概率看到此投票。换句话说，每个诚实节点对所有其他诚实节点投票有相同感知（概率较高）。在此情况下，如果对于所有$1 \le i, j \le N_h$，$\mathbf{AW}^{(h)}_{i,t}(\hat{x}) = \mathbf{AW}^{(h)}_{j,t}(\hat{x})$成立的话，我们可以列出交易$\hat{x}$诚实节点所看到的诚实 AW：

$$\mathbf{AW}^{(h)}_{t}(\hat{x}) := \mathbf{AW}^{(h)}_{1,t}(\hat{x}) \qquad (23)$$

对手节点可能改变其看法，尤其在接近阈值时间 D 时这样做，这样诚实节点可能对对手投票有不同感知，但该感知差异受到对手权重的限制。和文献[68]类似，对于每个 c∈C，我们将对手控制区域（或区间）定义为：

$$I_t(c) = [\mathbf{AW}^{(h)}_{t}(c), \mathbf{AW}^{(h)}_{t}(c) + q]; \qquad (24)$$

见图 13。此区间下限（或上限）刚好是当所有恶意节点对其投反对票（或赞成票）时，冲突的整体 AW。

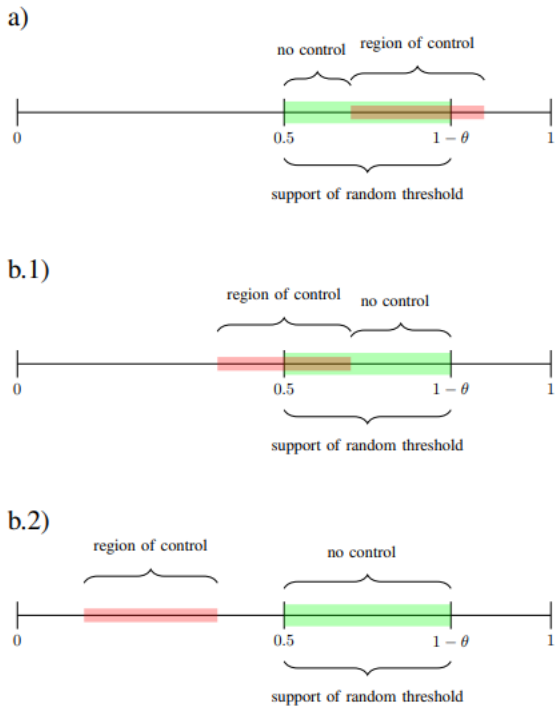
我们在以下语句概括上述考虑。

引理 10.1。假设诚实节点对诚实 AW 具有相同感知，则对于所有 i，$1 \le i \le N_h$，下式成立：

$$\mathbf{AW}_{i,t}(c) \in I_t(c). \qquad (25)$$

上述情况适用于每个满足假设 10.1.2 的对手策略。目前思路是以一种独立于诚实 AW 和对手策略的方式，选择 dRNG 支持，所有诚实节点均以正概率对相同现实做出决定。

为此，每 D 个单位时间，我们都有一个正概率，所有节点按照该正概率对相同现实做出决定。需要此类区间的几何分布数，直至所有诚实节点就相同现实达成一致。



定义 10.1 [收敛至共识状态]我们说当且仅当存在某个现实 R 和某个（随机）时刻 T，使得

$$\mathbf{AW}_{i,t}(R) > \theta, \forall i \in \{1, \ldots, N_h\}, \forall t > T. \qquad (26)$$

时，协议才会收敛到共识状态。

备注 10.2 定义 10.1 与共识状态的定义相似，见定义 7.6。虽然它描述协议的渐近行为，但没有提供切实可行的确认标准[14] 。不过，定义 4.6 所示「确认规则」总是容易受到账本状态可能「重组」[15]的影响；另见引理 9.1。对发生这种重组可能性的量化取决于精确通信和对抗模型，它超出本文的范围。

该讨论可以转化为用算法 4 编写的正式协议描述，我们得到如下定理。

定理 10.1（存活性和安全性-同步）。设

$$q < \min\left\{1 - \theta, \theta - \tfrac{1}{2}\right\}$$

[14] 这点的「概率性」原因是 T 并非停止时间。
[15] 重组指已经确认的交易不再处于偏好现实中的程序。

为对手权重，则根据假设 10.1，（算法 4 所述）协议收敛至共识状态。

证明。我们在 t_0=0 时刻，以大小为|C|的固定冲突集 C 开始协议，让节点交换投票，直至 D 时刻。我们设 ε>0 任意但固定，并且在证明结束时确定其数值。每个节点均等到 D+d 时刻。如果一个节点没有收到第一个随机数 X_1，则它将以 X_1 作为随机数运行算法 3。如果一个节点没有按时收到随机数，它将用 θ 的阈值（而非随机 X_1）运行算法 3。在 D+d 和 2D 之间，每个诚实节点均不会改变其偏好现实。设 A_1 为所有诚实节点都投给自己偏好现实的事件，而所有其他诚实节点都会看到这些投票。设 B_1 为所有诚实节点按时表达其偏好现实的事件，见假设 10.1.5。设 C_1 为所有这些区块在 2D 时刻前到达所有其他节点的事件。

由于 $A_1 = B_1 \cap C_1$，我们有

$$\begin{aligned}\mathbb{P}(A_1) &= \mathbb{P}(C_1|B_1)\mathbb{P}(B_1)\\ &\geq (1-\varepsilon)^{|\mathcal{C}|N_h}(1-\varepsilon)^{N_h}\\ &= (1-\varepsilon)^{N_h(|\mathcal{C}|+1)}.\end{aligned}$$

在 2D+d 时刻，所有诚实节点都以至少 $(1-\varepsilon)^{N_h}$ 概率接收新随机数 X_2。为此，所有诚实节点按以下概率

$$p(\varepsilon) := (1-\varepsilon)^{N_h(|\mathcal{C}|+1)}$$

就式（23）定义的诚实 AW，以及阈值 X_2 达成一致。我们写作$\mathbf{AW}^{(h)}(c) := \mathbf{AW}^{(h)}_{2\mathcal{D}}(c)$。这里注意到没有诚实节点可以感知诚实 AW。但为了便于分析，我们假设系统状态有完整视图或全部信息。

我们通过初始化 R=∅和 U=C，开始在冲突图中进行递归论证。将每次迭代时，第一个 while 循环内算法 3 选中的冲突定义为$\bar{c}^* := \arg\max\{\mathbf{AW}^{(h)}(c), \quad c \in \max_{\mathcal{C}}(U)\}$。我们区分两种情形。

A 情形：$\mathbf{AW}^{(h)}$（c*）>0.5。随机阈值支持度确实在 0.5 以上；另见图 13。更确切地说，每个节点（通过算法 3）将此冲突纳入其偏好现实的概率 ξ_A 满足 $\xi_A > \mathbf{AW}^{(h)}$（c*）−0.5>0。所有和 c*（即冲突图 N_C（c*）中的相邻节点）冲突的冲突都不被偏好。请注意，由于每个诚实节点均可能对实际 AW 有不同感知，它可能以不同「顺序」来运行算法 3。但由于冲突图中没有两个相邻节点诚实 AW 超过 0.5，算法会先处理所有「A 情形」，再处理下一种情形。

B 情形：$\mathbf{AW}^{(h)}$（c*）≤0.5。在这种情形中下，$c^* \cup N_{\mathcal{C}}(c^*)$所有冲突诚实的 AW 都小于 0.5。（这是因为在算法 3 中，节点是按「AW 递减」顺序处理冲突）。由于 q<θ−0.5，按正概率 ξB，所有这些冲突 AW 都不会超过阈值 X_2，也不会在算法 3 第一个 while 循环中被添加到偏好现实中。

现在我们从集合 U 中移除冲突，继续此过程，直到集合 U 为空集。

我们设置$\xi = \min\{\xi_A, \xi_B\}$。令 K 为冲突图中最大独立集尺寸。最后，节点就 A 情形偏好冲突达成一致，其正概率至少为ξ^K。节点必须用算法 3 中第二个 while 循环来填补最大分支。既然它们在 X_2 值上达成一致，那么它们也在偏好现实上达成一致。

总体而言，在长度为 D 的下个时期中，所有诚实节点均以至少$p(\varepsilon)\ \xi^K$的正概率投给相同现实。在此情况下，下个时期将获得大于 θ 的 AW。否则我们将重复此过程，直至它得到满足。所需时期数以几何随机变量为准。

备注 10.3 上述证明为估计「共识时间」T 提供可能。事实上，其预期值的上限为$\mathcal{D} \cdot (1 + (p(\varepsilon)^{\cdot} \cdot \xi^K)^{-1})$。

该定量分析是与定理 7.1 的主要区别之一。定理 7.1 并未获得「共识时间」的限值。另一个关键区别是，定理 10.1 不像假设 7.1 那样要求对数据包的随机性和发出进行假设。

备注 10.4 「冲突集是固定的」这一假设被简化为在协议运行时间内，冲突集是有限值的。为此结果也适用于随时间演变的冲突集。但对于较大冲突集，证明中的定量限值会变得更糟。

11. 展望-未来研究

将本文提出的共识机制与基于现实账本相结合，可支持多种流程并行化，比如处理、预订和投票。这样可以显著提高性能，因为其支持多线程并发。我们解决方案的多线程潜力、异步环境工作能力和无领导方法可提供高性能共识和账本解决方案。但仍需详细可靠的性能分析来验证通过理论预测的性能。

由于账本可以在缺乏对添加到账本的新交易全局知识的情况下进行，节点可以在不了解所有区块情况下与我们机制达成共识。该方法可能会在 Tangle 层上直接启用某些分区方案，其中节点只观察总账本中的一部分。但这种方法可能降低性能，降低安全性和/或存活性。为了解决我们方案对分区场景的可行性，一些关键问题（如必要假设和全面安全性分析）必不可少。

生成赞成权重的权重系统可通过多个来源，在不同环境下构建。比如，权重可通过令牌值派生，系统可被允许或不允许操作。另一种方法是通过信誉系统获得权重，但目前这一方法很少受到关注。

通过在第 6 节区块引用的基础上引入交易引用，我们可以利用算法 2 降低交易孤立性，但这并不能完全解决问题。比如，一笔诚实交易只能被最终拒绝的交易（直接）引用，可能永远无法具有可被视为确认的足够 AW。该缺陷可通过几种方式加以改进。首先，节点可保留其「自身」交易作为末梢，直至被确认，这类似于自动重新附加区块。其次，节点也可能将交易保留在其偏好现实中，但未在末梢池中投给它们。接着，我们可利用交易引用来支持交易。第三，对于给定区块，可以允许区块引用和交易引用产生冲突。这样，交易可以优先于交易中的区块

引用，使得从被引用聚合分支中移除部分分支成为可能。另一个更精准投票的解决方案是引入更多引用类型，最终使节点更显式地从被引用区块的获支持分支中移除特定分支。上述例子证明末梢选择算法求解可以缓解或减轻孤立性问题，但需要彻底分析，以涵盖边缘情形。

附录 A

汇合时间估计

本节给出汇合时间τ_c上限。

在网络处于低负载状态下，我们可以假设末梢池较小。经几次赞成后，所有新交易均间接引用此交易。在高负载状态下，末梢池大小$L_0 \gg$k 和汇合时间可以比较大。K(t)表示在 t 时刻赞成给定交易 x 的末梢数。在 t 时刻，一笔新交易根据 t-h 时刻 Tangle 状态选择 k 个末梢。为此，下式给出一笔新交易至少赞成 K(t-h)个末梢，而这些末梢赞成 x 的概率：

$$1-\left(1-\frac{K(t-h)}{L_0}\right)^k. \qquad (27)$$

如上所述，在时间间隔 h 内，我们有λh个新末梢到达，而λh个末梢被批准。于是，在 t-h 时刻属于末梢的交易在 t 时刻不再是末梢的概率为

$$\frac{\lambda h}{L_0}=\frac{k-1}{k}. \qquad (28)$$

因此，在 t 时刻，我们有(1)/kK(t-h)个之前末梢仍然是末梢，(k-1)/kK(t-h)个末梢已经被引用，不再属于末梢。对于从集合 A（或集合 B）中选择某一父节点的概率，我们用 A 表示仍然为末梢的引用 x 的末梢集，用 B 表示在[t-h,h]中被赞成的引用 x 的末梢。记为：

$$p_A=\frac{K(t-h)}{kL_0}, \text{ resp. } p_B=\frac{(k-1)K(t-h)}{kL_0} \qquad (29)$$

设 p_1 为赞成 B（而非 A）中至少一笔交易的概率，设 p_2 为从集合 A 中选择至少两个父节点的概率。设 Y_A 为集合 A 赞成的末梢数。注意到在第一个事件中，引用给定交易的末梢数增加 1 倍，二在第二个事件，末梢数减少 Y_A-1 倍。

第一个事件概率可以用二项式分布来描述。事实上，

$$p_1=\sum_{i=1}^{k}\binom{k}{i}p_B^i(1-p_A-p_B)^{k-i}. \qquad (30)$$

由于假设 p_B 很小，领先项是针对 i∈{1，2}而言，我们得到

$$p_1\approx kp_B+\frac{1}{2}k(k-1)p_B^2. \qquad (31)$$

随机变量 Y_A 服从二项式分布，因此

$$\mathbb{P}[Y_1 \geq 2] = \sum_{i=2}^{k} \binom{k}{i} p_A^i (1-p_A)^{k-i}. \qquad (32)$$

由于 K(t−h)较小，p_A 也比较小，上述表达式中领先项是针对 i=2 而言。为此，第二个事件的发生概率近似等于

$$p_2 = \frac{1}{2} k(k-1) p_A^2, \qquad (33)$$

末梢池大小实际减少 1。与文献[3]类似，此时我们写出 K(t)的微分式。只考虑 p_1 和 p_2 的一阶项，因为我们认为 K(t)很小：

$$\frac{dK(t)}{dt} = (p_1 - p_2)\lambda \approx \lambda \frac{(k-1)K(t-h)}{L_0} \qquad (34)$$

利用式（9），我们可以写出

$$\frac{dK(t)}{dt} \approx \frac{(k-1)^2 K(t-h)}{kh}, \qquad (35)$$

带边界条件 K(0) =1。根据文献[3]思路，我们得到以下形式的解：

$$K(t) = \exp\left(W\left(\frac{(k-1)^2}{k} \right) \frac{t}{h} \right), \qquad (36)$$

式中，W(·)是所谓的朗伯 W 函数。取两个边的对数，我们发现当 K(t)达到 εL_0 时，时间大致为

$$\tau_c \approx \frac{h}{W\left(\frac{(k-1)^2}{k}\right)} (\log L_0 + \log \varepsilon). \qquad (37)$$

对于较大 k，我们可以近似 $W\left(\frac{(k-1)^2}{k}\right) \approx 2\log(k-1) - \log k \approx \log k$，并得到

$$\tau_c \approx \frac{h}{\log k} \log(L_0) \approx \frac{1}{\log k} h \log(\lambda h). \qquad (38)$$

附录 B

术语表

赞成权重 计算赞成给定交易的网络「相关」部分的函数

冲突 消耗相同输出，但作为不同交易的交易

冲突交易 在其过去锥中包含两笔交易的两笔交易，这两笔交易消耗某笔交易的相同输出

锥 DAG 中顶点集，该顶点集可沿着 DAG 边的方向（过去锥）和相反方向（未来锥），从给定顶点到达。

分支 不包含冲突交易且为过去-闭合集的冲突集

分支 DAG 代表分支之间关系的 DAG

账本 DAG 以 DAG 形式存储所有交易的数据结构

Tangle DAG 一种以 DAG 形式存储所有区块的数据结构

投票 DAG 一种代表 Tangle DAG 和账本 DAG 组合的增强型 DAG，用于确定投票锥

生成 作为 UTXO 账本中任何交易最终前身的交易

区块 Tangle DAG 的一个元素，由引用至少两个区块的已识别数据组成

节点 作为网络一部分的机器。其作用是发出新区块，并验证现有区块

现实 最大分支

凝固 在过去锥给定区块中检索丢失区块的过程，可由节点提出请求

见证权重 计算赞成给定区块「相关」 网络部分的函数

致谢

作者在此感谢 GoShimmer 软件开发团队，该团队通过 IOTA 2.0 协议的原型实现支持本研究。同时感谢 IOTA 基金会工作人员和 IOTA 社区成员的反馈和批评。

参考文献

[1] S. Nakamoto，「比特币：一种点对点电子现金系统，」 2008.

[2] Y. Sompolinsky 和 A. Zohar，「确保比特币高速交易处理」，《金融密码使用与数据安全》，R. Bohme 和 T. Okamoto，主编，Berlin，Heidelberg：施普林格 Berlin Heidelberg，2015，pp. 507-527。

[3] S. Popov，「Tangle，」 2015。

[4] A. de Vries 和 C. Stoll，「比特币日益严峻的电子垃圾问题」，《资源、保护和回收》，第 175 卷，p. 105901，2021。

[5] I. Makarov 和 A. Schoar，「比特币市场区块链分析」，美国国家经济研究局，工作底稿 29396，2021 年 10 月

[6] P. Daian，S. Goldfeder，T. Kell，Y. Li，X. Zhao，I. Bentov，L. Breiden- bach 和 A. Juels，「Flash boys 2.0：去中心化交易所非法预先交易、矿机可提取价值和共识不稳定性，2020 年 IEEE 安全性与隐私研讨会（SP），2020，第 910-927 页。

[7] D. Rosenthal，「EE380 对话，」 2022. [在线]。可从 https: //blog.dshr.org/2022/02/ee380-talk.html 检索。

[8] M. Dotan，Y.-A. Pignolet，S. Schmid，S. Tochner 和 A. Zohar，「SOK：加密货币网络环境，最新技术和挑战」，第 15 届可用性、可靠性和安全性国际会议论文集，ARES '20。美国纽约州纽约市：美国计算机协会，2020 年。

[9] M. Belotti，N. Bozic，G. Pujolle 和 S. Secci，「区块链科技手册：时间对象和方式」，IEEE 通信调查教程，第 21 卷，第 4 期。第 3796-3838 页，2019。

[10] Q. Wang，J. Yu，S. Chen 和 Y. Xiang，「Sok：挖掘基于 DAG 的区块链系统」，预印本，第 abs/2012.06128 卷，2020

[11] V. Buterin，「以太坊：下一代智能合约和去中心化应用平台」，2013。

[12] M. M. T. Chakravarty，J. Chapman，K. MacKenzie，O. Melkonian，M. Peyton Jones 和 P. Wadler，「扩展式 UTXO 模型」，《金融密码使用与数据安全》，M. Bernhard，A. Bracciali，L. J. Camp，S. Matsuo，A. Maurushat，P. B. R0nne 和 M. Sala，主编。Cham：施普林格国际出版公司，2020，pp. 525-539。

[13] T. Rocket，M. Yin，K. Sekniqi，R. van Renesse 和 E. G. Sirer，「通过亚稳态实现可扩展性和概率无领导 BFT 共识，」 2019。

[14] S. Popov，H. Moog，D. Camargo，A. Capossele，V. Dimitrov，A. Gal，A. Greve，B. Kusmierz，S. Mueller，A. Penzkofer，O. Saa，W. Sanders，L. Vigneri，W. Welz 和 V. Attias，「Coordicide，」 2019。

[15] S. Muller，A. Penzkofer，N. Polyanskii，J. Theis，W. Sanders 和 H. Moog，「基于现实的

utxo 账本」，

2012（在线）。可从 https: //arxiv.org/abs/2205.01345 检索

[16] T. D. Chandra 和 S. Toueg，「可靠分布式系统的不可靠故障检测」，《美国计算机协会学报》，

第 43 卷第 2 期，第 225-267 页，1996 年 3 月。

[17] A. Penzkofer，B. Kusmierz，A. Capossele，W. Sanders 和 O. Saa，「IOTA 协议寄生虫链检测」，第二届国际区块链经济、安全性和协议会议（代币经济学，2020）2020，第 8：1-8：18 页。

[18] L. Lamport，R. Shostak 和 M. Pease，《拜占庭将军问题》，美国计算机协会编程语言与系统，第 4 卷第 3 期，第 382-401 页，1982 年 7 月。

[19] M. J. Fischer，N. A. Lynch 和 M. S. Paterson，「具有单一错误过程的分布式共识的不可能性」，2003。

[20] C. Dwork，N. Lynch 和 L. Stockmeyer，「部分同步背景下的共识」，《美国计算机协会学报》，第 35 卷第 2 期，第 288-323 页，1988 年 4 月。

[21] F. Cristian 和 C. Fetzer，「异步分布式系统的时序模型」，IEEE 并行与分布式系统交易，第 10 卷第 6 期，第 642-657 页，1999。

[22] A. Dembo，S. Kannan，E. N. Tas，D. Tse，P. Viswanath，X. Wang 和 O. Zeitouni，「一切都是一场竞赛，中本聪永远是赢家」。美国纽约州纽约市：美国计算机协会，2020，第 859-878 页。

[23] P. Gazi，A. Kiayias 和 A. Russell，比特币严格一致性约束。美国纽约州纽约市：计算机协会，2020，第 819-838 页。

[24] R. Pass，L. Seeman 和 A. Shelat，「异步网络区块链协议研究」，《 密码学进展-欧洲密码学年会》，2017，J.-S. Coron 和 J. B. Nielsen，主编，Cham：施普林格国际出版公司，2017，第 643-673 页。

[25] C. Natoli 和 V. Gramoli，「针对工作量证明区块链的收支攻击：以 R3 试验台为例」，CoRR，第 abs/1612.09426 卷，2016。

[26] R. Guerraoui 和 A. Schiper，《共识：重大误区》，1997。

[27] M. K. Aguilera，《在共识研究上跌倒：误解与问题》。柏林海德堡：施普林格，2010，第 59-72 页。

[28] M. Ben-Or，「自由选择（抽象拓展层次）的另一个优势：完全异步协议」，第二届美国计算机协会分布式计算原理年会论文集，ser. PODC '83。美国纽约州纽约市：计算机协会，1983，第 27-30 页。

[29] M. O. Rabin，「随机化拜占庭将军问题」，（sfcs 1983），1983，第 403-409 页。

[30] V. B. Misic，J. Misic 和 X. Chang，「比特币类分布式网络分叉与分叉特性研究」，2019 年 IEEE 国际区块链会议（区块链），2019，第 212-219 页。

[31] M. Iqbal 和 R. Matulevicius，「探讨区块链系统的 Sybil 和双花风险」， IEEE Access，第 9 卷，第 76153-76177 页，2021。

[32] M. Neuder，D. J. Moroz，R. Rao 和 D. C. Parkes，「sub-1/3 利益相关者发起的以太坊 2.0 低成本攻击」， 2021。

[33] J. P. T. Lovejoy，「基于工作量证明加密货币的链重组和双花攻击的实证分析」，论文，麻省理工学院电气工程与计算机科学系，2020。

[34] Y. Sompolinsky 和 A. Zohar，「加速比特币交易处理，快钱长在树上，而不是链上，」 国际密码研究协会预印本文档，第 2013 卷，第 881 页，2013。

[35] nxtforum.org 人员，「DAG：一种广义区块链，」 2014。[在线]可从 https: //nxtforum.org/proof-of-stake-algorithm/ dag-a-generalized-blockchain/检索

[36] S. D. Lerner，「DagCoin：一种无区块加密货币」，2015。[在线]可从 https: //bitslog.wordpress.com/2015/09/11/ dagcoin/检索。

[37] Y. Lewenberg，Y. Sompolinsky 和 A. Zohar，「兼容式区块链协议」， 《金融密码使用与数据安全》，施普林格，2015，第 528-547 页。

[38] Y. Sompolinsky，Y. Lewenberg 和 A. Zohar，「Spectre：一种快速可扩展加密货币协议」，密码学预印本文档，第 2016/1159 号报告,2016。

[39] Y. Sompolinsky，S. Wyborski 和 A. Zohar，幻影 DAG：一种中本聪共识的可扩展泛化：2021 年 9 月 2 日，美国纽约州纽约市：美国计算机协会，2020，第 57-70 页。

[40] A. Miller，Y. Xia，K. Croman，E. Shi 和 D. Song，「蜜獾 BFT 协议」，2016 年计算机协会安全、审计和控制特别兴趣小组计算机与通信安全会议论文集，ser. CCS '16，美国纽约州纽约市：计算机协会，2016,第 31-42 页。

[41] S. Duan，M. K. Reiter 和 H. Zhang，「节拍：异步 BFT 的实用化」，2018 年计算机协会安全、审计和控制特别兴趣小组计算机与通信安全会议论文集，2018，第 2028-2041 页。

[42] L. Baird 和 A. Luykx，「哈希图协议：高通量分布式账本的高效异步 BFT」，2020 年国际

全层智能系统会议（COINS），2020，第 1-7 页。

[43] A. Gagol，D. Lesniak，D. Straszak 和 M. Swietek，「Aleph：带有拜占庭节点的异步网络中的高效原子广播，」 CoRR，第 abs/1908.05156 卷，2019。

[44] S. Muller，A. Penzkofer，B. Kusmierz，D. Camargo 和 W. J. Buchanan，带有加权投票的快速概率共识，」2020 年未来科技会议（FTC）论文集，第 2 卷，2021，第 360-378 页。

[45] S. Popov 和 S. Muller，「基于投票的概率共识及其在分布式账本中的应用」，《电信年鉴》，2021。

[46] A. Cullen，P. Ferraro，W. Sanders，L. Vigneri 和 R. Shorten，「物联网分布式账本的访问控制：一种联网方法」，IEEE 物联网杂志，第 1-1 页，2021。

[47] B. Kusmierz，W. Sanders，A. Penzkofer，A. Capossele 和 A. Gal，「Tangle 在均匀随机性和随机游动末梢选择方面的特性，」 2019 年 IEEE 国际区块链会议（区块链），2019，第 228-236 页。

[48] P. Ferraro，C. King 和 R. Shorten，「智慧城市分布式账本、共享经济和社会合规性」，IEEE Access，第 6 卷，第 62728-62746 页，2018。

[49] W. Li，「齐波夫定律无处不在」， 《语言计量学》，第 5 卷第 14-21 页，2002。

[50] L. A. Adamic 和 B. Huberman，「齐波夫定律和互联网，」《语言计量学》，第 3 卷第 143-150 页，2002。

[51] D. Kondor，M. Posfai，I. Csabai 和 G. Vattay，「富者更富？比特币交易网络实证分析，」《公共科学图书馆•综合》，第 9 卷第 2 期，第 1-10 页，02 2014。

[52] B. Kusmierz，S. Muller 和 A. Capossele，「DAG 分布式账本和应用的委员会选择 」，《智能计算》，K. Arai，主编，Cham：施普林格国际出版公司，2021，第 840-857 页。

[53] B. Kusmierz，S. Muller 和 A. Capossele，「DAG 分布式账本和应用的委员会选择，」 第 840-857 页，2021。

[54] C. King，「共享账本随机图模型的流体限制」 ， 《应用概率论进展》，第 53 卷第 1 期，第 81-106 页，2021。

[55] E. Drasutis，「IOTA 智能合约」，2022 年 1 月检索。

[56] C. Dwork，N. Lynch 和 L. Stockmeyer，「部分同步背景下的共识」，《美国计算机协会学报》，第 35 卷第 2 期，第 288-323 页，1988 年 4 月。

[57] T. D. Chandra 和 S. Toueg，「可靠分布式系统的不可靠故障检测」，《美国计算机协会学

报》，

第 43 卷第 2 期，第 225-267 页，1996 年 3 月。

[58] J. Beauquier，P. Blanchard，J. Burman 和 R. Guerraoui，「人群协议中熵的好处」，第 19 届国际分布式计算原理系统会议(OPODIS 2015)，ser.Leibniz 国际信息学论文集（LIPIcs），E. Anceaume, C. Cachin, and M. Potop-Butucaru, 主编，第 46 卷，德国 Dagstuhl：Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik，2016，第 1-15 页。

[59] A. Penzkofer，O. Saa 和 D. Dziubaltowska，「延迟类对 IOTA 数据结构的影响，」CBT2021–第 5 届加密货币与区块链技术研讨会，2021。

[60] S. Popov 和 W. J. Buchanan，「FPC-BI：拜占庭架构内的快速概率共识，」《并行与分布式计算杂志》，第 147 卷，第 77-86 页，2021。

[61] A. Capossele，S. Muller 和 A. Penzkofer，「拜占庭架构内投票共识协议的鲁棒性和效率」区块链：研究与应用，第 2 卷第 1 期，第 100007 页，2021。

[62] I. Cascudo 和 B. David，「SCRAPE：经公共实体证明的可扩展随机性」，国际应用密码学与网络安全会议。施普林格，2017,第 537–556 页。

[63] A. K. Lenstra 和 B. Wesolowski，「带有 Sloth、Unicorn 和 Trx 的可信公共随机性」，国际应用密码学杂志，第 3 卷第 4 期，第 330-343 页，2017.

[64] S. Popov，「一种分散式去信任伪随机数生成算法」，《数学密码学学报》第 11 卷第 1 期，第 37–43, 2017。

[65] P. Schindler，A. Judmayer，N. Stifter 和 E. Weippl，「HydRand：实用连续分布式随机性」，

《密码学预印本文档》，第 2018/319 号报告，2018。

[66] E. Syta，P. Jovanovic，E. K. Kogias，N. Gailly，L. Gasser，I. Khoffi，M. J. Fischer 和 B. Ford，「可扩展抗偏差分布式随机性」，2017 年 IEEE 安全性与隐私性（SP）研讨会。IEEE，2017，第 444-460 页。

[67] B. Wesolowski，「高效可验证延迟函数，」《密码学杂志》第 33 卷，第 2113-2147 页，2020。

[68] S. Muller，R. C. Nitchai 和 S. Popov，「集合 FPC，」 2021。