阿维：一种经济可持续地永久保持信息的协议

Sam Williams
sam@aweave.org

Viktor Diordiiev
viktor@aweave.org

Lev Berman
lev@aweave.org

India Raybould
india@aweave.org

Ivan Uemlianin
ivan@aweave.org

草案-1
C0bQOaBVZrF4Y5Y4 -FU6DRle4yBhaMlNphglTrb06Hp

翻译：刘毅
liuyi3@gmail.com

版本：V1.0

摘要

从第一条区块链——比特币[47]以来，区块链已被作为存储历史记忆的机制。尽管区块链技术在构建无单点故障的弹性归档领域具有明显的潜力，但是链上数据存储技术的进展依然乏善可陈。本文通过介绍阿维协议解决这一问题的，阿维是一种基于机制设计的的新型方法，用于实现可持续的知识和历史永久账本。除了概述实现数据永久保存的激励机制，本文还概述实现可扩展链上存储的关键技术。

目录

1 介绍

1.1 动机

本文展示阿维协议，一种称为块织物（Blockweave）的类区块链新型数据结构。该协议旨在以可持续的方式提供可扩展的永久链上数据存储。块织物是永在网（Permaweb）的底层数据结构。永在网是一组驻留在块织物上的数据、网站和去中心化应用，可以通过普通Web浏览器访问。

在本文中，我们将介绍几项技术创新，包括：块织物、区块影（Blockshadow）、AIIA、去中心化内容策略和它们的机制设计。综合运用这几项技术创新，阿维能够在链上存储领域提供独特的效用。

我们在个人之间以及世代之间保存和共享信息的总体能力对人类的成功至关重要。尽管已尽力而为，但是通观历史，我们保存知识的方式容易遭受破坏，信息容易丢失或更改，有时是故意的，更多是因为事故。就像在古代世界一样，现代历史也充斥着重要信息被破坏，篡改和丢失的案例，从图书馆、档案馆的火灾[41，22，35]到专制国家焚书[57]，不一而足。

如今，被大量数字信息围绕的我们可以轻松地假设，由于信息可以从线上轻松获得，因此不会被更改或丢失。可惜事实远非如此[21，37]。尽管互联网作为分布式信息传播系统已获得广泛的成功，但尚有待去中心化的、永久知识存储系统弥补其不足。

如今，构成万维网的几乎所有页面都驻留在中心化数据存储中，这些存储设施通常由组织甚至个人控制。这意味着，当我们在线访问信息时，完全依赖这些组织和个人允许我们这样做。访问权可以随时被取消，数据可能无意间丢失或劣化。在互联网上提供信息会产生服务器和其他维护成本，这意味着一旦维护资金耗尽，整个网站、应用和信息存储都将消失。中心化数据存储的另一大风险是数据易受这些组织和个人的操纵。这些操纵诡计通常是在存储期间对文件进行修改[34，48]。阿维能解决这些问题。

一些政府正在加大审查力度，以杜绝对互联网上政治敏感信息的访问[68，6，1]。同样，对于媒体和新闻机构，我们曾经拥有实体出版物作为不可撤销的副本，但现在只限于访问数字信息，然后立即弃之不顾。由此媒体更改其过去的文章内容已不鲜见。这可能导致重要的上下文和内容丢失，或含混不清。

为使信息真正永久存在，存储必须既有弹性又去中心化。区块链技术显然具有实现弹性的、去中心化的信息保存的潜力[14]，因为区块链的一个关键特征就是所有数据一旦上链就不能更改。但是传统上，此类技术可伸缩性严重不足，限制了其存储大量数据的用途。

高昂的交易费用也抑制了典型的区块链扩展规模以容纳大量数据的能力。例如通过以太坊网络，尽管从技术上可以在链上存储数据，但高昂的费用令多数用例不切实际。

随着数据存储需求呈指数增长[56]，可扩展的、去中心化的、低成本数据存储协议已成为必需品。

数据一旦存入数据结构，就会与网络中每个先前区块密码化地编织在一起。这样可以确保任何更改文档内容的企图都会被自动检测，并被网络拒绝。因此，阿维提供了一种健壮的方法来永久地在链上保存数据，使之免于被有意或意外地丢弃和操纵。

归功于永在网去中心化的和密码学验证的本性，任何组织或群组都无法合谋审查其内容[36]。就其本身而言，阿维大大降低乔治.奥威尔所警示的“记忆漏洞” [51]发生的可能性。通过这种方式，我们把阿维协议看作维护信息自由的有用工具，也由此加强了依赖信息自由的机构和民主进程。

但是，为使阿维协议发挥其潜力，并使永在网成为新的去中心化互联网的基础，开发者的广泛采用是必要条件。就其本身而言，该协议旨在使开发者很容易地在链上低成本地、快速地存储内容。 阿维地HTTP API使在块织物上构建去中心化Web应用极其简单，开发者可以使用所有他们喜欢和熟悉的Web技术（包括HTML、Javascript和CSS），并在几分钟内部署应用到永在网。

对于最终用户，基于阿维协议构建的永在网具有几个主要优势。首先，正如我们前面所提到的，保证用户对永在网内容的可靠、不可变的访问权。尤其重要的是，用户能够可靠地永久保持对全部永在网应用和网站的访问，而不仅仅是它们所显示内容。这意味着，永在性应用一旦发布就无法取消发布，这有助于维护严格的应用完整性[5]。也使用户可以根据自己的个人喜好做出有意义的选择。

永在网内容的访问速度对于用户体验、阿维协议和永在网本身的成功都非常重要。 阿维协议是唯一可以提供真正的永久性和去中心化数据存储的系统。该协议被设计为激励阿维节点向网络快速共享数据，这是在第6章阐述的协议机制设计的关键方面，届时将介绍AIIA元博弈及其实现机制——野火（Wildfire）。

阿维协议及其技术家族均经过精心设计，以确保每个节点的行为都倾向于对网络本身效用做积极贡献。以这种方式，协议设计达到“主导策略激励相融”（DSIC，[58]），这是如阿维这样长期去中心化的激励驱动的网络的关键方面。

因此，阿维协议旨在解决许多传统归档系统中的缺点，以及在许多典型区块链协议中常见的可扩展性问题。该协议旨在最大程度地提升开发者和最终用户体验的质量和效率，这对于广泛采用至关重要。

1.2 本文档的结构

1. 简介
2. 阿维：关键贡献
3. 网络：机制设计
4. 节点：协议遵从行为
5. 去中心化内容策略

6. 阿维的自适应机制设计：自适应交互激励代理（ AIIA）
7. 阿维协议互操作性和永在网
8. 未来工作
9. 结论
10. 附录

本文档分为八章（原文如此），以及一个附录。这里是第1章——介绍，明确创建阿维协议主要动机和目标：提供永久的、弹性的存储。

第2章重点介绍阿维协议在去中心化数据存储领域的关键创新贡献。所描述的特定创新包括：块织物，访问证明共识机制，回忆块，状态记忆，区块影，自适应交互激励代理（AIIA）元博弈，以及野火——一种AIIA博弈的实现。此章解释了块织物与传统区块链数据结构的差别，以及这些差别的用意和好处。

第3章“网络：机制设计”确立了分布式去中心化存储对于满足永久性和适应性的需求是必要的。采用博弈论和机制设计方法，本章介绍阿维协议如何被精心设计，以平衡激励和约束条件，从而产生一个主导策略激励兼容（DSIC，[58]）的网络，最终最大化网络整体输出的亲社会（prosocial）效用。

第4章“节点：协议遵从行为”从节点在网络中的行为角度研究网络和协议。本章详细介绍了节点的主要活动：开采区块、接收和验证区块和交易，以及提供区块和交易。为了说明阿维协议如何促进亲社会行为，本节包括一些会发生的情况的示例，展示节点在尝试“欺骗”时会发生什么。

第5章——民主内容策略，描述阿维矿工，网络中数据存储的提供者，只接受、存储和共享他们选择的内容的机制。本节描述了内容策略的三个主要方面：投票阶段，存储阶段和激励（或机制）设计，在这里你将发现块织物如何使矿工集体决定希望或不希望存储的内容。

第6章涵盖自适应交互激励代理（AIIA）现象，介绍了阿维协议的机制设计和技术实现如何相结合，创造出互惠和激励平衡的强大网络。剖析通过这一交互涌现出的强大效应，展示协议如何产生高水平的积极外部性，包括节点和用户之间的亲社会性，而不必对系统中的代理实行强制约束。

第7章-阿维协议互操作性和永在网，描述了所有关键技术和基础组件共同构成阿维协议的，随即构成了永在网本身。本章从基础层面描述了协议如何以无信任、无服务器和去中心化方式运行。此外，还列举了该协议面向开发者和矿工的特定永在网技术：HTTP API、客户端服务器、网关节点、混合架构、ArQL、阿维DNS和TLS，以及一些真实的永在网应用。

第8章-未来的工作，探讨了阿维核心开发团队对于可能向阿维协议添加哪些技术以进一步增强其独特功能的看法。首先，本章描述了简洁访问证明，即数据存储可保持弹性、可验证和安全，但降低通过网络传播的交易中包含的数据量，从而提高效率。其次，考虑减少专用于维护完整钱包日志的存储空间的建议。最后，详细讨论了一种潜在的快速查找机制的建议，该提议将从根本上加快当请求发生时在网络定位数据的速度。

第10章-附录列举了一系列新颖的阿维数据结构的具体内容，包括区块、块阴影和交易，还有密钥数据集。

2 阿维：关键贡献

本节介绍了阿维协议的独特创新，以及这些创新如何影响阿维网络数据存储的可用性、弹性和永久性。

2.1 回忆块，访问证明和块织物

在阿维的块织物数据结构中，每个块都链接到两个先前的块：“链”中的前一个块（与传统区块链协议相同），以及来自先前历史的区块（“回忆块”）。因此，阿维区块链数据结构并不是严格的链（即单链列表），而是稍微复杂一些的图结构，我们称之为块织物。

回忆块是根据前一个块的哈希和高度确定的。这导致回忆块在历史中的选定既有确定性，又不可预测。

为了开采或验证新块，节点必须存有该块的回忆块。证明矿工能够访问回忆块是构造新块工作的一部分（另一方面，验证此证明是验证新区块工作的一部分）。访问证明（PoA）是工作量证明（PoW）的增强版，其间整个回忆块数据都包含在哈希运算的输入中，结果作为工作量证明的输入。有关区块构建的更多详细内容，请参见第4.1.1节。

PoA鼓励了存储，因为矿工需要访问块织物历史上的随机块才能开采新块并获得挖矿奖励。

PoA算法还激励矿工更愿意存储“稀有”区块，而不是复制良好的区块。这是因为当稀有区块被选定为回忆块时，拥有它的矿工能在更小竞争的情况下竞猜PoW谜题，争夺相同级别的奖励。结果是在相同条件下，倾向于存储稀有矿块的矿工长期来看会获得更高回报。

PoA采用概率和激励驱动的方法来最大化的网络中任何数据的冗余副本数量。相比之下，其他去中心化存储网络则为特定数据指定的确切数量的冗余副本，并使用“合同”系统对其进行调整[12]。阿维的基于竞争的方法经过量身定制，使其更简单、更通用，灵活适应良好和不佳的环境。

例如，网络中存储1个PB的数据，只有2PB的存储容量（一个'不健康”网络），PoA的激励结构推动矿工尽量存储更少副本的数据，以适应当前的状况。在良好的网络中，例如可用的100 PB存储只保存1PB数据，阿维还是推动矿工在网络中制作尽量多的的数据冗余副本。除了在这种情况下提高安全性和稳定性之外，这种通过激励实现最大化冗余的方法还可以显着加快查找和访问速度。

2.2 状态记忆

在标准的区块链链范例中（例如比特币区块链[47]），区块链是完全复制到网络中所有全节点的具体对象。每个全节点必须完整存储区块链。如前一节所述，阿维链节点不需要这样。阿维的新颖数据结构——块织物，不需要矿工存储所有先前的块。为此，处理新块和新交易所需的所有数据都存在于每个块的状态中（有关区块数据结构的详细信息，参见10.2.1）。

由于这种记忆技术，新用户加入网络仅需从其信任的对等方下载当前块，或者向后验证部分工作量证明（验证数量越大，加入网络所需的信任越低）。除其他数据项外，区块数据结构包括区块哈希列表（BHL）和钱包列表（WL）。拥有区块哈希列表使得旧区块可以被请求和/或验证。拥有钱包列表可以验证新交易，而无需处理钱包上一次交易所在的区块。区块哈希列表和钱包列表由矿工保持更新，并在挖矿和验证新区块时通过网络进行同步（有关挖矿期间节点行为的详细信息，请参见第4章）。

这减少了矿工进入的障碍——存储空间、算力和时间，使得块织物能够扩展到任何单个矿工都无力提供的容量。鉴于阿维网络的目标是超过传统Web的尺寸、范围和通量，该机制对于允许矿工切实地加入网络至关重要。

如果有兴趣验证从第一个块到当前块，并在本地重建整个块织物，可以采用几种方法。例如，通过每个块的指针到达前一个块，或者从可信的对等方请求区块哈希列表，对照当前块的不平衡梅克尔树哈希列表进行验证，然后直接下载这些块。

一旦加入并处于活动状态，节点完全不需要存储整个块织物，但是，PoA访问证明机制的奖励与存储块织物的量成比例。网络的主导策略激励兼容（DSIC）[58]特性为网络级的存储和复制提供了保证。这样就解放了每个节点，可以根据自己的偏好和资源情况对存储进行优先级排序和优化。

区块链通常是一个非常大的分布式数据结构，下载完整的区块链会花费很长时间，并且会消耗大量的计算资源[13]。与传统的区块链系统不同，阿维没有全节点和轻量客户端的典型概念，只有或多或少地下载了块织物的客户端。使用阿维，完全同步即不是风险也不是义务，而是矿工获得更多奖励的可选升级途径。

2.3 区块影

在传统的区块链网络中，当一个新区块被开采出来，整个区块都要传播给网络中所有全节点，无论它们已经持有了多少交易。这极大地限制区块能包含的数据量，因为所有数据需要在共识期间在网络中散播。如果在区块接受期间传输太多数据，达成共识所需的时间就很长，容易导致网络分叉。共识期间区块链网络出现分叉的概率是：

$$P(fork) = \frac{B_{dist\_time}}{B_{time}}$$

（公式1）

由于区块分发时间与区块尺寸成正比，由此可见区块尺寸跟分叉概率成正比。阿维协议将数据本身存储在区块里，因此的常规区块链分叉概率$P_{(fork)}$和区块最大尺寸B_{max_size}之间的折衷是不可接受的。

因此，阿维协议采用了一种新的数据分发方法：区块影，在Graphene [52]和紧凑区块(BIP-152)[19]的工作成果上构建。

区块影通过将交易与区块解耦来工作，仅需在节点之间发送最小的“区块影”，就使对等方重建完整的区块，而不必传输区块本身。区块影包含钱包列表和区块哈希列表，以及交易哈希列表（而不是区块内的交易）。根据这些信息（通常为几千字节），一个已经保存了所有交易的节点，再加上最新的区块哈希列表和钱包列表，可以快速重建几乎任意大小的完整区块。使用这种机制，区块大小的瓶颈就变成了可以包含的交易ID数量，以及从其组成交易中重建区块所需的时间。

为了促进区块分发并降低通信开销，节点会立即彼此共享交易，但是只有高度确定网络中其他节点也具有交易副本之后，才会尝试将交易打包进区块。区块影改进了Graphene [52]和BIP-152 [19]提出的工作，并引入了一种基于机制设计的AIIA博弈，用于计算网络中其他节点已经拥有区块数据的可能性。如果节点没有交易数据，它不会尝试从其他节点获取，因此节点被鼓励采取以下方式：

1. 不要过早地将交易打包进块，因为这会导致区块被拒绝。
2. 不要过晚地将交易打包进块，因为网络中其他矿工可能抢先一步打包交易。

应用区块影系统的结果是快速灵活的区块分发过程，该过程允许按照交易在网络中散播的速度尽快将它们打包进区块，并且以接近网络允许的速度对区块达成共识。

2.4 内容策略和抗审查

阿维协议避免将存储全部内容作为义务，允许每个节点自行决定存储哪些块和交易。节点没有被迫存储他们不想要的内容，这提高了网络的抗审查能力[3]。

网络的默认行为是每个被接受的交易都有大量副本。网络中当前的复制率超过97%。作为区块验证的必要部分，交易被存储在链上并在整个网络中复制，因此数据在地理上分布广泛。存储在网络层面（而不是单个矿工）得到概率保证，因此非常强韧并抗干涉。这些优势对最终用户和矿工都是有益的，最终用户得到比其他方法更快的数据访问速度和更可靠的存储，而矿工能够选择他们希望存储的区块和交易，从而能够实施符合自身要求的内容策略（有关此类策略的更多信息参见第5章）。

2.5 野火

野火机制是一种AIIA博弈形态（参阅第6章），阿维网络的每个节点都基于两个主要因素对其他节点进行排名。首先是对等方的慷慨度——发送新交易和区块，其次是对等方的响应性——迅速响应信息请求，其机制类似于Bittorrent的乐观针锋相对算法[18]。然后，节点优先与排名高的对手方通信。这使得节点能够合理化其带宽分配。考虑到对等方之间进行交互的实际影响，还具有促进节点的亲社会行为的效果。

请参阅第3.4以获取野火机制的更多信息。关于自适应互动激励代理（AIIA）元博弈的内容请参阅第6章，Wildre是AIIA代理的实例。

3 网络：机制设计

网络的参与者分为两类：矿工和用户。用户支付通证（AR）将数据添加到网络。网络中的矿工收取AR以开采新块，就需要存储和提供数据。一个钱包所有者既可以既是用户又是矿工。矿工从将数据添加到网络的用户间接获得奖励。由网络通过开采和验证新区块从总体上调节用户到矿工的通证支付。

打算添加到网络的数据封装在交易中，被开采成区块。区块既是一组交易的容器，又是处理这组交易之后的网络状态备忘表达。从用户的角度来看，网络中有两种交易类型：数据交易和价值交易。用户可以发起数据交易以将数据存储到区块中。价值交易仅包含两个钱包中AR余额的变化：发起交易的钱包余额减少，接收交易钱包余额增加。在技术实现层面，阿维网络中的所有交易都是相同的。每个交易有一个可选的收款人字段，和一个可选的数据字段。这意味着交易可以高度灵活，例如，可以用于在两个参与者之间发送邮件（更多永在网应用示例请参见第7.5节，包括Weavemail）。有关区块和交易内容的详细信息请参见第10.2，关于节点如何处理这些数据结构请参见第4章。

3.1 难度：调节区块生成速率

为了调节网络中的区块生成速率，访问证明（PoA）算法允许可变的难度设置。具有较高难度PoA挑战需要更长时间来计算。“难度”是一个网络统计信息，包含在每个模块中[47]。

如果网络中区块的生成速率超过目标频率，则未来用于区块生成的PoA谜题的难度就会增加。同样，随着网络中区块生成速度的降低，难度设置会向下调整。

通过这种方式，去中心化网络中的矿工网络能够调节区块生成速率（相应的影响各种奖励发放速率，详见下文），而与网络中节点的数量以及可用于解决PoA谜题的算力和存储空间无关。

3.2 通证经济

3.2.1 向网络支付矿工奖励

阿维网络使用通证，其稀缺性由块织物数据结构的共识机制保障。通证的主要单位是AR，子单位是Winston，其中1 AR = 1,000,000,000,000 Winstons。

由于系统中的通证是稀缺的，可用于两个有价值的功能，即将数据编码到系统和奖励矿工，因此通证本身具有非零的金融价值。尽管AR通证最初是因为作为支付永久数据存储费用的唯一手段而具有实用性的，它也可以用作价值交换的手段。

2018年6月8日网络启动时，创世区块创建5500万个AR。此外，还有1100万个AR（创世通证量的20%）将通过出块奖励逐步进入流通。因此，AR的最大发行量是6600万个。流通中AR通证要么在钱包里，要么在基金池中。有关增发函数和基金池的详细信息请参见3.2.3。

为了将交易写到区块中，用户必须支付一些AR作为交易费。不同于传统的区块链系统[47，67]，交易费用并未全部直接转移给开采该区块的矿工。大部分交易费进入存储基金，该基金根据3.2.3节描述的机制逐步分配到矿工钱包。

3.2.2 永久数据存储成本

由于阿维的核心功能是为其用户提供永久存储，因此必须明确对该存储定价的机制。

作为计算永久数据存储成本的前提，我们必须首先确定单个时间段的数据存储成本：

$$P_{GBH} = \frac{HDD_{price}}{HDD_{sz} * HDD_{mtbf}}$$

（公式2）

其中：
P_{GBH} = 在1个硬盘驱动器上存储1GB数据1小时的价格
HDD_{price} = 购买硬盘驱动器的最低市场价格
HDD_{sz} = 硬盘驱动器的容量
HDD_{mtbf} = 硬盘驱动器平均故障时间间隔

自从数字化数据存储技术出现以来，市售存储介质存储GB小时的成本一直在以惊人的速度下降（参见图1）。在过去的50年，GB小时成本每年平均下降30.57%（数据集在附录第10.1提供）。

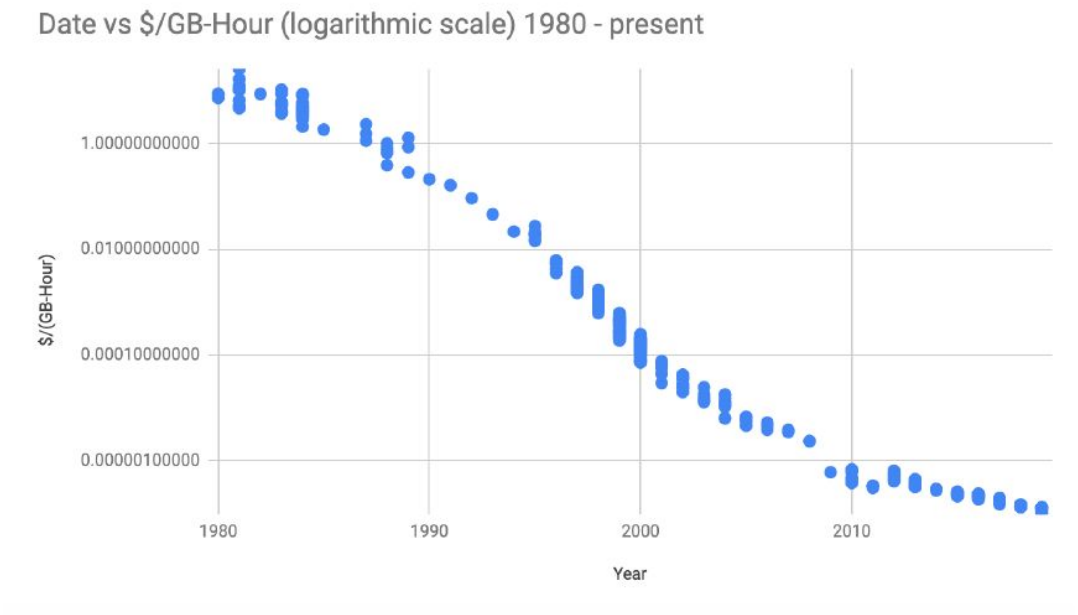

图1

仔细且保守地对数据存储成本做外推，展现出为无限存储提供有限成本的机会。

永久存储的成本可以建模为随时间下降的存储成本的总和：

$$Pstore = \sum_{i=0}^{\infty}(Data_{size} * P_{GBH}[i])$$

（公式3）

其中：
P_{store} = 永久存储的价格
$P_{GBH[i]}$ = 在时间i存储1GB数据每小时的价格
$Data_{size}$ = 要存储的数据量

3.2.3 交易定价

为符合上述成本模型，阿维协议采用了存储基金机制。该机制允许网络随着时间推移向矿工分配适当数量的通证，以可持续地激励永久存储任意量的数据。

阿维网络中的交易定价包括两个部分：对永久存储成本高度保守估计，以及为激励矿工接受新交易进入新区块所立即发放的交易奖励。

交易费用的计算如下：

$$TX_{cost} = TX_{size} * \sum_{i=BH}^{\infty} P_{GBB}[i]$$

$$TX_{reward} = TX_{cost} * C_{fee}$$

$$TX_{total} = TX_{cost} + TX_{reward}$$

(公式4)

其中：

TX_{cost} = 网络为TX交易永久服务的总成本

TX_{size} = TX交易的数据段尺寸 (以GB为单位)

$P_{GBB[i]}$ = 在区块高度i存储1GB小时的价格

C_{fee} = 定义矿工即时奖励的常量

TX_{total} = 用户为提交交易向网络支付的总费用

3.2.4 存储基金

为了让阿维挖矿保持盈利，并具有长期可持续性，必须遵守以下基本原则：网络在任意给定区块发出的奖励都必须大于维护当期的块织物的总支出费用。具体来说：

$$\forall B \in W_{blocks} \quad R_{total} >= W_{size} * P_{GBB}$$

（公式5）

虽然从基金持续释放通证自然地满足了此约束（假设按法币计价的通证价格稳定而且P_{GBB}的预测准确），当矿工已经通过其他方式达到盈利时，协议应避免释放基金通证。这一机制进一步增强了网络的经济稳定性，以抵御通证价格和存储介质价格的波动。为了实现这种稳定效果，挖矿奖励机制仅在维护块织物所需的价值支出超过增发奖励和即时释放交易费用的情况下才花费基金。

矿工开采区块的奖励由三部分组成：

$$R_{total} = R_{fees} + R_{inflation} + R_{endowment}$$

（公式6）

R_{fees}是区块内交易的交易费总和：

$$R_{fees} = \sum_{i=0}^{|B_{TXs}|} B_{TXs}[i]_{fee}$$

（公式7）

每个块的增发奖励是预先定义的，并仅随区块高度逐渐降低，如下所示：

$$R_{inflation} = \frac{G_{AR} * 0.2 * \ln 2 * 2^{-BH}}{B_{ny}}$$

（公式8）

其中：
G_{AR} = 创世块中的AR数量：55,000,000；
B_{ny} = 一年的区块数量：262,800；
BH = 当前区块高度；

【译者注：原文有误，修正公式如下。简而言之：创世区块增发约29.01个AR，此后无级地逐年减半。】

$$\mathrm{R}_{inflation} = \frac{G_{AR} * 0.2 * ln2}{B_{ny} * 2^{\frac{BH}{B_{ny}}}}$$

（公式8-勘误）

如上所述，仅在$R_{inflation}$加上$R_{rewards}$不足以在当前阶段维持网络的存储负荷时，才会从基金中提取通证。鉴于相对于哈希运算，存储成本极低（请参阅第3.3节获取访问证明价值支出均衡的详细信息），直到永在网比当前的万维网大很多倍为止（根据互联网档案馆每年的网络抓取规模估算[62]），存储负担都很可能不会达到令矿工从基金提取费用的程度。这意味着存储基金在实践中有必要被定期使用之前，将获得大量的“浮存”通证（可能积累多年）。

从基金中提取的的通证总数按以下方式计算：

$$R_{endowment} = Prop(\max(0, (W_{size} * P_{GBB}) - (R_{inflation} + R_{fees})))$$

（公式9）

Prop函数计算从基金提取的基本奖励，并使其正比于开采所需的回忆块的尺寸除以阿维网络区块的平均尺寸。这样可以确保在区块明显大于或小于平均值的情况下，均匀地适当激励矿工在网络中存储数据。

$$Prop(AR) = \min(Endowment_{[i]}, \frac{B_{recall\_sz}}{B_{avg\_sz}} * AR)$$ （公式10）

其中：
B_{recall_sz} = 回忆块的尺寸
B_{avg_sz} = W_{size} / B_{height}

最后，转移到下个块的基金可以这样计算：

$$Endowment_{[BH+1]} = Endowment_{[BH]} - R_{endowment} + \sum_{i=0}^{|TXs|} TXs[i]_{cost}$$ （公式11）

其中：
BH = 当前区块高度
基金[BH] = 在BH区块高度的基金金额，单位AR
基金[BH] = 在下一区块高度的基金金额，单位AR

3.2.5 未来数据密度和可靠性的期望

数据密度或存储介质的可靠性增加（导致P_{GBB} 下降）对于以上各节的论证很关键。从图1可以看出，这种模式已经持续了50多年。尽管仅凭历史分析并不能保证这一趋势的持续性，但我们注意到许多令人信服的因素，表明这一趋势足以持续到阿维网络本身所处的技术周期终结（请参阅3.2.6）：

1. 不同于CPU领域，计算时钟速度正在趋近于理论物理极限，因而摩尔定律正在减速[17]，数据密度的情况远非如此：

   当前消费级存储介质的密度最大值是1.66*10的12次方（比特/立方厘米）

   研究中获得的最大数据密度是2.5*10的25次方（比特/立方厘米）

   理论上的最大数据密度[11]是1.53*10的67次方（比特/立方厘米）

   从我们当前的位置，以每年30%的乐观数据密度增长速度，将需要434年才能达到最大理论极限，如果每年20%则需要697年，每年10%则需要1329年。

2. 即使数据密度提高速度放缓，存储介质的可靠性（平均故障间隔时间）仍将继续增加，并且可以说具有更光明的前景[33，39，69]。阿维挖矿可盈利性的度量核心指标GB小时成本，对数据密度和介质可靠性的反应相同。

3. 由于人类对数据需求的增长速度[56]，未来开发更低GB小时成本的存储机制具有很强的动机。因此数据密度/存储介质可靠性不能持续提升是小概率事件。

3.2.6 网络可朽，数据永在

所有技术都有周期[10]。尽管阿维的机制设计总体上是为了加强对新情况的适应性，但阿维核心团队并不期望该网络会按照当前规划持续产生区块，直到真正意义的永恒。但是，这并不意味着我们认为到阿维的最后一个区块被开采出来以后，存储在网络中的信息将丢失。在我们的预期里，一个更适合时代挑战的永久信息存储系统将会出现，阿维的数据将被“包含”到该网络中。在开采出最后一个区块之后，经济激励机制将让位于社会激励，以继续保存数据。这一结果将与极低的数据移植成本相伴，因为随着时间的推移，存储的相对成本已持续下降。

在人类历史上，存档退休时被“嵌套”进其他存档很常见。阿维永在网中就能找到Gopherspace的存档（“知识网” [4]，在基于HTTP的万维网[25]之前）。在Gopherspace存档里，人们能找到早期基于Telnet和电子公告牌的讨论系统。同样，美国国会图书馆的许多内容现在都保存在网上[38]，实际上其中许多书籍本身就是老故事/诗歌的合集。我们预计后网络时代阿维的数据将以类似的模式继续保存，这将受到网络中所有数据的密码学交织的支持（部分的真正验证需要整体的存在），以及网络鼓励创建信息的许多副本，其中的大多数不会被删除，即使矿工本身与网络断开连接。

3.3 访问证明：主导策略

节点在挖矿方面的主导策略[58]是能最大化挖矿收益的策略。正如我们在上面看到的，每个区块挖矿奖励很大程度上不在矿工的控制范围之内。最大化报酬的策略就变成了最大化节点首先开采新区块能力的策略，并要在其他节点在相同高度开采到新区块之前这样做。节点可以参与开采区块的概率等于节点存储了新区块的回忆块的概率，实质就是上就是节点存储的块占总区块的比例。节点首先开采到区块的概率，是该节点的哈希能力相对于所有其他也拥有回忆块的节点的总哈希算力比值。

$$P(\text{win}) = P(\text{has recall block}) * P(\text{finds hash first})$$

（公式12）

其中：

$P_{(\text{has recall block})}$ = 本地保存的区块数量 / 区块高度

$P_{(\text{finds hash first})}$ = 本地哈希算力 / 全网哈希算力

如上文第3.2节所述，提高本地存储的区块比例，或者提高节点的算力都能提高效用。第4章探讨了该战略中节点的战术。

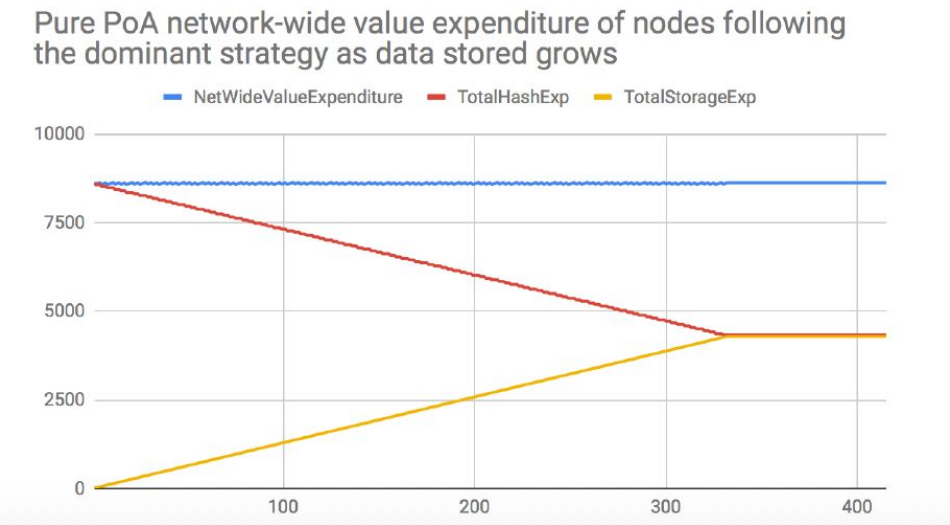

图2：矿工随着数据存储量增加而采取的价值分配主导策略

3.4 野火代理实现AIIA

阿维协议的参与者参与的是无共识基础的自适应元博弈，其中节点根据对等方提供的效用以任意方式相互评分。当前的阿维参考实现是名为野火的元博弈基础代理。野火是Bittorrent协议的乐观针锋相对带宽分享激励机制的衍生品。有关阿维网络自适应元博弈的更多详细信息，请参见第6章。

挖矿节点的可利用带宽总是有限的。因此，矿工必须合理地分配其带宽资源，以便最大化获得挖矿奖励的机会。参考阿维节点实现利用其野火代理参与AIIA元博弈（对AIIA的进一步说明见第6章），以鼓励矿工参与亲社会行为，例如，快速响应对等方的交易或区块请求。在野火中，挖矿节点对所有对等节点根据其响应性进行评分，并做排名，对排名高的对等节点优先考虑出站消息作为回报（例如，传播新区块和交易）。向最近对自己帮助最大的节点发送消息，是节点的效用最大化策略。它还具有激励接收请求节点提高响应性的副作用：节点需要避免被降级，甚至可能从其对等方的联系列表中被删除。最后，这会改善网络的整体响应能力，推动低响应性的节点改进，否则终将被排除在网络参与之外。

3.4.1 理性分配出站带宽

每个节点都维护一个对等节点列表。这些对等节点包括在启动时给定的“可信”节点，以及收到阿维API请求的远程节点。这些是节点发送交易、区块和信息请求的对等方。

每个对等方均获得一个评分，该分数代表其对评判者的API请求的响应速度和准确性。分数本质上是向对等方最近发出一些请求的响应速度（字节每秒）的移动平均值。新加入的对等方获得一定的宽限期，在此期间他们不参与排名。

对等方列表将定期修剪，运转不良的对等方被概率性地删除（即删除的概率正比于排名）。

对新对等方的宽大和概率性删除是考虑到对等方响应性会有短期波动。

在传播交易或区块时，对等方被分为两档：首先向运行最佳的对等方并行发送消息，然后按排名向其余对等方（运行不佳的对等方和新对等方）顺序发生消息。

信息请求也受益于野火的排名和裁剪对等方列表机制。

这种启发式方法确保节点可以合理化分配其拥有的任意带宽，并确保它们准确、及时地与对等方通信。这样可以最大程度地提高传播和响应的有效性，并减少时间和资源浪费（例如避免将消息发送到退休或恶意节点）。

3.4.2 提升响应性

接收信息请求的节点必须假定该请求是来自使用野火（或者其他AIIA代理，参见第6章）监控响应性的对等方（无论已知的或者新的）。高速连接性对于挖矿和避免分叉至关重要，迅速而准确地响应所有请求有利于节点的利益。

节点不对请求区别对待还有一个原因。因为某些请求不是来自矿工，而是来自边缘或外部用户。由于在此类用户使用网络应该是免费的（例如请求交易数据的用户），因此节点绝对不可对请求区别对待或者要求回报。

节点只能访问自己维护的排名，但不了解自己在对等方的评分。因此，可以给出一个净效用函数如下：

$$Utility = \overline{\forall P \in Peers \quad Rank(P, Self)}$$

（公式13）

但是，节点能体验到其等级降低的后果，包括：接收传播和信息请求的频率降低，来自新对等节点的请求减少，最终是挖矿机会减少且遇到更多分叉，以及其他影响。

3.4.3 网络生态学

在AIIA博弈中运行野火代理节点的网络得以实现吞吐量最大化，与Bittorrent蜂群的方式类似。该网络可以看作带有加权边的有向图，权重的单位是“最近N个响应的每秒字节数”，请参见图3。网络中的通路显示可能的区块或交易传播速率。在任意路径上最低的野火评分是该路径上的最大流通速率。由于阿维网络通常是高度连接的，单条慢速边不一定会形成瓶颈。

在这种情况下，排名较低的对等方从对等方列表中逐渐消失，节点间的连接性会增强。对于在多数对等方排名较差的节点，负面影响尤其强烈。希望继续有效挖矿的排名较低的节点具有强烈的动机来升级其带宽或以其他方式提高响应性。

野火的本质是节点的对等节点按照响应的准确度和速度进行排名。节点使用的特定实现甚至特定度量对其他节点都是不可见的。我们不能，也不需要，假设每个节点都使用相同的AIIA代理（有关这一动力学的更多解释，请参见第6章）。

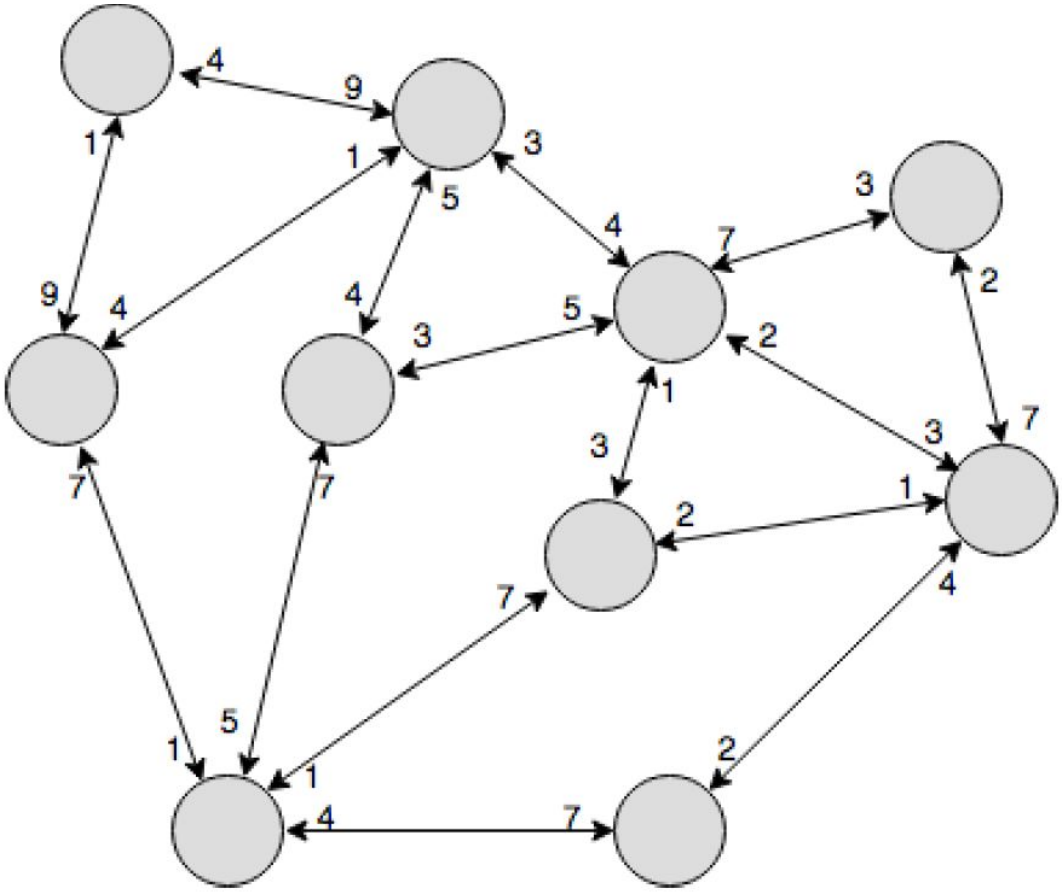

图3：可视为加权图的野火

3.5 抗分叉和恢复

当多个区块同时产生并被分发到网络，会导致多个候选块被不同节点接受，网络共识开始崩溃，形成“分叉”。快速恢复网络共识非常重要，本节讨论阿维协议解决此类问题的机制。

3.5.1 目标行为和激励措施

分叉会给单个节点和整个网络带来风险。理想行为首先包括避免分叉，还有尽快恢复到最佳分叉。

网络中出现分支的潜在原因之一是传播延迟[20]。因此节点被鼓励在最小验证成功完成后尽快散播区块和交易。节点将对数据执行严格的“边缘”验证，以保护网络免受攻击，然后传播这些数据。节点被激励在此阶段完成一些验证，因为传播无效数据可能会对排名产生负面影响。类似地，节点被激励快速开采区块，以最大化获得挖矿奖励的可能性，还要尽快传播第一

个有效的新候选块，如果节点接受一个新块，则它希望其他节点也接受该同块。除了这些激励措施之外，避免的分叉恢复机会成本的最好方式是在一开始就避免分叉。

但是，分叉总会发生，并且当发生时，节点必须能够有效地评估它是否在主分叉上，如果不是，则需要立即恢复到主分叉。阿维分叉选择规则类似于传统的基于中本聪共识的区块链协议。

每个块都包含一个“积累难度”字段，该字段代表快织物进入特定分支直到该块之前的工作量。节点可以立即将自己的当前块与收到的块进行比较，以评估收到的块（a）有效可接受则传播；（b）无效或者陈旧，应予以忽略；（c）分叉选择的证据。后者发生在收到的块具比节点的当前块有更大的累积难度，而且收到的块并不是当前块的后代。

当发现一个更优分叉块，节点请求得到从分叉点之后的全部块。逐一对它们进行验证（参见图4）。

在图4中，下面一条线的当前区块高度是39，当前块的累积难度为421。它接收的块高度为41，累积难度为529。接收的块能通过验证，可能预示着存在优选分叉。节点从新的高度41块开始上溯（从每个块依次获取“前一块”字段），直到到达自身历史记录中的一个块（即在节点自己的块哈希列表中）。在图中，在高度38处发现了分叉点（步骤1、2和3）。
这条新线成为节点的优选分叉，并采用累积难度为529的块作为当前块（步骤4）。

现在高度39累积难度421的块无效，如果该块中的交易不在优选分叉的三个最新块之中，则丢弃。

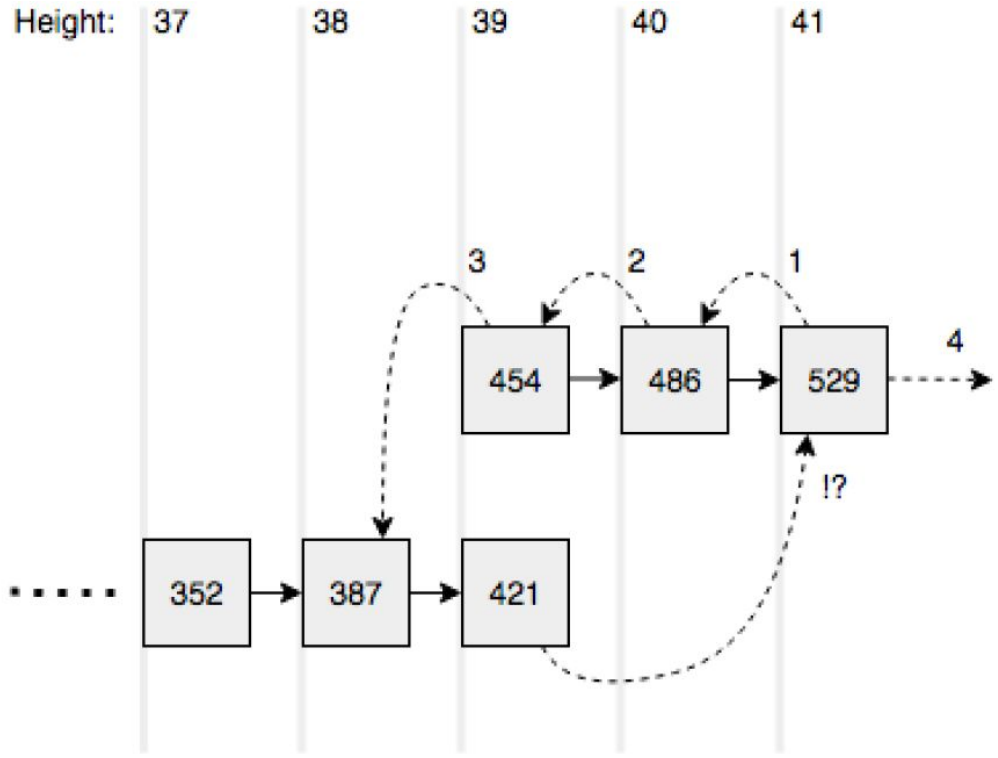

图4：具有累积难度的分叉

4 节点：协议遵从行为

本章描述阿维节点如何遵从上一章详述的阿维协议。节点的主导策略是优化挖矿奖励，这是通过开采和传播新区块来实现的。该节点还接收传入消息，并根据协议的激励结构做出反应。在这些消息中最重要的是新区块、将被打包进块的新交易，以及交易处理请求。

其他活动和其他激励措施会影响节点开采区块并使这些区块被网络接受的能力。也就是节点挖矿能力的先决条件或约束。

为了开采区块，节点的运营者必须考虑以下要求（参见图5）：

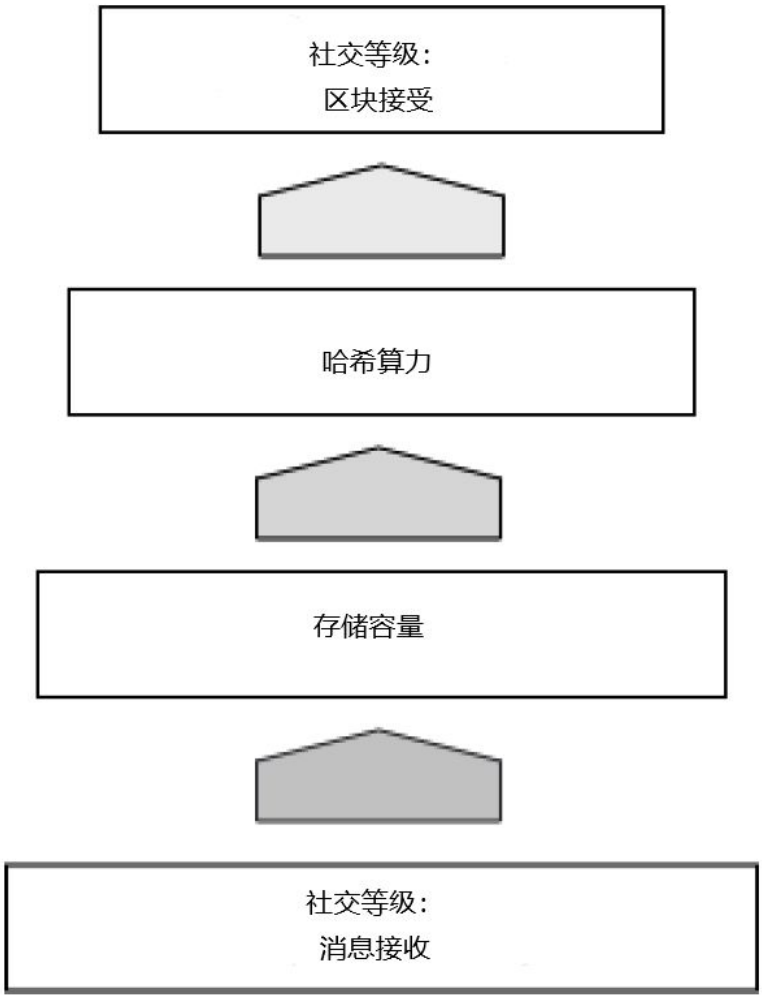

图5：挖矿的条件

1. **接收区块和交易。**节点在自适应互动激励代理（AIIA）元博弈中的社交等级将直接影响节点接收新区块和交易的延迟，甚至是否能收到它们。社交等级低的节点往往会比其他

节点更晚接收信息，并且可能接不到所有信息。最终，低社交等级的节点有被网络完全排斥的风险。如果节点未接收到区块和交易信息，就无法有效地挖矿。因此，合理的对等评分的是在阿维网络挖矿的基本前提条件。

2. **访问回忆块。**为了开采新块，节点必须拥有与新块关联的回忆块。由于$B_{n.recall}$无法在挖到B_{n-1}之前预测（请参见公式14），因节点拥有回忆块的概率与该节点存储的块织物的总量成比例。这受到节点的存储容量限制。

3. **生成候选块。**一旦完成第一阶段和第二阶段，节点就必须使用计算能力进行“竞争”，以生成候选块。节点为网络提供的计算能力越高，就越有可能首先产生候选块。

4. **获得对候选块的承认。**最后阶段这个又回到了网络的社交本质：AIIA社交等级低的节点有消息（包括新候选块）传播风险太慢而不被网络接受的风险。一旦生成，新的候选块必须被网络接受，开采它的节点才能收到奖励。除了AIIA评分外，候选区块中包含的交易也会影响它被网络其他节点接受可能性。例如，候选块中包含了网络中大量节点的内容策略黑名单所禁止的交易，则它不太可能被网络其余节点所接受（有关此过程的详细信息，请参阅第5章）。

考虑到这种情况，我们将在下一部分中首先探讨挖矿，然后探讨通信，最后关注分叉避免和恢复。

4.1 开采和传播新区块

开采和传播新块是连续的活动。如第3.2.1节所述，节点开采一个新区块获得的总奖励包括保证金奖励【译者注：原文如此，应为立即获得的交易费奖励】、增发奖励还可能有一小笔存储基金。由于将新交易打包到候选区块会立即获得奖励，因此矿工有很强的动力将有待执行的交易打包进候选区块（除此之外还有增加存储基金的长期激励）。即便没有交易费，开采新块的激励也足够，因为成功的矿工会获得增发奖励，在某些情况下，还会从存储基金中获得少量收益。

4.1.1 块构建过程

挖矿节点有两个交易池：等待池和开采池。当新交易到达时，将由节点验证并使用节点的内容策略进行扫描（有关此过程的详细信息，请参阅第5章）。一旦该节点有较大的把握认为其他节点也已接收到该交易（请参见第4.2节），则把交易从等待池移至开采池。一旦节点开采了一个区块并被网络接受，该区块内的交易将从开采池中删除。在协议层面，分叉恢复（请参阅下面的4.4节）可能导致交易从被放弃的区块重新进入开采池。

新块的区块数据段（BDS）是以下各项的哈希：

1. 上一块的独立哈希。
2. 回忆块的全部内容。
3. 候选区块的交易和其他元数据。

在阿维协议的工作量证明中，BDS充当了“谜题”。

下图（图6）显示了前一个块的信息、回忆块和交易信息如何并入BDS和新的候选块。新块包含重建BDS和验证工作量证明所需的所有信息，除了回忆块是节点必须已经保有的。

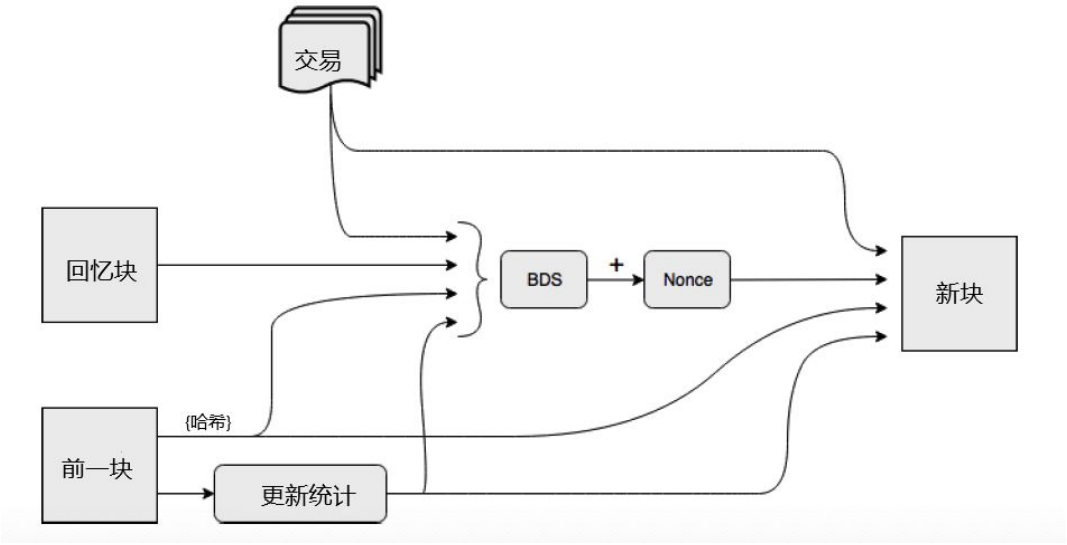

图6：从先一块、回忆块和交易构建新块

生成新的候选块有五个步骤：

1. 组装相关的元数据；
2. 收集一组交易打包进块，对这一交易集合整体进行验证，计算当前难度，并根据难度验证交易费用；
3. 生成新的块数据段（BDS）；
4. 根据给定的BDS，找到满足难度要求的Nounc [47]；
5. 以备忘的形式（区块影）打包并传播新块。

第一步：组装相关元数据

每个节点都在其状态（或等效地，在当前块中）中保有当前块哈希表（BHL）和当前区块高度（有关块结构请参见附录10.2.1）。当前块的独立哈希值模除当前区块高度得到回忆块的高度（参见公式14）。

$$BH_{recall} = BHL_{[B_{indep\_hash} \mod B_{height}]}$$

（公式14）

此高度将在[0-当前块高度)范围内。注意，这排除了当前块充当回忆块的可能性。因此，回忆块必然是块织物历史上的一个区块，并且始终需要两个不同的区块才能开采新区块。

下一个回忆块的ID（即高度或者和独立哈希）当且仅当先一区块已被开采后才能知道。因此，矿工们无法提前预测需要哪个回忆块才能开采下一个块。这意味着的任何矿工所能采取最有效的策略是存储尽可能多的区块历史（在第3.3节中所述的访问证明均衡范围内）。矿工能够参与某一轮次访问证明（PoA）哈希竞猜的概率，等于它保存的区块历史的比例。

如下面进一步详细介绍的。节点有权访问回忆块的证明（“访问证明”）包括在了生成新的块数据段的过程中。

第二步：获取和维护交易集

交易不断到达并不断被验证，在开采新块时，节点必须决定打包哪些交易。当交易第一次到达节点时，将被单独验证，然后其转移进等待池。在适当的等待时间之后（详细信息参见第3章），等待池中交易的一个子集（可能是全集）被选进开采池，在这里把这些交易作为一组进行验证。组中的所有交易都必须能在当前的网络状态和钱包列表上正常执行。

第三步：生成块数据段

回忆块（包括其中的交易及数据）被包含进新的块数据段（BDS）中（详细内容参见10.2.1节）。因此，要让新块通过验证并被接受，正确的回忆块必须被用来生成BDS。因此这构成了矿工在PoW挑战阶段可以访问到回忆块的证明。

第四步：找到有效的Nonce

接下来，矿工必须尝试找个一个Nonce随机数，以产生满足难度要求的的新块哈希。阿维协议中使用的挖矿算法是RandomX [64]。在参考实现中，默认的矿工数量是其主机上所有CPU内核减一。找到的Nonce被包括进新块数据结构作为工作量证明。

如果在第四步时节点接收到应该被打包进新块的新交易，则该过程将中断。在这种情况下，使用新交易集合从第二步开始重做，生成一个新的BDS。

第五步：散播候选块

在步骤五，区块被打包并按照排名的优先级散播给对等方（更多说明请参见第6章）。节点获得挖矿奖励的前提它的候选块先于其他候选块被网络接受。这意味着节点的块散播速度是其挖矿效率的重要决定因素。

4.1.2 激励

矿工因开采新区块而收到网络的奖励。虽然大部分交易费进入存储基金，但矿工会立即收到交易奖励和增发奖励，在某些情况下还会收到一部分现有存储基金（交易和奖励定价详细信息请参阅第3.2.3节）。奖励不是通过特殊交易支付给矿工（那样必须通过网络进行验证，造成块织物不必要的臃肿），而是直接扣减存储基金并更新钱包列表。更新的存储基金和钱包列表被集成进新的BDS和新区块的数据结，从而确定了新网络通证的所有权状态。

推论是，网络对新区块的接受事实上也是对奖励的接受。对矿工的报酬是特定分叉状态的一部分。如果在分叉恢复期间某些区块被拒绝，则这些区块的挖矿奖励也将失效。

4.2 接收、验证和散播交易和区块

4.2.1 交易

当节点接收到新交易（从另一个阿维节点，或从边缘客户端或应用程序），该节点将验证交易费用，并且对前一笔交易的引用与钱包列表中的内容匹配。如果成功验证了交易，则节点会尽快把交易散播给它的对等节点。与AIIA元博弈和避免分叉相关的激励机制促使了这种行为（参见第6章）。

收到交易后直接散播，而不是仅仅将其打包进区块，符合节点的利益。这是因为单个交易要被网络中的其他大多数节点接受为合法，包含该交易的区块才会被接受。

在进入交易池之前验证交易的步骤如下：

1. 格式不正确的交易（有关交易的完整结构请参见附录10.2.4）被接收节点忽略；
2. 已经处理过的交易被丢弃；
3. 与交易相关的钱包必须包含足够余额，以便处理交易以及来自该钱包的全部未决交易；
4. 交易发起者和交易目标不能指向同一钱包；
5. 交易成本必须高于动态最小值（有关交易成本和定价的更多详细信息请参阅第3.2.3节）；
6. 交易锚必须是当前钱包列表的交易发起者的最后被处理的交易ID，或者最后50个块之一的独立哈希，钱包的第一笔交易的交易锚为空值；

4.2.2 区块

在操作过程中，节点会从数据分发网络中的其他对等节点，即其他矿工或专用对等节点（例如，块织物浏览器节点[27]），接收到区块。接收节点需要尽快验证并接受该块，以便跟上网络共识并继续有效地挖矿。即时区块散播是避免分叉的一部分，节点必须确保其接受的块也被其对等节点接受。

至关重要的是，为了使向其他对等节点散播区块之前的区块初步验证速度快（增加区块共识时间，也就增加了分叉风险。参见第2.3节以了解区块影的详细信息），区块影中的工作证明可以独立于其他块进行验证。这避免了从区块影构造出完整区块并校验BSD的必要性，否则这会成为一个拒绝服务攻击漏洞。

在将接收到的块散播给对等方之前，必须进行以下步骤的初步验证：

1. 确保区块影结构格式正确（请参见附录10.2.3节）；
2. 确认该块是否已被处理（通过检查其BDS）；
3. 确保该块中引用的每个交易的完整副本都存在于本地交易池中；

4. 验证传入块的BDS和Nonce组合是否满足网络当前的难度要求；
5. 验证传入块的时间戳是在前一个块的可接受边界内（时间只允许增加）；

如果初步验证通过，从区块影生成出完整区块，并且将区块影在核心验证步骤之前进行散播。

接下来，在该块被本地节点接受之前，必须先执行核心验证步骤：

- 从该区块生成BDS，与区块影提供的BDS进行校验；
- 给定新块交易和时间戳，验证新块中包含的块织物元数据（例如钱包列表、块哈希列表、织物尺寸等）是上一个块提供的元数据的合法更新。

4.3 接收和响应请求

节点在挖矿的各个阶段都需要向对等方请求信息，包括第一次加入网络与对等方同步，以及需要访问回忆块以验证新块时。为了高效地分配稀缺的出口带宽，节点根据对等方社交行为（例如它们响应请求的速度）对其进行排名。通过AIIA元博弈对节点进行激励，以迅速准确地响应信息请求，从而从网络对等节点取得好的排名。有关AIIA元博弈激励设计的完整详细信息，请参见第6章。

除挖矿节点外，网络的其他用户也发送信息请求，包括但不限于对交易数据的请求。这些请求不一定会影响节点的AIIA评分，但是在没有特殊努力的前提下，无法把它们与对等节点的请求区分开来。有关阿维 HTTP API如何服务这些信息请求的详细信息，请参见第7.1.3节。

4.4 分叉避免和分叉恢复

传输延迟是区块链分叉的主要原因，因此迅速传播是避免分叉的重要策略。在阿维协议中，AIIA元博弈代理鼓励即时传播，例如通过野火机制（请参阅第6章）。

当节点收到比其已知的前一块还要晚两个或更多的新候选块时，意味着网络中存在分叉。因此，该节点必须立即采取行动，通过评估分叉的积累难度，确认这些明显的分叉中哪一个已经被最多的对等方接受。每个块数据结构中提供的“累积难度”，是封装在块织物分叉中的访问证明工作量的代理表示。在分叉中，优先选择产生了较高累积难度的路径。

当收到有更高累积难度的新块，节点将把新块作为首选分支，并通过分叉的区块哈希列表追溯到与其自身历史的分叉点。一旦分叉点这个共同块被找到，便会评估验证分叉所需的交易和区块。如果发现分叉中从分叉点到接收块的全部块都是合法的，并且当验证结束时它仍然是具有最大累积难度的分叉，则节点的上下文将切换到此分叉。

5 去中心化内容策略

鉴于矿工集体维护阿维网络，需要一种机制让他们对系统中应该包含哪些内容，不应该包含哪些内容发表意见。这是通过三个相关但不同的机制来实现的：

- 对内容进入块织物中进行民主投票的过程；
- 每个节点选择在其计算机上存储什么内容的能力；
- 网关节点过滤用户所访问的块织物数据的能力；

节点通过内容策略表达对内容的偏好。内容策略是可以在交易上执行的任意计算，以决定交易是否为本地节点所接受。在阿维的参考实现中，支持子字符串匹配和交易数据哈希匹配两种形式的内容策略。其他协议实现可以利用它们自己的内容策略机制，例如计算机视觉技术和模糊哈希匹配（比如PhotoDNA[46]）。

5.1 投票阶段

当交易被分发到网络，每个节点根据自己的内容策略对其进行扫描。如果交易包含内容策略禁止的内容，则该交易不会被接受到该节点的交易池中，也不会散播给其他对等方。

如果从对等方收到的新区块包含对已丢弃交易的引用，则该区块也不被验证直接丢弃。如果网络中的其他节点未丢弃有问题的交易，则会出现分叉。在分叉过程中，那些拒绝交易的节点与接受交易的节点“竞争”产生下一个区块。一旦下一个区块产生，另一个派的节点就会启动分叉恢复过程，以下载区块及其包含的交易，进行验证，然后从临时内存中删除有问题的交易。通过这种方式，网络能够保持共识，同时允许节点参加投票过程，表达他们是否希望特定内容加入到块织物。因此，希望拒绝所述内容的节点可以不把它们存储在硬盘上。

5.2 存储阶段

在节点投票接受数据进入网络之后，块织物的访问证明机制中内嵌了数据存储的激励。当新的参与者加入网络，他们会扫描并下载符合其内容策略的交易，回避他们不希望存储的交易。

当节点操作员更改内容策略，他们可以重新扫描本地存储，删除违反新策略的内容。

5.3 激励

去中心化内容策略机制产生了两个互补的激励：

- 激励不要过度狂热地拒绝太多交易，这会减少挖矿奖励；
- 激励拒绝被网络大多数节点拒绝的交易，因为包含这些交易的候选区块会被网络其余部分忽略。

这组激励措施迫使矿工在过度宽松和过度拒绝之间取得平衡，过度宽松会对网络造成长期危害，过度拒绝会导致网络效用降低。

5.4 折中

去中心化内容策略机制的后果之一是，交易的结算时间增加了一个区块周期。因此，交易被打包进区块，并不意味着该交易将被整个网络接受。但是，考虑到网络使用的是中本聪风格的最终一致共识[47]，这个额外的区块确认期不会显著地改变系统的用户体验。

如果内容策略导致当前回忆块的副本数太少，节点还可以选择丢弃网络的最后区块并重新开采它，以重新决定回忆块。重新开采区块的超时时间并不在协议级别强制决定。然而，越多算力被应用开采先前的块，头块被重采的可能性就越高。不鼓励节点过早地重新开采头块，因为这将要求他们不必要地花费算力，却没有候选块被其他节点接受的把握。

5.5 网关

就像去中心化内容策略机制影响内容如何添加到网络以及在网络中存储一样，网关也将这种机制应用于索引已经存储在网络中的数据。网关仅索引遵循其自己的内容策略的内容。随后，当用户查看例如阿维上的去中心化社交媒体应用时，他们选择的网关根据内容策略决定显示哪些内容。

因此从根本上讲，用户可以通过选择不同内容策略的网关，决定他们看到什么类型的网络内容。

此外，这些机制解决了应用开发者从其平台中删除或审查内容的问题。尽管用户可以选择自己看到的内容，但他们不能强行禁止其他人查看块织物上的内容，因为每个人都可以平等选择网关（对应内容策略）来查看块织物。

我们希望由志趣相投者构成的社区将根据其内容策略环绕在网关周围，团结起来维护和推进他们希望看到Web。当永在网达到适当的采用水平，我们预计这些去中心化的网关将代表不同的观念，从而使每个人根据他们自己如何看待Web而有广泛的选择。

6 阿维的自适应机制设计

6.1 引言

使中心化Web获得长期成功的力量之一是其协议的简单性和适应性[65]。例如，初始版本规范[25]中对超文本传输协议（HTTP）的描述纯粹是用于传输“知识网”。但是实际上，HTTP已经驱动了互联网上几乎所有以名称寻址的内容。这样的结果是协议的灵活性使然。鉴于阿维的目标是建立长期的基于经济激励的可靠数据存储机制， 灵活性和适应性均内建于其机制设计之中。

阿维经济机制的灵活组成是围绕“自适应互动激励代理”（AIIA）博弈构建的。 AIIA博弈是一种自适应机制设计[8]，可用于排名、展示和在随时间变化的网络环境中对亲社会行为进行奖励。 AIIA博弈中的每个玩家都制定了一种策略，用于对网络中与之交互的其他玩家进行优先排名。策略由代理执行，代理之间进行交互，根据其他代理提供的效用对它们进行排名，并依据排名分配玩家的稀缺资源以奖励它们。相互奖励亲社会行为的模式可以看作是BitTorrent创造的乐观针锋相对机制的泛化，从带宽共享泛化为一般性的有用行为[18]。

当代理互动并相互排名，以提供非通证社交奖励（例如数据分发或ArQL查询的优先级），他们自身对这些行为的反应也会社交化地被其他代理排名。即玩家彼此排名，以及他们自己的行为也被其他玩家排名。结果是在典型的针锋相对行为之上，创造出创建二阶“元博弈”。当代理们各自采用自己的特定策略互动时，他们就有动力激励彼此的行为朝着亲社会目标前进。

6.2 与基于共识的博弈相比较

与典型的区块链机制设计[47，67]不同，AIIA博弈玩家的确切行为不是通过全网共识强制实施的。这使得网络中的每个玩家本质上都可以参与元博弈之下的不同“游戏”（即他们对其他玩家及其代理的排名机制）。随后，元博弈表达的总体规则由每个正在进行单独游戏的总和来定义。这与阿维网络（和其他区块链协议）在共识层面的博弈形成鲜明对比。在典型的区块链博弈机制（例如，比特币的工作量证明）中，所有参与者都被要求按相同规则参与同一个游戏。随着时间推移，AIIA博弈中代理行为会随着它们所处网络的技术和社会环境变化，以反应每个玩家（即网络参与者）选择的激励变体。

AIIA博弈类似于BitTorrent网络[18]显示出的激励特征，它不会用程序强制每个网络参与者以相同的方式对网络中的其他参与者进行排名。相反，它允许每个参与者为其他对等方维护私有的本地排名。2007年，出现了一个经过修改的BitTorrent客户端——BitTyrant，该客户端专门利用大多数BitTorrent网络参与者使用的原始排名机制的弱点[54，55]。为了应对BitTyrant[26]，对标准的乐观针锋相对策略代理进行了改进，生态系统参与者可以自行选择使用它来避免由BitTyrant节点引起的问题。通过这种方式，成为类似于AIIA元博弈机制设计的BitTorrent数据分发机制已运转超过12年。这使BitTorrent网络能够适应新的挑战（例如BitTyrant的出现），而无需进行中心化的强制性协议升级。 阿维的AIIA元博弈有意地采用此类策略，为的是在网络的环境不断变化的情况下（例如网络使用模式、互联网架构、漏洞和激励机制出现问题），激励一般性亲社会行为。

6.3 实现概述和涌现属性

AIIA游戏的核心是鼓励网络参与者建立如此行事的代理：

1. 合理分配自身资源，以从其他代理获得最高的净效用；
2. 作为对第一点的折中，向节点运营者提供小额奖励令他们在当前情况下，对他们认为以亲社会方式运作的的节点表现出一定偏见。

一旦节点运营者对亲社会方式运行的其他节点表现出一定程度的偏见（即使它本身不以亲社会方式运行），经过多轮次博弈后，大体一致的激励行为将自行出现（详细信息参见后续仿真）。

例如，所有矿工都与阿维通证价格利益相关。由于价格一定程度上是需求的函数，引发更多通证需求的特性对网络中的所有矿工都是积极的。其中一种可能的增加需求的特性是阿维-IPFS桥。 阿维-IPFS桥需要一小部分阿维节点平行运行IPFS节点，从而向IPFS客户端暴露阿维数据。但是，每个实现阿维-IPFS集成的矿工都没有因为运行阿维-IPFS桥直接获得激励，尽管事实上这显然是亲社会行为。AIIA博弈的参与者，即使他们不亲自运行阿维-IPFS桥，能够通过为其他运行阿维-IPFS桥的节点提供边际奖励，从而对其他节点的行为施以“轻推”。表示此激励的将是AIIA节点运营商损失很少一点效用，来边际化地降低一点未运行桥接的节点的优先级。但是，运营商有动机（在某种程度上）这样做，因为亲社会行为将从长远来看，通过增加通证需求使他们受益。

形式上，当且仅当如下情况，节点被鼓励向其他节点的亲社会行为提供激励：

$$\sum_{i=0}^{\infty} Utility_{incentivised}(i) > \sum_{i=0}^{\infty} (Utility_{before}(i) - Cost_{incentivise})$$

（公式15）

每个节点仅模糊地了解它们在其他节点的AIIA排名中表现如何，因为随着时间的流逝，获得更高AIIA排名的奖励感知将随着网络游戏的发展而改变。于是理性的代理设计者不跟踪其他代理给他们的评分，而是衡量从其他代理获得的主观效用。通过尝试表达不同的行为（例如运行IPFS桥接节点、或者不运行它们、优先安排一个小组进行数据分发，或将数据分发给请求它的每个人等等），节点可以在没有完全感知全部规则的情况下优化他们针对 AIIA元博弈当前状态的行为。此外，在实践中，预计AIIA元博弈的代理设计者将在“链下”相互交流，以讨论激励策略和行为、交换代理代码等。通过这种方式，可以预期随着时间的推移，通过以无需可的方式由多方不断增加修改的和改进的交互代理，将形成一个生态系统。

6.4 仿真

为了演示AIIA元博弈随时间变化的特性，进行了仿真。在仿真中，每个参与者的内部代理的特征由一个浮点数向量表示，浮点数的取值是从0到1，并且在具有相同长度特征的偏向于偏好行为的向量中：

$$A_{Cs} = \{0, ..., n\}$$

（公式16）

$$A_{Bs} = \{0, ..., n\}|$$

（公式17）

每个代理表达的偏见是代理设计者对亲社会行为的认可，而特性则代表代理实际表现出的行为。在每个博弈回合中，每个节点计算相对于其他节点的净适应评分：

$$F_{net}(A, As) = \sum_{i=0}^{|As|} F(A, As_{[i]})$$

（公式18）

每个节点还执行随机变异并计算其新的效用得分，然后采用首选策略进入下一个博弈回合：

$$A_{next} = \begin{cases} M & \text{when } F_{net}(M, As) > F_{net}(A, As) \\ A & \text{otherwise} \end{cases}$$

where

$$M = Mutate(A)$$

（公式19）

节点根据混合了偏见的自己的行为与其他节点的行为之间的距离做相互排名。

$$F(A, B) = \sum_{i=0}^{|A_{Cs}|} diff(A_{Cs[i]}, mix(B_{Cs[i]}, B_{Bs[i]}))$$

（公式20）

这种行为模仿了AIIA博弈参与者通常是如何青睐那些用与自己类似的方式评价并分配资源的代理，遵照BitTorrent的乐观针锋相对数据分发机制表达的模式的抽象。

在执行时，仿真显示当同时使用随机和共享的A_{Cs}向量初始化时，经过足够长的时间，AIIA博弈的A_{Cs}倾向于在代理集合内形成共同的偏见。此外该仿真表明，只要亲社会策略的感知变化速率不超过每个偏见表达的平均速率，亲社会策略的趋同就总能实现。

6.5 局限性

尽管类似AIIA博弈的激励机制可用于构建机制对环境变化自适应的机制设计，但它们有三个基本局限：

1. 首先，从类似AIIA博弈涌现的自适应机制设计只能鼓励大多数网络参与者都认可的行为表达。这些机制本身不能在没有代理设计者的总量知道下计算亲社会行为模式。这与区块链网络的大多数节点（及其在Sybil抵抗中的相关权力）不诚实行事[59]所导致的攻击类同。

2. 第二个主要限制是类似AIIA的博弈不能用来改变全网络共识机制。只有链下声望评分才能服从于网络参与者个别采用的游戏；
3. 最后，如上文仿真部分所述，网络中偏见表达速率，通常对表达节点短期不利，限制了网络适应环境变化的速度。就是说，如果节点太自私不愿意表达亲社会偏见，他们可能就无法充分适应环境的变化。

迄今为止，阿维网络已经暴露给了两个AIIA代理：野火（在本文第3.4节中介绍），一种实现类似乐观针锋相对的带宽共享代理；和织工（Weaver）[27]，一种将消息随机转发到它在网络上遇到的前十个节点的代理。实际上，织工节点当前是少数，并且相比于其他节点，被占网络的多数的野火节点给予较低排名。我们期望随着阿维网络环境的进化，将对织工和野火（以及其他新发明的代理）进行修改，奖励那些诸如服务ArQL请求、最小化数据请求延迟以及帮助快速定位数据等行为。

7 阿维协议互操作性和永在网

本章我们描述永在网，一组建立在阿维永久知识账本上的通信协议，共同呈现一个永久性的、去中心化的Web。永在网在表面观感上与传统Web非常相似。但是，用户对永在网上的页面和应用的体验与传统Web在几个基本方面有所不同。

本节探讨了永在网和传统Web之间的一些差异，以及所涉及的协议如何协同工作以提供各种优势。

7.1 特性

本节将描述阿维永在网的关键特性，以及周边协议家族。

7.1.1 无信任且可证明的Web

阿维上存储的所有内容都带有时间戳，并且可以防止篡改。这是由块织物建构其上的基本区块链基础设施确保的。如经典的区块链论文[29，9]所述，区块时间戳不依赖于时间服务器，而是依赖于网络共识。时间戳的可靠性源于块织物的不变性，并且精度几分钟以内。将交易数据包含在区块哈希和回忆块中可确保数据在存储或检索期间保持纯粹，同时还可以确保所述交易数据随时在网络中保持可用。

钱包列表中的每个钱包都记录该钱包的最后一笔交易。这样可以确保每个钱包的完整历史记录始终可用。此外，由于阿维上的每条信息都由钱包签名，因此上传到永在网的数据始终与某种形式的身份（至少是假名）相关联。实际上，与其他内容寻址网络[12]不同，阿维中的每个网页在特定时间点与特定身份相关联。这意味着记录可以通过网络追溯到最初断言这些事实的身份。这是对中心化Web的重大范式转移，在旧范式中，可以断言、传播然后撤销事实，以达到传播难以确定来源的错误信息的目的[63、23、40、60]。

7.1.2 激励驱动的Web响应性

激励数据复制是阿维协议机制设计的基础部分。这种激励措施可确保网络整体具有抵御破坏或攻击的能力，并确保数据可靠地存储在网络中。这些激励措施还极大地提高了网络的响应能力，以及数据定位和检索的速度。首先，这是因为任何给定节点都有很高的可能性拥有请求所需的数据。因此，节点通常不必将请求转发到其他节点，而是可以自行满足所述请求，这一点与与基于“合同”的存储系统不同。

第3章详细介绍了阿维协议的机制设计。

7.1.3 开放HTTP API

为了运行网络，阿维节点使用阿维节点间协议相互通信，其参考实现是HTTP API，因为阿维网络节点用跟浏览器相同的格式相互分享数据，对台式机、移动设备和其他设备上的浏览器的网络内部数据的可访问性由协议本身保证。 因此，第三方应用程序和浏览器是网络中的“头等公民”，因为它们全都使用相同的、久已确立的基本协议进行通信。

7.2 应用程序架构

7.2.1 客户端-服务器

传统的Web或原生应用具有客户端-服务器架构。这种模式仍然可以兼容阿维，因为Web服务器可以作为网络永久账本的数据存储前端。这种“阿维使能的服务器”使用阿维HTTP API与一个或多个阿维节点进行交互，代表客户端读取和写入数据。这些服务可以是将用户作为游客的网站，也可以是原生应用将用户请求传递给开发者运行的服务器。在这种中心化的阿维App模型中，这些服务可以维护一个AR通证池，以便代表他们的客户来支付数据存储费用。使用这种应用结构仍然可以免费从块织物读取数据。

这种架构的获利潜力类似于中心化Web。开发者将需要通过广告、每月订阅或直接付款，从应用程序获得更多的价值，而不是他们用来为其存储提供动力的AR通证数量。永久不变存储有很多用例。

7.2.2 无服务器Web应用架构

去中心化应用可以直接驻留在块织物上，从其上直接运行，并且可以由普通Web浏览器访问。这是因为整个应用程序本身都作为交易数据存储在阿维网络上，当提供给浏览器或其他客户端后，就可以执行。由于块织物在上传数据和上传者（可以是假名）身份之间建立了不可变的链接，因此可以持续证明任何阿维应用程序的出处。

最流行的Web技术，包括HTML、JavaScript和CSS，通常作为阿维去中心应用的基础。如果用于访问交易数据的客户端包含其他语言的解释器/解析器，这些技术实现的应用程序也可以从块织物下载，并在客户端中执行，例如作为桌面应用程序。

阿维网络上托管的应用程序还能够将持久化且可证明的状态写入块织物。由于阿维并未强制特定的数据结构，因此开发人员可以自由地使用对他们来说最有意义的格式存储数据（交易数据本身和交易标签字段都可以保存任意键值列表）。如果高度优化的梅克尔结构能最好地满足应用程序需要，就像以太坊虚拟机那样，可以在阿维网络上轻松实现。

托管在阿维网络上的无服务器应用程序使用户可以直接为网络交互付费。这使开发人员不必补贴用户交互成本。

7.3 网关节点

阿维网关节点与任何其他节点相似，但集成了下列功能用于访问或查询网络：

- ArQL——一种用于查询块织物元数据的查询语言和索引，重点是交易、交易标签和交易链接的时间戳（更多详细信息，请参见下文）；
- 阿维DNS和TLS——使用相关钱包和交易ID作为域名访问存储在块织物上的应用的方法（更多详细信息，请参见下文）。此功能与现有的DNS和TLS系统兼容。

7.3.1 ArQL

ArQL是阿维协议家族中的一种可选协议，它是一种简单的查询语言，用于对阿维网络中存储的数据做基本的搜索。 ArQL旨在作为一种轻量级的机制，来帮助应用程序开发者在没有更高级的去中心化数据库的情况下，构建简单的永在网应用。ArQL通过索引用户和/或开发者定义的与交易相关联的标签来做到这一点。尽管ArQL索引通常与本地持有的区块和交易保持一致，它们显然不一定全网一致，因为并非所有节点都拥有所有网络数据的完全副本。这给去中心化数据库带来了三个不常见的特性：

1. 运行ArQL的节点返回的查询结果要服从该节点的内容策略。有关内容策略机制更多详细信息请参见第5章；
2. 由于查询仅在单个节点内执行，因此可以依靠它们及时提供结果。这是因为没有查询分片，也没有从其他节点整合结果的延迟等等；
3. 无法确定查询已返回所有可能的结果，因为查询的结果仅表节点所知的和愿意透露详情的网络交易。

第3点可能会在某些情况下限制ArQL的用途。但是，第1点对应用开发者和用户的影响更深远。

7.3.2 阿维DNS和TLS

阿维网关节点可以利用标准DNS基础设施来改进作为Web应用的阿维交易的显示和运转。例如，域名所有者只需将交易存储在阿维网络，并在普通外部服务提供商注册DNS记录，可以运行永久可用的去中心Web应用。

1. DNS

需要两条DNS记录才能将域名关联到网关节点上的阿维交易。例如，www.mycustomsite.com将需要以下记录才能将其指向www.arweave-gateway.net：

一条指向阿维网关的DNS CNAME记录：
www CNAME arweave-gateway.net

一条DNS TXT记录关联域名和特定的交易ID：
arweavetx TXT kTv4OkVtmc0NAsqIcnHfudKjykJeQ83qXXrxf8hrh0S

当浏览器请求www.mycustomsite.com时，用户的计算机将通过常规DNS进程将其解析到网关节点arweave-gateway.net的IP地址。当网关接收到具有非默认主机名的HTTP请求，例如www.mycustomsite.com而不是www.arweave-gateway.net，网关将查询www.mycustomsite.com的DNS记录，而`arweavetx` TXT记录将告知节点要提供的交易。

2. TLS

大量浏览器在未经TLS访问网页时不允许使用本机加密功能。这意味着阿维应用如果不集成TLS支持，就不能使用哈希函数、签名或验证功能。因此不支持TLS会限制阿维永在网应用的实用性，同时也使这些应用程序容易受到中间人（MITM）攻击。鉴于交易已签名，直接MITM攻击很困难，但是由于缺乏加密，也暴露出一类新颖的攻击向量。一个例子包括拦截和操纵`/ price`端点的可能性，可能被利用来返回不可接受的低价格，从而导致用户钱包缴纳过高的交易费用。

网关运营者为网关客户端生成并维护TLS证书。例如，使用Let's Encrypt自动证书管理环境 [7]，但是也可以使用其他TLS系统。在初始网关设置里，可以请求并生成通配的网关域名证书。这允许通过HTTPS / TLS使用网关顶级域名访问网关，还可以使用单级子域名访问网关（例如gateway.com和subdomain.gateway.com，但不允许sub.subdomain.gateway.com），反过来允许使用浏览器沙箱。

当浏览器从网关请求交易，例如https://gateway.com/6BdL...6VzZ，网关将返回301重定向到https://label.gateway.com/6BdL...6VzZ，其中label是从交易ID派生的Base32伪唯一地址。由于现在是从子域提供数据，因此浏览器同源策略被调用，网页被限制在沙箱，从而为其提供安全浏览器上下文。

对DNS和TLS的支持确保与这些技术"向后兼容"。除这些传统系统，阿维还可以与去中心化域名系统集成。

7.4 用例

阿维协议提供了真正可靠的、不变的和去中心化的数据存储，从而为广泛的潜在用例提供重要功能。

在解决现实用例时，阿维协议具有一些独特优点：

- 提供可靠的记录档案。一旦将数据添加到块织物中，就不会被删除或修改，无论有意还是无意；
- 真实性。块织物为特定数据在特定时点提供“存在证明”，基于相关交易的可验证和可靠时间戳；
- 出处。每笔交易都永久链接到相同钱包的前一笔交易，这意味着最终用户可以根据钱包地验证任何交易中数据的真实来源，包括阿维网络上托管的去中心化应用。
- 去中心化应用托管。与任何中心化应用托管平台相比，区织物更能确保可靠的应用访问，中心化平台通常会对应用完整性有负面影响[5]；
- 激励性数据存储和服务。 阿维独特的机制设计有力地鼓励向网络中的所有参与者快速提供数据，包括最终用户（更多详细信息，请参阅第3.4节）。

7.5 示例应用

如今，在阿维网络上运行着各种各样真实的去中心化应用，每个都得益于前文详述的阿维网络独特优势。在此，我们将简要介绍其中一小部分去中心化应用：

- ArBoard [61]。去中心化讨论板，用户可以在子类别中创建讨论主题，其他用户可以对这些话题进行回复，对热门话题投票，以及查看任何帖子的编辑历史记录。所有这些功能均由ArQL提供支持，而无需外部依赖或计算；
- AskWeave [44]。由阿维社区创建的托管在块织物上的应用程序，具有自由格式的问答式论坛功能。用户可以提出和回答问题，并直接在应用内对自己喜欢的回复使用AR支付小费。
- Weavemail [66]。托管在阿维网络的去中心化、健壮、永久的电子邮件客户端。除了在块织物上发送不可变的电子邮件外，用户还可以将其用为向其他Weavemail用户安全地发送和接收AR通证的方法。
- 阿维ID [43]。另一个阿维社区创建的应用，它使用户可以声明一个关联到其阿维钱包地址的唯一用户名。最有趣的是，许多其他社区开发者已将阿维ID集成到他们自己的永在网应用，包括前面谈到的AskWeave，这意味着用户可以利用“非孤立”的永在网数据（也就是说，块织物可以充当任何永在网应用的单例通用数据源）。在这种情况下，这意味着用户可以在所有永在网应用中使用一个通用用户名，而不必在每个单独的平台上注册单独的账号。

8 未来工作

8.1 简洁访问证明

当前阿维架构所面临的主要可扩展性挑战来自于，在将数据写入网络的过程中，必须将全部交易数据推送给全部节点。尽管当前架构的系统每年允许大约3.1亿个典型大小的数据存入（223,280个区块），在这一领域进行实质性的改进是可能的。 阿维核心开发团队计划为社区提供可选数据访问证明机制：简洁访问证明。

通过简洁访问证明机制保护的交易不直接保存数据，而仅需将梅克尔树根存入块织物。这需要引入多个权衡。尽管它允许参与节点不接收数据的完整拷贝也能就数据输入块织物达成共识（例如，节点没有多余的容量来存储所述数据），它也移除了传统块织物数据结构能提供的分发保证证明。这允许交易包含大量数据，但是丧失了所述数据曾经传播到整个网络的确定性。

对交易数据的简洁访问证明可以有外部节点在区块确认期间以很高的可信度提供，只需传输数据集的梅克尔树的一系列梅克尔路径。这样做的优点是可以将大量数据快速添加到网络中，但不能提供与传统访问证明相同的数据可用性保证。因此，简洁访问证明将作为可选功能，提供给那些要求长期高冗余存储大数据集的用户，他们不需要确保数据曾经完全公开可用。

8.2 钱包日志

有计划对阿维网络中的钱包列表系统做进一步改进。钱包列表（一种维护帐户、余额和防止重方攻击的机制）目前占阿维网络数据存储开销的很大一部分。曾经使用过的所有钱包出现在块织物的每个区块的钱包列表中。计划将这些钱包名单替换为只可追加的钱包日志。通过在每次余额变动时将新条目添加到钱包存储数据结构中，存储复杂性将从O（区块量*钱包量），降低到O（交易+区块量）。此外，钱包日志机制不会实质性地影响钱包数据的查找时间，因为日志结构可以每隔一定数量的区块就做一次有效地“压缩”。这种压缩机制使结构的尺寸相对于钱包数量可管理。此外，该机制还允许对新区块内容进行O1复杂度的验证。

8.3 快速查找

由于阿维网络激励所有网络参与者存储尽可能多的块织物，因此与典型的基于分布式哈希表存储相比，快速数据定位的问题不那么突出。通过确保网络中每条数据都有大量副本（撰写本文时大约为750），鉴于阿维网络中的数据副本如此充裕，查找数据已从“在大海捞针”变为“在谷仓里寻找干草”。尽管如此，到了某个时间点，仍然需要优化数据定位时间。我们希望可能通过改进的的AIIA代理来实现，即不仅奖励节点对本地存储的数据提供快速访问，还奖励把参与者重定向到能够提供数据的节点。

像典型的AIIA元博弈一样，网络中的每个节点都会被鼓励去激励那些表现出亲社会行为的其他节点。每个独立AIIA代理的实现都将精确地定义参与者如何维护路由表，用于将参与者的请求重定向到哪些可能拥有数据的节点。尽管如此，预计这些代理将最终形成类似于下一跳地址的路由层，就像现在的互联网一样。

9 结论

总而言之，我们介绍了阿维协议家族：一个模块化的用于实现知识和历史永久复制和回忆的系统，由可持续经济机制支持。除了促进信息永久存储的基金，阿维还通过强韧的链下激励机制来促进信息的分发。

在主要的阿维协议层之上，我们还展示了“永在网”的轮廓：一个永久的、去中心化的、有恢复能力的万维网。不同于传统的Web，永在网中的所有内容都是不可变的，带有时间戳并经过密码签名（确保强作者属性）。

永在网目前处于婴儿阶段。它已经运行了500天，并且每天以大约2500个Web应用程序和页面的速度增长。除了展示当前的协议实现，我们还概述了潜在的未来工作领域，以及确保这些进展的自适应机制设计。

引文

[1] Mustafa Akgul and Melih Kirlidog. Internet censorship in turkey. Internet Policy Review, 4(2):1-22, 2015.

[2] Eitan Altman, Rachid El-Azouzi, Yezekael Hayel, and Hamidou Tembine. The evolution of transport protocols: An evolutionary game perspective. Computer Networks, 53(10):1751-1759, 2009.

[3] Anderson, Moore, Nagaraja, and Ozment. Incentives and information security. In Nisan et al. [50].

[4] Farhad Anklesaria, Mark McCahill, Paul Lindner, David Johnson,Daniel Torrey, and B Albert. The internet gopher protocol (a distributed document search and retrieval protocol). University of Minnesota Microcomputer and Workstation Networks Center, Spring, 1993.

[5] Arweave. Avoiding consumer lock-ins with the decentralised web. https://hackernoon.com/avoiding-consumer-lock-in-with-the-decentralised-web-c618f28241ab,2019.

[6] Fernando Baez and Alfred J MacAdam. A universal history of the destruction of books: From ancient Sumer to modern Iraq. Atlas New York, 2008.

[7] R. Barnes, J. Homan-Andrews, D. McCarney, and J. Kasten. Automatic certificate management environment (acme). https://ietf-wg-acme.github.io/acme/draft-ietf-acme-acme.html,2019.

[8] Tobias Baumann, Thore Graepel, and John Shawe-Taylor. Adaptive mechanism design: Learning to promote cooperation. arXiv preprint arXiv:1806.04067, 2018.

[9] D. Bayer, S. Haber, and WS Stornetta. Improving the efficiency and reliability of digital time-stamping. In R. Capocelli, A. De Santis, and U. Vaccaro, editors, Sequences II, pages 329-334. Springer, 1992.

[10] Barry L Bayus. An analysis of product lifetimes in a technologically dynamic industry. Management Science, 44(6):763-775, 1998.

[11] Jacob D Bekenstein. Universal upper bound on the entropy-to-energy ratio for bounded systems. Physical Review D, 23(2):287, 1981.

[12] J Benet. Ipfs - content addressed, versioned, p2p le system (draft 3). https://ipfs.io/ipfs/QmV9tSDx9UiPeWExXEeH6aoDvmihvx6jD5eLb4jbTaKGps.

[13] A Berman. Bitmex research finds potential bug in syncing of ethereum parity full node. https://cointelegraph.com/news/bitmex-research-finds-potential-bug-in-syncing-of-ethereum-parity-full-node, 2019.

[14] Bitcoin genesis block. https://www.blockchain.com/btc/tx/4a5e1e4baab89f3a32518a88c31bc87f618f76673e2cc77ab2127b7afdeda33b,2009.

[15] Rene Carmona and Francois Delarue. Probabilistic Theory of Mean Field Games with Applications I-II. Springer, 2018.

[16] Anton-Hermann Chroust. Socrates{a source problem. The New Scholasticism, 19(1):48-72, 1945.

[17] D Clark. Intel rechisels the tablet on moores law. https://blogs.wsj.com/digits/2015/07/16/intel-rechisels-the-tablet-on-moores-law/, 2015.

[18] Bram Cohen. Incentives build robustness in bittorrent. In Workshop on Economics of Peer-to-Peer systems, volume 6, pages 68{72, 2003. [19] Matt Corallo. Compact block relay. bip 152, 2017.

[20] C. Decker and R. Wattnehofer. Information propagation in the bitcoin network. In IEEE P2P 2013 Proceedings, 2013.

[21] Robert P Dellavalle, Eric J Hester, Lauren F Heilig, Amanda L Drake, Je W Kuntzman, Marla Graber, and Lisa M Schilling. Going, going, gone: Lost internet references, 2003.

[22] Bianca M Federico. The patent oce re of 1836. J. Pat. O. Soc'y, 19:804, 1937.

[23] Steve Forbes. Web of deception: Misinformation on the Internet. Information Today, Inc., 2002.

[24] Anne Frank and Storm Jameson. Anne Frank's diary. Vallentine, Mitchell, 1958.

[25] Henrik Frystyk, Tim Berners-Lee, and Roy T Fielding. Hypertext transfer protocol - http/1.0. RFC 1945, RFC Editor, 1996.

[26] Pawel Garbacki, Dick HJ Epema, and Maarten Van Steen. An amortized tit-for-tat protocol for exchanging bandwidth instead of content in p2p networks. In First International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2007), pages 119{128. IEEE, 2007.

[27] GoldZeus. Weaver. https://github.com/GoldZeus/weaver, 2019.

[28] Olivier Gueant, Jean-Michel Lasry, and Pierre-Louis Lions. Mean eld games and applications. In Paris-Princeton lectures on mathematical finance 2010, pages 205{266. Springer, 2011.

[29] S. Haber and WS Stornetta. How to time-stamp a digital document. Journal of Cryptology, 3:99-111, 1991.

[30] Zhu Han, Dusit Niyato, Walid Saad, Tamar Baar, and Are Hjrungnes. Game Theory in Wireless and Communication Networks: Theory, Models, and Applications. Cambridge University Press, 2011.

[31] Adam Hayes. A cost of production model for bitcoin. Available at SSRN 2580904, 2015.

[32] Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. Eclipse attacks on bitcoins peer-to-peer network. In 24th {USENIX} Security Symposium ({USENIX} Security 15), pages 129{144, 2015.

[33] Lambertus Hesselink, Sergei S Orlov, and Matthew C Bashaw. Holographic data storage systems. Proceedings of the IEEE, 92(8):1231 - 1280, 2004.

[34] D Johnston. North's ex-secretary tells of altering memos. https://www.nytimes.com/1989/03/23/us/north-s-ex-secretary-tells-of-altering-memos.html, 1989.

[35] B Kahle. Fire update: Lost many cameras, 20 boxes. no one hurt. https://blog.archive.org/2013/11/06/scanning-center-fire-please-help-rebuild/, 2013.

[36] N Kuznetsov. As russian censorship increases, is a decentralized web the answer? https://thenextweb.com/podium/2019/05/30/as-russian-censorship-increases-is-a-decentralized-web-the-answer/, 2019.

[37] A Lafrance. The internet's dark ages. https://www.theatlantic.com/technology/archive/2015/10/raiders-of-the-lost-web/409210/, 2015.

[38] J Langston. Digital collections. https://www.loc.gov/, 2019.

[39] J Langston. With a hello, microsoft and uw demonstrate first fully automated dna data storage. https://news.microsoft.com/innovation-stories/hello-data-dna-storage/, 2019.

[40] Sangho Lee and Jong Kim. Early ltering of ephemeral malicious accounts on twitter. Computer Communications, 54:48-57, 2014.

[41] Birmingham Public Libraries. Notes on the History of the Birmingham Public Libraries: 1861-1961. Birmingham Public Libraries, 1962.

[42] Ziyao Liu, Nguyen Cong Luong, Wenbo Wang, Dusit Niyato, Ping Wang, Ying-Chang Liang, and Dong In Kim. A survey on applications of game theory in blockchain. arXiv preprint arXiv:1902.10865,2019.

[43] Lyner. Arweaveid. https://arweave.net/fGUdNmXFmflBMGI2f9vD7KzsrAc1s1USQgQLgAVT0W0, 2019.

[44] Lyner. Askweave. https://arweave.net/HhIjOjxgHYXJU5RVjRYfAR017vbZdujbCSlaA8NQ20U, 2019.

[45] Yuval Marcus, Ethan Heilman, and Sharon Goldberg. Low-resource eclipse attacks on ethereum's peer-to-peer network. IACR Cryptology ePrint Archive, 2018:236, 2018.

[46] Microsoft. Photodna. https://www.microsoft.com/en-us/photodna, 2019.

[47] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. http://www.bitcoin.org/bitcoin.pdf, 2008.

[48] The National Archives. Investigation into forged documents discovered amongst authentic public records: Documents purporting to have been created by members of the british government and members of the brit- ish armed services relating to leading nazis gures and axis power governments. https://discovery.nationalarchives.gov.uk/details/r/C16525, 2007.

[49] Mehdi Nikkhah, Aman Mangal, Constantine Dovrolis, and Roch Guerin. A statistical exploration of protocol adoption. IEEE/ACM Transactions on Networking, 25(5):2858{2871, 2017.

[50] Nisan, Roughgarden, Tardos, and Vazirani, editors. Algorithmic game theory. Cambridge University Press, 2007.

[51] G Orwell. Nineteen Eighty-Four. Secker & Warburg, 1949.

[52] A Pinar Ozisik, Gavin Andresen, George Bissias, Amir Houmansadr, and Brian Levine. Graphene: A new protocol for block propagation using set reconciliation. In Data Privacy Management, Cryptocurrencies and Blockchain Technology, pages 420{428. Springer, 2017.

[53] Steve Phelps, Peter McBurney, and Simon Parsons. Evolutionary mechanism design: a review. Autonomous agents and multi-agent systems, 21(2):237-264, 2010.

[54] Michael Piatek, Tomas Isdal, Thomas Anderson, Arvind Krishnamurthy, and Arun Venkataramani. Do incentives build robustness in bittorrent. In Proc. of NSDI, volume 7, 2007.

[55] Michael Piatek, Tomas Isdal, Tom Anderson, Arvind Krishnamurthy, and Arun Venkataramani. Building bit-tyrant, a (more) strategic bittorrent client. login, 32(4):8{13, 2007.

[56] J; Rydning J Reinsel, D; Gantz. Data age 2025: The digitization of the world from edge to core. https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf, 2018.

[57] Jonathan Rose. The holocaust and the book: destruction and preservation. Univ of Massachusetts Press, 2008.

[58] T. Roughgarden. Twenty lectures on algorithmic game theory. Cambridge University Press, 2016.

[59] Muhammad Saad, Jerey Spaulding, Laurent Njilla, Charles Kamhoua, Sachin Shetty, DaeHun Nyang, and Aziz Mohaisen. Exploring the attack surface of blockchain: A systematic overview. arXiv preprint arXiv:1904.03487, 2019.

[60] M Sa. Whatsapp 'deleting 2m accounts a month' to stop fake news. https://www.theguardian.com/technology/2019/feb/06/whatsapp-deleting-two-million-accounts-per-month-to-stop-fake-news,2019.

[61] sergejmueller. Arboard. https://bkxqaor2dlei.arweave.net/pvmiu4SZKQGWAYjrLWzE_mI70u1-v8zIzQ8WaxIYURk, 2019.

[62] A Souppouris. The internet archive is now home to 10 petabytes of data. https://www.theverge.com/2012/10/27/3563082/internet-archive-total-data-2012-ten-petabytes, 2012.

[63] Margaret Sullivan. Were changes to sanders article stealth editing. The New York Times, 2016.

[64] tevador. Randomx. https://github.com/tevador/RandomX, 2019.

[65] Dave Thaler and B Aboba. Rfc 5218: What makes for a successful protocol?, 2008.

[66] S Williams. Weavemail. https:/arweave.net/oEhzHOE2o9uZbi6O9cQatzhiHtc2EdFBGCQdqHsK-o4, 2019.

[67] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. Ethereum project yellow paper, 151:1-32, 2014.

[68] Xueyang Xu, Z Morley Mao, and J Alex Halderman. Internet censorship in china: Where does the filtering occur? In International Conference on Passive and Active Network Measurement, pages 133{142. Springer, 2011.

[69] Jingyu Zhang, Mindaugas Gecevicius, Martynas Beresna, and Peter G Kazansky. 5d data storage by ultrafast laser nanostructuring in glass. In CLEO: Science and Innovations, pages CTh5D{9. Optical Society of America, 2013.

10 附录

10.1 硬盘容量和MTBF历史数据

请参见：https://arweave.net/wufZ10dlzwfPFTNKr3uRAyeMRfMdkNx1iG9yjolRbv8

10.2 数据结构

本节将介绍一系列数据结构及其内容。

10.2.1 块数据结构

- **交易**：零到多个交易的列表（参见10.2.4交易数据结构）；
- **块尺寸**：交易数据的字节数总和；
- **时间戳**：十进制的Unix时间戳，表示开采区块的开始时间；
- **奖励地址**：最多32个字节。如果它与前一个区块的电子钱包列表中的某地址匹配，则该地址获得奖励；
- **前一个区块**：前一个区块的独立哈希；
- **高度**：此块在块织物中的序号，从零开始；
- **织物尺寸**：此区块尺寸与前一块的织物尺寸之和；
- **存储基金**：基金金额，以Winston为单位（参阅3.2.3）；
- **哈希列表**：从前一块到创世块的全部独立哈希列表；
- **哈希列表梅克尔根**：前一块的的哈希列表梅克尔根+前一块的独立哈希，取SHA-384散列计算结果。对于创世块，该值为空。基本上这就是从创世块到前一块的完全不平衡梅克尔树的根；
- **钱包列表**：所有曾经收到过AR的钱包的列表，即执行本块交易后的全部活跃钱包。该列表按钱包地址排序；
    - **钱包地址**：钱包公钥的SHA-256哈希；
    - **最近交易**：该钱包最近一笔被打包交易的ID，也就是最近一笔发起者字段的SHA-256哈希值等于此钱包地址的被打包交易的ID；
    - **余额**：钱包中AR的当前余额，以Winston为单位。经过以下规则调整：
        - 本块包含一笔发起者SHA-256哈希与钱包地址匹配的交易，则余额减去交易费，余额不允许为负值；
        - 该区块包含一到多笔交易，其目标与钱包地址匹配。余额要增加这些交易的金额总和。如果这是该钱包地址第一次收到AR，则将根据以下规则在钱包列表中创建其条目：
            - 余额将设置为交易总金额减去25 AR【译者注：原文如此】。这代表钱包创建费，充当创建垃圾钱包的过滤器；
            - 仅当金额大于等于25 AR时才创建钱包；
            - 交易按在块里的顺序执行；
        - 当奖励地址与钱包地址匹配，则将本块的挖矿奖励添加到钱包余额（参见3.2.3节）；
- **块数据段**：串接以下内容的SHA-384散列结果：
    - 前一块的独立哈希；
    - 前一块的依赖哈希；
    - 时间戳的十进制表示；
    - 最后难度调整的十进制表示；
    - 区块高度的十进制表示；
    - 哈希列表中全部值的串联；
    - 钱包列表的序列化表示；
    - 奖励地址；
    - 存储基金的十进制表示；

    - 本块的回忆块的序列化表示（参阅2.1）；
    - 本块全部交易的序列化表示的串联；
    - 哈希列表梅克尔根；
- **依赖哈希**：块数据段+Nonce的SHA-384散列结果；
- **Nonce**：PoW寻找合适的Nonce，以产生符合难度标准的依赖哈希，最大512字节；
- **难度**：PoW哈希必须大于的数字。该值是从前一个块继承而来，除非本块是难度调整块，则难度根据目标出块时间与实际出块时间之间的比例增加或减少；
- **最后难度调整**：最后一次难度调整的时间戳。如果本块是难度调整块，则该值应为本块的时间戳；
- **累积难度**：本分叉前面区块的尝试散列计算预期次数的汇总；
- **独立哈希**：对下列内容的深度哈希（参见10.2.2）：
    - Nonce；
    - 前一个区块；
    - 时间戳的十进制表示；
    - 最后难度调整的十进制表示；
    - 难度的十进制表示；
    - 累积难度的十进制表示。
    - 区块高度的十进制表示；
    - 依赖哈希；
    - 哈希列表梅克尔根；
    - 所有交易ID的串联；
    - 钱包列表的序列化表示；
    - 奖励地址，空值表示“无人认领”；
    - 空列表；
    - 存储基金的十进制表示；
    - 块织物尺寸的十进制表示；
    - 区块尺寸的十进制表示；
- **序列化表示形式**，以下内容的串联：
    - Nonce；
    - 前一个区块；
    - 时间戳的十进制表示；
    - 最后难度调整的十进制表示；
    - 难度的十进制表示；
    - 区块高度的十进制表示；
    - 哈希；
    - 独立哈希；
    - 按交易ID排序的全部交易的序列化表示的串联；
    - 哈希列表全部元素的串联；
    - 钱包列表的序列化表示；
    - 奖励地址；
    - 块织物尺寸的十进制表示；

10.2.2 深度哈希

深度哈希是一种散列算法，该算法将值的嵌套列表作为输入，并生成384位散列，其中任何值或结构的更改都会影响散列结果。

```
deep_hash ( List ) when is_list ( List ) -> hash_bin_or_list (List ).
%%% INTERNAL
hash_bin_or_list (Bin ) when is_binary (Bin ) ->
        Tag = <<" blob ", ( integer_to_binary ( byte_size (Bin ))/
        binary >>,
        hash_bin (<<( hash_bin (Tag))/binary , ( hash_bin (Bin))/
        binary >>);
        hash_bin_or_list ( List ) when is_list ( List ) ->
                Tag = <<" list ", ( integer_to_binary ( length ( List )))/
                binary >>,
                hash_list (List , hash_bin (Tag)).
        hash_list ([], Acc) ->Acc ;
        hash_list ([ Head | List ], Acc) ->
                HashPair = <<Acc/binary , ( hash_bin_or_list ( Head ))/
                binary >>,
                NewAcc = hash_bin ( HashPair ),
                hash_list (List , NewAcc ).
        hash_bin (Bin) when is_binary (Bin) ->
        hash_sha384 ( Bin).
```

清单1：深度哈希Erlang参考实现

10.2.3 区块影数据结构

区块影是完整区块的精简版，精简掉的数据可以从其他数据中重建。与完整区块相比，区块影极小，便于在网络中的节点之间快速低成本地传播。参见2.3。

区块影与完整区块的区别是：

- **交易**：区块影只包含交易ID列表。当区块影通过网络传播，接收到的节点很可能已经在内存池保存了交易，从而可以从交易ID列表重建完整的交易列表；
- **哈希列表**：哈希列表被精简为仅包含前50个条目。可以从前一个完整区块获取哈希列表，在前面加上前一个块的独立哈希，来重建完整的哈希列表；
- **钱包列表**：不包含钱包列表。可以通过前一个完整区块的钱包列表和本区块的交易来重建完整的钱包列表。

10.2.4 交易数据结构

- **数据**：介于0到10,485,760字节之间的任意数据；
- **发起者**：签署此交易的RSA密钥对的公钥；
- **金额**：发送到另一个钱包的AR数量，以Winston为单位；
- **目标**：指定的AR收款人的钱包地址，不能对有效性进行验证。最多32个字节；
- **奖励**：发起者向存储基金支付的AR数量，以Winston为单位（参见3.2.3）；
- **标签**：任意数量的标签，其中序列化表示最多2048个字节；
    - **标签**：表达任意元数据的键-值对；

        - **名称**：键，不限量的字节数据；
        - **值**：值，不限量的字节数据；
        - **序列化表示**：键和值序列化表示；
    - **序列化表示**：全部标签的序列化表示的串联；
- **交易锚点**：交易发起者的上一笔已打包交易的ID，或者最近50个块之一的独立哈希。第一笔交易该值为空；
- **签名数据段**：以下字段按顺序的串联：
    - 发起者；
    - 目标；
    - 数据；
    - 金额的十进制表示；
    - 奖励的十进制表示；
    - 最后一笔交易；
    - 标签的序列化表示；
- **签名**：签名数据段的RSA-SHA256签名，使用发起者RSA密钥对。
- **ID**：签名的SHA-256散列结果；
- **序列化表示形式**：以下内容的串联：
    - ID；
    - 最后一笔交易；
    - 发起者；
    - 标签的序列化表示；
    - 目标；
    - 金额的十进制表示；
    - 数据；
    - 签名；
    - 奖励的十进制表示；