**论没有区块奖励的比特币的不稳定性**

Miles Carlsten Harry Kalodner S。 Matthew Weinberg

[carlsten@cs。princeton。edu](mailto:carlsten@cs.princeton.edu) [kalodner@cs。princeton。edu](mailto:kalodner@cs.princeton.edu) [smweinberg@princeton。edu](mailto:smweinberg@princeton.edu)

Arvind Narayanan

[arvindn@cs。princeton。edu](mailto:arvindn@cs.princeton.edu)

**摘要**

比特币为矿工提供了两种奖励：区块奖励和交易费用。比特币网络最初时前者占矿工收入的绝大部分，但是随着区块奖励的减少，预计矿工收入将转向后者。有一种隐含的观点认为，矿工是否被按区块奖励或交易费用付款，并不会影响区块链的安全性。

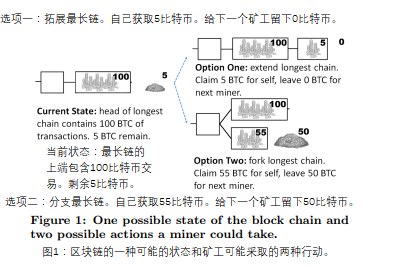
我们表明情况并非如此。我们的主要观点是，若只有交易费用，由于指数分布的区块到达时间，区块奖励的方差非常高，并且通过分叉“富有的”区块来“窃取”其中的奖励变得很有吸引力。我们证明，这导致了一种比特币安全性和性能的不理想特性的均衡，在某些情况下甚至是非均衡。我们也重新审视自私的挖矿行为，并且表明它可以为一个任意低哈希算力份额的矿工赚钱，即使这是在网络中随意地不良连接着的矿工。我们的结果来自理论分析，并由一个新的比特币挖矿模拟器证实，这一结果可能是独立的研究成果。

我们讨论了我们的结果对未来比特币安全性的不利影响，并为设计新的加密货币提供了借鉴。

1. **引言**

比特币共识协议的安全依赖于矿工的正确行为。通过挖矿收入的方式他们被鼓励做正确的行为，这假设他们是理性的实体。任何超越默认值的异常矿工行为因此都是对比特币安全的严重威胁。

比特币矿工获得两种类型的收入：区块奖励和交易费用。比特币网络最初时前者占矿工收入的绝大部分，但是随着区块奖励的减少，预计矿工收入将转向后者。（具体来说，他们每四年减半）。而从区块链的安全性的角度来看，这是一个未经斟酌的想法（包括挖矿比赛的奖励），矿工是否收到（比方说）每个区块25比特币作为区块奖励或者25比特币作为交易费用是不重要的。

示例（图1）

想象一下，一群理性的、自私的矿工。考虑一个区块以指数分布奖励的区块链，正如我们预期的固定区块的奖励耗尽。矿工在挖矿时有很多选择，但我们只关注两种可能。她可以拓展最长的链条（选项一），获得5比特币的奖励，并为下一个矿工留下0比特币的奖励（至少在更多的交易到来前）。 或者，她可以将其分叉（选项二），获得55比特币的奖励，同时留下50个无归属的比特币的奖励。 比特币协议规定了选项一，但一个快速的推理表明，选项二更好。

为了正确地解释这一点，我们必须考虑其他矿工正在使用的策略。例如，如果所有其他矿工在1-区块分叉的情况下对他们首先听到的区块上遵循启发式挖矿（如果没有网络延迟），那么分叉是无效的，并且显然方案一更优越。另一方面，由于其他矿工是理性的，也许他们会选择建立分叉而不是依赖老的区块，在这种情况下，选项二将有更多的奖励。

像这样的例子揭示了一些新的激励问题，这些问题在区块奖励是固定的时候根本不会产生。本文的目的是通过调查挖矿博弈来了解对比特币稳定性的潜在影响，在该机制中，区块奖励已经减少到可以忽略的程度，交易费用主宰着挖矿奖励。我们在交易费用制度中发现了新的令人惊讶的奖励问题，即便假设交易（和相关费用）达到了稳定的速度。要清楚的是：我们发现的奖励问题并不是因为交易费用可能会不规律地到账，而是因为交易费用随时间变化的性质允许更丰富的策略性偏差，而这在区块奖励模型中是不会出现的。

在一个较高的层面上，有一个关于矿池跳跃（指只在合适的时间在一个矿池挖矿的行为）的类比[21]。 对确定的矿池奖励计划，矿工对参与挖矿的预期报酬会随着时间的推移而变化，具体取决于自矿池发现最后一个区块以来贡献了多少份额。但值得关注的是，矿工们会用实时“跳跃”来回应这种计划，以最大化他们的预期回报。 关于这个主题的另一例子，请考虑一个有多种加密货币的未来，这些奖励随时间变化的加密货币可以用相同的硬件来挖矿。这也许会导致coinhopping, 即矿工们跳跃到有最大的交易费用矿池的加密货币。

**论文贡献1（Contribution 1）:一个挖矿策略模拟器（A mining strategy simulator）。**虽然我们在第5节和第6节中建立了许多理论结果，但是众多可能的参数和假设的多样性使得建立完全准确且易于理解的比特币博弈理论模型并不可行。为了填补空白并确认我们的理论结果，我们建立了一个挖矿策略模拟器。简单且根据事实的模型的理论结果提供了良好的直觉知识（intuition）来指导实践，并且对更复杂情况的模拟证实了这些结果适用于数学证明难以处理的更现实的模型。

在我们的模拟中，随着时间的推移，矿工们使用迭代更新使用策略的概率分布的不回溯的学习算法学习到了哪些策略是成功的（见第4。2节）。我们的模拟器是多功能的，可以模拟不同数量的矿工，哈希算力分配，网络延迟和奖励计划。我们展示了这个模拟器如何让研究人员在参数空间内快速建立原型和进行新的设置。 但它确实有局限性：它不能建立矿池模型或者非固定到账速率的交易的模型。 我们已经把模拟器开源[[1]](#footnote-1)。

**论文贡献2（Contribution 2）:削弱攻击（Undercutting attacks）。**本文的重点是分析交易费用制度中可能损害比特币安全性的异常挖矿策略。我们从观察得出的结论开始，如果有一个单区块分叉（1-block fork），那么下一个矿工通过扩展剩余最多交易费用的区块而不是最老的区块来打破平局是更加有利的。 我们把这个策略称为“PettyCompliant”。

一旦采用PettyCompliant策略的矿工的比例非零，那么矿工就可以实现各种更具侵略性且有害于比特币共识协议的策略。 我们称之为削弱攻击，其中矿工将积极地分叉区块链头，并且不完成交易，从而激励采用PettyCompliant策略的矿工建立他们的区块。

在某些情况下，我们的模拟揭示了一个越来越积极地削弱的非平衡。但是，随着策略空间的扩大和适当的假设，我们能够证明存在一个均衡。然而，这是一均衡中矿工只能把一部分交易放在他们的区块。这会导致规模随着时间无限扩大的交易的积压。我们的模拟证实了这一结果。

准确地预测稳态挖矿行为需要大量的变量建模，如矿工的成本结构，而这不是我们工作的目标。相反，我们的结果可以被看作是一个非正式的“底限”，可能在交易费用制度中是偏离合规行为的。基于现实地，我们可以预测，采用PettyCompliant策略的矿工一定会出现，这样的矿工的存在为各种更积极的策略开辟了领域（第5节）。

**论文贡献3（Contribution 3）:重新审视自私挖矿行为（Revisiting selfish mining）。**我们重温了Eyal和Sirer的自私挖矿策略[9]，并且表明，与直觉相反的是，它在交易费用制度中表现得比在区块奖励制度中表现得更好。接下来，我们提出了一个更复杂的自私挖矿策略，以说明奖励的不一致性，并且该策略优于默认挖矿和“经典”自私挖矿。更糟的是，与传统的自私挖矿不同，这种策略适用于任意低的哈希算力的矿工，无论他们在比特币网络中的连通性如何。此外，削弱攻击一旦部署就有利可图，而传统的自私挖矿只有在经过两周的困难调整期后才能盈利，这可能给社区一个关键的时间窗口来检测和应对这种攻击[10]。我们通过理论和模拟验证了这些结果（第6节）。

**对比特币安全性的影响。**无论我们探索出的任何异常挖矿策略被部署，对比特币安全**性**的影响都将是严重的。最好的情况是，由于持续不断的分叉，这个区块链会有很大一部分过时或孤立的区块，这使51％攻击变得更容易，同时增加了交易确认时间。最坏的情况是，由于区块扣留或越来越积极地削弱，比特币共识协议将被打破。 这显示了我们需要重新思考加密货币设计中区块奖励的作用。中本聪（Nakamoto）似乎认为，在去中心化的情况下，为了实现比特币的初始分配，区块奖励是必要也是暂时的坏事，而交易费用制度则是系统理想的无通货膨胀的稳定状态。但是我们的工作表明，在交易费用制度中激励合规矿工的行为是一个比在区块奖励制度中激励更为艰巨的任务。 也许恰恰相反，新的加密货币的设计者必须自己承担货币通胀的必然性，并让区块奖励永久化。交易费用仍然会存在，但仅仅是为了激励矿工将交易包括在他们的区块中。

1. **相关工作**

最近几篇著作分析了比特币挖矿的激励机制。一些例子包括[12]和[8]，在其中分析了有策略的挖矿池怎样以各种方式攻击与之竞争的矿池，和[15]，在其中分析了以太坊有策略的矿工如何欺骗他人浪费其计算能力来验证复杂脚本的有效性。了解矿工在比特币系统中的激励机制非常重要——有经验证据表明，矿工/挖矿池愿意攻击他人以最大限度地利用自己的利润（例如，通过对其他挖矿池发起DDoS攻击）[23]。

Eyal和Sirer发展了自私的挖矿攻击[9]，这是一种使矿工获得更多回报的异常挖矿策略。我们在第6节的基础上建立了他们的结果。其他著作，特别是Sapirshtein et al的[22]，利用马尔可夫决策过程（MDP，Markov Decision Processes）更详细地分析了自私挖矿。在马尔可夫决策过程中，一位玩家在一个离散的状态空间中移动，并试图最大化回报（状态转移函数和收益函数都是概率的）。这使得它非常适合模拟比特币挖矿。然而，在我们的情况下，状态空间（具体来说，是剩余的交易费用）是连续的而非离散的，所以我们不使用MDP的机制。

在了解市场交易费用及其与区块大小的关系方面，有一些工作要做（即，为了将交易包括在一个区块中，用户必须支付哪些费用？）[13,11,20,17]。我们的工作避免了这个讨论;我们发现，即使市场达到交易费用不可忽略的平衡点，并且稳定可靠地到达平衡点，也会出现不良行为。有趣的是，M¨oser 和 B¨ohme通过非常不同的方法得出和我们一样的结论（货币通胀是交易费用的一个更好的机制）[17]。在模拟方面，许多先前的作品都为比特币的某些方面开发了模拟器。 除了有策略的挖矿之外，有些模拟器是针对比特币的，比如隐私[3]，或者是对等网络[16]。 在[9]和[8]中开发的工作也集中在模拟异常挖矿策略上，但我们的理解是，这些模拟器是针对他们希望测试的内容偏离实际策略而量身定制的。相比之下，我们的模拟器允许在各种环境中轻松实施更广泛的策略。事实上，我们的模拟器的多功能性对于获得本文中每个结果的直觉知识（intuition）至关重要。我们已经开放模拟器的源代码，希望它能成为未来对有策略的矿工行为研究的有用工具。

1. **模式与策略**

在本节中，我们将介绍我们调查的比特币模型。 我们将使用这个模型来快速说明为何将奖励转换为以交易费用为主导可能会导致对比特币有趣却潜在的有害影响。我们还介绍了描述比特币策略的正式语言，我们将在整篇论文中使用该语言。

* 1. **系统的模型（Model of the system）**

简单地说，在进入具体细节之前，让我们描述一下我们模型的主题。 这项工作的目的不是要准确预测在实践中会发生什么样的挖矿行为，而是要发现纯粹由于交易费用与区块奖励的时间性质不同而产生的激励问题。为此，我们故意把模型简单化，因为我们想要分离随时间变化的影响与固定回报。作为一个例子，我们将假定交易（及其相关费用）以恒定和持续的速率到达。我们作出这种假设并不是因为我们必须预测它会在实践中保持不变，而是因为如果没有这一点，我们不能保证我们已经隔离了随时间变化的交易费用，而把这一交易费用作为我们发现的任何激励问题的原因。换句话说，我们的结论只是通过简化假设而变得更加普适，因为我们声称，即使人们愿意给予简化的假设，也只会产生奇怪和不希望出现的结果。

接下来介绍相关细节，我们分析的比特币模型是在区块奖励降到零之后的。也就是说，交易费用是矿工收入的唯一来源，我们将可用的交易费用模型以一定的比率作为到达比特币系统的模型。具体来说，我们假设任何时间间隔I的长度为t，在I期间公布的交易的交易费用的总和为t（选择t而不是引入某个常数c的ct只是为了标准化）。与现在的比特币不同的是，比特币与小额交易费用相比具有较大的区块奖励，但是我们假设的这种情况与最终所有比特币被铸造后比特币长期稳定时的预期情况是一致的。

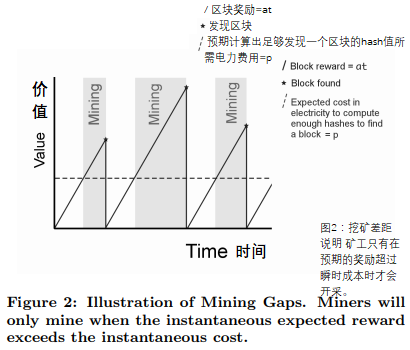
我们还假定挖矿难度被设置为：在期望中，每个单位时间内都恰有网络中的某一个人解决了一个hash难题（hash puzzle）（再一次，这只是为了规范化）。 此外，为了简单起见，在我们的理论结果和报告的模拟中，我们进行建模的网络是没有延迟的（除非另有说明）。 一旦矿工公布了一个区块，其他所有矿工立即知晓。 同样，一旦交易被宣布，所有的矿工立即了解其存在。我们的模拟器能够模拟这两种类型的延迟，但是，我们看不出随着延迟的变化会导致结果的任何实质性变化。

最后，我们假设当有R个交易费用时，矿工可以选择包括0到R之间的任何实数值的交易费用。也就是说，交易是足够细分的，矿工可以有选择地选择一组交易，其收费非常接近他们想要的任何实际价值的目标。我们认为这是一个合理的近似，因为每个区块的事务数量足够大。

我们还假定矿工总是有空间来包含所有可用的交易。即是区块大小不足以满足交易需求，我们仍相信所有结果的定性内容继续保持不变，但是定量影响会得到缓解。这种观点得到以下数据的支持：从截至2016年7月11日的最近1000个区块（约一周的价值量）中抽取：在这1000个区块中，702个区块已满。在全部区块中，交易费用总额为0。03 BTC至4。51 BTC。平均值为0。49 BTC，标准差为0。25 BTC，超过平均值的一半。目前还不清楚如何将这些数据外推到未来，但很明显，在一块区块内的可用费用确实会有波动。因此，如果区块大小不足以满足交易需求，即使发现区块之后的可用费用不会为零（如我们的分析），但可能会比（比如说）十分钟之后要低得多。所以即使我们的确切分析不适用于这种情况，我们的直觉知识（intuition）也会继续。

* 1. **什么可能出问题？****挖矿差距（What could go wrong? The mining gap）**

*如果没有区块奖励，在发现区块之后立即挖矿的预期收益为零，但电力成本不为零，这使所有矿工都无法从挖矿获利。*

******（图2）

为了提供有关随时间变化的奖励对比特币有害的信息，我们来看一个例子。想象一下，我们正在前面所描述的模型中，所有的矿工都在使用默认的合规策略（在最长的区块链头挖矿，授权所有可用的交易且立即发布），而且矿工有一定的电力成本来运行矿机（即，运行一台矿机t单位时间需要pt比特币的电力成本）。现在，在找到一个新区块之后，原区块中马上将不会有矿工进行交易，矿工都在开发下一个区块中。 这意味着，在发现新区块之后的那一刻，实际上挖矿的期望回报为零，但是这样做的电费不为零！ 图2展示了如何将这个推理扩展到之后的时间段。 从本质上讲，矿机每运转一次，您都会得到一些预期的回报，这取决于可用的交易费用。 但是，您的矿机每运行一次，您也必须支付一定的电费。因此，运行矿机的预期报酬将不会超过电力成本，除非可以包含一定数量的交易费用。如果a是单个矿机产生的总（有效）哈希算力的一部分，那么矿工在发现一个新区块，挖矿再次盈利之前必须等待t = p / a个时间步。

在本论文的完整版本中，我们更详细地讨论了这种挖矿差距的影响，并发现它会导致矿工在原有区块和新区块到达之间的、用来挖矿的时间越来越少（最终下降至难以补偿）。显然，这会对比特币安全性产生负面影响，因为网络中的有效哈希算力会下降，而恶意矿工会更容易将区块分叉。当然，每隔十分钟打开一次矿机，实际上可能是不可行的。尽管如此，这一分析表明，有策略的矿工可能会寻找方式背离默认协议，因为默认协议会让他们浪费电力挖矿近乎无价值的区块。

* 1. **挖矿策略的正式语言（Formal language for mining strategies）**

在本文的其余部分，我们将重点放在挖矿策略上，这些策略总是在同一加密货币中挖矿，但是在选择如何构建区块以及在发现区块后如何处理时可能会背离默认协议。我们考虑了各种已知的和新颖的比特币挖矿策略。 所有这些策略都可以形式化为总体相同的结构。 在每一个瞬间，每个矿工做出几个明确的决定：

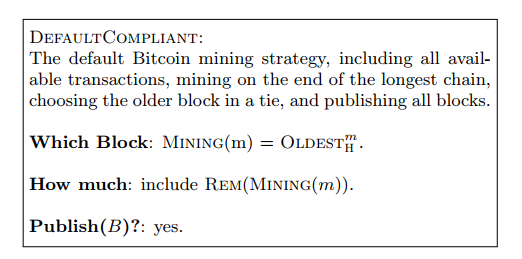
1. 要扩展哪个区块。
2. 在他们正在解决的区块中包含了多少可用的交易（和相关的交易费用）。
3. 对于每个未发布的区块，是否将其发布。

第一个决定是要扩展哪个区块。例如，默认合规的矿工选择在他们知道的最长的区块链上挖矿，而在有多个与最长链相连的区块的情况下，他们将倾向于在他们意识到的这些区块中的第一个里挖矿。这个决定形成了一个关于挖矿策略如何确定它想要支持哪一个分叉的基础，或者是矿工是否想要创建一个新的分叉。下一个决定是需要多少可用的交易费用。同样地，作为一个例子，默认合规的矿工将计入在他们的区块中的、他们知道的、所有未获得的（unclaimed）交易费用。最后的决定是何时发布区块。当一个矿工在一个区块挖矿，只有他们知道它的存在。在任意时刻，矿工可以选择是否广播给其他矿工他们找到的区块。这允许了这样的挖矿策略，即矿工故意选择不揭露他们的区块（比如自私挖矿[9]）

为了更加严格地描述挖矿策略，我们定义以下概念：首先，对于一组交易T，我们将滥用数学符号，并也用T来表示交易中包含的总交易费用。对于一个区块B， 我们将用Tx（B）表示为包含在区块B中的一组交易，并且用Rem(B)表示为区块B之后的剩余交易。也就是说，Rem（B）包含所有已经公布的交易，而不包括在B或B的任何一个前身（predecessors）（因此，这是一个随时间变化的集合）。 我们还将使用Height(B)来表示区块高度（即在区块B处结束的区块链的区块链高度），用H表示当前已经发布的[[2]](#footnote-2)最长链的高度和用Owner(B)表示产生区块B的矿工。

当一个矿工m决定在哪一个区块高度i来扩展的情况下，本文中考虑的所有策略矿工都首先选择一个他们自己挖矿的区块(Owner(B) = m)。 此外，本文中的所有策略都避免了在多个同一高度的区块上挖矿，因此如果存在Owner（B）= m的高度为i的区块，则它将是唯一的一个特例。 如果m在高度i没有生成任何区块，则默认客户端将选择m知道的第一个区块。所以我们定义Oldestm i是矿工m产出的高度i的唯一特例区块（如果这个区块存在的话），或者是矿工m知道的第一个高度为i的区块。注意，如果i = h，那么区块m将采用默认策略扩展。我们还定义了Mosti是高度i的区块的剩余的最大交易费（正式定义：argmaxB|Height(B)=i{Rem(B)}）。注意当REM（B）随时间的变化，只有当一个新的区块高度为i的区块被公布，Mosti才会变化。最后，如果m存在的话，我们用Mostm i表示由区块m产生的区块高度i，否则Mostm i表示高度i的区块的剩余最大交易费，。

现在我们可以正式定义我们考虑的挖矿策略。 我们将策略模拟是时间驱动的（而不是事件驱动）：在每个微小的时间步中，矿工必须决定扩展哪个区块（由Mining（m）表示），区块要包含哪些事务集，以及他们自己的每个未发表的区块是否公布。请注意，通过公布区块B，我们的意思是确保网络中的每个节点都知道B及其所有前身（predecessors），并且不关心具体的实现这点的物理措施。 在这种语言中，默认挖矿策略将被形式化如下：



（下为策略译文）

默认合规：

默认的比特币挖矿策略，包括所有可用的交易，在最长区块链的末尾挖矿，选择较老的区块，并公布所有区块。

**哪个区块**：: Mining(m) = 。

**多少交易费用**：包含 Rem(Mining(m))。

**是否公布区块（B）**:是。

（译文结束）

1. **挖矿策略模拟器**

为了更清楚地分析比特币挖矿激励变为基于交易费用而不是基于区块奖励的博弈理论，我们开发了一个多功能的比特币挖矿策略模拟器[[3]](#footnote-3)。在这里，我们讨论我们的模拟器能够实施的策略，我们的模拟器探索策略空间的过程，该模拟器的配置参数，及其局限性。

* 1. **策略、回合和比赛(Strategies, Rounds, and Games)**

首先，在进入细节之前，我们来描述我们模拟器的基本单元，以及它们是如何相互作用的。

**策略**。模拟器被设计为能够运行符合3。3节描述的策略空间中的任何策略。 也就是说，每个策略都由一个函数完全地定义，该函数输出一个要扩展的区块，一组要包含的事务以及一个规则决定是否发布找到的任何区块。所有的这些功能都可以被输入公共的任何信息，包括所有公布的区块和交易。

**回合**。我们的模拟器是时间驱动的，而不是事件驱动的。我们之所以做出这个决定是因为我们希望能够方便地向模拟器添加新的策略。在事件驱动的模拟中，新策略将受到当前可能事件列表的限制。然而，在我们的基于时间的模拟中，如何做出上述决定的任何策略的任何细节都可以轻松实现。一回合是我们的模拟器中的最小时间单位（目前，模拟器找到一个块的时间是整个网络的1/600）。在一个回合中，每个矿工首先接收（他们知道的）区块链和所有交易（他们知道的）并决定哪个区块（试图）扩展，以及哪个区块包含哪些交易。然后会有一个随机检查（作为矿工哈希率（hash rate）和当前网络难度的函数）来确定矿工是否成功找到一个区块。然后，矿工决定是否公布区块和将哪一些未公布的区块公布。一轮的持续时间是一个可配置的参数，我们将在4。3节中简要讨论。

**比赛**。一次比赛涉及到参数的设置，如选择一些矿工，分配他们的策略和哈希算力（hash power）等（详见4。3节）。一旦这些参数设置好了，比赛就会运行几轮，并跟踪每个矿工获得的奖励。

**模拟**。 一个模拟可能包括一次单一的比赛（看看某些策略如何相互抵消），或几个比赛之间的参数调整。 例如，为了模拟随时间学习的矿工，我们让他们参加几个比赛，根据过去比赛的结果决定在未来的比赛中使用哪种策略。 原则上，任何参数都可以在比赛之间进行调整。

* 1. **策略探索（Strategy exploration）**

在我们的几次模拟中，我们希望矿工能够利用好他们正在执行的策略，来模拟有策略的矿工随着时间的推移是如何适应的。为了实现这个目标，我们运行了几场比赛，每场比赛都有数百名矿工。矿工选择策略的概率与该策略在过去的成功程度成正比。正式地说，我们的模拟器中的矿工们进行不后悔学习（no-regret learning），这是一个在博弈论理论中广受欢迎的标准学习概念。这是因为在任何重复的比赛中，每个玩家分别进行不后悔学习（no-regret learning），重复的比赛都会收敛到粗略的相关均衡[1,2]。此外，众所周知的许多简单的不后悔学习（no-regret learning）算法可以快速地收敛（即在多个可能的策略的次线性回合中）[5,6,4,14]。如果一个矿工不后悔的话，他们总的回报总是至少和他们选择在每场比赛中使用的“最好”的战略一样好。同样地，一个粗略的相关均衡是策略概况上的联合分布，使得在这一相关均衡中每个矿工通过依照（follow）均衡来获得更多的期望收益，而不是偏离任何可能的策略来取得收益。

这些学习算法都为每一种策略保持一定的权重，并根据比赛结果的好坏调整策略的权重。我们的模拟器提供了两种更新规则的方案。第一个方案是用于对抗强盗（adversarial bandits）[ 5, 6 ]学习的EXP3算法的具体应用。这个更新规则为每个矿工的后悔（regret）提供了一个理论保证，把每个矿工的后悔（regret）作为比赛数量和更新规则中的可调参数的函数 （该符号无法显示，此处为截图。）。第二个方案是基于与专家学习的乘法权重更新规则（MWU） [4,14]。 我们发现MWU在计算上是昂贵的，所以我们使用了成本较低的替代方案。 这意味着对后悔（regret）的界限没有理论上的保证。 但实际上，这个更新规则明显更快，并且很快收敛到粗略的相关均衡。 有关这些更新规则的进一步讨论，请参阅完整版本。

* 1. **多功能性（Versatility）**

我们的模拟器有许多可配置的参数：

**策略**。重申一下：我们模拟器中的每个矿工都被分配了一个时间驱动的策略，它选择一个区块是否要扩展，一组要包含多少事务，以及是否发布找到的任何区块。适合这个框架的任何策略都可以在模拟器中实现。为了设计一个新的策略，用户可以创建一个新功能，将当前比特币网络的公开状态（区块链和可用交易费用）和使用该策略的矿工作为输入值，然后该函数将使用该信息确定要扩展的区块，以及在下一个区块中包括多少交易费用，最后，用户要去发布规则，并添加一个规则来决定是否公布找到的任何区块。

**哈希算力****（Hash Power）**。 每个矿工m都会被分配一个哈希算力αm。 任何数量的矿工，任何αm的值，比如，都可以支持。

**一回合持续时间**。可以设置一回合时间，这样每r回合网络就可以找到一个区块，对任何r > 0。

**奖励**。在每场比赛结束时，将根据矿工在最长区块链中的区块进行奖励。 他们收到的奖励是b每区块（固定奖励），加上交易费用。交易费用每回合都在系统中累积。这两个参数都是可配置的。

**成本**。每个矿工m都有一个可配置的参数，表示矿工的成本（即电力成本）。 对于我们的模拟，我们总是设置= 0，因为我们不关注这方面的挖矿。

**延迟**。如果需要，可以将延迟引入到模拟中。有一个可配置的参数λ，这样当区块被公布时，其他矿工意识到这个区块存在需要λ回合。其他矿工听到交易的延迟也可以配置——通过修改策略，随机地“假装”他们没有听说过一些交易，这是目前最容易做到的。

**学习参数**。我们的学习规则由参数化。对于EXP3，通常设定，其中n是所考虑策略的数量，T是所进行比赛的数量。对于MWU（以及我们的“类似MWU”的更新规则），通常设置。 更小的有助于更多的探索（exploration），更大的有助于更多的开发（exploitation）。

**原子矿工与非原子矿工**。 我们认为一些矿工是*原子（atomic）*矿工，如果这些矿工中有许多是有限的，而且每个矿工有一个有限的哈希算力（Hash Power）的一部分。 这样的矿工可能有兴趣牺牲与现在挖矿的区块有关的立即收益，以便为未来挖矿的区块获得更大的收益。而非原子矿工无限小，但有无限多个数。 当这样的矿工发现一个区块时，他们只关心最大化他们与这个区块相关的收益（因为他们将来再也找不到另一个区块）。

显然，我们的模拟不能创造无限多的矿工，但我们可以在功能上模拟他们。为了模拟非原子矿工的α分之一正在使用策略s，我们改为创建一个具有哈希算力（Hash Power）的α分之一的单个原子矿工，并确保这个矿工的所有策略决定仅将整个网络可用的公共信息作为输入，且处理“自己的”区块与普通区块没有不同。

当然，现实世界是原子的。但是，比较这两种模式之间的模拟结果是非常有帮助的，比如隔离只有当矿工是原子矿工的时候才会出现的行为（例如：自私挖矿），直观地说，这种行为会因为大矿工（的自私挖矿）而“恶化”。

* 1. **实施和性能(Implementation and performance)**

模拟器是用C ++编写的，而且模拟器运行时间与比赛数量，每场比赛回合数和矿工数量成线性比例。作为算法的一个优化，我们让矿工确定他们是否在每个时间步中找到一个区块，然后才真正计算他们将要扩展的区块以及它们将包含多少交易费用。我们发现，为了获得准确的结果，游戏需要包含足够的回合数，以便对于每个策略，能让使用它的矿工一起找到几十个区块。我们也发现，我们的MWU需要几百次比赛才能达到均衡状态。 在配备2.7 GHz英特尔酷睿i5处理器的商用笔记本电脑上，与200位矿工进行单一比赛模拟，平均间隔时间为600轮，共计600万轮（≈10000个块）需花费大约400 秒（6.5分钟）。

**局限性**。模拟器目前的局限性是交易费用只能按照统一的时间进行模拟。另外，模拟器不能模拟动态矿池，而是只能将其视为一个单一矿工，其哈希算力等于矿池动力。因此不允许考虑如[8]中所述的攻击。

1. **新的异常挖矿行为**

在本节中，我们将研究交易费用模型中可能出现的异常挖矿行为，这些行为在区块奖励模型中不会出现。具体而言，我们认为：

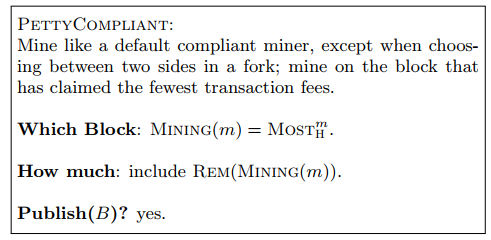
•一旦交易费用到达，自利的矿工选择采取PettyCompliant策略而不是DefaultCompliant（默认合规）策略是一个合理的期望。

• 采取PettyCompliant策略的矿工在网络中的存在开启了一系列侵略性的策略的领域，且对比特币的稳定性产生了不利影响。

* 1. **第一阶段：Petty compliant（Phase One: Petty compliant）**

*观察：默认客户端在最老的区块上挖矿的行为不是最优的。矿工们可以通过进行在大部分交易费用无人认领的区块挖矿的行为来做得更好。*

考虑有分叉的情况：两个区块被捆绑在最长的区块链上。传统的行为，以及默认客户端[[4]](#footnote-4)被编程的行为，是让矿工选择两个潜在区块头中较早的一个。然而，矿工任意地打破僵局（tie-break）的行为真的没有任何代价。特别是，如果矿工打算将所有无人认领的交易费用包括在他们的区块中，那么矿工的收益最大策略就不是在最古老的区块挖矿，而是在无人认领的交易费用最多的区块挖矿。因此，一个有策略的矿工会希望在而不是来挖矿。 我们把这个策略称为petty compliant，因为它仍然是在一个最长的区块链上进行挖矿，也包括所有可用的交易，并且公布所有被发现的区块（就像默认合规的矿工一样）。这只是为了获得更大的收入而以一个“petty”的方式在最长的区块链之间打破僵局（tie-breaking）。



（下为策略译文）

PettyCompliant：

像一个默认合规的矿工一样挖矿，除了在一个分叉的两边进行选择时；会在交易费用被认领最少的区块挖矿。

**哪个区块**： Mining(m) = 。

**多少交易费用**：包含 Rem(Mining(m))。

**是否公布区块（B）**:是。

（译文结束）

如果分叉曾经存在，那么 PettyCompliant策略严格地优于DefaultCompliant（默认合规）策略. 除了矿工需要在两个相同区块高度的区块之间进行选择的情况外，两者是相同的。在这种例外情况下，PettyCompliant策略总会决定在最大化矿工的奖励的区块挖矿，而DefaultCompliant策略可能不会。在我们的挖矿策略模拟器中，我们比较了DefaultCompliant策略和PettyCompliant策略，并且事实上看到执行PettyCompliant策略的表现总要优于DefaultCompliant策略，不管其他矿工在任何一次模拟中有多少（足够学习新区块和事务）的延迟。

请注意，执行PettyCompliant策略的矿工的存在本身并不一定是有害的：那么如果矿工在自然发生分叉的罕见事件中以不同的方式打破僵局（tie-breaking）呢？ 当其他有策略的矿工注意到执行PettyCompliant策略的矿工的存在并选择以更激进的策略利用这点时，问题就出现了。 在本节的其余部分中我们将看到一些例子。执行PettyCompliant策略的矿工的存在也以惊人的方式影响其他的异常策略。例如，一个自私矿工（在第6节中会有更多的讨论），对付执行PettyCompliant策略的矿工比DefaultCompliant更好。

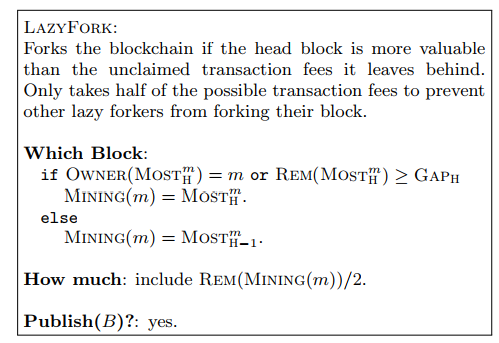
* 1. **第二阶段：懒惰削弱（Phase Two: Lazy Undercutting）**

*观察：一旦一部分矿工执行PettyCompliant策略，其他的矿工可能通过故意将链分叉而获利。*

更侵略性的策略的重要见解是，一个偏差的（deviant）矿工可以激励执行PettyCompliant策略的矿工扩大他们的区块，即使在几分钟前发现了一个相同高度区块的老区块，例如扩大该区块的直接前身（direct predecessor），包括略少的交易费用。如果当前未经授权的交易费用大大低于当前的MostH区块所包含的交易费用，那么尝试用一个新的区块高度H的区块替代MostH，而不是继续在区块顶部，是符合矿工利益的。我们把这称之为削弱（undercutting）。

那么一个有策略的矿工可以怎样利用这点呢？ 首先他们可能比较他们通过继续（continuing）或削弱（undercutting）（虽然区块仍然会成为新的MostH）两种方式可以获得的最大回报，而在能获得更大回报的区块头挖矿。那么，为了保护自己免受未来采用同样的规则的削弱者（undercutters）侵略，他们可以收取余下交易费用的一半。由于用来选择这些参数的松散的推理，我们称之为LazyFork策略。由于用来选择这些参数的推理是松散的，我们称之为LazyFork策略。

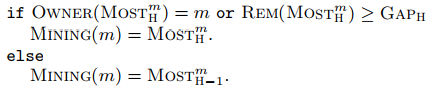
虽然执行PettyCompliant策略的矿工的存在本身是相对良性的，但执行LazyFork策略的矿工的存在是不好的：他们经常会决定故意孤儿化区块，以获得更大的回报。除了令何时区块在最终的最长区块链中变为“安全的”产生不确定性，这还会降低网络的有效哈希算力（Hash Power），使比特币更容易受到双重支出攻击（double spend attacks）。为了正式定义LazyFork和其他削弱（undercutting）策略的清洁程度（cleanliness），我们引入了符号，表示一个矿工在将变成的区块顶挖矿时可以包含的最大交易费。



（下为策略译文,伪代码不译）

LazyFork：

如果头区块比其剩下的无人认领的交易费用更有价值，则将区块链分叉。只收取可能的交易费用的一半，以防止其他执行LazyFork策略的矿工（lazy forkers）将自己的区块分叉。

**哪个区块**：

**多少交易费用**：包含 Rem(Mining(m))/2。

**是否公布区块（B）**:是。

（译文结束）

* 1. **第三阶段：积极削弱（Phase Three: Aggressive Undercutting）**

*模拟结果：当矿工制定策略时，日益激进的**削弱（undercutting）行为将会发生。*

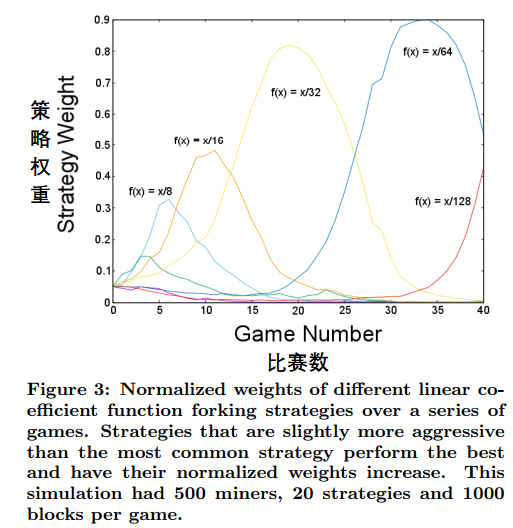
一旦矿工考虑削弱（undercutting），他们也可能试图侵略性地优化：最大限度地利用他们区块中包含的交易费用之间的权衡，并最大限度地减少系统中其他矿工削弱（undercut）他们区块的机会（而不是使用较少原则推理出的 LazyFork）。 我们这样定义这些策略，使得当它们被提交给Rem（Mining（m））= x时，它们将授权f（x）交易，对于所有x的f（x）∈[0，x]的那些f(·) ，并把他们称为分叉者（forkers）。

虽然原则上，forkers可以考虑退回几个区块来削弱（undercut），但我们研究的策略只考虑在区块高度为H或H – 1的区块挖矿。当然，这将是一个有趣的方向，看看是否可以通过考虑区块高度为H – 2或更小的区块来获得额外的收益，但是当forkers回归一个块时，我们已经发现了有趣的行为。

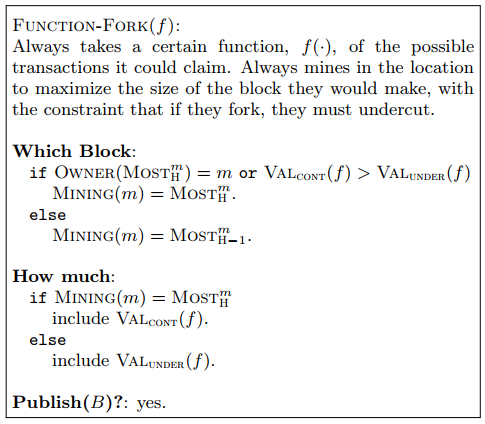
一个正在分叉矿工查看区块高度为H的他们可以扩展的区块的函数，并且在这个集合内只考虑扩展，因为它留下了最多的剩余交易费用。 如果矿工确实选择在区块顶挖矿，我们称之为继续（continuing）。矿工也研究区块高度H -1的潜在区块，再次考虑仅从该组扩展区块。如果矿工确实选择在区块顶挖矿，我们称之为削弱（undercutting）。当决定是否继续（continue）或者削弱（undercut）时，一个分叉者（forkers）只是简单地观察到他们会选择通过继续（continue）来认领，或相反的，如果他们（取最小值是因为他们必须真正地削减才能激励未来的矿工选择他们的区块）。 所以对于给定的f，我们可以定义：



**图3：一系列比赛中不同线性系数函数分岔策略的归一化权重。 比最普通策略稍微更有侵略性的策略表现最好，并使其标准化权重增加。 这个模拟每次比赛有500名矿工，20个策略和1000个区块。**

**

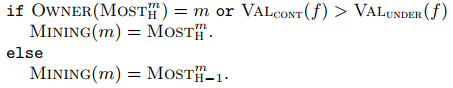
如果，则可以通过继续（continue）获得更多的奖励。 否则，通过削弱（undercut）可以获得更多的回报。形式上，对于任何函数f（·），归纳如下正式策略：

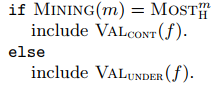


（下为策略译文）

Function-Fork(f):

f（·）对它可能认领的交易总是采取一定的功能。总是在最大化区块的大小的地点挖矿，并限制：如果他们分叉，他们必须削弱（undercut）。

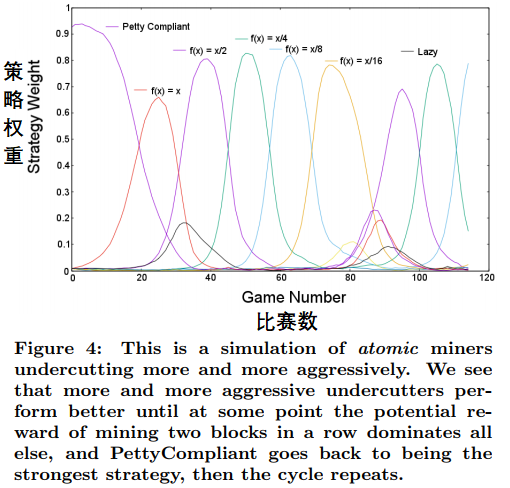
**哪个区块**：

**多少交易费用**：

**是否公布区块（B）**:是。

（译文结束）

f（·）的任何合理选择将单调递增, 这意味着  会一直大于 , 所以关于是否继续或者减少的决定将归结为与GapH的比较。



**图4：这是一个原子矿工越来越侵略性地削弱的模拟。我们看到，越来越侵略性的削弱者表现更好，直到在某一时刻，连续挖矿两个区块的潜在回报超过其他所有的回报，PettyCompliant又变回最强大的战略，然后这个循环重复。**

f（·）的一个自然族（natural family）是线性函数（即，对于某个k∈[0,1]，f（x）= kx）。如果我们采取一些策略，并且让非原子的有策略的矿工学习许多次比赛中性能最好的策略，我们会得到图3中的情况。我们所看到的是：当大多数矿工使用Function-Fork(kx)时，最好的反应是使用，因为比k小一些（即更有侵略性地削弱一点）。所以最终，在我们的模拟中最小的系数成为主导。如果我们考虑原子矿工，我们观察图4中的行为：一旦Function-Fork(kx)在k足够小的情况下占主导地位，那么矿池的可用交易费用将变得相当大（稳定在1 / k左右）。因此，如果采取PettyCompliant策略的矿工有幸能连续挖矿两个区块，他们将获得巨额回报（请注意，对非原子矿工，这种事件的概率为零）。 随着主导的K变得越来越小，最终从这个不太可能的事件中获得的巨大回报会占主导地位，PettyCompliant又成为主导策略，循环重复。

* 1. **削弱均衡（An Undercutting Equilibrium）**

*分析结果：所有矿工都使用相同的削弱策略。这导致越来越多的交易积压。*

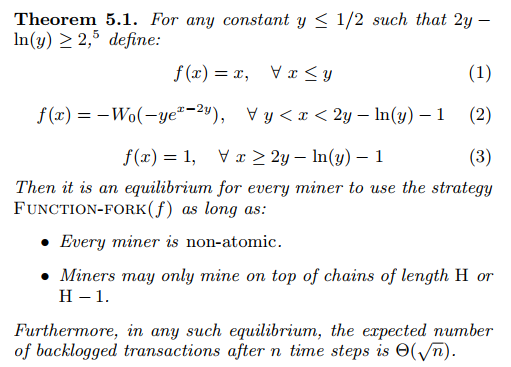
线性函数分叉当然是一种自然的策略类型，但是我们在前面的模拟中表明，如果矿工只考虑这些策略，他们的长期行为可能是不稳定的。 我们在这一部分的目标是了解哪些削弱行为是稳定的。

我们的方法是找到一个函数f（·），使FunctionFork（f）是一个均衡。也就是说，只要其他矿工正在使用策略Function-Fork（f），这样做也符合你的利益。换句话说，我们希望找到一个函数fork（f）是所有其他矿工自己使用Functionfork（f）的情况下的最佳选择。我们提供现在的直觉知识（intuition）为什么我们提出的f（·）会产生一个均衡。

那么一个策略对其他矿工行为的最佳选择意味着什么呢？ 回想一下，一个策略提出了哪个区块要扩展，要交易多少交易费用，以及根据当前持有的信息公布哪些区块。 如果一个策略能最大限度地提高矿工的预期报酬（考虑将来的事件，尤其是当前区块是在最后一个最长链的可能性），那么这个策略就是最好的选择。尤其是，最佳回应必须至少与在同一地点挖矿并发布相同区块的所有其他策略一样好（但在包括哪些交易方面有所不同）。

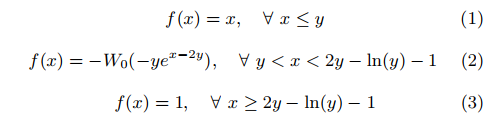
为了得到潜在平衡点必须满足什么条件的直觉知识（intuition），我们首先考虑一个矿工已经决定在最长区块链上继续（continue）的决定和决定包括多少交易费。如果F表示包括交易费用的数量，则将π（F，f，x）定义为该区块被包括在最终的最长区块链中的概率，其条件是在该区块中包括F 比特币的交易费用值，所有其他的矿工使用策略Function-Fork(f)和（注意π是明确的）。那么矿工的预期奖励，如果他们幸运地找到一个区块的话，现在将是F·π（F，f，x）。

那么最佳选择就是包含argmaxF≤x{F·π（F，f，x）}交易费用。 Function-Fork（f）策略将建议包括f（x）交易费用。所以要让Function-Fork（f）成为其他使用Function-Fork（f）策略的矿工的最佳选择，那么对所有x 有。 请注意，这是对f的一个强有力的条件，因为其他矿工正在使用Function-Fork（f）的事实会影响π（F，f，x），而我们也希望这个矿工的最佳选择有。

在这一点上，我们表明有一个连续的分片（piece-wise）可微函数f（·）满足这个条件。我们还表明，结合f（·）是单调递增这一事实，在一些假设（我们将在定理后讨论）下，这足以使Function-Fork（f）成为一个均衡。在下面的定理陈述中，是所有满足的和的Lambert W函数的上分支。 通过显示积压的交易数量与无偏一维随机游走（an unbiased single-dimensional random walk）之间的联系证明了定理的“更多...”部分。

***定理5.1***

对于任何常数y≤1/2，2y - ln（y）≥2[[5]](#footnote-5)，定义：



那么只要每个矿工使用策略Function-fork（f）就是一个均衡：

• 所有矿工都是非原子的。

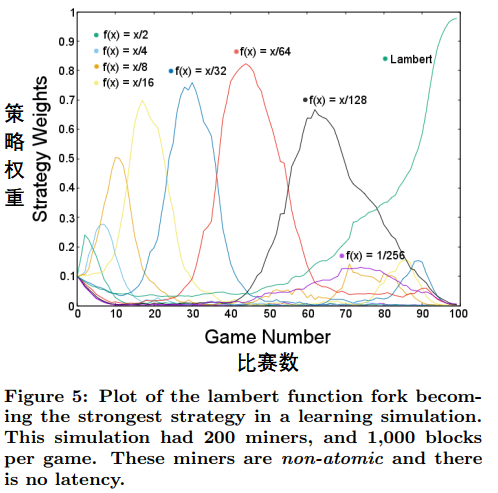
• 矿工只能在长度为H或H - 1的区块链上挖矿。

此外，在任何这样的均衡中，在n个时间步之后的预期积压交易的数量是Θ(√ n)。

定理5.1的证明出现在完整版本中。为了理解定理5.1的影响，首先考虑区块奖励模型。对于非原子矿工，DefaultCompliant策略非常平衡，而且这个结果对延迟的一般模型（见完整版中的证明）是强健的（robust）。但是当我们转向原子矿工的时候，像自私挖矿这样的策略就会出现，而均衡就会变得混乱（如果原子矿工存在的话）。

现在，在交易费模型中，即使矿工是非原子的，平衡行为也是复杂的，而不受欢迎的，正如我们刚才所表明的那样。因此，我们应该认为对原子矿工的分析结果会更加混乱。

**图5：在学习模拟中，lambert函数分叉成为最强的策略。这个模拟有200名矿工，每场比赛1000个区块。这些矿工是非原子的，且没有延迟。**

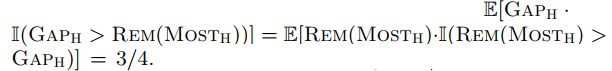


* 1. **削弱没有策略的矿工（Undercutting Non-strategic Miners）**

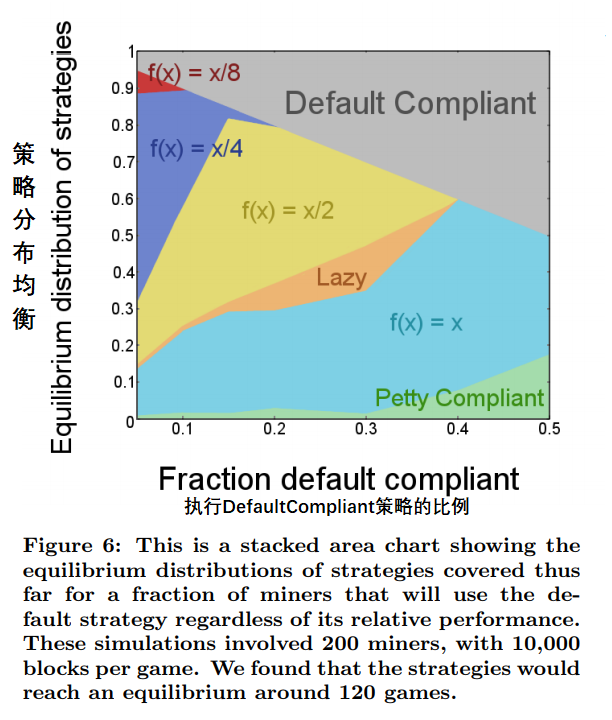
*分析和模拟结果：即使66％的矿工保持default compliant（默认合规），削弱（undercutting）策略同样是盈利的。*

我们在前面的分析和模拟中假定所有的矿工都是策略的学习者。虽然我们从这个分析中清楚地学到了很多东西，但是现实情况下还需要考虑一些矿工会固执地（或者诚实地，取决于你的观点），甚至在欠佳的情况下继续执行DefaultCompliant策略。如果很大一部分矿工是没有策略的矿工，那么功能分配就会立即变得不那么有利可图，因为实际上只有一小部分的网络挖矿是在你削弱（undercut）的时候在你的区块顶挖矿的。特别是，如果其他100％的矿工是没有策略的，那么削弱（undercutting）根本没有用处。

在本节中，我们详细介绍了当不同的比例的矿工是没有策略的时，我们的模拟结果。在这些模拟中，我们惊奇地发现，如果我们修复一小部分的网络来让其总是使用DefaultCompliant策略挖矿，那么经过一系列的比赛之后，由我们的更新规则而产生的剩余策略的策略分布将会稳定下来。此外，多种策略将以稳定的分布呈现（即，这种策略分布似乎形成一种均衡）。 图6显示了我们的不同比例的矿工拒绝放弃DefaultCompliant策略的均衡模拟结果的堆积面积图。这个图有很多有趣的特点，但我们关注的是：即使大多数矿工选择保持DefaultCompliant（其余的矿工则是有策略的），那么分叉策略开始变得可行。

理论分析的确预测了FunctionFork（x）的持续存在，直到2/3的矿工仍然保持DefaultCompliant策略。为了看到这一点，假设每个矿工都是系统目前的DefaultCompliant或PettyCompliant，我们想看看这是否有利于一个执行PettyCompliant策略的矿工切换到FunctionFork（x）。在任何时候，考虑当前的MostH。然后，如果矿工执行PettyCompliant策略，他们将总是尝试继续（continue），如果他们发现一个区块（因为网络中没有其他人在削弱（undercutting）），将会得到Rem（MostH）。如果相反地，他们运行FunctionFork（x），那么每当Rem（MostH）> GapH，他们将继续（continue），否则削弱（undercut）。当他们继续（continue）时，他们将永远得到Rem（MostH）。当他们削弱（undercut），他们将包括GapH交易费用。如果下一个找到一个区块的矿工是执行PettyCompliant策略的（或者这个矿工是），那么这个削弱（undercut）就会成功，矿工将获得奖励GapH。但是，如果是执行DefaultCompliant策略矿工发现下一个区块，则削弱（undercut）会失败，并且他们什么也得不到。因此，如果y是执行DefaultCompliant策略的网络的比例，我们可以看到FunctionFork（x）获得的预期回报与以下成正比：[[6]](#footnote-6) 最后，因为和都是独立同分布的随机变量的均值为1的指数，我们有 因此，每当y≤2/3时，FunctionFork（x）的回报至少为1，因此其比PettyCompliant策略（获得期望的回报只有1）更好。

**图6：这是一个堆积面积图，显示了一些矿工的策略平衡分布，这些矿工将执行默认策略（DefaultCompliant），而不管其相对的性能如何。 这些模拟涉及200名矿工，每场比赛10000个区块。我们发现这些策略将在120场左右达到平衡。**

****

1. **利用交易费用的自私挖矿**

自私挖矿是Eyal和Sirer [9]首先确定的一种偏差策略。从本质上讲，一个自私的矿工选择不要一发现区块就立即公布，而是扣留区块，以希望欺骗网络的其余部分，浪费他们的算力在将将被孤立的（orphaned）区块挖矿。

令人惊讶的是，交易费用模型中的自私挖掘策略比区块奖励模型表现的更好。验前（A priori），没有理由期待这一点。在本节中，我们提供了模拟结果，以及一些直觉知识（intuition）和证明这一点的理论分析。从本质上讲，发生的事情是，尽管自私的矿工在任何奖励模式中都在相同比例的区块中挖矿，但是自私矿工的区块往往会变得更大。在区块奖励模型中，这并不重要，因为所有的区块都是相同的，但是在交易费用模型中，这意味着自私的矿工获得更大的回报。

* 1. **自私挖矿策略（The Selfish Mining Strategy）**

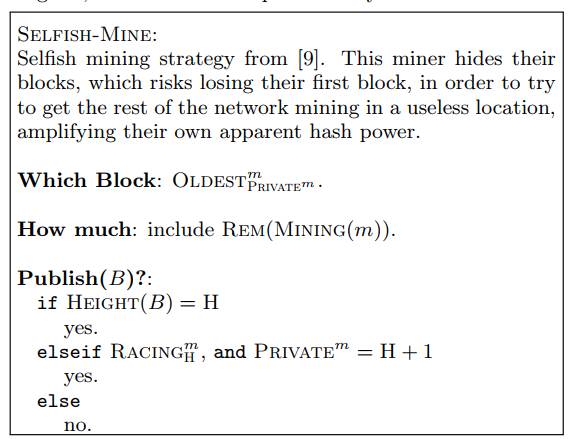
*分析和模拟结果：自私挖矿在交易费用模型中表现得稍微好一点。*

采用自私挖矿策略的矿工的目标是从根本上欺骗比特币网络中的其他矿工，让他们在一个将被孤立的（orphaned）区块顶挖矿。通过让其他的矿工们浪费他们的算力，这个自私的矿工能够夸大他们自己的整个网络哈希率（the overall network hash-rate）。自私的矿工通过保持一个只有他们自己知道的私密区块链来做到这一点。当这个自私的矿工最初发现一个区块时，他们就不会将他们的区块公布到网络的其他部分。他们将继续在他们的私人区块挖矿，希望在网络的其他部分找到区块之前自己找到第二个区块。

如果自私的矿工成功了，现在他们处于非常强大的位置：他们知道一个区块高度为H + 2的区块，而网络的其余部分只知道一个区块高度为H的区块。如果网络的其余部分发现下一个区块的区块高度是H + 1，自私的矿工可以透露自己的私人区块链，公共区块将立即成为孤立的。当然，也许这个自私矿工也会找到第三个区块。在这种情况下，他们处于更好的位置，可以浪费更多的网络算力。但重要的是，有两个或更多的区块领导，这个自私的矿工可以保证剩下的网络正在浪费算力。

当然，自私的矿工也可能在网络的其他部分找到第一个区块之前找不到第二个区块。在这种情况下，他们立即公布他们的区块，以希望其他人首先找到他们的区块。显然这不是很理想：如果他们立即公布这个区块，他们可以保证先被找到。所以有一个权衡 ——阻止这个区块有机会给这个自私的矿工一个长度为二或更多的私人区块链，在这种情况下，自私的矿工会受益，但也有可能导致他们自己的区块被孤立，造成自私的矿工获利较少。

假设这个自私的矿工拥有不到网络整体哈希算力的一半，他们最终将需要发布他们的私有区块链。为了保持我们对交易费用和固定区块奖励之间的差异的关注，我们认为他们只是“纯粹的”自私挖矿，尽管考虑自私的矿工也可能削弱（undercut），或各种其他概论（generalizations）（如[7，19，22]）与[9]类似，我们研究一个自私的矿工将会得到的潜在回报，假设其余的网络是默认挖矿。在我们的分析中，我们也用α来表示自私矿工拥有的总挖矿算力的比例，而用γ来表示在一场竞速（race）中（自私的矿工被触发公布长度为1的私人区块的过程）以网络的其他部分找到下一个区块作为结束的概率，那个自私的矿工的区块并不是孤立的。我们引入记号来表示矿工m知道的最长链的高度（至少和H一样高，如果m保持任何区块私有的话，可能更高）。我们还引入记号作为布尔变量（boolean variable），如果存在两个带有和的块B1，B2，则布尔变量为真（true）。换句话说， 表示是否有两个区块高度为i的竞争区块，其中之一是由矿工m产生的。



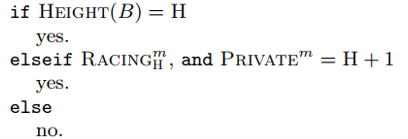
（下为策略译文）

Selfish-Mine（自私挖矿）:

自私的挖矿策略来自[9]。 这个矿工隐藏自己的区块，这冒着失去他们的第一个区块的风险，为了试图把剩余的网络放在一个无用的位置挖矿，从而放大他们自己的哈希算力。

**哪个区块**：

**多少交易费用**：

**是否公布区块（B）**: 

（译文结束）

**分析**

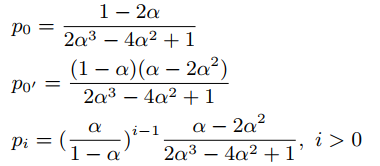
现在我们来分析一个自私的矿工在交易费用模型中得到的回报。部分看起来与[9]中的分析类似。对于到达的每一笔无限小的交易费用，我们希望计算它在自私矿工开采的区块中卷入（winds up）的概率。请注意，如果这个自私的矿工只是使用默认挖矿，不使用自私挖矿，那么这个概率就是α。

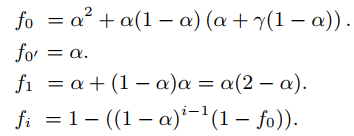
这个概率的决定因素将是自私的矿工的私人区块链的大小。为此，我们定义以下状态（与[9]中使用的状态相同），我们将分别计算每个状态的概率。

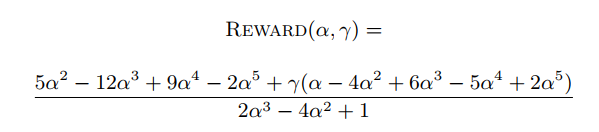
• State 0: 所有人都一致在最长区块链上——= false.

• State i > 0: 自私的矿工m有一个长度为i的私人区块链——  = H + i.

• State : 有区块高度H的竞争区块，其中之一是由自私的矿工产出的，自私的矿工没有私人区块。——= true and  = H.

假设表示在最终的最长区块链中被自私的矿工开采的被卷入（winds up）的交易概率，当交易被公布时系统处于状态s。我们宣称我们可以计算这些概率，并在这样做。如果我们将定义为系统处于状态s的概率，那么我们可以观察到，私人矿工所期望认领的交易费用的部分正好是。 Eyal和Sirer [9]已经计算了对所有s的。 的值是：

为了完成分析，我们只需要为每个s计算。 本文的完整版本包含了所有s的的推导，如下所示：

最后，当α∈（0，.5）和γ∈[0，1]时，我们在完整版本中展示的私人矿工的奖励是

我们提出以下意见：

•模拟证实了以上Reward(α, γ)的解析公式（图7）

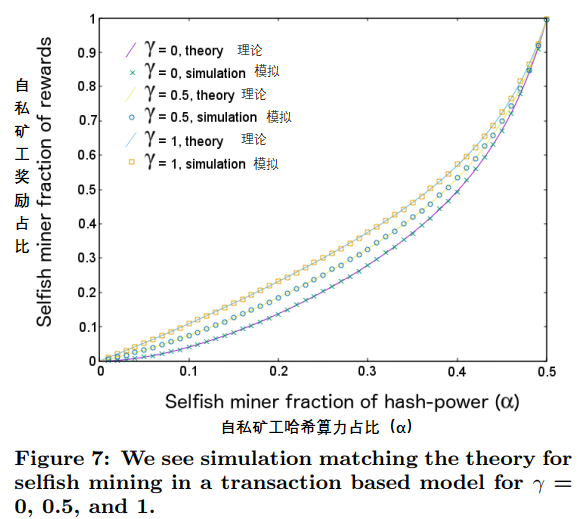
• 这个函数与[9]中有区块奖励的奖励函数非常接近。我们发现，在数值上，在感兴趣的区域内绝对差值绝不会超过0.026。

•对于0≤γ<0.55（特别是对于γ= 0），和所有的α∈（0,0.5），交易费用模型中的回报显然大于区块奖励模型的。

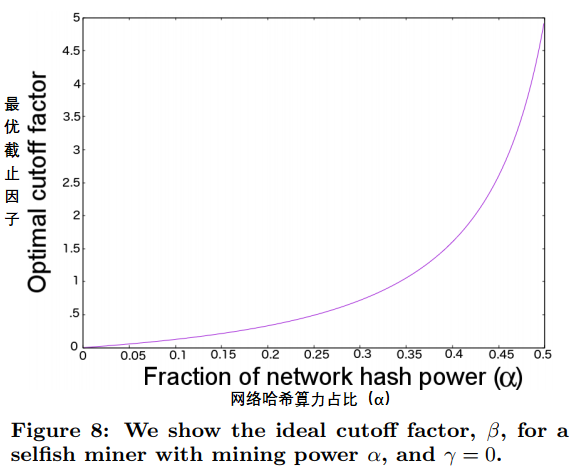
我们为最后这一点提供了一些直觉知识（intuition）。首先，很明显，自私矿工与默认矿工开采的区块的比例与奖励模型无关。所以这个差距必须来自各个矿工发现的区块大小。假设我们处于状态100（我们只是为了举例说明），而自私的矿工的挖矿能力α= 1/10。几乎可以肯定的是，下一个非孤立的区块将被自私的矿工发现。这个区块需要多久才能找到？答案是大约10个时间步。这是因为整个网络大概每一时间步都会发现一个区块，因为这个自私的矿工是唯一一个延伸他自己的区块链的矿工（他的速度是整个网络的十分之一），所以要花十倍的时间。这意味着，当自私的矿工拥有巨大的领导（lead）时，自私的矿工发现的区块与自私的矿工没有领导（或微小的领导）时所发现的区块相比，不相称地过大。所以即使这个自私的矿工赢得相同比例的区块，其中一些区块也比默认矿工赢得的大得多。

**简短的讨论**。本节的重点是强调一个关于令人惊讶的激励问题的例子，这个激励问题在交易费用模型和区块奖励模型之间存在差异，而不是争论自私的挖矿更好（提高得不大）。尽管如此，我们还是希望指出两种模式中自私挖掘之间可能存在的显着差异。首先，在区块奖励模式中，自私的挖矿实际上从来没有立即获利——只有在因网络中有效的挖矿能力较低，问题难度调整时，它才会变得有利可图。这是因为在难度调整之前，这个自私的矿工实际上只是把区块扔掉，而欺骗其余的网络以更快的速度扔掉区块。在交易费用模式中，自私的挖矿立刻就是有利可图的——每一笔到达的交易都在某个地方，所以自私的矿工和默认矿工都不会扔掉奖励。还要注意的是，我们的分析决不会需要难度调整，在它变得准确之前——无论哈希问题（hash puzzles）的难度如何调整或不随时间调整，我们的分析都将持续。而且，如果网络的其他部分已经切换到执行PettyCompliant策略，那么当竞速（race）被触发（因为它被更早地开采并且因此包含更少的交易）时，自私的矿工的区块更有可能获胜。因此，在交易费用制度中执行PettyCompliant策略的矿工的存在通过增加γ间接提高了自私挖矿的表现。

**图7：我们看到模拟与基于交易的模型中的自私挖矿的理论相匹配，γ= 0,0.5和1。**

****

**图8：我们给出了一个挖矿能力为α的自私矿工的最优截止因子β，γ= 0。**

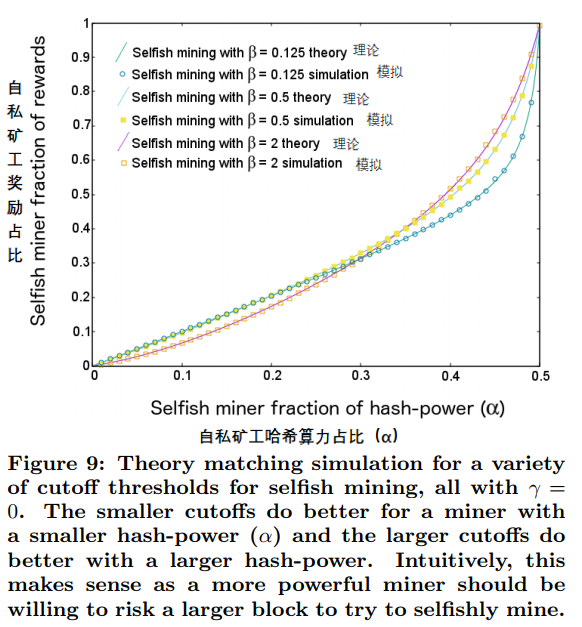
****

* 1. **一种改进的自私挖矿（An Improved Selfish-Mine）**

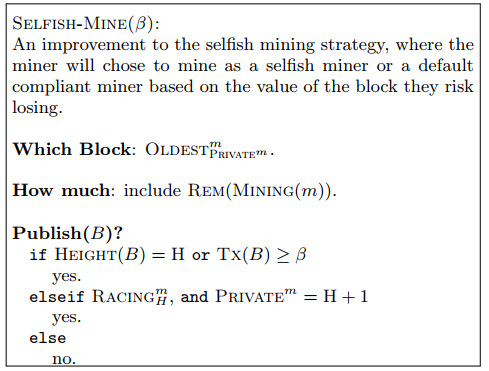
*分析和模拟结果：在交易费用模型中，自私的矿工可以根据区块的价值做出是否隐藏他们的第一个区块的决定。这种自私的挖矿方式得到了严格的改善，并且始终优于默认挖矿和传统的自私挖矿策略。*

在这一节中，我们开发了一个改进的自私挖掘策略。从本质上讲，我们观察到，在交易费用模型中，一个自私的矿工在决定是否隐藏或发布他们的私有区块链（即包含多少交易）时有额外的信息。我们证明，对于所有α，γ<1，我们的策略在交易费用模型中都严格优于默认挖掘和“纯粹的”自私挖掘策略。我们的策略将决定只隐藏“小”区块，包含最多β（包括策略选择的截止参数作为α，γ的函数）交易费用，但会立即发布任何交易费用大于β的“大”区块，以避免失去他们的风险。

直观地说，假设你正在挖矿，并发现自己解决一个新的区块后并在任何新的交易已经公布之前，前一个区块被公布。那么这个区块是毫无价值的，所以不要发布它，为什么不用它来试图自私挖矿呢？没有任何成本，但是无论你的哈希算力如何，你都有一个正向的概率来建立两个区块的领导。同样，想象一下，自从找到最后一个区块以来，一个小时过去了，并且您刚刚解决了一个新的区块，包括在此期间到达的所有交易。 这个区块大概值六个“正常”区块，所以为什么要冒这个风险呢？除非你的哈希算力非常接近50％，否则自私挖掘所带来的预期收益就会因为失去这个异常富裕的区块的可能性而变得相形见绌。所以诀窍就是选择适当的截止因子β作为你的哈希算力α和网络连接γ的函数。

**图9：自私挖矿的各种截止阈值的理论模拟匹配，γ均为 0。对于具有较小哈希算力（α）的矿工而言，较小的截止值会更好，而较大的截止值对于较大的哈希算力效果更好。直观上，这是有道理的，因为一个更强大的矿工应该愿意用更大的区块冒险来试图自私挖矿。**

请注意，，而。所以很显然，如果选择β的最优选择，将会产生一个等于或者优于两者的策略。使用类似于6.1节的分析，我们可以计算矿工获得的期望报酬，其中α为挖矿算力的占比，γ为赢得比赛的成功概率，并使用策略Selfish-Mine（β）。在完整版本中包含派生。



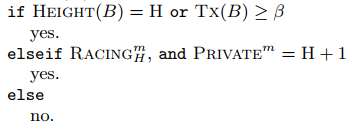
（下为策略译文）

Selfish-Mine(β):

自私挖矿策略的改进，矿工将选择作为一个自私的矿工挖矿或作为一个默认合规的矿工根据他们用来冒险的区块的价值。

**哪个区块**：

**多少交易费用**：

**是否公布区块（B）**: 

（译文结束）

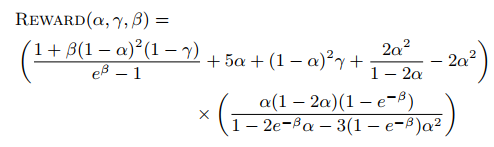
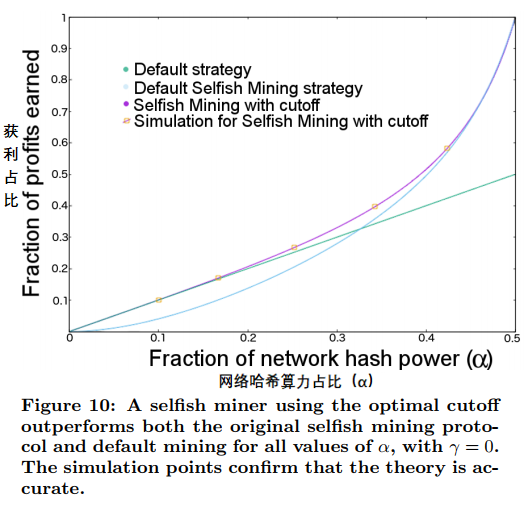


图8包含的曲线图显示了当γ= 0时β的最优选择关于α的函数从该曲线中可以看出一些值得注意的点：当α→0时，最佳的β也趋向于0。 当α→1/2时，最优的β→∞。 图9绘制了我们的理论预测与模拟结果的对比，证实分析是正确的。

****

**图10：一个自私矿工使用最优的截止时间，对于所有的α值和γ= 0，优于原来的自私挖矿协议和默认挖矿策略。模拟点证实，理论是准确的。**

我们用图10来总结这一部分，图10显示了默认挖矿策略，自私挖矿策略和有一定范围内的α和γ= 0的最佳截止值的自私挖矿策略的（理论）性能。注意，在一些范围内，收益是相当显着的。具体来说，当α= 1/3时，自私挖矿策略和默认挖矿策略都可以达到≈1/3的预期回报，但是利用最佳截止的自私挖矿策略可以获得≈.38的预期回报，提高了13.6％！

1. **对比特币的影响和对加密货币设计的教训**

我们认为交易费用制度中的异常挖矿策略可能会损害比特币挖矿的稳定性并危害整个比特币生态系统。在一个具有不断分叉的分支区块链中，攻击者的有效哈希算力（hash power）被放大，因为他总是会在自己的区块挖矿，而其他的矿工则不统一。这将使用远少于51％的哈希算力进行“51％”攻击成为可能。

同时也可能会出现许多意想不到的副作用。在区块大小的争论中，人们常常认为或者假设区块链中的空间是稀缺的资源，市场将会出现，用户可以通过支付足够大的交易费用来加速交易的确认。但是，如果矿工们在解决区块中问题时故意“把钱留在桌上”，就像在削弱攻击（undercutting attacks）的情况下一样，这样的话就打破了这个假设。这是因为削弱（undercutting）矿工并不是想要最大限度地提高他们认领的交易费用，也没有强烈的理由来优先处理高额交易[[7]](#footnote-7)。换言之，区块大小对区块中的交易总规模施加了限制，并且因为受到削弱的威胁而对总交易费用施加了另一个限制。两者以复杂的方式相互作用。我们认为，我们的结果将在一个可用区块的尺寸远小于需求，但在数量上减少削弱的影响的世界中继续存在（见3.1节结尾）。然而，更深入地理解这一联系是未来研究的一个重要方向。

尽管我们的结果多种多样，但我们相信，我们只是抓住了交易费制度中出现问题的表面。 就我们而言，我们并没有提出这样一个分析，矿工的策略选择同时包括削弱挖矿和自私挖矿，这主要是由于模型的复杂性。

对交易费用制度的过渡一直没有得到足够重视。中本聪（Nakamoto）的论文简单地说：“激励也可以用交易费用来支付......一旦预定数量的虚拟货币进入流通，激励可以完全转换为交易费用，并且完全没有通货膨胀”[18]。比特币维基百科和其他地方的类似评论表明，社区认为这一过渡并不显眼。一些山寨币（altcoins）（如Monero，Dogecoin）甚至选择加快奖励减半的时间。

我们的结果表明了不同的观点。我们将区块奖励看作是挖矿游戏稳定性的一个组成部分。至少，在交易费用制度中分析均衡似乎比在区块奖励制度中要困难得多，这是一个令人担忧的问题。像以太坊一样，让区块奖励永久性导致货币通胀从而确保一个加密货币的稳定性。相比货币的稳定性，通货膨胀可能只是一个很小的代价。

1. **致谢**

我们非常感谢Jiechen Chen, Kira Goldner, Anna Karlin, and Rainer B¨ohme对本文早期草稿的详尽反馈。

1. **参考文献**
2. Calibrated learning and correlated equilibrium。 Games and Economic Behavior, 21(1):40–55, 1997。
3. A simple adaptive procedure leading to correlated equilibrium。 Econometrica, 68(5):1127–1150, 2000。
4. E。 Androulaki, G。 O。 Karame, M。 Roeschlin, T。 Scherer, and S。 Capkun。 Evaluating user privacy in bitcoin。 In Proceedings of Financial Cryptography, 2013。
5. S。 Arora, E。 Hazan, and S。 Kale。 The multiplicative weights update method: a meta-algorithm and applications。 Theory of Computing, 8(1):121–164, 2012。
6. P。 Auer, N。 Cesa-Bianchi, Y。 Freund, and R。 E。 Schapire。 The nonstochastic multiarmed bandit problem。 SIAM Journal of Computing, 32(1):48–77, 2002。
7. A。 Blum and Y。 Mansour。 From external to internal regret。 Journal of Machine Learning Research, 8:1307–1324, 2007。
8. N。 T。 Courtois and L。 Bahack。 On subversive miner strategies and block withholding attack in bitcoin digital currency。 CoRR, abs/1402。1718, 2014。
9. I。 Eyal。 The miner’s dilemma。 In Security and Privacy (SP), 2015 IEEE Symposium on, pages 89–103。 IEEE, 2015。
10. I。 Eyal and E。 G。 Sirer。 Majority is not enough: Bitcoin mining is vulnerable。 In Financial Cryptography and Data Security, pages 436–454。 Springer, 2014。
11. K。 Hill。 Bitcoin is not broken。 Forbes, 2013。

http://www。forbes。com/sites/kashmirhill/2013/11/06/bitcoin-is-not-broken/#55d4a8812568。

1. N。 Houy。 The economics of bitcoin transaction fees。 Working Paper GATE 2014-07。 halshs-00951358。, 2014
2. B。 Johnson, A。 Laszka, J。 Grossklags, M。 Vasek, and T。 Moore。 Game-theoretic analysis of ddos attacks against bitcoin mining pools。 In Proceedings of the First Workshop on Bitcoin Research, 2014。
3. J。 A。 Kroll, I。 C。 Davey, and E。 W。 Felten。 The economics of bitcoin mining, or bitcoin in the presence of adversaries。 In Proceedings of the Twelfth Annual Workshop on the Economics of Information Security (WEIS), 2013。
4. N。 Littlestone and M。 K。 Warmuth。 The weighted majority algorithm。 Inf。 Comput。, 108(2):212–261, 1994。
5. L。 Luu, J。 Teutsch, R。 Kulkarni, and P。 Saxena。 Demystifying incentives in the consensus computer。 In Proceedings of the ACM Conference on Computer and Communications Security (CCS), 2015。
6. A。 Miller and R。 Jansen。 Shadow-bitcoin: scalable simulation via direct execution of multithreaded applications。 In Proceedings of the eighth workshop on Cybersecurity Experimentations and Test (CSET), 2015。
7. M。 M¨oser and R。 B¨ohme。 Trends, tips, tolls: A longitudinal study of bitcoin transaction fees。 In Workshop on Bitcoin Research, pages 19–33, 2015。
8. S。 Nakamoto。 Bitcoin: A peer-to-peer electronic cash system, 2008。
9. K。 Nayak, S。 Kumar, A。 Miller, and E。 Shi。 Stubborn mining: Generalizing selfish mining and combining with an eclipse attack。 In IEEE European Symposium on Security and Privacy (EuroS&P), 2016。
10. R。 Peter。 A transaction fee market exists without a block size limit。 2015。
11. M。 Rosenfeld。 Analysis of bitcoin pooled mining reward systems。 CoRR, abs/1112。4980, 2011。
12. A。 Sapirshtein, Y。 Sompolinsky, and A。 Zohar。 Optimal selfish mining strategies in bitcoin。 In Financial Cryptography and Data Security, 2016。
13. M。 Vasek, M。 Thornton, and T。 Moore。 Empirical analysis of denial-of-service attacks in the bitcoin ecosystem。 In Proceedings of the First Workshop on Bitcoin Research, 2014。

1. https://github.com/citp/mining simulator [↑](#footnote-ref-1)
2. 例如，如果一个区块链高度为2的区块链已经被发布，

   但是某个矿工正私下存储一个长度为10的区块链，我们将定义H = 2。 [↑](#footnote-ref-2)
3. 虽然这才是我们开发模拟器的原始动机，但它确实能够用来模拟非零的区块奖励 – 详见4.3节。 [↑](#footnote-ref-3)
4. 备注：这不是协议的自我执行部分，而完全是客户端行为。 [↑](#footnote-ref-4)
5. 这样的y存在。其范围是（0，≈0.2 ]。 [↑](#footnote-ref-5)
6. E [X]表示随机变量X的期望值，I（E）表示事件E的指标随机变量（E出现时为1，否则为0）。 [↑](#footnote-ref-6)
7. 他们确实有一个很弱的原因：矿工从创造尽可能小的区块并认领一定权重的总交易费中受益，因为小区块通过网络传播得更快，也不太可能成为孤立（orphaned）区块。 [↑](#footnote-ref-7)