关于 τ-CHAIN

OHAD ASOR.

概要。τ-chain（读法： tau-chain）是一个由规则，证明和电脑程序组成的去中心化的点对点网络。τ-chain 可以把几乎任何中心化或去中心化的点对点网络结合起来，形成一个富有许多新功能的网络。

"如果制定法律是一款游戏的话，那么它就是一款把改变游戏规则作为下一步走法的游戏" - Peter Suber 呈现 Nomic [10]。

文件版本：
0.2 2015 年 2 月 6 日
0.21 2015 年 2 月 9 日
0.22 2015 年 2 月 12 日
0.23 2015 年 2 月 21 日
0.24 2015 年 2 月 25 日

1.前言

1.1.**概述**。我们（1）建议用本体语言统一以下语言：

- 知识
- 规则
- 逻辑
- 电脑程序
- 网络协议

本体是通过 RDF 语言家族（资源描述框架）表达。

我们（1）提出一个客户端软件以储存本地规则的本体，和利用推理器决定本体的行动。推理器可以智能地使用纯逻辑推理从现有的规则和结论生成新的规则和结论，此外，推理器也为生成的结果提出证明。所以，身为一个智能代理，τ-chain 节点利用非常易读和相同的语言和其他代理进行沟通。

当其他语言，比如 HTTP 被实现在 RDF 后，τ-chain 也可与它们沟通。

τ-chain 透过区块链演算法把时间箭头纳入网络中。大致上物体们可以被整合到由矿工签名的 Merkle 树中。此外，τ-chain 网络将成为一个用 RDF 语言沟通的分布式储存系统。此系统，即是 Kademlia DHT，能透过规则选择是否把物件们的散列值记录在一个机制的时间戳中。

（1）我的逻辑老师（把这个概念介绍和设计给我）HunterMinerCrafter https://bitcointalk.org/index.php?action=profile;u=245263 和我本人，Ohad Asor

网络的规则是由其使用者决定的。相反地，很多独立的”网络世界“可以在 τ -chain 上创建，它们可能会或可能不会分享 τ -chain 的时间戳。由于我们用的是统一语言，所以规则自身就是代码。规则和规则之间可以互相参考，籍此达到代码重用的目标。除此之外，我们用 RDF 实现可判定的电脑软件，即是 DTLC 语言，而不是图灵完备的一种。我们之后会介绍这些语言的含义。

1.2. **历史简介：什么是去中心化？为什么我们需要去中心化？**为何人性不是与生俱来的，而是被制造出来的？因为世界上的有一小群人制造和操控了>人性。这群人按照自己的意愿控制了我们的媒体，教育，经济，法律，政治和伦理。虽然他们尝试灌输”我们正被启蒙“的观点，但事实上情况正好相反。Immanuel Kant 在他的名著“何为启蒙？”中给了以下的定义：

> *”启蒙就是人类脱离不成熟状态。不成熟指的是缺少他人的教导就没有能力运用自己的理智。这种不成熟状态之所以是自己造成的，原因不在于缺少理智，而是没有他人的教导就缺乏运用自己理智的决心和勇气。勇于认知！（Sapere aude.）“要有勇气运用你自己的理智！”这就是启蒙运动的格言。“*

那些在 1784 年写的句子在今天看来更适合用于当下的社会，它们让我们打破压制，发出自己的心声。

一个世纪后，Franz Kafka 描述了个人无法对抗不合理和不道德的官僚系统。虽然法律从来没有真正地赋予公民，但人们仍然须要服从法律。没有法律的社会将导致暴力冲突。而这个系统是有矛盾的：一方面，他们声称他们从来没有私下定义法律，另一方面，他们总会想办法利用法律为自己的行为辩解。如今，人们也责怪电脑和程序中被允许保留或修改的记录或判断能力。也许法律不能为了我们被形式化，但可以为了他们被形式化?

法律可以和应该被形式化。法律系统中最需要的是一致的伦理基础（如宪法）和一致的意义，包括从此基础到法律本身。但是谁应该形式化法律呢？

如果法律由集权者来形式化的话，这将产生更大的危机。显而易见的是，民主是无法帮上忙的：现在的投票方式要达到保证一致，有道德和实用系统的目标还有一大段距离，这个情况在近代是无法避免的。

如今，我们能够利用数学和技术的方式合作创建法律，同时保留我们为自己设定的一些特性，比如一致性，投票，或最低要求。

法律中心化是可能发生的，那资讯中心化呢？想象一下谷歌拥有全世界最值钱的信息，就是“人们需要什么”，然而谷歌并不开放它的数据库。但试想如果我们能够访问谷歌的数据的话：我们不仅得到重要的信息，而且还能进一步搜寻。打个比方，我们可以查询和指定主题有关的其他主题，或者从数据库生成聚合数据，它的使用方式是无穷的。

沟通也是中心化的。当我们利用电子的方式沟通时，很多时候沟通内容往往被第三方拦截。我们的私隐是极度脆弱的。虽然一般上我们没有自己的网站来发表意见，但是我们更倾向于使用中心化平台，比如脸书来表达自己的看法。我们这么做是因为中心化平台向我们提供了更好的技术，长远的支持以及更少的漏洞。我们牺牲了自己的私隐，接触层出不穷的广告和营销垃圾来换取服务，我们甚至让他们决定谁是我们的朋友。我们是否可以保留种种的好处和拥有一个有道德的高端软件呢？

1.3. **愿景**。τ-chain 的目的为把人类的知识，思想，诀窍，沟通，法律和意见聚集在一个巨大的共享数据库，但同时保留去中心化的特性。这个连贯和一致的数据库可被有意义地查询，有效地重复使用信息/代码/数据，不受他人干预进行社交运作和沟通，允许每个用户设置自己的规则和成为社区的一部分以便分享同样的想法或目标或需要。τ-chain 不会强制任何人订阅或跟随任何规则集，但它结合了思想和技术上的优势使人性更美好。

显然地，世界上并非所有问题都是可以解决的，一些问题由人类解决，另一些由机器解决，还有一些则由许多机器和知识解决。一旦某一方能够解决一个难题，即有办法证明某个主张的话，那它就可以进入网络，反之亦然。

这不是一个新的构想，而且可能从我们将在第 5 节谈到的语义网开始。但在没有去中心化的前提下，这构想将无法达成。去中心化也给予公平地奖励参与者的能力。这个去中心化版的构想也不是新的，只是之前没人认真地去实现这个目标。

像中本聪开发的货币系统那样，现在是时候开始着手解决许多困扰人性的问题。有一点需要指出的是 τ-chain 本身不是货币，而且将从一个去中心化的网络开始做起。由于系统本身必须从零规则开始，因此开发者就像每个人，并没有所谓的预挖或手持任何币。

2.缩写和定义

- db：数据库
- P2P：点对点网络
- prop[s]：主张 – 一个在严格的逻辑下能够让人明白的建议或提倡（2）
- DHT：分布式哈希表
- DTLC：依赖性类型化演算
- TFPL：完全函数式编程语言

- RDF：资源描述框架
- N3：Notation3 语言

（2）"猫是花或哺乳动物”是一个用纯逻辑诠释的真实主张。

3.概述

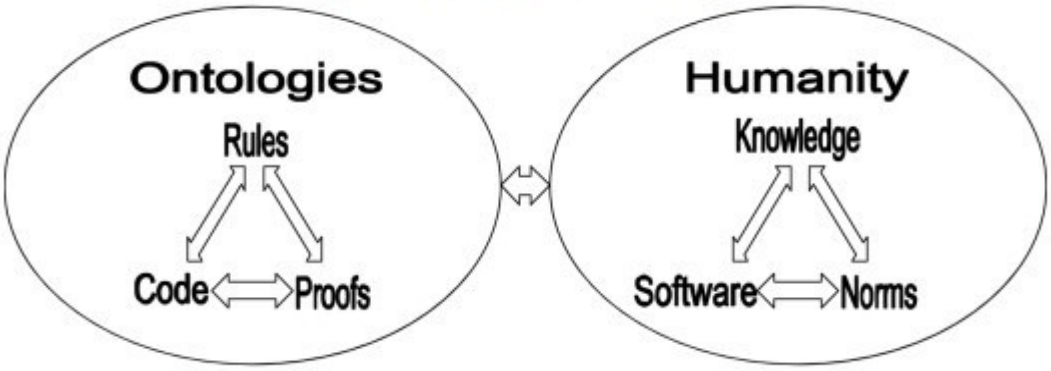
我们提出 τ -chain，一个完全去中心化的点对点网络。τ -chain 能结合许多中心化或去中心化的点对点
网络，包括区块链在内。我们打算尽量实现我们所发现的概念，并给予用户在 τ -chain 上实施任何点对点网络的能力。τ -chain 的意义，用途和目标不止是仅仅的一个点对点网络，它还包括了软件开发，法律，游戏，数学，科学，咯及，加密货币经济，社交网络，规则制定，民主和投票，软件库（像去中心话的 Github + Appstore / Google Play），软件审批和验证以及很多其他方面的用途 3。由于 τ -chain 能够推理信息，所以除了语法搜寻，你还能用语义搜寻和更多的功能等等（3）。

3.1.五个同等的定义。τ -chain 可以成以下的概念：

- 一个共享的规则数据库，包括了一个能够改变规则，遵守规则，从现有规则生成新的规则，和确保规则的一致性的一个客户端。
- 一个共享的电脑程序的代码数据库，它包含了共同组成的版本控制和自定权限（如 git），和具有可以运行代码，重复使用现有代码和针对程序本身的正式规范对程序进行验证的一个客户端。
- 一个共享的主张数据库，和一个可以生成新的主张，用主张本身的自定派生规则证明主张并验证主张的证明的客户端。
- 一个共享的本体数据库，包括类型的定义（分类）和它们的关系，以及能够提出新的本体，确保本体的一致性，和查询本体数据库的一个客户端。
- 一个去中心化的 Nomic 游戏。

关于 τ-CHAIN

FIGURE 3.1. τ-chains



3.2.**τ -chain node 节点的本质**。原则上，τ -chain 只不过是由多个含有四个单字，即上下文，主语，谓语和宾语的元组组成的一个数据库。我们在这里会展示规则，程序，证明和本体是如何通过统一的方式被表现出来，和此系统的一些长远用途。

（3）Zennet 超级电脑和 BitAgoras 的设计已经改成在 τ -chain 上实现。
每个节点包含了知识和非常聪明的判断力，即是用逻辑作出推理和启示。虽然这不意味着它们代表着人类智慧，但它们的知识体现了如何沟通和传达可靠的信息。技术上我们可以通过形式化本体算法，如 DHT 和区块链来做到这一点。换个角度来看的话，它们就像能够提问而不是获取，能够了解而不是储存，能够交谈而不是沟通，能够考虑而不是验证，只有嘴，耳朵，本体和推理器的虚拟生物。

4.基本相等关系

我们给了一个规则等于程序和证明这样的一个非正式说明。关于正式的说明请参考类型理论的文献。

直觉上，“执行一个程序”就像“遵守规则”那样简单，而规则就是代码本身。它的不同之处在于：所有规则是否可以被形式化成程序呢?答案就某个意义来说是正面的，我们将通过了解证明来解释它。

4.1.**什么是证明？**从抽象逻辑的角度来看，证明是一个从被提出的假设和公理开始，到最后产生一个被证明的结果的一个过程。而整个过程必须依照所给予的派生规则和蕴涵的概念（a ⇒ b）。

例子 1. 以下是派生规则的例子：

- 分离规则：如果 A 蕴含 B 和 A 为真，B 则是正确的：
$$((A \Rightarrow B) \wedge A) \Rightarrow B$$
- 加密签名：如果没有存取私钥权限的话，就几乎不能生成一个有效的签名（取决于假定的验证过程）
- 一个从数据到其哈希值的映射是一对一和不可逆转的。显然的，纯数学逻辑上这是不正确的。但作为一个实用科学，它在密码学是有效的。
- 希尔伯特的形式推理规则和根岑正式推理规则。

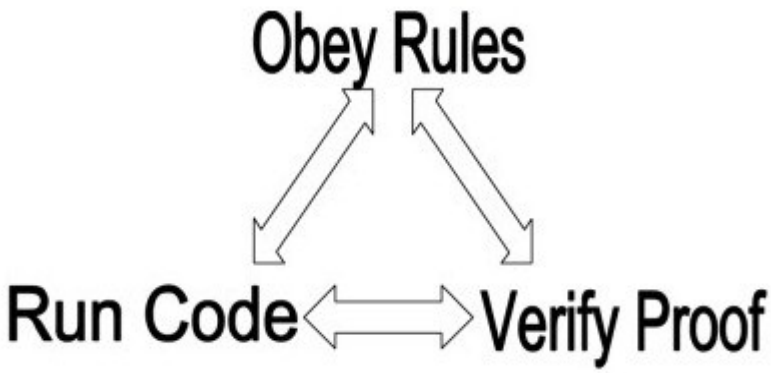
值得一提的是，派生规则可以被诠释成公理，而公理则是假设。因此，一个证明是从其他语句推断语句，或从其他规则派生出一个规则。如果吧定义（分类）看作规则本身的话，那规则就是证明，证明就是规则，证明就是推断。

4.2.**证明=程序。**这就是是著名的 Curry-Howard 同构（译者：电脑程序和数学证明之间的密切联系）。在众多的多程序和证明中令我们感兴趣的是程序经常停机的部份。大致上，这一类型的编程语言属于 TFPL 类型，如 Idris，AGDA，COQ（4）。它们和其它的类型与 DTLC 对应，并且和证明类型是同构的。

TFPL 不是图灵完备的。这让我们避开了许多有图灵机形式主义（保证程序经常停机，和赋予我们主张的权利和证明有关程序的的主张）产生的矛盾（5）。通过这样的方式，我们可以证明相关的代码通过相关的单元测试（6）。我们也可以证明程序的运行（7）。

（4）虽然 COQ 不是 DTLC 而是构造演算，但是这些分别并不重要

FIGURE 4.1. Obeying Code



需要注意的是，身为非图灵完备反而具备了一些优点：任何人开发所开发的软件不需要图灵完备，但只需要 DTLC 就够了。图灵完备语言只能在极端（主要是无限的）理论的情况下做到 DTLC 所不能做的事情。

推论 2。逻辑性主张可以被诠释成规则，反之亦然。公理的证明可以被诠释成从其他规则推断规则，反之亦然。函数式编程语言编写的电脑程序可以被诠释成建设性证明，反之亦然。

值得一提的是，证明和电脑程序之间的对应是很强大的，它促使了数学新基础的诞生，如 Cantor 的/ZFC 集合论 即范畴论。事实证明，证明和程序在某种程度上是和范畴论同构的。这就是 CurryHoward-Lambek 同构，又称计算性三位一体（8）。

5.本体和语义网

5.1.**简介。**长话短说：互联网之父 Tim-Bernes-Lee，提出了所有在线数据将可被机器解读的愿景，和塑造一个物件和类型之间的关系网络。简单来说，这就是语义网络的概念。为了达成这个目标，目前有许多工具已被开发出来或正在开发中。

如今我们可以看到巨量和增长中的各种本体，比如基本的网络组件和逻辑实体和关系（OWL），法律（参见 LKIF 本体），软件，医药，电子商务，加密技术，社交媒体，地理，新闻等等（9）（10）。另一个值得关注的项目就是 dbpedia,它从维基百科摄取数百万个概念并用一致的本体形式化它们。此外，市面上有很多工具可以被用来处理 RDF 数据，更多详情可参考语义网维基（11）。

（5）有时候就好像“程序正在做 X 但没有在 Z 步骤中做 Y”
（6）正式规范的一种
（7）请参考 http://amiller.github.io/lambda-auth/
（8）https://ncatlab.org/nlab/show/computational+trinitarianism
（9）https://www.w3.org/standards/semanticweb/ontology
https://www.w3.org/wiki/Ontology_Dowsing#Lists_of_ontologies_and_services
（10）EulerGUI 手册：本体和本体的搜寻引擎
http://eulergui.sourceforge.net/documentation.html##Finding
同样的增长也在数学和为 COQ 形式化的其他科学中出现，它们包含了数以千计充分证明的数学定律和可以被翻译成 RDF。

5.2.**语义网络的语义。**语义网络概念的基本表述方法是由 RDF（资源描述框架）来规划。RDF 是一种表达本体的语言。一个本体包含类型的定义，比如“狗是一种动物”，和它们之间的关系，如“所有的狗都有四条腿”。当然，“四”的定义必须合理，就像“有”的定义。

Notation3（N3）是一种用来编写 RDF 本体的语言。它不仅更方便，还具有人类可读的特征。逻辑上它足以代表 DTLC。此外，像刚才在简介部分所提到的，它可以被转换成四元组（τ -chain 的格式）,反之亦然。

在某种程度上，身为纯逻辑的 RDF 也可以被转换成英语，反之亦然。它比较像是机器可读的英语，而不是人类可读的，但它还是很自然的。我们当然会提到 Attempto 项目。

5.3.**四元组。**τ -chain 的储存系统只包含了四元组的列表，比如四个字即是上下文，主语，谓

语，宾语。上下文给予空间之间的分离，因为我们需要支持独特和非混合规则集。所以我们保持了一个由主语，谓语，宾语组成的三元组。三元组可以包含所有的知识，在某种意义上可以证明逻辑的句子（因此包括了规则和电脑程序）可以通过这种方式配置，反之亦然。让我们用三元组写“敏捷的棕毛狐狸从懒狗身上跳跃过”这个句子。为了更方便，我们会为宾语命名，比如狐狸弗雷德和狗戴夫，所以：

弗雷德是狐狸
弗雷德是敏捷
弗雷德是棕色
戴夫是狗
戴夫是懒惰
J 是跳
J 越过戴夫
J 靠近弗雷德

最后两个句子将更正式地写成"跳过弗雷德的领域“，”跳过戴夫的范围“，但是任何词汇是合理的，因为我们不介意单词的真正意义。我们更注重他们的具有实际意义的结构依赖。

5.4.**最先进的。**EulerGUI 是各种推理器引擎的 IDE 元老，它可以被用来证明本体的强大性。它支持的格式包括 RDFS，N3，OWL，UML，SPARQL，Attempto（英文对照），甚至是 Java 的 jar。通过基于规则的 SWING，它可以输出 Java 代码以建立本体的用户界面。它支持了四种不同且强大的推理器引擎，逻辑性查询，解释证明，一致性检查，模糊逻辑，可视化图形等等。它也可以和推论引擎（12）整合，即自身被写入本体。它在其网站被描述为”通过 N3 + OWL 本体和规则利用一阶逻辑被应用在一般软件的任务的一种人工智能技术"。EulerGUI 和推论引擎是与推理器引擎 Drools 兼容的。在 Drools 的网站上，我们可以看到以本体为基础的开发的优点和缺点（13）。读者可以从相关的链接获得更多的资料。最近推出的一个工具叫 Protege，它是一个开放的本体 IDE。

（11）http://semanticweb.org

5.5.**规则的本体。**由于我们把规则形式化成本体，那我们需要用它来干什么？我们把 cwm（14）作为工具范例。我们可以问在规则范围内它所能回答的问题。我们也可以验证规则的一致性（15）。

读者可以看看它的各种工具，包括推理器和 OWL 描述器（OWL 是包含基本本体定义的一种 RDF，也是一中 W3C 的标准）。请参考附录以获取截图示范。

5.6.**程序作为本体。**基于 DTLC 的语言几乎可以实现一切。虽然图灵完备语言收到不可判定性的影响，而通过 DTLC 语言，我们可以吧代码当成我们要操作的逻辑。我们也可以直接从代码证明各种有用的主张，比如这样的程序不使用互联网，或只能访问某些文件，显示它停

止操作，证明其执行路径等等。

从 DTLC 过渡到 RDF 是不平凡的，但它如何被实现的两个例子可以在（13）（14）中找到。程序的结构（几乎）不变和它的命名可被保留。其 RDF 格式可以维持人类可读性，但对于机器证明我们不仅使用 RDF 方式表达，而且还应用了 SMT 求解器进行推理。

6.点对点

6.1.**DHT。**如上所述，τ -chain 可以概括任何点对点网络。分布式点对点客户端是一个根据规则决定如何处理各种输入的一个状态机。

例子 3。DHT 是点对点结构的一个例子。其中一个著名的产品即是 BitTorrent，而 DHT 中最普遍的协议算法就是 Kademlia。比特币网络本身就是 DHT，它的每个节点
都是完全复制的，而一般的 DHT 中复制的数量是可以被控制的。一个 Kademlia 的正式范例（即规则）的样本可以在[11]中找到。要注意的是，这个范例包含一个 TFPL，所以它能够被轻易地转化为本体。我们可以看到它有四个原函数：Ping，Store，FindNode，和
FindValue。它的规则定义了如何处理每个例子。

（12）http://deductions.sourceforge.net/
（13）http://docs.jboss.org/drools/release/5.4.0.Final/drools-expert-docs/html/ch01
（14）http://www.w3.org/2000/10/swap/doc/cwm.html
（15）一个一致性检查的重要例子是，一个指定的定理的逻辑证明作为规则，验证证明是等于验证规则的一致性。
（16）http://attempto.ifi.uzh.ch/site/tools

例子 4。比特币身为一个去中心化的点对点网络使用了证明系统，当然它也可以被写为规则。

在 τ -chain 中，每个节点储存了三个本地本体：其路由表，其用户输入和其他节点的的输入。它们与此节点订阅的本体被结合在一起。从共享数据库中，一起推断客户端应该做些什么，比如是本地储存操作或将信息发送到（多个）节点。另一个比喻就是：我告诉你我知道的部分，你想想看，并告诉我你要和我分享的结论。

关于 τ-CHAIN

推论 5。*任何点对点网络是一个从收到的信息中定义应该发送些什么的规则集。*

一个点对点网络可以设置自己的规则和很多规则的背景，通过持续性地实施新的网络，而用户可以决定他们要参与哪个环境。这就是 τ -chain。

6.2。**时序逻辑：区块链。**我们如何将一个时间概念置入 τ -chain？毕竟有用的程序或规则必须能够意识时间的。同步更是去中心化设置的一大问题：点对点节点如何对时间戳达到共识？这个难题被中本聪发明的区块链算法解决了。通过工作证明的方法，它把时间划分成区块。”下一刻“指的是下一个区块。本体可以通过暂时的词汇连接一个区块链，甚至实施一个区块链作为本体。

其数据被分层为 Merkle 树：最深层次代表了被加时间戳的 DHT 项目。树的上部组成一个区块链中的一个区块。因此 DHT 的项目可以通过它们的哈希值被加时间戳。

7.τ -chain

7.1。**节点结构。**本质上客户端是一个推理器，像 EYE 一样，被一个 C++程序封装起来指定它要做的的事。该封装器给予了推理器额外的网络能力，即连接和接受连接，发送和接受。

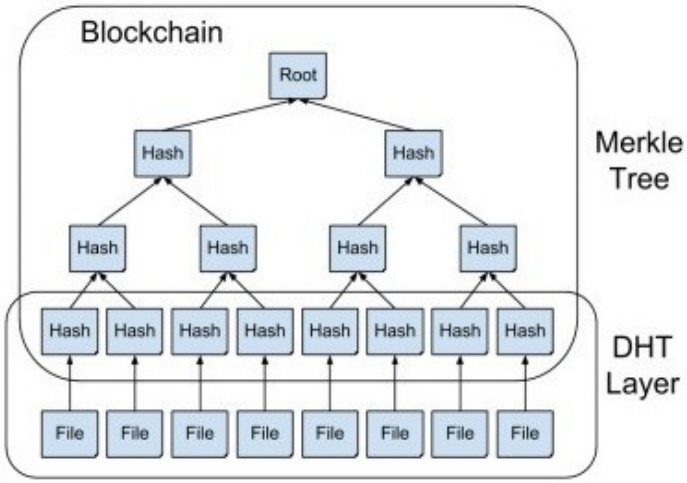
事件是由封装器写成一个 N3 的本体，然后它会调用推理器来统一违反规则的事件。如果推理器的输出包含动作的调用，那封装器将会解析它们并作出相应的行为。

通过给予本体网络的原函数，推理器可以实现一个 HTTP 服务器，并通过它把 GUI 提供用户。

我们从一个提供 DHT 语法的原函数的 C++封装器开始，而其语义性则有本体决定。

7.2。**引导。**该网络将由以下的原始客户端开始：它将拥有一个实现简单的 DHT 的本体，而且将利用 CWM 来查询本体，比如”我现在该做什么“，或者更准确地说，“我应该给其他每个节点发送些什么”和“我应该用我的本地储存做些什么”。从这一点，我们可以通过去中心化网络用本体来共同地实现一切。

FIGURE 7.1. τ-chain Structure



在这个引导阶段，我们打算植入很多本体，和举办由专业人士组成的圆桌会议以便一起策划此网络的全球性规则。下列是一些可能在引导阶段被实现的功能：

- 避免恶意使用网络的规则。
- 能够创建独立的环境，每个环境都有自己的规则，并互不干扰。这样的话，用户可以创建许多程序和选择他们要使用的程序，就像一个去中心化的 Appstore/Google Play。
- 奖励做出贡献的节点。
- 实施比特币作为概念验证（17）。
- 大规模的分布式储存，这对系统本身是必要的，因为按照计划它会处理大量的数据。
- 投票系统。
- 如何从现在开始设置规则。

7.3.**成熟性。**一旦系统被认为有能力推出市场和供日常使用的话，我们将谈到它可以做到哪些东西。

- 去中心化的代码储存库（比如去中心化的 GITHUB）。
- 去中心化的应用储存库。
- 安全地和自动地向人们或机器提供代币（18）以证明一个定理或根据正式规范来编写一个软件。
- 一个庞大的代码片断数据库将被自动地重复使用以验证正式的要求或检测本体之间的同构（19）。
- 任何目的的投票系统，比如开发团队的投票系统以供有效和真实的软件发布，或去中心化的民主。
- 共同的社会/企业规则制定，并有能力找出矛盾，和提问“为了得到 X，我们还需要什么”，和必要时投选那些规则。

（17）我们不打算在现实生活中使用它。当然，未来我们计划推出改进的虚拟币，特别是在

Agoras 的范围内。
（18）τ-chain 将不会从第一天开始推出代币，但它是系统可以和需要制造的东西。
（19）那些同构可以连接科学之间的概念。人们可以编写一个在图形游走的代码，这将解决一个生物科学家正面对的问题。

- 安全，去中心化，私人和可完全客制化的社交网络，以及私人的俱乐部（甚至有会员费，入会测试，接受规则等等）。
- 询问几乎所有的人类可读的问题。比如“Aristotles 住在哪里”。CWM 能够从本体回答这样的问题，并且为其答案附上证明。当然，其他工具比如 COQ 也是兼容而且更强大的。

我们将会在 τ -chain 上建立一个称为 Agoras 的金融市场，以实施一个拥有各种服务（比如银行，咨询，编程或证明）的智慧市场的虚拟货币。

7.3.1.*软件安全性。*

7.3.2.*代码复用。*

7.3.3.*共享式有意义和可被查询的知识。*

8.结论

我们已经表明 τ -chain 是可以概括任何的协同工作，尤其是点对点网络。它提供了丰富的信息共享功能，和查询数据的多种方式，推断出新的信息，并共同行动。现有的点对点网络，比如比特币的区块链可以被移植到 τ -chain，使它们能够被可即时改变的规则（通过任何改变规则的规则）所控制。它可以成为一个可靠的信息来源，作为一个共享的知识来源，源码，和规则，并且是机器和人类可读和可处理的。它提供了一个可以统一科学的平台，更重要的是，人类的思想能得到满足和被统一。

参考

（1）“比特币：一个点对点的电子现金系统”，2008，中本聪
（2）“直觉类型论”，Per Martin-Löf, 1980, http://www.cip.ifi.lmu.de/~langeh/test/1984%20-%20Loef%20-%20Intuitionistic%20Type%20Theory.pdf
（3）”相关类型的认证程序“，Adam Chlipala, http://adam.chlipala.net/cpdt/cpdt.pdf
（4）“W3C RDF1.1 概念和抽象语法“，W3C 推荐标准，https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/
（5）”RDF1.1 语义学“，W3C 推荐标准，2014 http://www.w3.org/TR/2014/REC-rdf11-mt-20140225
（6）”RDF 概要 1.1“，W3C 推荐标准，2014 http://www.w3.org/TR/2014/REC-rdf11-mt-20140225

（7）”Notaion3 （N3）：一个可读的 RDF 语法” W3C 团队提交，2014
http://www.w3.org/TeamSubmission/n3
（8）”OWL 声明在控制性英文中的可理解性，语义网“，Tobias Kuhn, 2013,
http://attempto.ifi.uzh.ch/site/pubs/papers/kuhn2013swj.pdf
（9）”马丁-洛夫的类型理论的编程
（10）“Nomic：一个自我修正的游戏”， Peter Suber, 1990,
http://legacy.earlham.edu/~peters/writing/nomic.htm
（11）“Kademlia 路由表和 Maude 语言的 Kad 路由表的正式规范”，Pita and Camacho,
2012,
federwin.sip.ucm.es/sic/investigacion/publicaciones/pdfs/KademliaSpecificationSummary0212.pdf
（12）“科学家范畴理论”，Spivak, 2013, http://math.mit.edu/~dspivak/teaching/sp13/CT4S--
static.pdf
（13）“一个归类的类型理论，归类元，和叙述的本体信息”，Norihiro Ogata
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.20.3116&rep=rep1&type=pdf
（14）“朝向基于类型理论的概念结构“”，Richard Dapoigny and Patrick Barlatier
http://ceur-ws.org/Vol-354/p63.pdf