加密货币网络分布式共识算法

用户johnstuartmill*及匿名用户供稿

2016年10月22日

摘要

本文旨在呈现加密货币网络“分布式共识算法”。共识机制的对象为某些数据（可以是添加至区块链的候选区块）的哈希码，共识的主体为分布式节点，分布式节点就最正确的哈希码达成共识。本文旨在为分布式节点提供一组最优规则，确保各节点在共识决策期间遵从规则，最终使节点：1.快速达成共识；2.消耗最少的网络流量；3.能抵挡大规模、有组织的恶意节点协同攻击。这种算法是可扩展的，其算力消耗较低，很好地取代了工作量证明（Proof-of-work）方案。因此在价格低廉、能耗低的硬件上就能运行共识算法、出块，从而解决加密货币网络被垄断以及中心化的问题。

目录

0.1 前言 . . . 2
0.2 网状网络 . . . 2
0.2.1 疏连接性 . . . 2
0.2.2 节点的隐私 . . . 2
0.2.3 信息传播与公共广播 . . . 2
0.2.4 控制数据内容 . . . 3
0.3 网络共识算法 . . . 3
0.3.1 共识算法的必要性 . . . 3
0.3.2 算法的设计 . . . 4
0.3.3 网络规模的扩展性 . . . 4
0.3.4 达成共识前的准备 . . . 5
0.3.5 最终达成共识 . . . 5
0.3.6 时钟同步的独立性 . . . 5
0.3.7 执行共识算法 . . . 5
0.3.8 共识算法举例 . . . 6
0.4 模拟 . . . 6
0.4.1 网络参数 . . . 6
0.4.2 网络拓扑学的作用 . . . 8
0.4.3 被动节点的作用 . . . 9
0.4.4 数据样本规模的作用 . . . 9
0.5 算法伪代码 . . . 10
0.6 致谢 . . . 10
0.7 法律声明 . . . 10

*由[ ]资助。

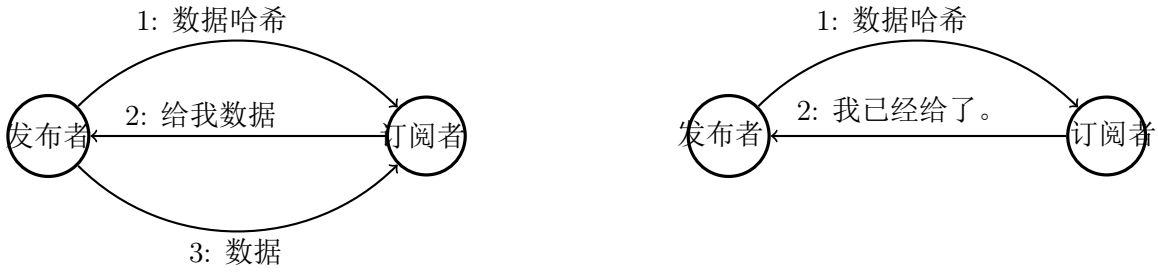

图1)数据通知及请求。左图：订阅者“ Sub ”未看到哈希，因而向发布者“ Pub ”发出要求，随后发布者发送数据。右图：由于订阅者早前已看到相同的哈希，因而不向发布者发出相应数据请求。

0.1 前言

本文基本架构如下：章节0.2对网状网络进行简单介绍，章节0.3介绍相关算法，章节0.4展示算法模拟结果，章节0.5呈现算法的伪代码。

本文要求读者具备以下两项基础知识：1.基本了解哈希函数；2.理解加密公/私钥如何用于签署哈希及如何验证签名。本文将解释区块和区块链的概念。

0.2 网状网络

0.2.1 疏连接性

网状网络的设计是为实现疏连接性。其设计原理是每个网络节点（“节点”）直接连接上、下游数个节点[1]。每个节点扮演两个角色：发布者和订阅者。作为订阅者，接收与之直接连接的上游节点（“发布者”）发布的数据。作为发布者，节点需（1）转发其收到的数据；（2）向与之直接连接的下游节点（“订阅者”）发送其独立生成的数据。因而，理论上每个节点可以从任何一个节点处接收数据，也可向网络上所有节点发送数据。参见图2：物理连接与逻辑连接对比；图1：数据传播。

0.2.2 节点的隐私

加密公钥（“公钥”）对节点编址。节点的IP地址仅向与之直接连接的节点公开。

0.2.3 信息传播与公共广播

所有节点均可发布信息。下游节点收到信息，通过加密验证及检查信息是否为重复发送，核实后将信息转发至其各自的下游节点。通过这一机制，信息得以传送到整个网络。

[1] 额外的链接生成路由，其跳转更少，能加速信息传播。

0.2.4 控制数据内容

若节点想要停止接收与之连接的其它节点发送的数据或恶意节点提供的虚假、垃圾数据，可采取以下方法进行控制：

切断连接

一个节点可以与其直接连接的其他节点切断连接。

常见的情况包括（1)节点Y转发大量信息到节点X，超出了节点X的负荷。（2)节点Y大量转发不恰当或恶意的信息。

若存在通向Y但不涉及X的路径，网络上其它节点依然可以接收到来自Y的信息。

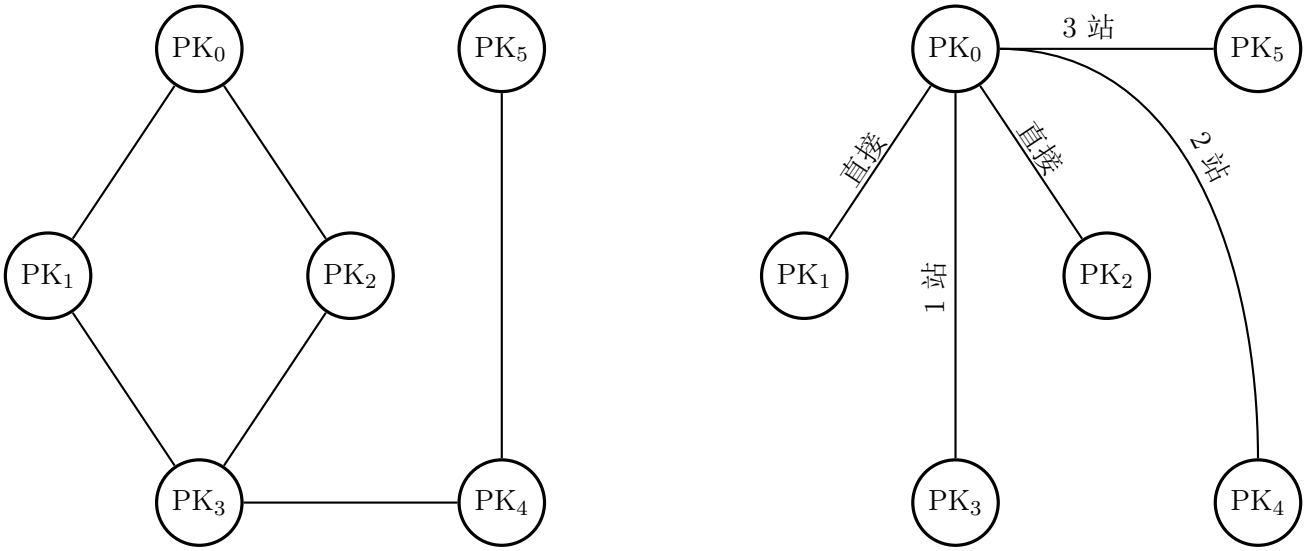

图2：小型网状网络节点物理连接 (左)，节点PK_0逻辑连接。

0.3 网状共识算法

0.3.1 共识算法的必要性

加密货币系统完整、运行良好的重要基础是：所有参与者能存储一份包含最新交易数据及过去交易数据的统一账单。而去中心化系统要实现这一目的，需要特别的机制，其中一种机制为网络共识算法。

共识算法的主要任务是在所有网络节点中推送区块链的状态，实现同步。同步区块链能确保账目记录一致，即计算指定公钥的代币余额时，每个参与计算的节点都得到相同的结果。

0.3.2 算法的设计

我们对一些共识算法进行试验。在这些算法中，节点与（1）最近的节点或（2）局部一小部分节点持相反意见。但我们发现在模拟中，相当小的一部分恶意节点可以操纵上述行为良好的节点，轻松达成共识。这种弊病在（直接）连接循环图[3]中尤为突出。意识到这些算法存在根本性的缺陷，不聪明的节点会仿效其它节点的行为，小部分有组织的聪明的节点可任意改变群体的意见，所以我们选择放弃这些算法模型。

另一类算法涉及节点间投票选举领导节点，我们同样放弃使用这类算法。我们认为同意选举一位领导（或临时统治者）不是明智的做法。原因是：群体需要智慧才能求得生存，但是一般的群众普遍都不聪明[4]。因此选举领导是很自然的事情，群体为了生存必须找到能为群体做出聪明决策的人。这种行为效仿羊群或那些倾向于被领导的物种的行为，并不符合加密货币社区的要求。另外，一位领导者也可能存在缺陷，被恶意实体挟持，侵害社区的利益。因此，我们也放弃了这种基于领导节点的算法模型。

基于上述种种局限，节点应满足以下要求：

- 聪明。每一个节点应该能够通过充分的统计分析独立做出决定（如决定最佳的下一个区块）；
- 善于质疑。节点必须质疑它所收取的资料，处理资料时必须经过多番验证，确保资料安全。同时，检验出虚假数据；
- 具备自主意识。除参考其它节点意见外，具备自主意识的节点不应服从某些群体或权威，更不得为了收取利益而支持某些观点；
- 具备创新能力。节点能接收原始数据（未经处理的基础事件，如交易），进行独立研究，提出新观点（如区块哈希）。

上述要求可确保算法执行。详情请参考章节0.3.7。

0.3.3 网络规模的扩展性

如章节0.2.1中提及到节点在理论上能接收网络上任何节点发布的信息，即使节点仅直接连接到少数节点。随着网络规模扩大，与共识相关的信息数量也会增长。大量数据点（如意见）也利于获得统计结果。但为了做出高质的共识决策，节点想要获得大量数据样本，会引发一些问题，其中涉及等待信息在网状网络沿多跳路径进行传输，反过来会对出块规则带来限制。为解决这个问题及其它相关问题，我们限定取样数目，该参数在下文中定义为Z。就网络延迟而言，这种决策的后果是形成节点特定影响区域，它由与问题节点恰好最近的Z个节点中的一些节点组成[5]。

[3]将分析限制为非循环图，其约束很大：即使是极权主义社会的通信网络也会有循环。

[4]我们很好奇，人类学家或人种学者是否能证实这一命题，或是利用他们尚未发表的研究成果来证明这一命题。

[5]影响区域及其边界随节点间的延迟而变化。

根据网络面临破坏攻击时获得的共识实验结果，我们探讨Z的作用，发现Z不需要很大，其值约为100属可接受的范围。详情请参考章节0.4.4及图6。

在实际的应用上，我们预期每一个节点应该从我们建议的范围之中根据实际情况选取选一个特定的Z值，从而使该节点的共识决策水平最优化。

另外，我们也探讨增加网络节点数量N的作用，同时成块节点数量B保持不变。结论是，共识结果的质量与N的增长无关。详情请参考章节0.4.5及图5。

本文并未探讨网络规模对信息传播时间的影响，详情请参考后续的文章。

0.3.4 达成共识前的准备

参与共识决策前，节点需下载区块链的“尾巴”（即最新的区块n)。

在一般的操作过程中，节点不时检查网络上有多少节点与自身的区块决策一致。

这两种相关的任务由独立的算法执行。

0.3.5 达成共识

对多个区块序列号的共识决策可以同时进行。如同犯罪学同时收集多个案件的证据：当案件证据收集足够多，能做出定论时就可以结案。

若节点已获得Z个对指定区块序号的意见，或区块时钟比区块序号超前许多，节点会结束共识决策，计算获选的候选哈希码。

回顾共识决策过程，节点与节点间会交换包含哈希码的细微信息。节点决定获选的哈希码后，网络要求（更大）的包含该区块的信息（图1）。决定获选哈希码后再传送区块可减少网络流量。

最后获胜的区块会被添加到区域区块链上。

0.3.6 时钟同步的独立性

该算法不采用“墙钟”（日历日期/时间）。相反，该算法从验证共识与区块链相关的信息中提取区块序号，使用序号来计算节点内部时间，俗称为“区块时钟”。

0.3.7 执行共识算法

执行共识算法需指派一些节点为“造块者”，所有节点均可成为造块者。造块者需要（1）观察其它节点公布的交易，（2）遵照某些规定，将观察到的交易打包成一个区块，即候选区块，及（3）向网络广播包含以下四点内容的信息：

- 候选区块哈希码，
- 哈希码签名（使用节点秘钥签署），
- 节点公钥，及
- 区块序号。

需要时，候选区块自身会作为独立信息进行广播。

多个节点生成候选区块，是为了获得大量基于观察的加密签署的独立意见。具体而言，若所有造块者（1）以区块链的同一备份作为起点，（2）接收同一组交易记录，及（3）遵照相同的造块规定，则其生成的候选区块就会一样。

在现实网络环境中，一些节点不会接收到当前已发布的所有信息，因此它们生成的候选区块会与完全掌握信息的节点生成的候选区块不同。尽管节点内部延迟（就概率分布而言），信息送达率相对不高时，大部分造块节点还是可以就给定的序号生成相同的候选区块。因此它们会广播相同的哈希码。正确的候选哈希码广播越多，越能防止攻击性实体挟持大量恶意节点广播选定的欺诈性候选区块哈希码，抵御大规模有组织的网络欺骗行为。

所有节点接收候选区块的哈希码广播，形成意见数据样本，根据数据样本推断出正确的哈希码。

网络连接较差的节点有可能无法发现所有的交易，因此不能指派其为造块者[6]。尽管如此，这些节点依然可以收到含有候选区块哈希码的信息；这些信息发布的频率比交易信息发布的频率低，且只包含几点内容。但因此非造块节点仍可保留一份最新的区块链区域备份。

模拟结果显示本文所阐述的共识算法会在系数0.4和0.5间失效，这意味着，攻击者需要大量恶意造块者才能使虚假哈希码赢得共识。详细模拟结算，请参考章节0.4.4。

无法保证所有节点达成共识，如，当一个节点（或一个节点群）中断与整个网络的连接，一旦网络生成新的区块，其区块就会过时。另外一个例子涉及诚信节点在物理上连接到恶意节点。恶意节点忽略收到的网络广播，且向周围的节点发送恶意数据。最后一个例子是，当发布者存在恶意嫌疑，订阅者可将发布者公钥拉进黑名单，将区域区块链重新同步至网络中。

0.3.8 共识算法举例

我们以一个节点举例阐述共识算法，可获得比较直观的理解。节点收集共识相关的信息，侦查出虚假的意图，最后决定获胜的哈希码（从而决定获胜的区块）。图3展示数据流向，表格1为数据分析，并附有详细说明。

0.4 模拟

0.4.1 网络参数

假设N个节点构成一个网络。每个节点有上游和下游连接节点。平均来说，每个节点直接连接到S个下游节点，因此上游连接节点平均数目也为S。该网络有B个造块者，B≤ N。每个节点从Z个造块者中收到候选哈希后，会决定获选的哈希。

总体来说，连接拓扑并没有限制。下文将对规模不同的两个网络进行模拟。

在第一个网络中，每个节点随机连接到S个发布者，所以每个节点自动获得分派平均S个订阅者。该网络模拟现代城市，城市中的单一ISP或多个ISP降低其他ISP请求的优先处理级别。该网络可作为测试算法的基础方案。

在第二个网络中，节点排在环状的网络上，平均连接到最近的S个发布者，获得分派S个（最近的）订阅者。值得注意的是，尽管这些节点仅直接连接到最近的节点，由于网状网络的转发机制，这些节点同

[6]由于可能丢失部分交易，候选哈希赢得共识的概率很低，因此广播候选哈希将浪费网络带宽。

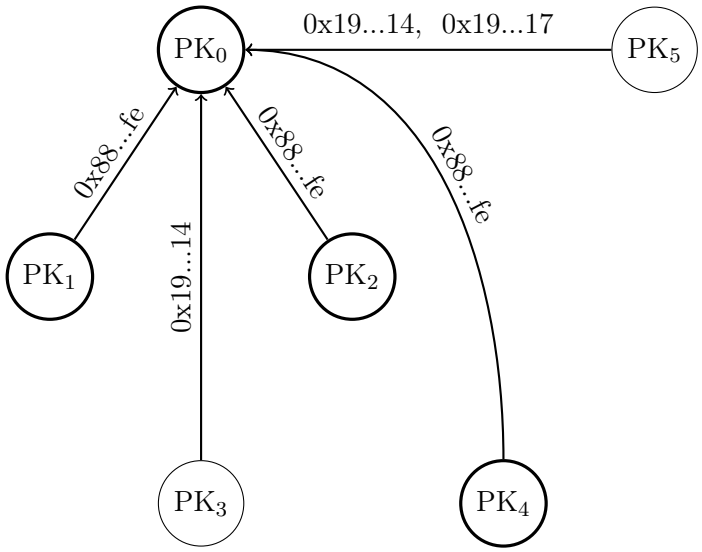

图3：如图2所展示的，网络中围绕节点PK_0的共识数据流向，呈现了所有直接发送和转发至PK_0的信息。图中使用线条连接造块节点至节点PK_0，以准备发送的哈希码作为标记。箭头表示数据流的方向。依据（1）区块链状态，（2）观察到的交易，及（3）造块规则，正确候选区块哈希码应为0x88...fe。节点PK_3和PK_5谋划将错误的交易插入到区块链上，正在推送哈希码为0x19...14的区块。另外，PK_5就同一区块序号计划发送哈希码0x19...17，企图降低正确候选哈希码的胜出概率。详见表格1，如何决定获胜的哈希码。

候选区块的哈希码	发送哈希码的不同公钥列表	作为独立意见代理列表长度	是否赢得当前共识决策？	注释
`0x19...14`	PK_3 PK_5	2	否	
`0x19...17`		0	否	PK_5因发送垃圾信息被屏蔽
`0x88...fe`	PK_1 PK_2 PK_4	3	是	

表格1：图3展示了最终的共识决策。就未来生成的区块序号，该算法标定出一组公钥（第2栏）发送的哈希码（第1栏）。算法针对每一个节点，独立地决定什么时候停止接收新数据，决定选中哪一个哈希码。第3栏展示每组的公钥数目，数目最大的哈希会被选中。第四栏注明指定的哈希是否赢得共识决策。在这个特定例子中，我们从两个公钥中收到0x19...14，从三个公钥中收到0x88...fe。后者0x88...fe赢得60%的选票，比其它候选哈希码的选票都要多。根据算法，哈希码0x19...17被判定为恶意攻击，被踢出竞争，其发送者公钥PK_5也被屏蔽一段时间。我们举例公钥PK_5将0x19...17传播至PK_0来阐明如何进行哈希验证；0x19...17应被第一个有效运行的节点（如PK_4）拦截和销毁。

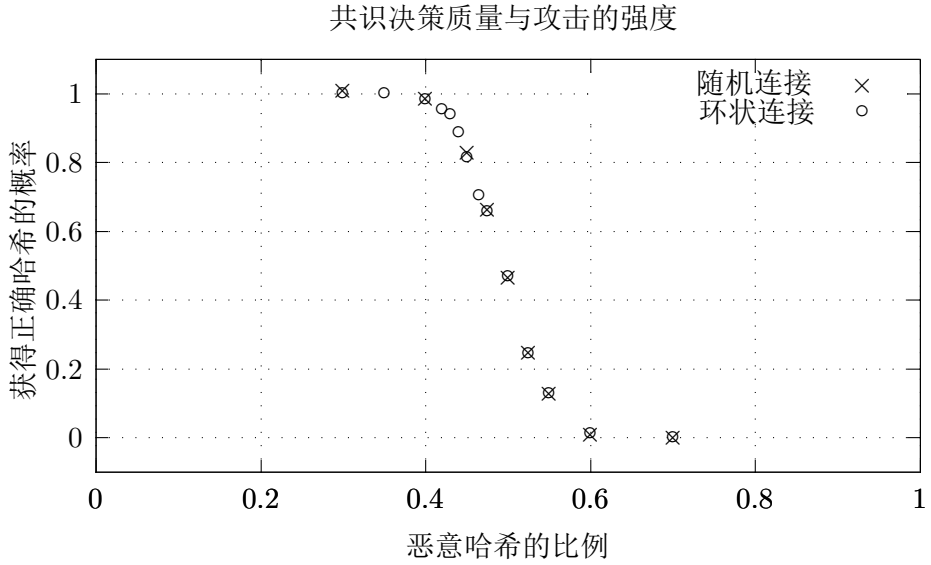

图4：网络拓扑学对共识决策质量的影响。该图对比两种网络获得正确哈希的概率（垂直轴）与恶意哈希比例（水平轴）。网络参数N=10000, B=1000, S=5, Z=100，但网络连接方式不同。这两种网络应对攻击的方式基本相同，其关键原因是网状网络的转发机制。更多探讨，详见章节0.4.2。

样可收到远处节点的信息。选择这种网络拓扑是为了便于检测出可能出现[7]的意见不收敛。

在这两个方案中，我们采用一致分布的节点间延迟概率分布，节点延迟介乎100到400毫秒。模拟实验同样包含造块和验证签名所需的加密计算。

0.4.2 网络拓扑学的作用

为了研究网络拓扑学对共识决策的作用，我们使用参数N=10000, B=1000, S=5, Z=100对两个网络进行模拟。如图4所示，尽管网络连接方式不同，参数N、B、S和Z的作用很小。两种网络应对攻击的方式基本相同，其关键原因是网状网络的转发机制：节点间无需直接连接，也能从其它节点处接收信息。因　此，若网络具备抵御破坏攻击的能力，但对连接问题反应较慢，那么每个节点只可直接连接其信任的公钥，从而无需担心构成异常的连接、受到攻击。

该结果十分鼓舞人心，原因有两个：其表明在目前的敌对环境中，网状网络一直是部署加密货币的不错的选择；网络连接作为技术参数，无需用到某些（并非所有）技术分析中。

虽然网络拓扑学对共识的作用很小，但涉及节点收集充足的候选哈希的时间要求，其影响不小：环状网络转发更多，花的时间也更多。

[7]以下举例阐明非收敛。将节点排成环状，其连接方式为：每个节点顺时针方向的下一个临近的节点是它的唯一订阅者，其上一个临近的节点是其唯一发布者。选择一个共识算法，在算法中节点赞同/复制其发布者的意见。意见限定为两个值，0和1。节点任意初始意见值有的是0，有的是1。然后执行共识决策，同步更新节点意见。显而易见的是，0和1的环状顺序是顺时针旋转的，节点改变它们的意见，且永远不会达到稳定状态。

[8]延迟分布对共识的作用将在后续文章中进行探讨。

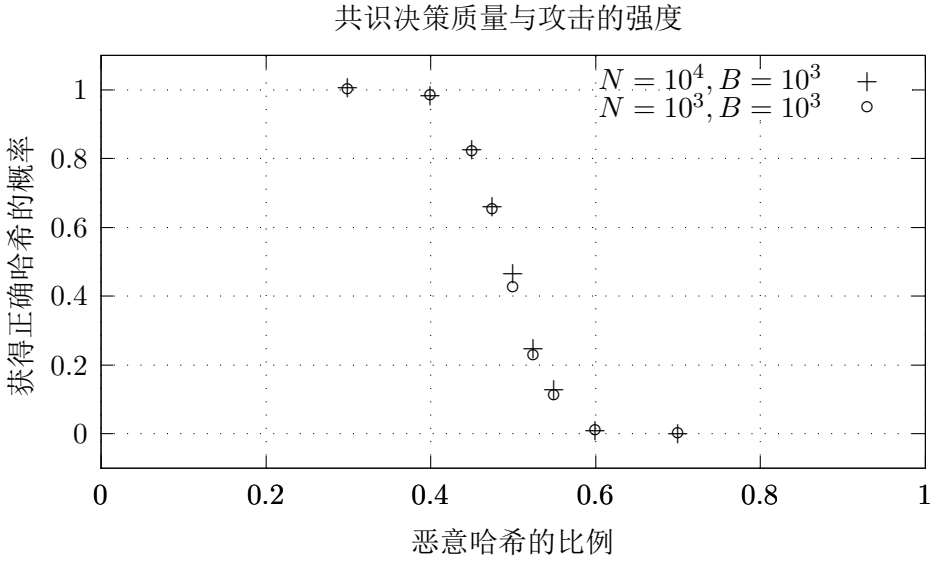

图5：被动节点对共识决策质量的影响。两种网络属无结构的随机连接，B=1000, S=5, Z=100 及 $N \in \{1000, 10000\}$。值得注意的是，两个网络的网络参数是一样的，但第二个网络有900个非造块节点。该图说明了共识决策的质量不受被动节点，即非造块节点的影响。

0.4.3 被动节点的作用

为探讨非造块节点对共识决策的作用，我们模拟两个无结构的随机连接网络，B=1000, S=5, Z=100 及 $N \in \{1000, 10000\}$。两个网络的网络参数一样，但第二个网络与900个非造块节点。图5展示了模拟的结果。从图中可以看到，共识决策不受被动节点影响，但取决于诚信节点与恶意节点的比例。

模拟结果很具有启发意义：增加被动节点（只交易币和转发信息，但不制造区块）不会使网络抵御攻击的能力受损。这也意味着使用硬件性能低、成本低的用户不会构成系统的负担。

0.4.4 数据样本规模的作用

为探讨样本大小Z对共识决策的作用，我们模拟环状连接的网络，N=1000, B=1000, S=5, 及 $Z \in \{25, 100, 200, 1000\}$。

为加快得出结果，即完成共识决策，需符合两个条件：（1）至少80%节点同意一个哈希码；（2）一致认同的哈希码不是虚假的。同样，若恶意节点（或发送的恶意哈希）比例达到最大，但仍能成功达成共识，该最大恶意节点比例称为“故障系数”。系数越大，网络就越能抵御破坏攻击。

图6的实验结果表明每个节点仅仅收集Z=25个候选哈希后做出决策，尽管企图破坏共识的虚假候选信息达40%，共识决策还是能成功完成。在这种情况下，故障系数为0.4。当Z增加到100时，故障系数上升为0.45。若样本规模更大，Z=1000，故障系数接近0.5；只要虚假候选哈希比例小于50%，几乎所有节点能达成共识，选出正确的哈希。

从而，我们得出结论，该算法足够稳健，能抵御上述类型的攻击。

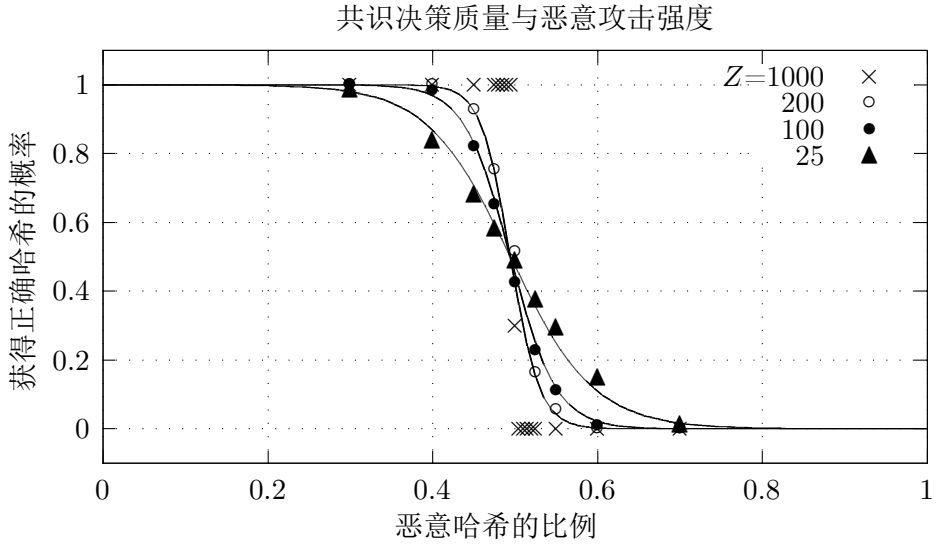

图6：获得正确哈希的概率（竖轴）对比恶意节点的比例（横轴）。Z是一个变量参数，代表节点用以做出决策的数据样本数量。恶意节点通过广播同一个虚假的哈希码，对网络进行协同攻击。图中说明决策数据样本Z规模越大，共识决策系统就越稳健。该网络为环状连接网络，N=1000, B=1000, S=5, 及 $Z \in \{25, 100, 200, 1000\}$。

0.5 算法伪代码

图7为算法的伪代码。

0.6 致谢

本文作者对参与探讨、做出贡献的社区成员（按他们意愿，不公开姓名）致以感谢。

0.7 法律声明

```
handle_consensus_data(pubkey,hash,sig,seqno) {

  if (!(is_consensus_data_valid(pubkey,hash,sig,seqno)))
    // Ignore data. Consider banning 'pubkey' for failure to follow data specification.
    return;

  if (!(verify_cryptographic_signature(pubkey,hash,sig)))
    // Someone is trying to impersonate 'pubkey', so ignore the data.
    return;

  if (!is_seqno_relevant(seqno, blockchain))
    // Either trials completed for this seqno, or the seqno is too large.
    return;

  // Get consensus data we collected so far for the given block sequence number:
  ConsensusInfoType& info = db[seqno];
  if (info.is_consensus_closed())
    return;

  if (info.Z.contains(pubkey))
    // Public key 'pubkey' has already sent its opinion. This could be a malicious attempt.
    // Ignore data. Consider banning 'pubkey' for failure to follow consensus protocol.
    return;

  info.X[pubkey] = ConsensusDatum(pubkey,hash,sig,seqno);
  info.Y[hash].insert(pubkey);       // This maps hash --> set of unique pubkeys
  info.Z.insert(pubkey);             // These are unique pubkeys

  if (info.Z.length() < cfg_max_sample_size)
    return; // Sample size is too small to infer population parameters.

  // At this point we have collected the amount of data that we think is sufficient to
  // make statistical inference. The following steps are verbose when expressed
  // programmatically, so we describe them in words:

  // 1. For each key in info.Y map, extract data into a list of pairs
  //    (key, info.Y[key].size()).
  // 2. Sort the list according to the 2nd element, resolve any ties using the 1st.
  // 3. The 1st element of the last record in the sorted list is the hash code that has
  //    the largest number of unique pubkeys. Call it best_hash.
  // 4. Get the data that corresponds to best_hash. The 'info.X' is used here.
  // 5. Prepare the data to be appended to the local blockchain.

  db.erase(seqno); // No longer needed.
  return;
}
```

图7：共识算法伪代码。为清晰地呈现出来，上述代码去掉了出错处理和欺诈检测等一些功能。算法的完整版可通过以下链接获得https://github.com/johnstuartmill/consensus or https://github.com/skycoin/skycoin/consensus。