Tau-Chain 和 Agoras

Ohad Asor

（ohad@idni.org）

编辑：Avishy Carmi 教授博士

（avishy@idni.org）

2020 年 8 月 20 日

社区审查草案

摘要

我们描述了系统 Tau-Chain 和 Agoras，前者是一个完全由用户有效定义的对等网络，后者是一个建立在这些能力之上的经济体，促进了知识经济和其他支持方面。基于一种新提出的范式（即人机交互），Tau 是一个新一代智能社交网络和讨论平台，为大规模讨论、决策、软件开发、人工智能、法律哲学等相关问题提供一套基于逻辑的解决方案。

智能去中心化网络倡议

www.idni.org

编者按

智能化在很大程度上（也可能完全）是一个需要适当选择的问题。聪明的人寻求在不同的环境下（有时在重要的约束下），增强他们做出盈利决策的能力。除了自然对信息的操纵和存储的限制外，没有什么能真正阻碍一个足够智能的实体。这种观察结果使得智能化成为唯一有价值的资源。

当谈到智能化时，我们常常区分人和机器。在每种情况下，智能似乎都有不同的含义。事实上，通用人工智能的最终目标是将机器智能升级到人类水平。智能化是一个相当复杂的术语，包括与心智和智力有关的范畴。然而，有一点是明确的：智能化随着时间而发展。在这里，我们提问两个问题。第一个问题是为什么智能化要发展？第二个问题是智能化通过什么方式发展。第一个问题的答案可能是，正如自然界中的所有其他过程一样，智能化也会通过不断面对新的条件而追求发展，这些条件用以解决新难题。 第二个问题的答案是，在很大程度上，智能化是通过人类、文化和物种之间的交流而发展。

人类擅长沟通。他们开发了多样化的语言，用于讨论问题并给出可能的解决方案。但是最紧迫的问题，如涉及社会和国家等大型社会结构的问题，无法通过讨论解决。如果我们遵循上述推理，讨论规模的限制是人类进化过程中的一大障碍。

Tau-chain 或 Tau 系统，旨在最终解决扩大讨论所固有的问题。这一系统是借助机器来实现的。然而，机器需要花费巨大的成本。Tau 网络中的机器应该能够访问。换句话说，人类和机器使用他们都能理解的语言。本文论证了为什么在人机交互中应该使用形式逻辑。重要的是，本文假设没有一种适合所有的语言，因此提出了一种（Tau）元语言 TML，用于构建语言和其译者。这里所称的是语言互联网概念的基础。

Tau 确实是一个由社会选择过程支配的自我修正系统。在 Tau 系统上，用户讨论并决定各种问题，包括 Tau 自己的代码。因此，Tau 基于其用户的集体决策而发展。在本文中，这方面与立法过程平行。事实上，社会通过投票作为选举制度，不断改变其统治者、法律和行为。与 Tau 的关键区别在于，在 Tau 中，用户可以平等地讨论选择什么。Tau 可以平等地进行讨论，因其设计保留了小规模讨论的宝贵特征，例如在大规模环境下分享意见和作出选择。

为了处理自我参照的悖论，TML 的设计考虑到了三条定律。它们是法律形式逻辑的特征，因为如上所述，Tau 过程类似于立法。可判定性、布尔运算下的闭包和自我解释这三个定律使得可判定性和无限制递归共存，这两个特性在立法过程中备受期待。他们还指出了遵守这三条定律的具体逻辑。

其不会在这里结束。为了维持社会结构，人们需要一个经济基础。Tau 上形式化的知识适用于维持经济基础设施，即知识经济。引用本文所述：

> *将知识形式化为机器可访问的格式比书籍和搜索引擎具有优势：数据不再是一个比特流，而是比特背后的含义，因此，即使是在大型著作汇编中，也可以对小块知识执行自动语义查找。*

在 Tau 上，知识经济通过 Agoras 实现，Agoras 是一个借助加密货币的网络。Agoras 将利用 Tau 的知识表示和协作形式化功能，以允许知识所有者和创造者提供有偿使用。

本文深入描述了 Tau 和 Agoras 的这些特征和其他许多特征。

目录

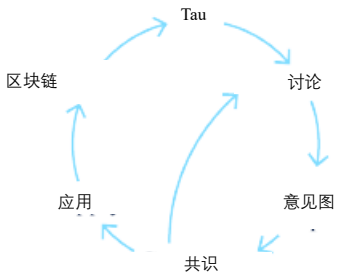
1 前言

Tau-chain（简称 Tau），是一个去中心化的区块链网络，提供创新的社会选择方法，以确定下一版本，并由其众多用户单独有效地控制。Tau 的用户指定他们希望系统是什么样的，然后 Tau 根据用户的共识自动更新。

软件每天更新和自动更新。但是谁来决定下一个版本是什么样子呢？通常只有一个开发团队决定这样做，其会考虑到用户的输入，但是我们怎么能拥有一个完全由其众多用户控制的软件呢？这就是 Tau 的意义所在。在 Tau 上，开发人员不是其他人，而是用户自己，在某种程度上，他们实际上可以大规模工作。此外，在 Tau 上，软件的需求和规范由其代码标识。用户不需要描述“如何”，只需要描述“什么”。

我们的重点是如何在大量用户的情况下实现这一点。事实证明，实现这样一个目标需要解决某些难题，特别是可扩展性差、避免逻辑悖论以及由自我修正产生的不可能性。本文提出的解决方案本身很有趣，可用于许多其他类似环境，例如现代民主、健全立法、协作软件开发和管理大量知识，如学术研究或大型组织中产生的知识。本企业的一个关键要素是正式语言的使用，如下所述。

图 1.1：**TAU 的工作流**



我们认为，用户合作决定 Tau 后续编码的过程与立法过程相同。法律不仅是法律的一个方面，而且是就法律应该是什么达成协议的过程，更重要的是，法律如何随着时间的推移而变化。Tau 是一个立法过程，其中法律只是 Tau 的编码。因此，我们将重点关注法律和立法。读者可能会记住，这些过程只是下一段中定义的 Tau 过程的另一种方式。

在本文中，我们简单的定义了 Tau：一个过程，用 X 表示，由【许多】人组成，通过正式语言的讨论形成并遵循另外一个过程，用 Y 表示。Tau 是 X=Y 的情况。

简而言之：Tau，是关于 Tau 的讨论。特别是，Tau 是一种机器辅助的相互理解过程。

Agoras 是一个通过 Tau 实现加密货币的网络，允许实现以下功能：[1]

- 自我定义：Agoras 建立在 Tau 技术之上，是一个由众多用户有效控制的软件。
- 知识经济：利用 Tau 的知识表示和协作形式化功能，以允许知识所有者和创造者提供有偿使用，并让知识探索者高效地查找和购买知识。
- 计算资源市场租用和出租各种形式的计算资源（CPU、磁盘等）的能力，允许在成本和计算效率较低的情况下进行大规模计算。
- 衍生市场： Agoras 拥有一个没有中间人的点对点衍生市场，通过使用零增量投资组合的概念，在不创造任何新硬币的情况下实现无风险利息。

应该注意的是，这里的设计包含了[11]中提出的错误设计，并且忽略了与自我修正系统相关的重要方面，例如，如何扩大社会选择，同时避免自我修正悖论。 旧的设计必须被彻底抛弃。对读者来说，试图通过重新审视图片来补充图片没有好处。此外，计算资源市场不同于[10]中描述的 Zenne 市场，因为周围的功能（主要是逻辑、契约和证明）导致了更复杂的设计。

[1]与 Tau 相反，Tau 没有货币或任何其他货币方面。

另一点是，本文有时用现在时态提到 Tau 和 Agoras，例如“Agoras does…”，但 Tau 和 Agoras 还没有准备好写这些。

本文的组织结构如下所示。第一节讨论 Tau。首先，介绍人机交互的范式和系统的主要特征，以及如何扩展讨论。然后我们继续讨论决策的各个方面。之后，我们讨论协作软件开发，描述我们的主要理论贡献，即承认它们的定律和逻辑。我们以语言互联网作为结束。

第二部分是关于 Agoras 的，首先描述合同，然后是知识经济学，最后是计算资源市场和衍生市场。

在最后一节中，我们介绍了该系统的长期影响。

五个附录如下：附录一总结了本文的主要问题及其答案。附录二介绍了一些经典的选择理论，并将其与我们的环境进行了比较。附录三的观点与苏贝尔关于自我修正的观点相一致。附录四描述了 P-DATALOG 语言，第五个给出了带有求值运算符的 P-DATALOG 折叠为 P-DATALOG 的证明。

2 Tau-Chain

2.1 人机交互

一个庞大的群体如何达成集体决策？Tau 提供了一个新的范例（人机交互范例）在很大程度上解决了这个问题。因此，Tau 对于各种协作理论的形成和决策非常有用，而不仅仅是其自身的代码，我们将在后面强调这一点。我们甚至可以说，我们的解决方案可彻底解决问题：与目前讨论的规模形成鲜明对比的是，多出 10 个参与者确实会产生至少多出 10 个结果。[2]这套解决方案植根于两个概念，此后将对其进行解释：第一种是使用机器可理解的语言，第二是能够在大范围内保持小规模的讨论和决策。但在讨论这两点之前，让我们先讨论一下比特立法和投票。

当考虑大规模的立法合作（或任何其他理论形成）时，通常会想到投票。事实上，国家法律的立法通常涉及投票形式。但是，与投票相关的某些主要比例限制：即使有可能让每个人都有有效和平等的投票权，每个人都不可能有有效和平等的权利来提议投票，因为即使每个人都有平等的权利来提议，一天怎么可能阅读数百万份提议？这使得平等提议的权利无效。我们的结论是，投票不能同时有效地扩大规模和保持公平。

此外，即使我们假设所有参与者都彬彬有礼、聪明、思想开放、彼此相爱、表现完美，讨论的难度仍然存在。应该注意的是，处理帮助讨论的其他项目确实倾向于处理这种现实的缺失，即每个人都是“完美的”。然而，我们处理的是一个基本瓶颈，即使“完美”假设成立，这个瓶颈仍然存在。

[2]讨论的规模甚至比经济学中出现的其他环境更小。典型的经济设置规模下的“收益递减定律”：两倍的工人通常不会产生两倍的结果，而是更少。这通常是通过说明效用函数的二阶导数是负的，或者换句话说，是凸的来形式化的。这是凸性在经济学中发挥重要作用的主要原因之一，任何经济效用函数的主要假设之一就是凸性。然而，讨论的规模更大：它们的效用（例如，知识的数量，或协议，变得对参与者可用和有用）不仅仅是一个负的二阶导数 W.R.T.参与者的数量，但通常甚至是负的一阶导数，因为在讨论中有太多的参与者通常会减少结果，不仅不是按比例增加。
注意这里的可微性和凸性是广义的：对于离散变量，取其差异将为相同的目标和收益递减定律的论点服务。

在日常生活中，大规模决策通常通过建立决策者的等级来解决，特别是决策者可以决定人们可以投票选择什么。但是，大量的知识和选民的不同偏好并没有反映在提案中，因为由于信息处理瓶颈，数据无法向上传播层次结构，从而阻止人类以任何形式处理大量信息。[3]因此，对几乎没有真正相关考虑的备选方案进行投案。

在小团体中，我们很少考虑投票。人们只需说出他们必须说的话，这本身就足以让参与者理解彼此的观点以及他们同意或不同意的地方。参与者不需要投票，因为他们已经知道投票结果是什么，所有信息都已经隐式或显式地呈现。换言之，意见图自然来自讨论中所说的话。我们如何才能在更大范围内实现这一点？在这个范围内，我们在大量参与者中进行了讨论，但我们仍然能够恢复意见图。

这一过程在机器的帮助下实现，但它带来了巨大的成本：讨论的参与者必须使用机器能够理解的语言。[4]在这种情况下，机器就能够计算意见图（以及协议/分歧/共识）。我们把这种语言称为形式语言，更具体地说，我们认为逻辑公式是可译的语言。关于 Tau 语言方面的其他内容将在第 2.5 和 2.6 小节中强调，但我们首先要描述系统的其他元素。

[3]更不用说其他瓶颈，这些瓶颈可能通过强加某些流程和行为来解决，也可能无法解决。
[4]事实上，并非所有这样的语言都适合这种情况，我们将在后面强调实现这种工作所需的逻辑方面。

2.2 大型讨论平台

当大范围内保持小规模讨论时，我们希望用户简单地说出他们的观点，而不需要对其进行任何组织，但是所有观点以及它们之间的关系都应该自动推断出来。

Tau 的形式是一个讨论平台和一个类似的社交网络，有朋友、帖子和评论。主要区别在于用户使用正式语言编写，以及由此衍生的其他方面。因此，该系统的主要组成部分是：

- 讨论：使用正式语言进行讨论的能力。讨论包括观点、问题和答案。讨论也可以隐式：spho-radic 在网络上发布的帖子和评论，但关于同一主题的帖子和评论，可能会被计算在内。
- 世界观：作为用户配置文件，代表系统中所有用户意见的总和。
- 团队：为共同目标而协作的一组用户。
- 权限：仅允许特定人群（如朋友）访问用户的世界观或其部分内容。团队也是如此。
- 查询：用户能够对在世界观和讨论以及平台上的任何数据进行语义查询。这相当于机器以精确的方式回答问题。
- 同意/不同意：用户可以在其他用户的帖子上单击“同意”或“不同意”。这被认为仅仅是一种速记，就好像用户发表了相同的意见（或否定意见）。同样，用户可同意/不同意哪一个问题有趣，请参阅下面系统中的问题与答案。
- 信任：自动与某些用户就某些主题达成一致。
- 共识计算所有或大部分讨论参与者同意的内容。用户可以对“共识”的含义进行更广泛的定义。

- 意见图：在讨论中表达的所有观点的更详细的报告，按逻辑意义上的暗示排序。[5]请注意，虽然我们可以预期四个人的讨论会包含四种不同的观点，但我们并不期望一百万人的讨论会包含一百万种不同的观点，但许多参与者会持有相同（或重叠）的观点和/或同意彼此的观点。
- 矛盾解决方案：当世界观内部或用户之间（无论是否在同一讨论中）出现矛盾时，系统将报告矛盾并提供解决方法。
- 自动评论：Tau 能够根据同一用户发表的关于该主题的帖子，根据用户的审查和批准，自动对主题进行评论。[6]
- 合成：在讨论某个所需软件的规范时[7]，要综合符合该规范共识的代码。

此外，我们还拥有语言互联网的以下特点，该设计允许定义新语言并在它们之间进行翻译，详细讨论见 2.6：

- 添加语言：将文档翻译成现有语言的翻译人员，通过网络支持新语言或现有语言的新版本。
- 翻译：翻译（编译）一个文档，从一种通过语言互联网定义的语言翻译成另一种语言。

2.2.1 世界观和团队

用户的世界观是该用户以帖子、讨论和同意/不同意的形式表达的所有观点的集合。类似地，世界观包含用户标记为“有趣”的问题。[8] 然后，用户不仅可以查询自己的世界视图，还可以查询其他用户的世界视图，但他们授予其他用户查看其世界视图或其一部分的权限。通过这种方式，用户可以找到具有类似想法或常见有趣问题的其他用户。人们能想象其对招聘、研究、商业合作甚至恋爱观的影响。

[5]例:“X 和 Y”意味着 X 和 Y，换句话说，“X 与 Y 相交”是 X 和 Y 的子集，或者“a 是 B 的一个特例”。
[6]Tau 永远不会猜出你的意见，甚至是有根据的猜测。这证明了基于逻辑的人工智能的力量和必要性，而机器学习在本质上是概率的。
[7]包括系统本身的规范。
[8]关于 Tau 的问题的性质将在下面讨论。

团队是一群有共同事业或共同问题需要讨论和回答的人。团队创建者可以任意选择其初始规则。一个特殊团队将决定下一个 Tau 代码。谁将参加这个团队以及如何参加这个团队将由这个团队来决定，我们将在后面详细描述。

2.2.2 真理与意见

值得注意的是，Tau 不可避免地无法获得真理，只能获得意见。有人可能会争论人类是否能够获得真理。我们也不会讨论这一点。我们只是说，显然，机器无法获得我们直觉上认为是真理的东西。[9] Tau 无法验证“太阳是由氢构成的”或“太阳是由奶酪构成的”等说法的正确性。[10]这样一来，Tau 永远无法说出谁是对的，哪种观点是正确的。[11]然而，如果出现重言式和矛盾，它可以推断出某些种类的真理和谬误。一个相互矛盾的观点总是错误的。 在这种情况下，我们确实可以指出一个错误。重言式也是如此。

对某一话题发表的意见越多，出现矛盾的可能性就越大。[12]因此，我们可以通过阐述来近似真理，并选择通过辩论的方式将某些观点视为真理或谬误，要求每一方就这一主题发表更多的声明，从而增加每一方产生矛盾的可能性。与 Agoras 相比，Tau 也可能有货币方面的影响。

[9]与数学真理相反，数学真理与同义反复论没有什么不同，也就是说，“真实世界”除了“矛盾可能永远不存在”之外，没有什么重要的。

[10]即使我们在机器上装备了传感器，比如摄像头和麦克风，也总有可能通过让机器看到任何预先设定的输入来“欺骗”它，它永远也不会知道它的感官输入是“真实的”、“虚假的”还是“妥协的”。

[11]特别是，Tau 不应该被视为避免“假新闻”的工具，除非是我们马上解释的特殊情况。我们之所以提到“假新闻”，是因为在听到 Tau 及其逻辑性和真实性的人当中，一个普遍存在的问题是，Tau 能否解决这个问题，而答案实际上大多是否。

[12]事实上，由于合取的乘法性质，这个概率以指数的方式增加。

2.2.3 问题与答案

我们从关于问题与答案的哲学观察开始。我们有一个正确答案的概念，我们甚至可以编写计算机程序来判断给定问题的给定答案是否正确。我们对问题的正确答案感兴趣。

正如答案可能是正确的，也可能是错误的，同样，问题也可能是有趣的或不有趣的。我们如何判断哪个问题有趣？这不是机器能回答的问题，因为这纯粹是人类的事情。

有趣的问题不仅是主观的，而且本质上源于提问者的逻辑上偏好。[13]没有“一个有趣的问题”这样的东西，而是“一个在特定时代对某些存在感兴趣的问题”。我们的问题来自我们的人性和我们的个性。我们的个性由我们感兴趣的问题决定，而不是由我们给出的答案决定。同样，与对相同答案达成一致的人之间的合作相比，对相同问题感兴趣的人之间的合作更有意义。许多人会更感兴趣地发现人们提出与他们相同的问题，而不是发现人们给出相同的答案。

问题还指导知识表示中的一些理论考虑，例如开放世界假设的设置与封闭世界假设的设置[14]，因为问题可能被视为开放世界假设下的疑问，而答案与封闭世界假设更相关。因此，问题是 Tau 的一个主要方面。

为清楚起见（仅针对本小节），我们可以区分“问题”和“ 查询”。“查询”指的是我们希望立即得到答案的问题，例如，我们向机器提供信息，然后查询该信息的情况。机器不应返回任何新信息，它将只使用我们提供的信息。[15]

[13]为了让我们相信自己欲望的随意性和非逻辑性，我们可以承认，自己甚至不知道晚餐想吃什么。没有任何逻辑公式可以得出这一点。它与真理的本质无关，而与人性有关。

[14]开放/封闭世界假设是数据库和知识表示领域的基本概念。

[15]这与机器学习的人工智能形成了鲜明的对比。Tau 是关于逻辑人工智能的，而机器学习是关于对测量人口一无所知的统计，因此即使是对整数相加这样的简单问题，也经常反馈错误的答案。与逻辑相比，机器学习的表达能力相对较少，这一点还有很多值得探讨的地方。从描述复杂性的角度来看，机器学习被认为属于复杂性 P 类，而到目前为止，普通逻辑时不时地包含该类。

相反，我们所说的“问题”指的是我们还没有得到答案的问题。问题是定义所需知识的工具。问题通常按时间顺序排在知识之前，即使没有，这些知识也会经常被忽略，并被视为“不感兴趣”。问题是界定讨论范围或探索某些知识领域的工具。这是一种机器永远无法模拟的工具，只有考虑到人类对哪些问题感兴趣，它才能通过组织对这些问题的讨论和执行自动推理任务帮助我们找到正确答案。特别是，不管谁或多少人给出了答案，Tau 都能找到感兴趣问题的正确答案，这在整个网络中取决于用户设置的可见性权限。到那时，好的答案和想法将不再被浪费和忽视。

事实上，问题没有类似于答案的“真实值”，而是“兴趣值”[16]。这一事实引导我们在拥有足够的形式化知识后，总结出人的角色与机器在 Tau 上的角色：人是用来提问的，机器是用来回答的。更广泛地说，我们认为这是一个哲学真理，应该指导任何人工智能的实现。

2.2.4 相互理解

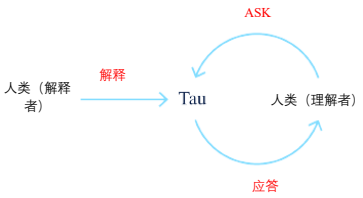
在 Tau 上，机器在对话中不是平等的部分：它只是一台机器。Tau 只是组织我们所说的，因为我们以一种可访问的方式对我们的信息进行编码。一个用户可以向另一个用户传播想法。在这种人与人之间交流想法的狭窄范围内，我们已经可以享受三个好处：易于解释，易于理解，并将知识形式化为副产品。我们首先提出了“理解”的有效定义，以此证明这一说法的合理性：完全理解某一点意味着回答与该点相关的所有问题的理论能力。[17]

[16]引起了问题的排序。一个正确答案并不比其他正确答案“更正确”，但一个问题可能比其他有趣的问题更有趣。

[17]特别是，由于哥德尔的不完备性定理，我们可以有把握地说，没有人会完全理解算术。

在 Tau 上，解释者不需要让其他用户（目标用户）理解，而只需要让机器理解。这项任务在某些方面可能更简单，在另一些方面可能更复杂。因为机器不像人类那样受组织和规模的约束，而且可以进一步帮助形式化过程，但另一方面，机器除了无法接近人性之外，还有语言障碍。实现了语言形式化后，目标用户现在不仅可以将其翻译成其他知识表示语言，或者按照他们认为合适的方式对其进行组织，或者将其与其他形式化想法进行比较，而且他们还可以向机器询问他们得到的所有问题。由于机器理解了上面的主题，它也可以帮助目标用户理解它，因为它可以回答所有用户的问题，而无需将问题提交给最初作者。因此，Tau 是一位值得信赖的思想大使。

图 2.1：相互理解



2.3 大规模去中心化社会选择

2.3.1 自我修正程序概述

我们现在将 Tau 的流程绘制为一个自我修正的平台。使用法律的正式语言（定义见下文），用户通过 Tau 讨论 Tau 的下一个版本应该是什么。Tau 然后计算其下一版本的可能替代方案。在 2.3.2 和 2.3.3 中，我们将讨论选择哪个备选方案。这些备选方案是 Tau 区块链的下一个区块候选方案，其将在下文讨论。一旦选择了一个替代方案并添加到区块链中，Tau 就会下载下一个区块并根据其新定义调用。正如我们将看到的，我们的选择逻辑将包括一个自解释器和内置的引用和评估操作符，这样 Tau 就不需要替换其代码并从外部运行。Tau 只需在区块链内引用的代码上调用求值运算符，然后将区块链从 genesis 区块重放到当前区块，从而证明整个区块链直到最后一个区块的历史是正确的。[18]

[18]如果没有额外的逻辑成分，Suber 的 Nomic 并不支持这一点，正如“Tau vs。Nomic”。

此外，本文还介绍了共识和下一版本之间的一个特殊步骤，称之为“应用”。在用户发表意见并达成共识后，更改不会立即发生，但用户也必须同意应用更改。如果用户不同意，将继续进行讨论，可能会或可能不会达成不同的共识。这一“同意应用”步骤被用作保障措施，并作为在应用共识之前审查共识的另一个机会。原因是，共识总是浮动的——这是所有用户在某个时间点上达成一致的声明。如果达成共识，并不意味着在下一分钟仍保持一致。我们必须有意识地接受协商一致意见的形式，而协商一致意见正是以这种形式出现的。此外，由于共识是许多人意见的重要集合，用户可能第一次看到了提议的共识（以及提议的下一个版本）。然后，他们需要对其进行审查，并同意完整的情况确实是可取的。

此外，共识在某种意义上可能不是决定性的，因其并不要求程序的每个状态都有一个唯一的操作，因此无法将其作为程序执行。因此，完善共识是这一进程的一个重要方面。

最后，我们指出，用户可以在接受一致意见之前在应用阶段配置更多的测试和条件，如下一节所述。我们在此提到一种可能性：衡量一组规则与另一个规则的方法，如果每个协议具有相同的强度（例如 50-50），则要考虑每一套规则都会回答标记为“有趣”的问题，而回答更有趣问题的答案则会受到青睐。

2.3.2 选择困境

“如果两个过程被判断为相等，那么医嘱就无法打破僵局，它所能做的就是暂停判决，直到情况发生变化，并且行动方向明确。”

-Jean Buridan, c. 1340

我们解释了 Tau 是一个由用户有效决定的软件。为了实现这一点，用户将不得不就 Tau 的下一个代码需要做什么进行大规模讨论。 随后，共识将自动计算，所有客户都将更新为 Tau 在该时间点定义的新规范。 但应该使用哪种方式达成共识？全体一致、多数一致或其他方式。

在去中心化软件网络中，多数投票的主要技术问题是无法检测到同一个人的多张投票。如果我们给予一些人信任和投票权来减轻权力下放的要求，那么上述情况可以避免。另一种方法是让用户通过集中的第三方（“电子公证人”）证明其身份的唯一性。

在去中心化环境中，一致性也会带来安全风险，因为参与者可能会诚实或不诚实地阻止整个网络达成共识，并将网络置于危险的 Buridan 的 Mule 情况下。

我们在上面提到了一种解决矛盾的机制，以及辩论，并测试哪些标记为“有趣”的问题得到了回答。这些都不是完全决定性的方法。因此，在下文中，我们假设矛盾没有得到解决。[19]

经验法则之一是不要限制用户做任何不影响其他用户的事情。如果一组人希望系统以某种方式运行，而另一组人希望系统以不同的方式运行，那么我们可以检查这两组人的偏好是否可能共存，并且每个人对系统的看法是否不同。但事实上，这两个变化并不总是共存，因为它们可能包含一些关于系统基础的矛盾，这可能会使系统之间不互相兼容。

[19]在这里，模糊性被称为矛盾，因为结构良好的程序或功能的要求与我们对程序的每个状态都有非唯一的操作的情况是矛盾的。

处理两个（或多个）相互矛盾且不能兼容共存的备选方案的一种方法是，要求每个组计算其提案（挖掘）的哈希值，直到哈希值足够小。这会优先考虑具有更多计算（或其他挖掘）资源的用户，这就是比特币的机制：计算能力更长的链是网络首选。许多基于区块链的算法都是可能的，每种算法各有优缺点。类似的方法是在备选方案中随机选择。这当然是我们希望避免的解决方案。然而，在纯粹的 Buridan's Mule 的情况下，随机性是唯一的出路。[20]

我们在这里涉及的主题实际上是改变法律的定律，这是下一节将要讨论的一个非常重要的主题。有很多计算共识的方法，但没有一种是完美的。因此，它需要用户之间进行讨论，在讨论中，他们提出了各种可能性，并强调了各自的优势和劣势，最终形成一套由社区支持的初始规则。因此，我们建议在半集中式环境下（与区块链网络相比）开始讨论 Tau，让用户决定去中心化网络的初始治理机制，从而暂时避免 Buridan Mule 情况的灾难性后果。

2.3.3 区块链

写这些文章的时间，解释区块链的用途很重要，因为目前世界上对区块链可以做什么有很多误解，而其他最佳实践和加密原语无法做到这一点。答案是区块链解决了去中心化时间戳的问题。许多形式的数据安全可以在没有任何区块链的情况下完成，但可以使用公钥基础设施、哈希和其他加密手段。然而，在不信任的环境中，确保事件的时间顺序的问题被证明是无法解决的，参见拜占庭将军问题。

Satoshi Nakamoto[8]第一个提出基于密码假设的近似解。这种解决方案非常昂贵（而且不仅仅是在财务方面），因此对于不需要去中心化时间戳的情况，非常不推荐。但是，这种解决方案在 Tau 的范围内是必需的。

[20]Buridan 提出的等待也是一种可能的解决方案，但不能保证有效。

Tau 的区块链是其自身代码的分类账。每个块包含当前 Tau 代码，下一个块包含下一个代码，方式类似于“自动更新”。[21]这些代码的时间顺序至关重要：如果客户端在选择哪种代码版本方面不一致，它们甚至可能会破坏通过网络彼此通信的能力，例如，如果网络协议已以非向后兼容的方式修改。

有人可能会问：Tau 将使用哪种区块链算法？答案很简单：它将使用用户希望它使用的任何东西。如果有一天会发明一种更好的去中心化时间戳算法，那么它就可以自动更新。

2.4 协同软件开发

Tau 有四个与协作软件开发相关的主要准则：自动建构校正软件、面向知识的编程、面向共识的编程和代码重用。

通过使用声明式编程（一种程序员只定义“什么”而不定义“如何”的范例）来获得自动建构校正软件。例如，假设我们有一个代码体，我们想给它添加额外的约束，例如“从不通过私有网络发送数据”。在非声明性编程中，程序员必须分析代码的控制流，查看数据发送到哪里，然后确保数据不是私有的。当然，这导致了许多人为错误。[22]

但是，如果我们可以在程序中添加一个句子来表示上述约束，会怎么样呢？如该约束定义良好，懂得该约束的机器应该能够跟踪程序流并防止发送私有数据。为此，我们需要对程序进行数学推理，因此语言在这些任务中必须具有可判定性。

[21]正如我们在上文提到的，这就是“改变法律”的过程，下文将进一步强调。
[22]在软件开发过程中，人为的错误和 bug 太多，非程序员可能会很大程度上低估了它们。

因此，我们观察到，声明式编程在很大程度上消除了处理程序控制流的需要。我们可以关注程序的规范，而不是程序的实现。获取程序规范并将其转换为实现的领域称为软件综合。[23]因此，程序员只需要关心规范，其余的可以而且应该是自动的。

本规范，或换句话说：软件的需求大部分不是由程序员定义的，而是由程序员的客户端定义的。客户端需要说出他们想要的软件，程序员需要编写代码。一旦通过声明式编程，我们用代码识别需求，程序员和用户（或从程序员那里订购编程工作的实体）之间的边界就会模糊。然而，规范必须完整，因为机器总是能够推断出下一步要执行的唯一动作。在我们的例子中，我们需要一个“私有数据”和“通过网络发送”的完整规范。

这就给我们带来了面向知识的编程。在 Tau 上，大量的知识将被形式化，要么直接为了拥有一个形式化的知识体系，要么作为讨论的副产品。一旦有了足够的形式化知识，即使是非程序员也能说“根据我从一位我信任的专家那里调整的安全定义，我希望这个软件是安全的”。只要说出这句话，软件就会变得安全。

我们现在转向协作方面，即面向共识的编程。考虑到目前为止我们所解释的一切，这很直截了当：人们只需要陈述他们想要软件做什么，他们的共识是商定的规范。这不仅有助于彼此达成一致，而且有助于不同的人承担软件不同（甚至相同）方面的责任，而不必相互干涉。这在现实生活中是一个痛苦而昂贵的软件开发难题。

代码重用也是一个非常有用的概念。实际上，所有常用算法都已经实现了很多次。 这是因为使用现有代码有很多困难，所以重写代码通常更简单、更便宜。[24]但是，由于我们将编码任务简化为知识表示任务，因此知识很容易重用。我们不需要对同一事物进行两次定义，也不再需要进行集成工作。

[23]参见第 11 页还可以观察到，给定一个完整的规范，可以在每个步骤中查询下一步应该是什么来执行程序，并且无需任何代码转换就可以执行程序。这种方法在实践中可能工作进度非常缓慢，有点不切实际。因此，综合是一种优化。 也就是说，此优化可能有更多方面不涉及代码合成方法。

[24]这是软件开发中非程序员可能无法完全理解的又一特殊事实。事实上，如果是由不同的人编写的，或者是由同一个人在一段时间后编写的，重写通常比维护现有代码要便宜得多，而这段时间通常非常短。

此外，这不仅与代码有关，也与知识形式化有关。总体而言，我们获得了一个类似经济异常的特征：因为随着系统中已经存在许多代码和知识，编写新代码或形式化新知识变得更加容易，所以收益递减定律变成了收益递增定律。

2.5 法律逻辑

“法律门前坐着一位守门人”。

- Franz Kafka，“法律门前”，1915 年。

我们生活在一个没有人知道法律的世界里。除了琐碎的案件，你不知道什么是合法的，什么是不合法的。你所能做的就是采取行动后看看法官或警察的意见是什么。当然，不同案件的意见是不同的。或者你可以咨询一位律师，他会告诉你没有绝对的答案。说到底，这是一个概率问题，不幸的是，没有人知道如何计算。你可以尽最大努力过上富有的生活，遵守你所理解的规则或律师的指导，但没有人能保证你不会被视为罪犯，或者不会对你采取法律行动。同样，一个人可以过着伤害这么多人的生活，即使法律体系意识到所采取的行动，也没有任何法律体系会阻止它。这种悲观的情况不是新情况，也不是局部情况。Franz Kafka 的描述符合我的情况。

自从人类掌握了语言技能以来，我们总是有意或无意地玩弄文字并应用到任何地方。最坏的谎言只包含真相，最坏的罪行都记录再按，几乎任何事情都可以用论据来证明。这种“真理危机”是后现代哲学流派的基础，尤其是解构主义的方法，论证了文本如何以多种矛盾的方式解读。“没有一个真理”是后现代主义的基础。但我们至少能有一些真理之岛，让社会契约变得有用和有意义吗？

接下来，我们得出了法律和立法的逻辑背景，从而提出了实施法律和立法的合理方式。回想一下，这不仅适用于一般法律，也适用于 Tau 代码。

2.5.1 法律的定律

出于我们的目的，我们将法律定义为一个函数，其对一种情况进行描述，并返回“合法”、“非法”或“未定义”。[25]这可能令人惊讶：如果法律没有进一步假定这些法律的性质、目的或范围，那么如何能说一些不重要的话呢？考虑到法律表述的有效性，以及法律变更的情况，我们确实得出了重要的陈述。

此外，我们绝对不谈论法律本身，只问哪种语言可以被视为法律语言，或者更准确地说，因为语言细节无关紧要，我们寻求一种能够合理地代表法律的逻辑。我们从法律的逻辑（“法律的三个定律”）中提出三个要求，这将导致我们走向某种逻辑形式主义。

我们寻求一类适用于表示法律的表达式 L，现在我们列出了 L 中的一些要求。[26]前两项要求非常自然，但对所需语言构成了重大限制：

- 可判定性：回答某事是否合法不应该花费很长时间。我们需要始终有能力得到答案，特别是在有限的时间内。这使我们认为 L 是一组图灵机。
- 布尔运算下的闭包：等式为 0 时，语言应该在 if-then-else 下关闭。因此如果 f，g，h 是 L 中的程序，那么“如果 f（x）=0 那么 g（x）else h（x）”也是如此。换句话说，L 在并、交和补下应该是封闭的。[27]

下一个要求是在考虑修改法律的情况时提出的。假设我们有一条法律，现在我们想随着时间的推移进行修改。因此，我们最好制定修改法律的定律，否则，法律在任何情况下都不可能被修改，从而使其变得毫无意义。为了打破法律，人们只需要修改它，而在没有修改法律定律的情况下，没有任何东西会阻止人们这样做。

因此，我们不仅需要法律，而且需要改变法律的定律。但是我们也需要改变法律的定律，这样我们就可以改变法律，再次使法律变得无效和无意义。

因此，我们必须有无限多的定律（无限变化定律）吗？如果不是的话，一个独裁者立法者可以随时把法律变成任何东西，仅仅从最简单的法律开始？我们马上就会看到，有一个办法可以解决这个难题。

考虑以下定律：“所有法律，包括这项法律，都可以通过多数票修改”。是定律，还是改变法律的定律，还是改变法律的定律的定律？好吧，它是同时发生的，因为它指的是自己。它指的是这个定律本身是如何改变，它实际上是一个无限改变定律的定律。要注意，这条定律很有道理，不会引起自相矛盾，这种问题通常出现在自我参照的陈述中，[28]参见附录 C。

---

[25]“未定义”值的必要性将在下面说明。“未定义”不等于“未指定”。它更像是一台永远不会停止的机器，因此根本没有返回值，所有可能调用它的机器也永远不会停止，因此也将处于“undefined”的状态。另一种看法是： 除零不是“未知”，而是真正没有定义。任何包含除零的公式都不为真或假，只是没有定义。

[26]我们将开始治疗 L 图灵机作为一组,一个视图下面的可判定性的要求是合理的,但是,同样,我也可以被认为是一种编程语言（以及它所代表的组图灵机是在那语言）的所有程序集,或者,作为一个可决定的（因此可计算的）逻辑。我们的目标是找到一种逻辑语言来捕获这组机器。

[27]另一种看法是： 法律语言应该允许使用逻辑连接词“and”、“or”和“not”。

[28]自我参照定义有一个不合理的负面名声。如果我们打开字典，看到某个单词的定义，而这个定义包含并依赖于同一个单词，人们通常会认为这个定义是矛盾的或没有意义的。然而，我们从可计算性理论和数学逻辑中知道，有时递归定义不仅有意义，而且有时是必要的。 例如，考虑阶乘函数的定义。

我们之所以能够在一个有限句中获得无限变化定律，仅仅是因为我们使用了自引用，或者换句话说，递归。我们已经证明，如果法律不涉及自身，那么我们将永远无法健全地保护法律不受意外变化的影响，因为我们确实需要无限多的法律，如上所述。

事实上，我们从法律逻辑上要求的不仅仅是递归：我们实际上需要自我解释。[29]支持自解释的语言还必须支持无限制递归。[30]我们需要进行自我解释，因为现行法律必须能够解释拟议的法律，以便了解其语义，然后决定是否接受或拒绝拟议的法律。

这要求我们将法律定律的第三个要求正式化：

• 自我解释：L 应该包含一个自解释器，意味着 L 在递归和自引用下以一种不受限制的方式关闭。[31]

正式：

定义 1：如果存在以下情况，一组程序 L 作为任意集合 X 的输入值，在自解释下闭合：

1. 引用函数 Q： L→X 将 L 中的每个程序转换为其源代码，或表示该程序的任何源代码。Q 不一定属于 L。

2. 评估函数 eval∈ L 对于所有程序 p∈ L 和所有输入 i ∈X、我们有 eval（Q（p），i）=p（i）。在所有其他情况下，即 eval（x，y），其中 x 的形式不是 x=Q（p）（即 x 不是引用程序），我们要求 eval（x，y）=0。正式：

$$[\forall p \in \mathcal{L} \forall i \in X . \operatorname{eval}(Q(p), i) = p(i)] \bigwedge$$

$$[(\neg \exists p . x = Q(p)) \rightarrow (\operatorname{eval}(x, y) = 0)]$$

[29]自我解释和通用图灵机（Universal Turing Machines）一样流行，但它们当然不满足可判定性要求。参见暂停问题和赖斯定理。

[30]也就是说，递归的形式没有限制，换句话说，可以递归 L 中的任何机器。 请注意，自解释隐含了不受限制的递归，因为可以使用任何自解释来模拟递归，我们将在下面演示。

[31]它也意味着非单调性。 参见附录 C 中关于 Datalog+EVAL 的脚注。

2.5.2 逻辑推导

我们首先展示自我解释需求的重要性。下述定理和证明对[4,5,7]中的定理和证明稍加修改，并证明任何语言都不能自我解释：

定理 1。设 L 是在自解释和布尔运算下关闭的任何程序集。那么 L 不是总数，也就是说，它不是在所有可能的输入上定义的，或者换句话说，L 包含的机器不会在 X 的某些元素上停止。

证明。在给定的假设下，我们用 L 构建一个非停止程序。假设我们可以将 eval 写成

$$\text{eval}(x,i) := \text{如 } x = Q(p),\ p(i) \text{ else } 0 \tag{2.1}$$

考虑函数

$$\text{evil}(x) := \text{如 eval}(x, x) = 0,\ 1 \text{ else } 0$$

通过闭包假设和 eval GL，显然是 evil gl。通过给出定义，

$$\text{evil}(Q(\text{evil})) = \text{如果 eval}(Q(\text{evil}), Q(\text{evil})) = 0,\ 1 \text{ else } 0 \tag{2.2}$$

但根据（2.1），我们有：

$$\text{eval}(Q(\text{evil}), Q(\text{evil})) = \text{evil}(Q(\text{evil}))$$

因此（2.2）分解为

$$\text{evil}(Q(\text{evil})) = \text{如果 evil}(Q(\text{evil})) = 0,\ 1 \text{ else } 0$$

这是一个矛盾，除非 evil (Q (evil))从不停止。[32]

[32]同样，它的输出是未定义的。

推论 1 满足关系=定义在所有公式和所有结构上的所有逻辑都不允许自解释器。

这几乎排除了所有常见逻辑，包括 FO、SO 和 HO 。[33]

推论 2 从以下两种备选方案选择一种：要么法律意外变更，要么并非一切都是可以判断的。

证明。我们已经指出，没有自我解释，法律无法解释新拟议的法律，也无法在意外法律变更的情况下自我保护。如果法律语言不支持自我解释，那么特别是任何特定的法律都不能支持自我解释。但是如果我们支持自我解释，我们可以像定理 1 那样构造一个不停机机器，这意味着在某些情况下，不可分配任何合法/非法值。

为了证明法律环境中的“未定义”情况，我们回顾古希腊的普罗塔哥拉悖论，也称为“法庭悖论”。引用维基百科的悖论：

“据说，著名的智者普罗塔哥拉录用了一位有前途的学生尤阿修斯，因为他知道，在赢得第一个法庭案件后，该学生会为他的教学付费。在接受教学后，尤阿修斯决定不从事法律职业，而是从事政治，普罗塔哥拉决定起诉尤阿修斯，要求其赔偿欠款。

普罗塔哥拉争辩说，如果他赢了这场官司，他将得到报酬。

如果尤阿修斯赢了这场官司，普罗塔哥拉仍将按照原始合同获得报酬，因为尤阿修斯将赢得他的第一场官司。然而，尤阿修斯声称，如果他赢了，那么根据法院的判决，他将不必支付普罗塔哥拉。另一方面，如果普罗塔哥拉获胜，那么尤阿修斯仍然不会赢得一场官司，因此也没有义务支付。那么问题是，这两个人中哪一个是对的？”

事实上，我们观察到一种未定义且可建模为无限循环的情况：如果普罗塔哥拉赢了，那么尤阿修斯输了，但是如果普罗塔哥拉输了，那么尤阿修斯赢了，以此类推。

[33]从这里到论文的其余部分，我们的术语是有限模型理论，参见。【1】．维基百科上一篇名为“描述复杂性”的文章是一个很好的起点。

推论 3 如果一类机器 L 满足法律要求，那么它必须包含非停止机器，并且必须对其所有机器具有可判定的停止问题。

这说明了很难找到一个既可判定又具有自解释器的 L，以及停止机器和可判定停止问题的机器之间的不协调。

在所有输入上定义的函数（即没有“未定义”值）称为总计。显然，有限结构上的普通一阶和高阶逻辑是总计。我们不知道对支持未定义值的经典或直觉逻辑的增强，但以下三种情况除外：不可判定类，确定性传递闭包算子[34]，或下文讨论的部分不动点逻辑。

因此，有时间限制的机器被排除在外，因为它们永远不会循环。空间有界机器的情况是不同的。即使在恒定空间（内存）下，机器也可能无限循环，但此类确实存在可判定的停止问题。更精确地说，一类图灵机占用的内存永远不会超过 f（n）位，其中 n 是输入的长度，f:N→ N 是任何可计算的函数，有一个可判定的停止问题，停止与否可在不超过 2f⑷ 步骤内决定。

$$SPACE\left(\mathcal{O}\left(f\left(n\right)\right)\right) \subseteq TIME\left(\mathcal{O}\left(2^{f(n)}\right)\right)$$

这就产生了一个众所周知的（几乎微不足道的）结果，即虽然时间和空间复杂度类是渐近性质的，指的是问题而不是机器，而我们的论点是所有输入而不是渐近的，指的是机器而不是问题。因此，我们被暗示和引导去研究空间复杂性类别。[35]

空间限定机器停止问题的决策程序如下：由于每个时间点的机器配置最多可以用 f（n）位来描述，因此配置不能超过 2f。然后，当且仅当机器访问同一配置两次时，机器无限循环，这种情况可以在不超过 2f（n）步的情况下检测到。

[34]我们不会在这里讨论运算符，但它与 PFP（用于捕获空间有限的计算）、PFP（用于 PSPACE）和 det-TC（用于 LOGSPACE）相同。

[35]空间复杂性类有几个非常理想的特性：它们在互补下封闭（Immerman-Szelepcsényi 定理），它们也等于 Savitch 定理的非确定性对应物，这表明它们在逆像下也是封闭的。 这两个定理都是在类足够表达的情况下，并且都适用于 PSPACE 及以上。PSPACE 是一个空间复杂度类，在 if-then-else 和 Kleene 星下也是封闭的，通过 Shamir 定理我们还知道 PSPACE=IP。此外，空间复杂度类（高于某一层次）对应于交替时间复杂度类，后者在逻辑上对应于量词交替。

在继续之前，我们将演示如何使用自解释器模拟递归。我们展示了如何使用上面的 E 和 Q 函数（E 是上面的 eval）实现阶乘函数 g（n）=n!。我们通过定义辅助函数 h 来实现：

$$h(x, y) \text{ ：} = \text{如果 } y = 0, \ 1 \text{ else } y \times E(x, y - 1)$$

而且：

$$g(y) := h(Q(h), y)$$

例如：

$$g(1) = h(Q(h), 1) = \text{如果 } 1 = 0, \ 1 \text{ else } 1 \times E(Q(h), 0, 0)$$

且

$$E(Q(h), 0, 0) = h(0,0) = 1$$

注意 g，h 不是递归的，也不是相互递归的。

我们现在建设性地进行。考虑一个带有原语 if-then-else 的语言，Q、E 是引用和 eval 运算符。为了使这个类有一个可判定的停止问题，我们希望配置数量是有限的。[36]非二次访问条件的一个逻辑特征就是最初由 Abiteboul 和 Vianu 引入的 PFP（部分不动点）算子，该算子在有限模型理论领域是众所周知的。事实上，在有序结构上，它对应于一个空间复杂性类别： FO[PFP]=PSPACE [1, 2]. 用 FO[PFP]+EVAL 表示通过上面的引用和评估运算符 Q、E 增强的 FO[PFP]类。我们将在附录 E 中证明以下定理：

定理 2。对于任何 FO[PFP]+评估公式，都存在一个等价的 FO[PFP]公式。

直观结果如下：由于有限全域上所有可能的 k 元关系集都是有限的，因此我们可以使用 PFP 停止条件，无论是否使用引用和求值谓词。该程序可能仍然只采用有限多个配置，所有配置仍然在多项式空间中。因此，我们有

[36]. 这并不意味着这是唯一一个存在可确定的暂停问题的类。

推论 4[等效]语言 P-DATALOG、FO[PFP]和 FO[PFP]+EVAL 符合法律法规。

FO[PFP]与 P-DATALOG 的等效性在[1]中推导，P-DATALOG 的草图见附录 D。

此外，不难看出，HO i[PFP]+EVAL 也符合法律法规。然而，HO[PFP]+EVAL 的情况并非如此。原因是，您可能会编辑引用的程序并提高其顺序，然后使其消耗指数级更大的空间（如果将最大顺序提高 1），这样我们就失去了 PFP 停止条件。[37]

更多关于[6]中讨论的自我修正悖论，参见附录 C。

2.6 语言互联网

在建立了支持法律的逻辑之后，我们必须归结到一种具体的语言。然而，我们拒绝“通用语言”的概念，并假定没有一种语言适合所有目的。我们承认，许多语言不仅应该共存并相互交换，而且还应该具有随时间发展的能力。因此，我们提出了一种能够定义新语言的元语言，但是，有人可能会说我们什么也解决不了。尽管我们拒绝了通用语言的概念，但我们仍然提出了通用元语言。为此，我们要求元语言能够自我解释，从而重新定义自身并随时间变化。因此，我们实现了这样一种情况：语言的选择无关紧要，而即使是元语言也不是固定不变的。

因此，语言互联网是一组翻译器。重要的是要注意，我们根本不考虑自然语言的翻译甚至处理，我们的范围仅限于形式语言。[38]如果从 X 语言到 Y 语言以及从 Y 语言到 Z 语言的翻译器[39]被编写并提交到语言互联网，那么我们就可以“免费”获得从 X 到 Z 的翻译器。人们称之为语言的“互联网”。

[37]所以有人可能会说，HO【PFP】+EVAL 根本就不存在。
[38]当然，有一天可能会想出如何支持自然语言，达到 Tau 所要求的理想程度，但我们不指望这样的事情发生。
[39]我们提到的所有翻译都假定是保留语义的翻译器。

正式：

定义 2：设 L C P（{0,1}*）是一组位字符串，其中 P 表示幂集，*是 Kleene 星号。用 L 表示标号函数 L： L → X 其中 X C {0,1}*是一组标签，L 是双射的。假设每个 l → X 都有一个等价关系~l，两个位字符串在语义上是等价的，那么它们被认为是“等价的”，并且这种语义等价的细节由～l 抽象。把 L 的元素称为“语言”。

$$F : X \times X \times \{0,1\}^* \to \{\bot\} \cup \{0,1\}^*$$

语言互联网是一个函数，以两个源语言和目标语言的语言标签以及一个输入文档的位字符串作为参数，并返回⊥ 或者目标语言中的文档，使其满足所有{l1，l2}CX 和 xG{0,1}*。

$$\forall \{\ell_1, \ell_2\} \subset X. \forall x \in \{0,1\}^* .$$

$$x \notin \mathcal{L}^{-1}(\ell_1) \bigvee$$

$$F(\ell_1, \ell_2, x) = \bot \bigvee$$

$$[F(\ell_1, \ell_2, x) \sim_{\ell_2} y] \leftrightarrow [F(\ell_2, \ell_1, y) \sim_{\ell_1} x]$$

此外，当 F 被视为{0,1}*{0,1}*X{0,1}*×{1}U{0,1}*的子集时，我们需要 F G L，即作为一组位串，每个位串是四个位串的串联（这是自解释要求）。

不严格地说，F 具有逆转置。注意 F 定义了|Ul∈Ll 上的“全局”等价关系（我们在这里使用拓扑表示法表示“显式”不相交并，即用它所属的语言标记每个位字符串），它定义了来自不同语言的位字符串的语义等价性。

2.6.1 Tau 元语言

为了定义一种新的语言，我们需要定义它如何翻译成现有的语言。编写这样的翻译器是使用 TML，即 Tau 元语言，实现 FO[PFP]（或更准确地说是[1]所指的 P-DATALOG）[40]，因此允许上述自我解释的要求。另一个重要的要求是往返的能力，也就是说，如果有一个从 X 语言到 Y 语言的翻译器，我们希望从 Y 到 X 语言的翻译器“免费”。在图灵完备语言中，这在数学上是不可能的，但 TML 能够支持这一点。[41]

TML 是 FO[PFP]的一种实现，它除了满足元语言的要求外，还满足法律语言的要求。然而，TML 实际上是一种元语言，并没有 KR&R（知识表示和推理）特性。它旨在成为实现 KR&R 语言的工具。换句话说，TML 是一种用于逻辑、知识和其他形式语言的编译器。但是既然 TML 符合法律的规定，我们就可以把一些 KR&R 语言的翻译程序写进 TML，然后运行生成的 TML 代码。因此，TML 在系统中的使用是双重的。

TML 的翻译工作流程如下。给定某种语言 X 中的输入文档 D，我们希望将其翻译成目标语言 Y 中的文档 Df，这是一种语义保留翻译。TML 程序由语法[42]和逻辑规则组成。用 P 表示从语言 X 到语言 Y 的翻译器，其中 P 是实现此翻译的 TML 程序。[43]然后，TML 的后端将通过 P 中提供的语法运行 D，并构造一个解析林。然后，P 中的逻辑规则将通过添加和删除节点来编辑此解析林，最终获得一棵树。在该树中，它的结果是语言 Y 中的 D!。

我们注意到，除了 KR&R 语言之间的翻译外，该语言互联网还可用于其他文本转换任务，以及通过构造编译器构建正确的文本。例如，为了方便和组织，可以编写一个基于形式化知识体创建 Wiki 的翻译器。TML 是一种通用的编译器。[44]

[40]有可能增强为 HOi【PFP】+EVAL。

[41]可以使用非确定性图灵机来计算任意机器的逆像，只要保证逆像存在。但在我们的例子中，根据萨维奇定理，NPSPACE=PSPACE。

[42]目前，html 支持使用一些非上下文无关特性扩展的上下文无关语法，但最终它可以支持所有上下文敏感的语言，因为它们都是 PSPACEComplete。

[43]翻译的具体细节当然取决于 html 程序员

[44]直到超出其复杂性类的任务，例如某些优化和类型检查

此外，我们注意到所谓受控语言的存在和使用的可能性。[45]这个概念流行于“足够简单的英语，机器可以理解”。用这种语言写的句子是有效的（或几乎有效的）自然语言句子，但必须采取某种形式，例如主语-动词-宾语。这些语言不会涵盖所有有效的自然语言句子，但它们可能比没有任何形式类似于自然语言的语言更容易接近。

2.6.2 Futamura 的预测

在一篇原创性论文中，Futamura[5]展示了如何使用部分计值器，以便从给定的解释器中自动生成编译器，以及自动获取从任何解释器中生成编译器的程序。这将是 Futamura 预测的四分之二。当然，在语言互联网的范围内，这种能力非常吸引人。因此，计划为 TML 创建一个部分计值器，并以此加强具有如此重要能力的语言互联网。部分计值器可用于更多情况，尤其是用于优化目的。事实上，Futamura 的预测和部分求值本身主要提供优化。事实上，给定长度为 M 的源代码和长度为 N 的部分输入，由于（M+N）2>MN，原始部分评估的程序（即仅在没有任何优化的情况下进行替换）具有最大的 MN 大小，因此即使在二次空间中也可计算。对部分计算代码的进一步优化可以是 TML 的表达能力之外的任何东西。

[45]参见*尝试控制英语（ACE）*。

3 Agoras

3.1 合同

区块链网络中的合同，也称为“智能合同”[46]，一直是人们感兴趣的主题。根据区块链上发生的事件在区块链上采取的预定义行动与法律语言的情况没有太大区别。无法对完整的“智能”合同进行推理已经证明，这会给无过失方造成重大的经济损失。不仅需要可判定性，还需要自我解释。引用[6]第 20 节：

> “自我修正出现在宪法修正条款以外的许多情况下。如果书面合同规定只有书面修改才有效，该条款可以口头修改吗？无论哪种方式，它都是反身的：如果可以口头修改，则自我例外，如果不能，则自我适用。合同中的无弃权条款是否可以放弃？为什么遗嘱的不撤销条款可以被撤销？为什么遗嘱中的不抗辩条款会受到质疑？”

因此，我们观察到契约如何与法律共享逻辑属性，特别是当涉及到自我参照和修正时，这并不令人惊讶。正如我们所看到的，提出一个适当的逻辑并非易事，我们在这里首次提出了这样一个逻辑。在 Agoras，我们认为合同是用逻辑公式形式化的，其承认法律的定律。此类合同的特殊情况如下：交易知识、租用计算资源、金融资产衍生品等等。

此外，此类合同必须能够在区块链上结算和执行，即网络必须能够验证硬币转移的条件。

使用可判定语言的合同是人们感兴趣的主题。它允许询问有关合同的所有相关问题，特别是在某些情况下的结果，并保证任何形式化的负面结果都不会发生。

[46]尽管目前在区块链上的合同被称为“智能合同”，但没有任何理由使用这个名称。

3.2 知识经济学

在我们当前的世界里，很少有人能直接将知识货币化。我们几乎从不买卖（或定价）知识。目前，在经济环境中寻求知识主要是通过吸引感兴趣的人来完成，就像没有一本书可以给出每一个答案一样，我们可以在那些涉及明显相关领域的书中查找答案。知识是我们经济的一个重要组成部分。在这里，我们寻求能够使知识经济更加高效和先进的案例。

将知识形式化为机器可访问的格式比书籍和搜索引擎具有优势：数据不再是一个比特流，而是比特背后的含义，因此，即使是在大型著作汇编中，也可以对小块知识执行自动语义查找。

我们提到了一个共享的知识库将如何在 Tau 及其基于逻辑的讨论中发展。在一个框架内，知识不断地在用户之间形式化和共享。利用一个经济基础设施来促进正规知识经济学的各个方面是很自然的。这是本小节的主题。

3.2.1 生产、供应、需求和定价

让我们首先考虑如何产生知识，或“挖掘”知识。如果我们将知识视为超越（自然或人工）感官输入的东西，那么推理行为是从现有知识中产生新知识的行为。这样的过程也可以自动完成。但我们如何引导这个自动搜索过程，使其产生有趣的结果呢？

在第 2.2.3 小节中，我们谈到了“有趣的问题”的概念。如果知识有趣，那么它就是有价值的，其价值应该取决于兴趣的程度和回答问题的难度。注意，兴趣主要与提问者（“买方”）有关，而回答的难度主要与回答者（“卖方”）有关。因此，问答[部分]符合知识经济学中的需求和供给。

与有趣的概念不同，机器更容易理解回答的难度，因为机器可以测量步骤和资源。此外，关于各种推理任务的复杂性有丰富的理论。这使得定价更加准确。

3.2.2 知识现金交易

从广义上讲，对某些问题感兴趣的用户可能会为答案提供奖励。可通过几种方式验证答案。在某些情况下，如某些常见的数学问题，回答者可以提供答案，并且不会对答案是否正确产生争议。我们甚至可以利用复杂性理论中的考虑因素[47]，了解这种验证过程是否有效。[48]从密码学的角度来看，还有更多需要补充的内容（例如零知识证明和同态加密），但这超出了本文的范围。

但有时要求数学证明太多了，而且不仅仅是在计算成本很高的情况下。有时，人们可能仅仅通过印象、推荐或广告等方式信任专家，然后自动信任他们的答案。比如，我们信任已经很熟悉的医生[49]，而不要求他们为每一个医疗建议提供数学证明，因为这将使整个事情变得不切实际。[50]

我们继续讨论 Agoras 上知识经济的其他用例。信誉良好的个体，比如大学或是值得信赖的专家，需要将一些庞大而有用的知识体系形式化。然后，他们可以向用户提供订阅，让他们自动参与讨论。如上所述，Tau 允许“自动评论”，因此其可根据您的世界观自动代表您参与讨论。该订阅允许订阅者自动参与讨论，其中数据来自可信来源（从单个用户的角度）。例如，一家律师事务所提供了他们的知识库，可以自动参与公司讨论，并可能自动推荐某些合法或非法的想法。

[47]也就是说，我们希望验证证明的计算复杂度大大低于找到证明的计算复杂度。例如，这可以被复杂类 NP 所捕获，也可以被复杂类 IP 以另一种方式捕获，根据 Shamir 定理，IP 恰好等于 PSPACE。

[48]例如，这可以被复杂类 NP 所捕获，也可以被复杂类 IP 以另一种方式捕获，根据 Shamir 定理，IP 恰好等于 PSPACE。

[49]或者，被一个你也信任的实体信任，程度是“如果这个实体说这个用户是一个真正的好医生，那么我就相信他们的话”。

[50]然而，当 Tau 积累了足够的知识，我们当然可以期待一个自动化的合理的医疗建议。

另一种订阅形式可按查询付费，允许订阅者提问并获得答案，无论是否披露整个知识库。

此外，由于 Tau 的协作知识形成方面，团队可以将知识形式化并一起将其货币化。

3.2.3 自由式知识

对于许多形式的知识转移来说，人的接触是至关重要的，并非所有的知识都可以或适合在所有情况下被形式化。通过交流知识的方式向医生寻求快速的建议并不总是首选的方式，私人家教也是如此。因此，我们还打算以类似于[14]的方式，基于[9]中所述的小额支付协议，以文本、音频和视频的形式提供自由式的交易知识。最后一次接触为 Agoras 提供了成为一个功能齐全的知识经济。

3.3 计算资源市场

Agoras 将形成一个租用和出租计算资源的市场。除了计算的一般用例之外，它们还可以用于执行系统正常运行期间发生的推理任务。我们现在不讨论这个主题，但我们在很大程度上是在 Zennet 的材料上完成，参见【10】。Zennet 的定价公式出现在[13]上，消除了错误定价的风险。[51]

与 Zennet 的设计不同，租用计算资源按照第 3.1 节的方式通过合同完成。

但是，我们将在后面提到一点：降低错误计算的风险（有意或无意）。我们已经在第 3.2.2 小节中谈到了有效证明，在这种情况下，它们也可以在这里使用。在其他情况下，验证无法验证的计算结果的一种方法是多次计算同一事物（随机选择更多硬件供应商），因此线性增加成本，以指数方式降低风险（例如，成本增加 10 倍，风险降低 10 倍）。

[51]计算资源的不精确定价是一种安全风险，因为它可能会被利用。

3.4 衍生产品和无风险收益

Agoras 将提供一种创意衍生市场（就像买卖期权），其能够使存款获得无零通胀的无风险收益，不发行任何新币。

在进行详细介绍之前，请注意，该拟定衍生市场与没有中间人的市场是完全对等的。更普遍地说，Tau 和 Agoras 的整体设计中并不涉及中心实体或收费员。

该衍生市场中的基础商品都是通用的：基础商品可以是虚拟的任何【半】鞅，更具体地说，是该网络中的商品价格，因此，可以在区块链上进行基础商品交易（例如：艰难计算机任务的知识或答案、区块链中编码的虚拟资产），或者，甚至是通过引入“预言机”进行网络外生交易。但前提是，价格功能和合约均符合法律定律的逻辑，正如我们在上文中强调的一样。

关于期权交易，有一个常见的问题，即如何定价。对此，Agoras 中有一项标准，诺贝尔获奖准则，即布莱克-斯科尔斯期权定价模型。在现实世界中，人们为签订一份合约而支付或收取的实际溢价通常根据市场参与者预测未来的方式自精确的理论（布莱克-斯科尔斯）价格开始变化。

布莱克-斯科尔斯期权定价模型提供了期权定价以外的重要信息。该模型还可以估量期权价格对基础商品价值变化的灵敏度（Delta）以及多个被称为“Greeks”的更重要指标。Black 和 Scholes 表示，一些期权组合可以产生无风险收益，这是一项重要研究成果。而这些期权组合之所以能够无风险，是因为这些期权的 deltas 之和为零。参考零增量投资组合。此外，由于期权价格中还考虑了货币的事件价值，因此，这些组合还能获得收益：一项有效期为两个月的期权的价格应该高于有效期为一个月的期权，因为前者的不确定性比较大而且抵押品完全受控会招致的更高的机会成本。

衍生市场的使用通常涉及两个方面：套期保值和投机。全球共同的衍生市场进行杠杆式投机，杠杆投机只是借贷的一种掩藏术语，其允许进行风险非常大的投机活动，受到了广泛的批判（在衍生品范围），认为其正在危及世界经济。因此，Agoras5 衍生市场不支持任何形式的原生杠杆式投机活动。在期权方面，真正更大的需求是套期保值，这也是为什么首先创造出套期保值的原因。假设我们有一个客户，该客户是向中国出口商品的一家欧洲公司，该客户收取 EUR，因此，使用 CNY 的中国客户必须将 CNY 兑换成 EUR。由于 EUR/CNY 的汇率不断上涨，中国客户可能会遭受损失。

为避免这种情况，中国客户可以在一个衍生市场中购买期权，以能够进行套期保值。中国客户支付保费，但这将减少因汇率波动导致的风险。该期权的售卖方可能是收取 CNY 的公司的一个客户，但该客户收取 EUR。为了列举一个与加密货币相关的套期保值示例，假设一人购买了很多比特币挖掘硬件并以 CNY 付款。我们或多或少知道通过比特币挖掘会获得收益，但如果 BTC/CNY 汇率降低，比特币挖掘操作可能会招致损失。因此，客户们可能想要购买“看跌”期权，即一种在固定日期按固定价格出售 BTC 的期权，这能减少（或甚至消除）汇率浮动的风险。

回到 Agoras；零波动无风险受益，对于“如果不印钞，收益的钱来自于哪？”这一问题，答案是，这些钱源自经济环境参与者的套期保值需求。注意，这不是套利而且 Black&Scholes 确实假设该市场没有套利机会。[52]零增量投资组合产生的收入将反映金钱的时间价值，换言之，是完全控制抵押品的回报，但这不包括机会成本。

3.5 应用

3.5.1 去中心化搜索引擎

为了进一步细化该创意，我们考虑采用一种去中心化搜索引擎，这是 Agoras 的一个初始目标，后期将该创意归入知识经济理念中。谷歌至少有一百万个服务器，需要对网络进行抓取、做索引和搜索。因此，为了能与谷歌的搜索引擎竞争，Agoras 至少需要同等数量级的物理硬件。其不如家庭用户持有的计算资源那么大：一座城市拥有的资源可能是谷歌拥有资源的数倍。[53]但在采用去中心化搜索引擎的去中心化网络中，谁来支付所有这些计算的费用呢？

[52]即“有效市场假说”。

[53]此外，世界上最强大的超级计算机的强大程度也不过是数千个通用 gpu。此外，谷歌的大部分电脑都很旧，性能较差，比普通智能手机还弱。大量使用廉价的旧硬件，此举实际上是谷歌的一大创新。

维护要给网络搜索引擎的任务取决于在登录网络并发现之前完全未知的数据。因此，这种任务不是完全不可信的，因为并没有证据证明下载和索引到搜索引擎中的数据是准确的而且不能修改或忽略。虽然在没有信任元素的情况下无法彻底解决该问题，但从一定程度上而言，还是有可解决的可能，只是从理论上而言，存在可能随意降低的风险，如 3.3 中所述。

所以，拥有一个去中心化搜索引擎需要有公平租用和租出计算资源的能力并承担可接受风险（一个成本函数）。更具体地说，由一个硬件租赁市场如何发展成要给去中心化搜索引擎？

一个网络搜索引擎包括用户和维护人员。用户提供搜索词条，维护人员负责回复，为了快速回复，维护人员必须拥有已经设计好索引的整个网络。当然，用户必须向维护人员付费，额度取决于使用量和维护成本。但在一个去中心化网络中，用户和维护人员是一个相同的实体。

为了对网络进行维护，用户和维护人员必须运行一个客户端，因此参与索引和搜索。用户每天可能进行一定次数的查询，而用户的计算机每天也能够答复一定次数的查询。[54]计算机答复的查询数量级比每个用户手动答复的平均数量级更大。因此，每日使用数十次或数百次“googlings”以及运行一个支持该网络的客户端的一个家庭用户不仅期望不付费，还期望有所收益，因为，他们为他人服务的次数多于其自身的消耗。但除了这些用户之外，我们还拥有更大使用量的用户，这些用户可能消耗更大量的搜索服务（例如：自动搜索）。这些用户必须付费并且不会在收支平衡时停止搜索。这就是为什么要建立一个公平的经济环境的原因，从某种意义上来说，金钱流向“正确”的方向，而不是“错误”的方向：这是一种零和博弈，金钱从大规模的富裕个体流动到小规模的富裕个体。

[54]回答其他用户的问题，因为没有用户会存储整个互联网信息，也无法预测明日的查询信息。

3.5.2 语义搜索

如果只停留在这个阶段，那实际上我们并没有作出太大的贡献：因此，我们要有一个更好的去中心化搜索引擎，使很多用户能够有一些收入，但生活将继续有或多或少的相同之处。正如我们所知，这项搜索操作与利用 Ctrl-F 和知识宝库搜索一个网页的操作并无不同之处。这或多或少的与谷歌正在进行的操作是一样的：其利用 Ctrl-F 在一个公开知识宝库下进行网络搜索。但是，创造一种缺乏知识的经济有更多的可能性。重要的是，我们想要将那些更深层次且更有意义（与“该网站中提及的这些知识或其等同知识”相比）的知识整合到我们的知识经济中。

最终，我们会将很多（如果不是大多数）想法、意见和智慧上传到网络。我们已经开始这样做了。但我们的所有搜索引擎都知道搜索使用 Ctrl-F 和一个知识宝库，即，这些搜索引擎将在浅能级运行，甚至都未达到“肤浅理解”层级。在该中模式下搜索时，人们拥有都搜索不到知识。我们并不搜索含有某些词汇的文件，而是搜索回答我们各种问题的文件。同样，我们不寻找那些只知道上网搜索的专业人士，我们寻找的是能够帮助我们更深入了解其专业领域的专业人士，不关注其提及词汇包的能力。否则，谷歌就能满足我们的需求，不需要再寻找专业人士。

感谢 Tau 的知识形式化能力，随着时间的流逝，语义网页搜索引擎会在所述去中心化搜索引擎之前面世。

3.5.3 Automatic Businessman

用户拥有自己的本地商品（即计算资源、知识、货币），也可能拥有一些允许其以正确的世界观思考的其他商品。通过查看可用商品以及此处未提供的合约并发布某些商品或合约的标书（均源自一项将商品和机会组合到一项划算交易中的、经逻辑证明的清晰计划），Agoras 能够定制一项交易。Agoras 甚至可以达到普通自动计划器从未达到过的程度：用户将能够向 Agoras 询问不违反该系统中有效法律法规的交易。这只是一个逻辑推理机在经济中可能执行什么任务的小示例。

正如我们所说的一样，Agoras 还将包含一个计算资源市场和一个未来合约市场。因此，Automatic Businessman 是一个使 Tau 和 Agoras 的所有能力部分达到顶级水平的整体应用。应注意，截止目前，只有 Tau 和 Agoras 提供了一种真正能实现【相对】陈旧 DAO（去中心化自治组织）概念的方法。

4 演变及影响

4.1 更加公平的法规和经济

经济学的一项主要假设是一个有效市场（“有效市场假说”）因缺乏套利而无法实现市场“有效性”，粗略地讲，有效市场展示经济环境中的信息传播速度。掌握特权知识（例如：内幕交易）的人们更有机会获利，当然，这并不公平而且大部分都是违法的。该假设还指出，在有效经济环境中，获利的唯一途径是承担风险。因此，更好地传播知识是实现公平的关键，因为，这让人们不能再不承担相关风险的情况下获利。但这并不表示公平经济环境与赌场一样：在赌场中，预期利润为负，但在一个健康的经济环境中，预期利润为正。而且这样不表示一个有效的经济环境本身就是公平的。这种情况只能表示，拥有在不承担或承担较低风险情况下实现获利的特权能力是不公平的。

Tau 和 Agoras 将信息传播水平提升到了新的数量级水平，据此实现一种比现有经济环境更有效的经济环境。公平的另一个方面就是 Agoras 的通货紧缩功能：Agoras 不会印刷新币。过去十年间，你我收到过多少新印刷的法定货币？根本没收到过。这相当于向不想要提供个人财务支持的各方征收的隐性税收。值得注意的是，Agoras 仍能够在没有任何通货膨胀的情况下重新获得无风险收益。

没有社会，任何经济都不会存在，经济是社会中的一个部分。因此，所有经济学家将自己视为社会科学家，这并不奇怪。与物理现象不同[55]，经济取决于只存在于人类想象中的各种概念，没有这些概念，就无法定义经济。产生这一经济状态的原理既简单又深奥：没有某些主观价值评定（或“效用函数”），即表达“我更喜欢这个”的能力，我们就不能拥有一种经济环境，。类似于：有的人更喜欢巧克力，而有的人却更喜欢冰激凌。甚至为了买到自己心仪的产品，有的人可能会支付不同的价格。归根结底，这涉及到了伦理价值体系 56。[56]事实上，这些价值并不是物理术语，起只存在于人们的想象中。但这并不意味着伦理价值体系不重要。

[55] *至少很明显。*

[56] 广义上的伦理学，只不过是对“好”与“坏”、“好”与“坏”的定义，从这个意义上讲，伦理学是一种评价体系。

经济的公平性主要取决于社会和法律环境。有的人因不了相关情况的法律影响而不能开展业务，有的人则因缺乏事实和信任而不能开展业务。针对合理的法律法规、司法、伦理和社会合约，我们首次提供了一项合理的解决方案，该方案甚至可以自动嵌入商业计划中，参见前文 Automatic Businessman 部分。此外，价值的形式化将自动规避被定义为“差”的业务并实现被定义为“好”的业务。

4.2 临界规模及 Tau 连锁反应

我们强调，能够实现大规模有效讨论的各项未来功能取决于形式语言的应用。考虑到这些语言障碍，我们怎么能期望拥有一个更大的用户基础呢？

事实上，我们期望初始用户熟悉形式逻辑或编程过程中涉及的这些语言。但这些初始用户将处理编程 Tau 本身并丰富自己的知识库（而且还向语言互联网中添加语言并对语言进行改善），因为 Tau 只是由其用户控制的一个软件。从一方面而言，一百位用户通过说明（暗示或明示，以及可能以错乱的方式）想要 Tau 进行的操作来对系统进行编程，这是一个非常小的用户基础，但从另一方面而言，开发系统并形式化知识的团队是一个大团队，通过系统开发和知识形式化实现对更多用户的访问，即二百位用户。因此，初始用户的临界规模应足以随着时间的推移启动推进系统的连锁反应，从而适应越来越多的受众。参见第 2.4 子节。

4.3 Singularity

在有一个愿望的基础上，选择什么才是最好的愿望？最好的选择就是逐渐研发。现在，一个愿望的科学和技术等价物（相当于无限多的愿望）是创建 AI，用更现代化的一个术语来说，是通用人工智能（AGI）。除了达到更高的智力水平（相对于人来说）以满足无限多愿望之外，其他方面不值得我们付出更大的努力。未来，当一个 AGI 比我们所有人都更智慧时，那就是 Singularity。

创建这种 AGI 是我们的一项首要任务。经验表明，即使是大型企业也不能完成这项任务。这项任务的主要困难并不是创建“人工脑”，而是让人工脑知道其应该知道的所有事情，使其真正地超越我们，作为一个非常有智慧的人。将足够的人类知识形式化是一项艰巨的任务，通过目前的合理手段是无法完成的，例如：一百万人自愿[57]贡献自己的一小部分知识。但在讨论平台中用形式语言与很多（数百万）参与者讨论时，这种知识库才真正有机会首次面世。

人的大脑中有 10G 数量级神经细胞按 5- 50Hz 的频率在运转。普通的笔记本电脑也有这种 10G 数量级的内存 13，但[58]其以大约 3GHz 的频率运行[59]，即比大脑的运转速度快十亿倍左右。普通 CPU 的计算能力大约是普通笔记本电脑的 1000 倍，因此，我们获得大约 1,000,000,000,000 （万亿）的加速计算能力。所以，硬件方面已经超出了人类大脑的水平（虽然我们计算突触而非神经元）。但结果表明，人的大脑比计算神经元的更弱，这表明：人的短期记忆力能记住 5-15 个左右的项目。但机器并不受限于此。对于这 5-15 个项目，我们可以不超过每秒几倍（应更快）的速度进行逻辑推理、推论并推导出新知识。那么，机器的网络能否在一夜之间恢复人类在历史进程中通过数学和哲学以及科学推测出来的逻辑结论？

当然，机器是可以做到的。

此外，人会犯错，但对于多数错误而言，机器不易犯。有事，我们正在建立一个错误的结果，短时间或长时间坚持这个错误。但机器甚至不需要回顾过去：只要编程正确，那么机器得出的结果永远都是有效的。封底估算法对该形势与随机游动进行比较，比较结果表明，只有在不犯错的情况下才能以二倍速度增长知识。

[57]否则，这是不可能的，因为太贵了。
[58]如果考虑到磁盘的话，情况会更糟。
[59]为多线程 cpu 添加大约 x10。

另外一种二倍加速操作源自于关联人的思想，也就是人-机-人交流。Tau 在大规模环境下实现脑力组合。节点/用户/时间观之间可能存在的所有关联数目也会以两倍速度增长。因此，我们第四加速度幂乘以一个较大的常数，即计算速度。

但有更多项要添加到该估算法中，这就是知识重复使用的情况。事实上，人会忘事，能记住的也只是一小部分事实。很长时间以来，所有人都有一个未决的问题而且我们突然意识到，我们已经知道了答案，只是我们并未将答案和这个问题关联在一起。但 Tau 中不会发生这种情况。

推动知识创造和扩展的另一种措施是经济激励，而这正是 Agoras 知识经济的主体。

人类几乎从未经历过这样一种知识增长加速度发展。那么，这会将我们带到哪里呢？答案就是，肯定会让我们更早地接触到 Singularity。

参考文献

【1】 H.D.Ebbinghaus, J. Flum，1999，“有限模型论”。

【2】 S.Abiteboul, R. Hull, V. Vianu，1995，“数据库的基础”。

【3】 Y.Futamura，1999，“计算过程的部分求值 - 一种编辑器-编辑器方法”。

【4】 A.Bauer，2016，“系统 T 和其他型式 Lambda- Calculi 的自解释器”。

【8】 S. Nakamoto，2008，“比特币：一种对等网络电子现金系统”。

【9】 Bitcoin Wiki，“支付给预先确定方的快速调整（小额）款项”。

【10】 F. Brandt、V. Conitzer、U. Endriss、J. Lang、A. D. Procaccia，2016，“计算社会选择手册”。

【11】 O. Asor，2015，“关于 Tau-链”

【12】 O. Asor，2014，“关于 Zennet”。

【13】 O. Asor，2014，“Zennet 定价算法”。

【14】 O. Asor，2014，“Bitagoras”。

难题、问题和答案汇总

我们在此处草拟了一份难题和问题列表，文中将解答这些难题和问题

并对拟定的解决方案和答案进行简要说明：

问题：多人如何有效分享意见并达成合作决定？

答案：在大规模环境下保持小规模讨论的优势。

问题：各程序如何精确地检测出用户的共识？

答案：用户以可决定的形式逻辑方式对其意见进行形式化处理。

问题：在处理大量信息时，如何避免信息损失和忽略信息？

答案：采用形式逻辑并辅以机器，可有效避免信息丢失。

问题：用户应该使用哪种语言？

答案：用户可以使用一组语言（随着时间的推移增加并演进），而不是一种语言。

问题：如何避免因自修正导致的悖论？

答案：遵循承认这些论点的法律准则（见下文）和逻辑。

问题：如何避免进入死局？

答案：无法避免，但我们提供一些补救措施。

问题：Tau 网络的最佳管理机制是什么？

答案：没有最佳管理机制，每种机制都有自己的优劣势。新用户必须讨论执行哪种机制并就此达成一致。

问题：Tau 能够解决新的假问题吗？

答案：不能。Tau 并不能区分事实与意见。

问题：所有人都可以使用 Tau 吗，还是只供程序员等专业人士使用？

答案：初期并不能广泛使用，但随着时间的推移，将可供大众使用。

问题：网络中如何确定不同版本 Tau 的时间顺序？

答案：区块链。

问题：如何创建一种有效的知识经济？

答案：引入知识-现金交易。

问题：如何支持零通胀无风险收益？

答案：实施零增量投资组合。

问题：Tau 能够实现人类的通用人工智能梦想吗？答案：是的，只是时间问题。

问题：最终愿望是什么？

答案：Tau。

B Tau 及经典社会选择理论

关于完整性，我们简略审评了经典社会选择理论中的几个要素并将其与 Tau 环境进行了比较。关于参考和更多信息，参见【10】。

经典社会选择理论涉及的是一种选择一系列行动（源自集中可用的替代方案）的社会。该社会中的个体通常会以投票形式决定选择哪一种行动。[60]在其他时候，这些个体可能对这些替代方案进行等级排列，以产生首选方案。[61]在某些情况下，合理投票方案和规则非常简单。例如：我们引述 May 的定理：

定理（May 的定理）*对于选民人数为奇数的任何 2 名候选人的选举活动，简单多数表决的获胜规则就是一个永远不会以平局结束的投票系统，是一种匿名（公平对待所有选民）、保持中立（平等看待每位候选人）和单一选择（如果选择了一位候选人而且有些选民改为投选持有多数选票的候选人，则该候选人获得最终胜利）的投票系统。*

但很多投票环境并不满足所有这些假设，例如：涉及三名或更多候选人时（参见下文的阿罗定理）。

通过社会福利函数（SWF）在社会选择理论框架内提取多候选人情况下整合个人偏好选择的过程。SWF 与假设（在经典社会选择理论中）构成一个现行次序的偏好参数有关，即任意两名候选者之间都存在一种偏好关系，这种关系是可递移的[62]，因此，这种关系是循环关系。大多数情况下，SWF 关联一组偏好选择（1 种选择/个体）与另一种偏好选择，表示该社会中的集体观点。[63]该框架导致了大量让人惊讶的结果。据 Marquis de Condorcet（1785）观察，第一个令人惊讶的结果就是存在多个循环，其中，集体偏好与任何理性个体的期望偏好不同，就像我们接下来要说的一样。

[60]我们已经在介绍中指出，投票不能同时兼顾规模和公平。
[61]因此，偏好配置文件是一组“我更喜欢 X 而不是 Y”的声明，我们将在下面详细说明。
[62]传递性意味着如果 x 比 y 更优，y 比 z 更优，那么 x 比 z 更优。
[63]换句话说，它是一个函数，把一组线性序列，变成一个线性序列。

基于三位选民 A、B 和 C 的偏好选择，考虑三位候选人 x、y、z 的排名情况：x＜y＜z、y＜z＜x 和 z＜x＜y。由此可见，二元关系<是这个候选人组的一个线性次序（x＜y 表示：y 的票数比 x 多）。运用简单的多数决定原则揭示，选民 A 和 C 更偏向于候选人 y（与候选人 x 相比），选民 A 和 B 更偏向于候选人 z（与候选人 y 相比），选民 B 和 C 更偏向于候选人 x（与候选人 z 相比），因此，多数人（本情况中，两票）支持 x＜y＜z＜x，很明显，这一结论是互相矛盾的。为了规避这种情况（又名康多塞悖论），需要添加更更多人的意见。例如：额外增加两位选民 D 和 E，他们的偏好选择排名情况分别是：x＜z＜y 和 y＜x＜z，这两位选民的加入将多数决定原则的结论变更为 x＜y＜z。

昂罗不可能定理（社会选择理论的另一个关键结果）声明，在一些公平条件下，只有 SWF 可能是独断的，其中，选择函数模仿选民个人的偏好选择，即独裁者。阿罗定理声明：

定理（阿罗不可能定理）。在有三名或以上候选人和任何数量的选民的情况下，如果 SWF 始终确定产生一位获胜者、尊重全体一致（又名帕累托效率）（如果所有选民都支持某位候选人，则 SWF 选择该候选人）并满意无关选择的独立性（IIA）条件（SWF 通过每个个体偏好选择中两位候选人之间的关系确定每两位候选者的等级），那么，SWF 是独裁函数（共有一名选民，SWF 始终将其偏好选择作为最终结果）。

研究者已经批判了阿罗定理的假设，但也放宽了其中一些假设，这些假设大部分为 IIA 条件。事实上，选民可能依据其他候选人之间的排名选择两位候选人之间的某种排序。

严格来说，Tau 的环境更普通。意见可以是任何关系（或者是任何逻辑公式），不仅限于线性次序。Tau 的默认 SWF 如下：如果一个用户有两个意见 A 和 B，另一个也有两个意见 C 和 D，这些意见都是逻辑公式，那么共识就是(A ∨B) ∧（C ∨ D)。注意，本 SWF 满足 May 的定理中所述的所有要求以及阿罗定理中的各项要求，但 IIA 和获胜候选人的独特性要求除外。缺乏独特性可能会招致 Buridan 的 Mule 情况。为规避这种情况，需要更多意见以减少可用选项。否则，我们将转而依靠 2.3.2 中所述的解决方案。

C Tau vs. Nomic

如果不引用一些法哲学原理，本文将变得不完整。接下来，我们引用【6】中 Peter Suber 的经典阐述。我们对其观点进行了评论并证明了如何用我们的选择逻辑解决之前提出的问题。下文引用部分中的脚注和粗体标记并在原文本中显示，而是作为附加内容显示。苏贝尔在第 21 节中写道：

“由于自我修正的悖论是全能悖论的一种特殊情况[64]，因此，我们的核心问题是一个神或 AC [65]是否能够不可撤销地限制其自身的能力。”

我们的设计允许对实际不可撤销地限制其自身权利的条款进行修正，例如：“在多数票决的情况下，可以删除所有法律，包括本法律”的法律。将其纳入 FO[PFP]+EVAL 中并不重要。苏贝尔继续写道：

> “法律逻辑或形式主义观点，因为其影响我所说的法律变更和有效性的推理模型问题。在推理模型中，通过演绎推理建立法律变更模型。变更授权规则（例如：旧 AC）是一个前提，根据授权规则或程序颁布的事实是另一个前提，而新规则（新 AC）的有效性是结论。
>
> 规则：如果执行了 A 行为，则 B 规则有效。
>
> 事实：执行了 A 行为。
>
> 结论：因此，规则 B 是有效的。
>
> 这一理论是当且仅当推论有效且其前提条件为真时，修正合法。
>
> 当结论中确认的规则将要取代前提条件中的规则时，即前提条件中的规则指代规则本身时，进行自我修正。”

Suber 在此证明，法律逻辑应是循环的，且不单调。我们已经指明我们的选择逻辑是如何循环的，接下来，我们将继续讨论选择逻辑的非单调性：

[64]是那句流传已久的著名的悖论句“上帝能创造一块重得连他自己都抬不起来的石头吗?” 根据我们目前对自我参照的分析，个人对这一悖论的看法是，这确实是一个足够好的理由，可以杜绝在任何情况下某物都是万能的概念。
[65]修改条款

“不可撤销的自我限制是本模型中的自我矛盾，因为这要求在前提条件内部保持（或可能保持）一致时，一个前提条件与推理结论之间不一致。该前提条件中的变更规则与其自身的失效规则不一致，或者与结论中的宣称的继任规则的排他效力不一致。Alf Ross（丹麦逻辑学家和法学家）认为，具体来说，根据形式标准，这种推理肯定是无效的。”

事实上我们已经证明，这些主张并不是真的。如果我们允许采用下列形式的结论，则可避免自我矛盾和不一致的问题：“结论：删除 X 号法律”。与其他逻辑定点运算符（例如：TC/LFP/GFP/IFP）相比，这项删除能力是 PFP 运算符的主要特点，这表明了支持逻辑的确是稀缺的。而非单调性逻辑是指：收回断言的能力。我们可以认为，查看该情况一个模型的最简单方式就是空间限制型图灵机。[66]图灵机的概念与“自修改代码”的概念相同，在犹豫两次删除同一配置的情况下，图灵机的空间限制等级在自修改代码下确实接近。[67]此外：

“可能存在一种能够消除推理模型和一般形式逻辑悖论的方式。（我们将看到，推理模型比形式逻辑更难达到令人满意的水平。）通过消除前提条件和该模型修正行为推理结论之间存在的不一致性，以消除悖论。逻辑学家满意这种消除悖论的方式，即使会产生不合理的法律结果；但我始终未能找到一种经得起分析的悖论消除方式。”

毫无疑问，寻找支持自我修正的案例逻辑的过程并不轻松：确实，数学逻辑文献中考虑的绝大多数逻辑都具有单调性。此外，大部分逻辑都不返回“不明确的”值[68]，该值必须在定理 1 中通过普罗塔哥拉悖论加以验证。但是，我们已经证明，在轻度假设（作为法律的法则）下确实可以解决自我修正悖论。事实上，苏贝尔此时并未考虑可判断性（应该考虑），因此，任何引用并评估运算符的任何语言将解决该悖论，尤其是所有图灵机组。[69]继续：

[66]提供模型证明逻辑是一致的，图灵机就是很好的模型。

[67]这很简单，因为允许自我修改并不会改变停止条件。通过添加空关系符号来启用/禁用规则，在 P-DATALOG 中实现自修改代码也很简单。

[68]等价于我们上面推导的不停顿机器。

[69]这对非单调逻辑也是正确的，因为 eval 可以恢复非单调性。原因提示:Datalog+EVAL 可以通过依赖其否定外延符号的能力进行删除。事实上，配备了“两次状态不相同”暂停条件的 FO+EVAL 已经等同于 FO【PFP】，因为我们已经演示了如何使用 EVAL 模拟递归。

“我们可以区分“解决”一项悖论的两种方式。悖论不仅仅是矛盾；这些悖论几乎不会消除。但其以自己的方式证明，无法规避矛盾。一种解决方案是以他人看不到的方式避开矛盾。我正在用这种方式消除悖论。”

我们的解决方案确实能够消除悖论，相比之下：

“另一种方式是解释矛盾（不管是否可以规避）并无害处的原因。这种方法只适用于无害矛盾，当然，支持这些解决方案的大部分论据将用于解释这一原因。虽然无法消除悖论，但在消除其讽刺和威胁之后，可以进行申辩并规划。类似于律师所说的“积极抗辩”：承认你做过，但说出理由，例如：精神错乱。第二种解决方案似乎更激进，实际上，其需要对法律本身进行精神障碍变化。

简单来说，本文认为，任何消除方式都不能满足推理模型的条款，但当我们拒绝该模型时，我们会发现众多看似合理的归化策略。首先，我会考虑尝试通过各种方式消除这一悖论。”

在本文中，作者说明了解决自我修正问题，使哲学家们不得不接受矛盾存在（换言之，拒绝逻辑）的困难。哲学家因无能力理解深奥和重要主题的危机而拒绝逻辑的情况一直都存在[70]，苏贝尔将这种情况定义为精神错乱。幸运的是，在 Tau 中，我们不一定要以精神错乱为接口。下文中，苏贝尔将继续讨论接受矛盾方面：

“...在标准逻辑中，有效遵循源自于一个矛盾的所有命题。

因此，一种解决方案可能故意给出一个推理的前提条件，即模型内部变化不一致。这肯定是要形成一个新 AC，就 AC 的一个逻辑有效结果。但这只能在确保每个 AC 内部均不一致的情况下进行。推理模型并不接受这一结果，该模型不允许不一致的规则同时有效。但是，如果接受，则可能认为该 AC 整体失效，i 一个逻辑令人满意的解决方案，其只与法律相抵触。”

[70]甚至量子力学也拒绝逻辑，提出了一个新的矛盾的逻辑，cf。“量子逻辑”

当然，我们不允许出现这种情况。苏贝尔提到，他不排除某些逻辑支持自我修正的可能性，但通过他语调，我们不难听出其不合理的悲观态度：

“...接触到该问题（主要通过全能神学悖论）的多数哲学家显然都希望找到一种不违反或不要求形式逻辑不适用性的解决方案。

我并不赞同应抑制这一希望的任何结论。我认为，形式逻辑在法律中的应用非常有限，但这并不表示通过逻辑标准绝对不可能消除法律形式的悖论。因此，我承认 - 没有上述诚挚的希望 - 可能存在令人满意的形式逻辑悖论消除方式。我认为，（1）在这方面进行的明显尝试是失败的，和（2）在任何情况下，法律能够免除这种消除方式。第二点是更为重要的一个论点，即使未来发现了一种令人满意的逻辑消除方式，也支持（如果完全支持）这一论点。

从这个意义上来说，当前的自我修正状态类似于介于 Leibniz 和 Weierstras 之间的微积分学状态。未来可能会达成一致，但目前，我们缺乏达成连贯和一致性的理论。与此同时，我们使用有好结果的自我修正。差别就在于，对于法律而言，自我修正的连贯理论并不是必要的，但对于数学来说，微积分学的连贯理论确实是必要的。

如果只在改写条款和处理自我变更的过程中观察到了某些无关紧要的细节部分，如果自我修正的逻辑没有问题，那么法律系统将继续完全忽视这些细节。无论需要更改的地方是很简单还是极其复杂，情况都是如此。如今，议员们只有在逻辑学家在选区中成为很有影响力的投票团体的情况下，才会听取逻辑学家们的意见。没有理由认为，如果逻辑学家的观点有朝一日变得正确，他们的立法行为将不再有效，正如更有力的坚定的反对派的声音正确时，这些立法行为将不一定失效。法律效力是有关权力和社会实践的问题，而不是抽象概念上的正确性。

我们再次反观“左撇子与矛盾共存”的主题，但更糟的是：他声称即使存在矛盾，立法者的行动仍然有效！***这在很大程度上解释了我们所处的反乌托邦社会。这种方法被证明是危险的：***从一个矛盾中，人们可以推导出任何东西，正如苏伯本人所说：“在标准逻辑中，所有命题都有效地从矛盾中推导出来。”.[71] ***更糟糕的是，“什么对逻辑学家有益”与***“什么对立法者有益”之间的区别远不止苏伯上文所说的荒唐，人们甚至可以称其为邪恶。没有人可以拒绝逻辑，不能拒绝量子力学，不能拒绝法律，甚至不能拒绝立法者和执法者让我们不要忘记由于不成体系的司法系统夺走了许多无辜的生命。

“……如果形式逻辑禁止所有的自参照，比如说，还需要结合类型理论，且如果自我修正不可避免地需要自参照，那么形式逻辑将没有令人满意的解决方案。”

的确，类型理论将恢复悖论“改变规律的规律改变规律的规律……”无限延续。这是基于类型理论和全部语言的旧 Tau 设计中的一个错误。然而，在这里，苏伯使我们的解决方案变得可能：

罗斯区分了逻辑矛盾和法律矛盾。任何命题与其对立面之间都存在着逻辑上的矛盾。我们不需要断言其中一个或两个命题，矛盾才存在。任何两个同时有效的不一致的法律之间都存在法律矛盾。如果它们不能同时有效，它们将是逻辑上而不是法律上的矛盾。有了这一区别，

罗斯有一个最常见的解决自我修正悖论的方法。”

事实上，时间方面，即 PFP 的分时间段方面（这里指的不是确切的“时间”，但对这件事来说是一样的），解决了悖论。 但这一点也没有造成“逻辑矛盾”和“法律矛盾”的区别。我们只需要找到一种适合法律的逻辑，一种可以塑造自我改变的逻辑，作为我们的选择逻辑。苏伯继续强调这一点：

消除这一悖论最常见的尝试是，确保或假设新老 ACs 永远不会同时既合法又至高无上。可以防止时间重叠; 但阻止它只是避免了法律上的矛盾，而不是逻辑上的矛盾。它使不一致的规则不能同时享有法律效力，但因此以其逻辑不一致为前提。 这种不一致性使模拟自我修正的推论无效，因为前提和结论 [72]（前提是内部一致）之间的逻辑矛盾足以使推论无效。正如罗斯经常说的，无效在于我们试图从一个规范衍生出与第一个不一致的第二个规范

[71]又名“爆炸原理”或简称为 quodlibet。
[72]X - >不是 X。

这确实是正确的观点，同时在 PFP 中写和删除等于矛盾（如附录 D 中暂停条件的解释）然而，苏伯认为这并不能防止逻辑上的矛盾，我们不同意这种说法，因为我们已经指出了一种合理的模拟情境的逻辑，以及无法接受逻辑不合理的的法律世界。苏伯继续提到对于法律和逻辑上的矛盾没有区分的案例：

“如果一个人拒绝区分逻辑和法律上的矛盾，但仍然坚持推理模拟法律变化，那么他没有更好的条件来证明模拟自我修正的推理的有效性。

> 如果矛盾消失是因为旧的 AC 在新 AC 获得有效性的同时失去有效性，那么在模拟这一过程的推理中，肯定在推理过程中忽视了关键前提。即使这成功地消除了模拟自我修正的推论中的不一致性，它也会用一个新的谬论取而代之。逻辑学家没有为此谬论正名，因为它不能在普通的论证中执行，在普通论证中，推理（从逻辑上考虑，而不是心理层面出发）是瞬时的或非瞬时的。”

它基本上认为非单调逻辑是不合逻辑的，这一点当然是不正确的。我们的选择逻辑有一个模型，这一事实就证明了这一点。

我们本可以继续评论苏伯所作的总结，但为了简洁起见，我们还是下次再说吧。现在让我们将苏伯的 Nomic 游戏（出现在【6】的附录中）与我们的解决方案进行比较。

Nomic 允许将规则声明定为“可变的”或“不可变的”，以及选民“改变”规则的能力，从而来实现自我修正，即将规则从可变的转变为不可变的，反之亦然。简而言之，在每一轮中，每位参赛者可以提出一个新规则，或对现有规则的改变，或对现有易变规则的修改进行投票。但是，本规程不适用于：

- 人数众多如果我们有 100 万名参与者，我们是否需要等待 100 万次才能让每个人都说出自己的观点?

- 可判定性在缺乏可决性的情况下，我们甚至如何推断一个规则是否与另一个规则相矛盾?

- 逻辑一致性。 在每个时间点上，当前的法律体系可能在逻辑上是一致的，但在 Nomic 中改变法律的过程是在法律本身之外的。*因此，在 Nomic 中，法律和立法不是在相同的逻辑形式主义下出现的。*

事实上，Nomic 并没有解决这些悖论正如苏伯在附录中所写：

"……在 Nomic 的情况下，很容易出现这样的情况，即很难确定某一举动是否合法。[73] 此外，在 Nomic 中可能会出现使判断瘫痪的悖论。偶尔，这可能是由于规则起草得不好，但也可能是由于规则起草得很好但带点恶作剧性质。 这类悖论的种类之多，实在难以预料。规则 213 是为了尽可能地处理这些问题而设计的，而不是用太多的法律限制来混淆初始集。需要注意的是，第 213 条规则允许狡猾的玩家创造悖论，并让其通过（如果其他玩家认为该规则没有问题），从而获得胜利。"

[73]甚至在没有可判定性的情况下是不可能的。

D. P-DATALOG 语言

我们提供了教程，介绍符合定律的建议逻辑是什么样子的。正如我们提到的，在【1】上，ifs 被称为 P-DATALOG，相当于在有序结构上的 FO【PFP】。它还捕获了在有序结构上的复杂类 PSPACE。它也是 html 语言的核心部分。html 在逻辑上等同于 P-DATALOG，并包含 P-DATALOG，但为了方便起见，使用了保守的[74]扩展进行了增强。但在我们开始描述这种语言之前，我们首先描述 Datalog 语言。

数据记录程序是一组事实和规则，每条规则都有一个头和一个主体。 头是结果，主体是前因。 事实是没有主体的规则，表示它与其他事实无关。 事实也是程序的输入。 所以规则“if it 's raining then I should take an umbrella”的主体是“it 's raining”，而头部是“I should take an umbrella”，如果输入包含了"“it 's raining”这个事实，那么输出就会包含“I should take an umbrella”。

Datalog 操作关系数据库[75]，即由表[76]组成的数据库。表中列的顺序很重要，而列的顺序不重要。[77]表格上的每个单元都可以从一些有限的集合中取值，这些集合被称为宇宙，或话语域。原则上，宇宙可能是无限的，但只有在有限的情况下才能恢复可判定性。

让我们以文献中关于 Datalog 程序的主要例子为例，图的传递闭包：

T（?x ?y） ：- E（?x ?y）；- E（?x ?y）；

内容如下：给定一个图作为边的列表 E（<vertex1><vertex2>），那么得到的图 T 也是一个边的列表，其中：

1. T（xy）也就是在结果图中 x 和 y 之间有一条边，如果 E（xy）也在输入图中，或者，
2. 对于某个顶点 z，在原始图上有一条从 x 到 z 的边，我们已经得出结论，一条从 y 到 z 的边出现在 T 上。

[74]即保持逻辑表达的完整性。
[75]它可以被看作是一台具有多维内存的随机存取机器。
[76]一个有 k 列的表只不过是一个 k-ary 关系。
[77]也就是说，一个表就和一个关系一样，是一组元组。

因此，结果图 T 将包含输入图上任意两个顶点之间的一条边，这两个顶点之间可以互相到达。注意，T 和 E 是表的名称，也称为关系符号。

这样一个程序的输入可能如下所示

E（1　2）. E（2　3）. E（3　4）.

对于这个输入，输出将是：

E（1　2）. E（2　3）. E（3　4）.

T（1　2）. T（2　3）. T（3　4）.

T（1　3）. T（1　4）. T（2　4）.

Datalog 程序的执行由所谓的“前向链接”的方式进行。[78]在每个阶段中，每个规则“触发”一次且仅一次。规则的“触发”意味着我们在规则的主体中替换当前数据库上的所有事实，然后我们可能推断新的术语（在规则开头已给出）添加到数据库中。在第一阶段，所有事实（在程序或其输入中已声明）都被替换到主体中，我们得到要添加到数据库中的事实列表。添加它们后，才进入下一个阶段，以此类推。当没有更多的新事实要添加到数据库时，或者换句话说，当数据库在两个连续的阶段中没有发生更改时，程序就终止了。

在数据表中，我们区分了外延关系和内涵关系。 在规则的头部出现至少一次的关系符号称为内涵的，否则称为外延的。 在我们的例子中，T 是内涵的，E 是外延的。数据记录只允许对有意符号进行否定。P-DATALOG 是 Datalog 的扩展，其中否定可以出现在任何地方，包括在规则的头部或正文中。 有很多方法可以用否定来给数据日志程序赋予语义，最明显的是基础良好的语义（WFS）和分层数据日志。这两个是 PTIME-Complete，而 P-DATALOG 是 PSPACE-Complete。

在 P-DATALOG 中，头中的否定表示删除。正如正头意味着向数据库中添加记录一样，负头意味着删除记录。 物体中的否定自然是指所有不满足关系的变量的替换。然而，对于头部和主体的否定，程序的执行顺序变得很重要。 主体中的否定在每个阶段都加以解释。所以可能会出现这样的情况，一个被否定的词在一个阶段是真的，但在以后的阶段不是真的。类似地，通过否定头删除，是只相对于现阶段而言删除的。例如,在

[78]有更多的方法来评估 Datalog 程序，我们只是用可用的案例来演示 P-DATALOG。

E（?x ?y） ：- E（?x ?z）, E（?z ?y）, ~E（?x ?x）.

变量? x 将绑定到所有值，如 E（？X？X）没有出现在当前步骤数据库中。 同样，

~E（?x ?x） : — E（?x ?x）.

将使下一步数据库不包括表单 E（？X？X）包含在当前步骤数据库中。

与 Datalog 不同，固定点并不一定存在。P-DATALOG 的终止条件如下： 如果在同一阶段插入和删除相同的术语，程序就会停止，并计算为“false”，或一个矛盾。 否则，它将继续执行下一个阶段，再次对每个规则执行单个求值。这一进程最终必须以下列任何一种形式停止:

1. 从当前步骤中得到的数据库与上一步得到的数据库相等。 这是一个有效点。当这种情况发生时，结果数据库被认为是最终结果。

2. 或者，在一个步骤中获得的数据库等于前一个状态中的数据库状态，而不是前一个状态。 在这种情况下，程序将永远循环，除非我们检测到它并停止程序，因此被计算为“undefined”，因为不存在固定点。

注意，这两种选择中只有一种可能发生，因为属性和宇宙大小是固定的。最终，对于宇宙大小 n 和最大大小 k，它们将在不超过 2n 个步骤中发生。

应该指出的是，PFP 的停止条件和主体中的否定造成了严重的性能困难，甚至达到合理的性能。我们已经通过合并二元决策图解决了这个问题，但是细节超出了本文所述范围。我们只提到二元决策图也提供了非常重要的数据压缩，通常出现在逻辑推理期间，没有这种压缩，通过这可能执行的推理任务是远远不可行的。

还应该指出，主体可以是任意的一阶或二阶逻辑公式。*它仍然等价于 P-DATALOG，并且仍然是 PSPACE-Complete【1】。*

E 定理证明 2

我们假设 Q 将一个公式转换为一个关系结构，该关系结构表示一个字符串，该字符串又表示公式。 由于 FO【PFP】中包含或不包含 Q+E 的所有公式集都可以在上下文无关的语法中给出，因此识别一个结构是否确实表示了一个公式可以在 FO【PFP】中表示（很容易通过使用“确定子句语法”或通过复杂性理论考虑来看到）。给定 FO【PFP】+EVAL 中的公式，我们将所有出现的 E 替换为检测结构是否确实是格式良好的公式的代码，如果不是，则返回 0。 否则，我们将 E 的出现转换为纯 FO[PFP]公式。 我们通过公式上的例子来概述接下来的步骤

$$F := R(x,z) \wedge R(z,y) \rightarrow R(x,y)$$

首先，

$$Q(F) = R(0,x,z), R(0,z,y), R(1,x,y)$$

额外的 0 和 1 的告诉我们这是一项为一体的一项,它是一个术语,和额外的这些常量应该添加指定的“规则”是佛（PFP）采取的形式 P-DATALOG（见附录 D）但我们省略了这个和其他小细节为了演示的主要思想，剩下的留给读者作为简单的练习。我们还隐含地假设了一个附加关系 V，它告诉我们哪些符号是变量，哪些是常量（这个 V 应该明确地添加到翻译后的公式中，但我们在这里只概述了一些高级细节）。现在是评估函数

$$E(Q(F)) = E(R(0,x,z), R(0,z,y), R(1,x,y))$$

在对其应用 PFP 运算符后，可以用公式 G 代替：

$$G := \forall a,b,c,d,e,f,g,h,i,j.$$

$$R(a,b) \leftarrow R(1,a,b) \wedge R(0,c,d) \wedge R(0,e,f) \wedge R(g,h) \wedge R(i,j) \wedge T(a,b,c,d,e,f,g,h,i,j)$$

T 的定义是

$$T(a,b,c,d,e,f,g,h,i,j) \leftarrow (a=c) \wedge (b=d) \wedge$$

$$(c=d) \rightarrow (g=h) \wedge$$

$$(c=e) \rightarrow (g=i) \wedge$$

$$(c=f) \rightarrow (g=j) \wedge$$

$$(d=e) \rightarrow (h=i) \wedge$$

$$(d=f) \rightarrow (h=j) \wedge$$

$$(e=f) \rightarrow (i=j)$$

注意，结果平移不仅适用于 F，而且适用于更多的公式，因为 T 适用于主体中任意变量的选择。我们现在可以用 PFP【G,R】替换 E （Q（F），R）的每一次出现。然后我们将得到的 FO【PFP】公式转换为 P-DATALOG，如【1】所示。