理性神谕系统的共识环境

Akis Chalkidis akis@ecoc.io

February 2020

Contents

1 前言

本文的目的是在特定的假设下分析神谕系统协议。神谕系统是一个可以访问真是数据的实体，它可以访问真实世界的数据，也可以单独（集中式）或在与其他神谕系统（去中心式）达成共识后改变机器的状态。

一个明显有趣的例子是去中心化神谕系统化，也就是说，我们需要一组不可信任的神谕系统，和共识来决定一个值或事件的真假性。如果是真的，那么交易就可以发生，并且可以在账本（区块链）上记录价值。但本文的结果可用于在任何存在不诚实参与者的系统（不仅是区块链），其中需要达成共识，参与者的行为是基于回合制的，并且同时在发生（而按照顺序）

2 基本假设

在提出协议之前，必须命名关于环境的假设。

1. 神谕系统行为是理性的。这意味着他们想要最大化他们的利润，所以他们可以遵循或（试图）违反协议，根据他们自己的利益。“拜占庭”(Byzantine) 本术语被许多人使用，但理性和拜占庭的参与者并不完全相同。理性参与者的唯一动机是最大化其效用，而效用具有财务意义。拜占庭式参与者的动机（以及行为）可以是理性的，也可以是恶意的。所谓恶意，我们指的是那些愿意接受经济处罚、试图破坏系统的参与者。换句话说，拜占庭参与者的实用程序可以是财务上的，也可以是对系统恶意的。我们的假设是神谕系统有理性的行为，不是拜占庭性的行为。

2. 神谕系统可能有缺陷。他们的行为仍然理性，但对于第三者来说，他们的行为可能看起来不理性。例如，检测事件或提取值的设备功能不正常，或者存在连接问题，无法在特定回合采取行动。

3. 所有神谕系统的行动同时发生。这意味着神谕系统知道除本轮以外其他神谕系统的全部行动历史。这使得他们无法考虑其他神谕系统在本轮的行动。他们必须在没有这些信息的情况下做出决定。

4. 他们的决定是不可更改的，这就保证了投票的不可更改性。

5. 因为神谕系统是理性的，他们的数学期望值必须是正数的（简单地说，他们期望利润）。否则，他们一开始就不会参与。

6. 神谕系统是聪明的。这意味着他们有一个非近视的观点。他们试图使最大化长期的利润。他们可以做出一个短期看来不理想的决定，但实际上从长期来看，支出是最大的。

7. 没有 “最后一轮”，也就是说，轮次是无限的（或者至少期望是无限的）。

3 协议目标

最终的目标是消费者能够以很高的概率得到正确的答案。消费者指的是提出问题的实体。消费者可以是任何人，对结果的评估可以：

1. 决定区块链上（例如，由智能合约功能自动提取）。

2. 由算法决定区块链下，为消费者进行评估区块链上（中心化版本）。

3. 决定区块链下，绝不记录在区块链上（该值可以在集中应用程序中使用）。

4. 不经评估就被记录在传统数据库中，可能供将来使用或提取统计结果。

   从上面可以清楚地看出，“结论” 可以在议定书之外达成。协议只输出每个神谕系统的投票和可信度。评估，无论是自动的还是手动的，甚至不进行（只是保存以备将来使用）可以考虑 a）多数票 b）票数比率 c）每个投票人的可信度（权重）。神谕系统的可信度 “分数” 可以从其投票历史中提取出来，因此被视为输出。

4 威胁

系统存在两个威胁、两个障碍必须进行解决

- 联盟：由于无限循环的期望，神谕系统有形成联盟的动机。这是因为联合的的盈余可能是正的。如果他们能够形成多数，那么他们就可以确信他们每次都会得到回报。因此，协议必须防止神谕系统进行联盟，否则产生的结果将是无用的。神谕系统会投票而不管其真价值。

- 搭便车：个别神谕系统可能会试图模仿多数人的投票。这样他们就能得到奖励。这会降低输出的质量。

现在必须清楚的是，由于上述威胁，投票必须隐藏，直到所有神谕系统结束投票过程。这可以通过提交-揭示示策略来实现。所以每次投票都有两个步骤，提交（哈希值）和揭示（显示实际投票）。哈希值和显示必须匹配。单靠这一战略并不能完全消除这些威胁。神谕系统被认为具有智力和非近视逻辑。因为他们可以接触到投票历史，所以他们可以组成联盟做出结论，或者尝试猜测其他人的本轮投票，然后搭便车。协议中的一个内置惩罚系统必须激励神谕系统为了避免这种行为。

5 博弈论

协议可以在博弈论的数学领域中形式化。在博弈论中，我们讨论的是参与者、规则、参与者的效用函数、动作集和策略（一系列动作）。在我们的案例中：

- 神谕系统为参与者：o_i
- 效用函数是每个神谕系统的经济利益：u_i
- 每一轮的行动都是投票。因为这个值是布尔值，所以它是集合 s : $\{true, false\}$
- 神谕系统的策略 o_i 是一系列动作 σ_i. 对于每个神谕系统 o_i 有一组策略 $\sigma_i : \{\sigma_{i,1}, \sigma_{i,2}, \sigma_{i,3}, ..., \sigma_{i,n}\}$. 集合的基数是 $|\sigma_{i,n}| = 2^n$ 其中 n 是一个循环序列，因为 s 是布尔值 $(|s| = 2)$.

该系统可以形式化为游戏 G 通过以下元素：

- $G :< O, S, U >$
- O 是神谕系统集合 $O : \{o_1, o_2, o_3, ..., o_n\}$
- S 是每个神谕系统 $\{\sigma\}$ 的一组策略可能性 o_i.

  $S : \{\sigma_{o_i,1}, \sigma_{o_i,2}, \sigma_{o_i,3}, ..., \sigma_{o_i,n}\}, \forall i \in O$
- $U : \{u_i\}$ 是参与者实用函数 o_i, 其中 $u_i = B_i$. B 是参与者的支付函数。

注意：

- 神谕系统是理性的参与者，因此他们的效用函数只具有财政性质，即 $U \leftrightarrow B$. 所以我们不考虑任何恶意行为（除了财务目标）。

- 对于每一轮，函数 B 只接受当前属于回合集合 O 的所有神谕系统的动作作为参数。这意味着每个神谕系统 o_i 的收益 b_i, s 与上一轮完全独立的。从数学上讲，对于每一轮 r

$$U_{i,r} = B(\sigma_{i,r})$$

- 显然，每个神谕系统 o_i 将遵循的策略是最大化其效用 u_i. 我们已经假设所有参与者都有非变形的观点。因此，他们将采取行动和策略 $\sigma_{i,max} \in S$ 这样会给出最大的总收益，而不是本轮的最大收益。

  更准确地说，对于圆形 r 和神谕系统 o_i 最佳动作 $s_{i,r}$ 而不是

  $\forall\ s_{k,r},\ s_{max,r}$:

  $B(Es_{1,r}, Es_{2,r}, ..., s_{max,r}, ..., Es_{n,r}) > B(Es_{1,r}, Es_{2,r}, ..., s_{k,r}, ..., Es_{n,r})$

  但是

  $\forall\ s_{k,r},\ s_{max,r}$:

  $\sigma_{i=1}^{r} B(\sigma_1, \sigma_2, ..., s_{max,r}, ..., \sigma_n) > \sigma_{i=1}^{r} B(\sigma_1, \sigma_2, ..., s_{k,r}, ..., \sigma_n)$

  (5.1)

  在第一个不等式中，$Es_{i,r}$ 是所有其他参与者 o_i 的预期动作，因为他们当前动作的动作是未知的。

  第二个不等式显示了每个神谕系统的最佳非近视策略。这是理性参与者将遵循的策略。注意，这个等式考虑了其他参与者的整个历史（策略）。因此，神谕系统收集前几轮的信息，对其他神谕系统的未来行为有一个预期，并据此形成自己的策略。

  以上的不等式会变成等式，如果：

  - 神谕系统不能接触到其他神谕系统的投票历史。

  - 神谕系统的观点是短视的。

  - 出于任何其他原因的可能性导致阻止它们结成联盟在未来几轮协调行动。

  - 回合不是无限而有一个已知的最后一轮。

  如果上述任何一个条件成立，那么神谕系统的最佳行动将是相等的，使给定的回报由两个的任何一个相等。但由于上述条件不成立，任何神

谕系统都可以组成或加入联盟。因为有些联盟有正盈余，所有理性的参与者都会试图创建或加入一个联盟。在下一章我们将讨论到联盟。

- 支付功能 B 必须具有阻止形成联盟的属性。为了达到这个目标，支付函数 B 必须使任何联盟盈余完形填空为零。让我们把所有可能的联盟的集合表示为 C 将联盟的剩余部分 i 表示为 Cs,i。因此，支付函数 B 具有如下性质：

$$\lim_{r->\infty} Cs,i = 0 \ \ \forall i \ \in C \tag{5.2}$$

6 联盟

我们可以将联盟 C_i 定义为一组神谕系统 $O_c \subseteq O$ 和 $|O_c| \geq 2$ 个的集合寻求提高他们的利益 $\sum_{i=1}^{\infty} B$。为了实现此事，很明显，盈余必是正数值：$C_{s,i} > 0$, 其中 $C_i \in C$。盈余是联盟中每个个参与者的总利益之上：

$$C_{s,i} = \sum_{i=1}^{\infty} EB_{C_c} - (\sum_{c=1}^{\infty} EB(o_{1,c}) + \sum_{i=1}^{\infty} EB(o_{2,c}) + ... + \sum_{i=1}^{\infty} EB(o_{|C_i|,c})) \Leftrightarrow$$

$$\Leftrightarrow C_{s,i} = \sum_{i=1}^{\infty} EB_{C_i} - (\sum_{i=1}^{|C_i|} \sum_{i=1}^{\infty} EB(o_{i,c}))$$

因为 $C_{s,i} > 0$, 所以：

$$C_{s,i} > 0 \Rightarrow \sum_{i=1}^{\infty} EB_{C_i} - (\sum_{i=1}^{|C_i|} \sum_{i=1}^{\infty} EB(o_{i,c})) > 0$$

$$\Leftrightarrow \sum_{i=1}^{\infty} EB_{C_i} > \sum_{i=1}^{|C_i|} \sum_{i=1}^{\infty} EB(o_{i,c}))$$

EB_o 为每联盟的参与者回报的预期，EB_C 为回报预期总数。因此：

$EB_C = \sum_{i=1}^{\infty} EB_{C_i}$ and $EB_o = \sum_{i=1}^{\infty} EB(o_{i,c})$ 我们最后有：

$$EB_C > \sum_{i=1}^{|C_i|} EB_o \tag{6.1}$$

在 r 是轮数，回报函数 $B()$ 必须存在 $C_{s,i} - > 0$ 的属性，所以 $B()$ 函数定义为：

$$\lim_{r->\infty} EB_C = \sum_{i=1}^{|C_i|} EB_o$$

(6.2)

让我们回头看一下，考虑下在 $n.$ 个神谕系统情况下，可能存在多少联盟。我们知道组合联盟的可能性为 $2^n.$ 但是我们不关心那些单个或没有神谕系统而组成的 {} 因此，我们有：

$$|C_n| = 2^n - n - 1$$
(6.3)

用数学归纳法来证明这个方程是很容易：

- 对于 $n = 2$ 我们有 $|C_2| = 2^2 - 2 - 1 = 4 - 3 = 1$ ，这是实际的。只有一个联盟是可能的：$\{o_1, o_2\}$
- 我们假设 $|C_k| = 2^k - k - 1$
- 为了对方程：$n = k + 1$ 进行证明。所以我们必证明：

  $|C_{k+1}| = 2^{k+1} - (k + 1) - 1$

  联盟的总数 $|C_{k+1}|$ 是联盟的 $|C_k|$，此外，$k + 1$ 元素可能创建更多的 $|C_k|$，因为它可以加入所有现有的联盟。此外，它还可以创建任何其他元素包括它本身的联合：$\{\{o_1, o_{k+1}\}, \{o_2, o_{k+1}\}, ..., \{o_k, o_{k+1}\}\}$。所以我们有：

  证明：

$$\begin{aligned}
|C_n| = |C_{k+1}| &= |C_k| + |C_k| + |\{\{o_1, o_{k+1}\}, \{o_2, o_{k+1}\}, ..., \{o_k, o_{k+1}\}\} \\
&= 2 * |C_k| + (k) \\
&= 2 * (2^k - k - 1) + k \\
&= 2 * 2^k - 2k - 2 + k \\
&= 2^{k+1} - k - 2 \\
&= 2^{k+1} - k - 1 - 1 \\
&= 2^{k+1} - (k + 1) - 1
\end{aligned}$$

  证明完毕

  不幸的是，方程 (6.3) 的复杂度为 $\mathcal{O}(2^n - n - 1) = \mathcal{O}(2^n)$ 这意味着即使是少数神谕系统 $|O| = n$，联盟的组合的可能是非常大的。该协议必须以这种方式建立，以鼓励形成任何规模的联盟。

7 单神谕系统战略

7.1 动机

在对协议体系结构做出任何决定之前，我们必须分析神谕系统的合理行为。一个神谕系统，为了使自己的利润最大化，他会试图加入一个有最大盈余的联盟，这对神谕系统来说是最大的利益。这是因为，虽然系统基于 NTU（不可转让的效用），奖励均分给每个赢家。也就是说，不可能有其他交付。在联盟中，神谕系统有更大的机会获得奖励。

但是为什么联盟有盈余呢? 很容易证明这一点。它是基于这一个事实，即神谕系统获得错误事件值的概率是正的，$p_{sf} > 0$。加入一个联盟总是会得到奖励，而诚实地投票，他们将在每一轮中以概率 $1 - p_{sf}$ 获得奖励，以概率 p_{sf} 获得惩罚（或至少零奖励）。

不幸的是，神谕系统加入大联盟还有第二个原因。即使在没有错误的情况下，也就是说 $p_{sf} = 0$ 他们最好组成一个联盟。这是因为在现实中，提取事件的价值是有代价的（运行服务器、支付第三方使用其 API、维护基础设施、开发和更新软件等）。但在一个联盟，投票总是一个布尔变量的（不管是真是假，这无关紧要），所以神谕系统可以降低运行成本，只需反复投票相同的价值。

当然，要加入一个联盟，神谕系统首先必须发现一个联盟。如果它检测到许多联盟，神谕系统将加入一个有最大概率形成多数的联盟，换句话说，就是拥有最多成员的联盟。

7.2 假设

首先，我们将分析神谕系统如何检测联盟。对于每一个神谕系统来说，唯一可用的信息是其他神谕系统的投票历史和以前事件的真实值（布尔值）。我们所说的“真实”是指神谕系统从前几轮投票中发现的值。我们可以假设如下：

1. 所有的值和选票都是布尔值
2. 因为投票记录在一个不可变的账本上，所以它拥有完美的投票历史信息

3. 神谕系统知道每个回合的真实值，但这个信息不是完美的：神谕系统可能无法知道某些事件（甚至是所有事件，如果其基础设施出现严重故障）的正确值

4. 神谕系统认为其他神谕系统都是理性的（参考 §2.1）

5. 神谕系统之间的直接沟通渠道不存在，也就是说，发送和接收信息与其他神谕系统互动是不可能的。这保证了 NTU 是有效的。

6. 每个神谕系统出错的概率都很低。这是一个合乎逻辑的假设。我们所说的“低”是指远低于 50

7.3 指标

神谕系统发现其他可能属于联盟的神谕系统的指标是：

- 神谕系统的历史表明，其投票率与正确值的比率非常高。如果是这样的话，这些神谕系统出于目的“错误地”投票（反对实际价值）的概率很高。

- 在错误的投票回合中，相应的奖励结果是正的，而且频率非常高。这是为了排除可疑的神谕系统有缺陷的可能性（持续不完善的信息，使其无法看到事件的正确值）。

- 考虑到上述情况，神谕系统必须知道事件的真正价值。不幸的是，它不能确定，因为我们假设这些信息是不完美的，也就是说，有可能是做检测的神谕系统本身是有误的。它还必须尝试计算它本身不断地得到错值（它发生故障）的概率。所以神谕系统也必须考虑到自己的历史。

7.4 联盟检测

乍一看，上述指标是没有用的：每个神谕系统都不能确定它是有缺陷的，所以它不能断定其他神谕系统是属于一个联盟，还是他们只是诚实和投票正确的价值。但有一点打破了这种困境的对称性：历史的起点。我们假设神谕系统之间不存在侧面交流，也就是说，系统之外的交流是不可能的（§7.2.5）。因此，在第一轮中，联盟是不存在的。传

递信息的唯一途径就是投票。不断地建立并发送错误的信号至希望建立联盟的神谕系统。它就像一个灯塔，每轮都发出而接受惩罚的信号。经过多轮谈判，可能会成功，从长远来看，如果成功，联盟盈余将弥补并超过神谕系统的初始的亏损。因此，神谕系统可以安全地假设第一轮没有任何联盟。

由于（§7.2.6），上述逻辑而成立。如果每个神谕系统都有接近 50% 的故障概率，那么即使是开始的历史也不能使用。但是，如果每个神谕系统都有很高的机会出错，那么这个系统无论如何都是无用的。在 $p_f = \frac{1}{2}$ 错误的神谕系统机中，系统熵非常高，而且无论如何也无法提取出有用的信息。在实践中，假设大多数神谕系统没有错误为一个真实。

如果神谕系统认为它本身经常出错，那么加入联盟的紧迫性就更高。以前的得票者总是愿意改变他们的投票结果。我们可以将这种策略命名为*信息搭便车*。神谕系统完全忽略了认为是正确的价值观，并复制了前几轮优胜者的投票结果。

让我们更具体一点。神谕系统发现联盟的第一步是自己判断它本身是否经常出错。他唯一能找到答案的机会是在最初的几轮，那里肯定还没有形成联盟。我们用 p_{cf} 来表示神谕系统不断出错的概率，而 p_{sf} 则表示只在一轮中出错的概率。我们还假设神谕系统 $|O|$ 的总数不是很低。在这种情况下，神谕系统严重的出错而且不断出错概的率可以用二项分布来计算，因为该值是布尔值，并且动作是以轮（离散分布）的形式进行的。神谕系统没有动机在第一轮投票中投错票，因为它将受到惩罚。只有在它想建立一个联盟的时候才会这么做，但在这种情况下，它会知道自己故意投错的。我们假设它会正常工作。前 k 轮神谕系统可以安全地假设没有形成大型联盟。所以，如果它不是经常出错的话，它必须几乎所有时候都正确地投票。几乎不是所有时候，因为 $p_{sf} > 0$

利用二项分布的累积分布函数（CFD），我们得到：

$$CFD(c; k, 1 - p_{cf}) = P(X \leq c) = \sum_{i=1}^{c} \binom{k}{i} (1 - p_{sf})^i p_{sf}^{k-i} \tag{7.1}$$

- c 为正确猜测的极限

- k 为神谕系统认为安全的最后一轮数字
- p_{sf} 为公司神谕系统每轮得到错误值的概率（远低于 $\frac{1}{2}$）
- X 为正确猜测的次数
- $P(X \leq c)$ 为正确猜测低于或等于极限 c 的概率

对于二项分布的正确猜测，期望值平均值为: $E[X] = k * (1 - p_{sf})$ 而方差为: $Var(X) = k * p_{sf} * (1 - p_{sf})$

从理论上讲，二项置信区间依赖于对二次分布的观测值 P 的误差分布的近似，即正态分布。使用正态近似，$P(X \leq c)$ 估计为:

$$\frac{c}{k} \pm \frac{z}{k}\sqrt{\frac{c(k-c)}{k}} \tag{7.2}$$

在我们的情况下，$z = 1 - \frac{p_{cf}}{2}$ (7.3)

结合（7.1)、(7.2）和（7.3)，我们得到

$$\begin{gathered}
P(X \leq c) \approx \frac{c}{k} \pm \frac{z}{k}\sqrt{\frac{c(k-c)}{k}} \\
\Rightarrow \sum_{i=1}^{c} \binom{k}{i} (1-p_{sf})^i p_{sf}^{k-i} \approx \frac{c}{k} \pm \frac{1 - \frac{p_{cf}}{2}}{k}\sqrt{\frac{c(k-c)}{k}} \\
\Leftrightarrow k\sum_{i=1}^{c} \binom{k}{i} (1-p_{sf})^i p_{sf}^{k-i} - c \approx \pm\ (1 - \frac{p_{cf}}{2}\sqrt{\frac{c(k-c)}{k}}) \\
\Rightarrow \tfrac{p_{cf}}{2}\sqrt{\tfrac{c(k-c)}{k}} > 1 - k\sum_{i=1}^{c} \binom{k}{i} (1-p_{sf})^i p_{sf}^{k-i} + c \ \ \vee \ \ \frac{p_{cf}}{2}\sqrt{\frac{c(k-c)}{k}} < \\
1 + k\sum_{i=1}^{c} \binom{k}{i} (1-p_{sf})^i p_{sf}^{k-i} - c \\
\Leftrightarrow 1 - k\sum_{i=1}^{c} \binom{k}{i} (1-p_{sf})^i p_{sf}^{k-i} + c < \frac{p_{cf}}{2}\sqrt{\frac{c(k-c)}{k}} < \\
1 + k\sum_{i=1}^{c} \binom{k}{i} (1-p_{sf})^i p_{sf}^{k-i} - c \\
\Leftrightarrow \tfrac{2\sqrt{k}}{\sqrt{c(k-c)}}(1 - k\sum_{i=1}^{c} \binom{k}{i} (1-p_{sf})^i p_{sf}^{k-i} + c) < p_{cf} < \\
\frac{2\sqrt{k}}{\sqrt{c(k-c)}}(1 + k\sum_{i=1}^{c} \binom{k}{i} (1-p_{sf})^i p_{sf}^{k-i} - c)
\end{gathered} \tag{7.4}$$

从（7.4）中，神谕系统机可以高精度地计算出不断发生故障的概率 p_{cf} 。（7.4）有用的限制条件是：$c \neq 0$, $c \neq k$ and k is not very small. Also, c, k and p_{sf} 不是很小。另外，计算 p_cf 时，神谕系统必须知道 c,k 和 p_{sf}。正确的投票数 c 是可以知道的，因为在第 k 个起始回合中没有联盟，所以大多数投票就是正确的值（我们已经假设 p_{sf} 很低并且远远低于 $\frac{1}{2}$）。最后，$c \neq k$ 。在 $c \neq 0$。在 $c = 0$ 或 $c = k$ 的任何一种情况下，都不能使用（7.4）。当 $\hat{p}$ 接近 0 或 1 时，即 $\frac{c}{k} \approx 0$ 或 $\frac{c}{k} \approx 1$ 时，不能使用正态近似区间。这两个限制虽然失效（7.4），但实际上并不是问题。如果 $c = k$ 且 k 不是很小，则根本不需要（7.4）。神谕系统可以得出结论，它并不是经常出错的。一个神谕系统经常出错，并且偶然猜对所有选票的概率是 $\frac{1}{2^k}$。对于正常值 k，假设前 20 轮 $k = 20$ ，概率小于百万分之一。$c = 0$ 的情况在实际中永远不会出现。一个经常出错的神谕系统应该平均有一半的情况是正确的。总是“不幸”的可能性的概率也是 $\frac{1}{2^k}$，这是意味不可能的，正如我们以上分析过。

最后要说明的是，p_{sf} 的概率越低，p_{cf} 根据（7.4）精度就越高。p_{sf} 低是我们所望的，因为我们假设 p_{sf} 为很低。

在神谕系统检测到自身不断出错结果的情况下，它有以下选项：

(a) 试着搭便车。不幸的是，在我们的系统中，这是不可能的，因为本轮投票神谕系统总是直接沟通的。

(b) 试着发现并加入一个联盟。如果它希望留在系统，这是唯一的选择。

(c) 离开系统，因为它认为他未来的预期回报将为负 $EB(O_{i,\infty}) < 0$ 。

因为一个经常出错的神谕系统会降低输出质量，我们不希望他们留在系统中。稍后我们将研究如何预防（7.4.b）

因为一个经常出错的神谕系统会降低输出质量，我们不希望他们留在系统中。稍后我们将研究如何预防（7.4.b）现在让我们来研究一个常见的情况，即神谕系统并不是经常出错。他的策略很简单：

(d) 如果它发现一个获胜的联盟，也就是说，一个盈余为正的联盟，它就会加入该联盟。

(e) 否则，它会在每一轮都诚实地投票，因为它认为没有联盟，所以诚实地投票在大多数时候都会得到奖励。只要奖赏期望值高于惩

罚期望值，它将保留在系统 $EB(O_{i,\infty}) > 0$ 。

(f) 最后，它可以尝试成为一个领导者，组成一个联盟。在这种情况下，第 k 轮投票结束后，它将开始有目的地投票反对实际值。它知道每轮都会受到经济处罚。这是神谕系统的一个选择，只有当它相信联盟盈余的未来红利将超过起始惩罚时。

显然，作为系统的架构师，我们不希望（7.4.d）和（7.4.f）发生。我们要永久性地禁止经常出现问题的神谕系统，并防止功能神谕系统建立联盟。在**系统架构**，我们将分析系统如何实现这一点。但首先我们需要了解清楚联盟的战略。

8 联盟共识

联盟有三个阶段：创建阶段、过渡阶段和维持阶段。

8.1 创建阶段

创建阶段是一个发起者，我们称之为“领导者”，在每一轮中向所有其他神谕系统发出信号，让他们跟随它的阶段，测信道通信不存在，但领导只需要广播一位信息（“加入我!”）. 它可以通过反复投票反对正确值来做到这一点。或者，它根本不能投票（或者显示与提交的值不匹配的值）。当然，我们的协议同样处理所有这些情况，它们被认为是错误的值。我们必须首先计算领导人和他的追随者在联盟达到多数之前的预期损失。在这里，我们可以安全地假设该神谕系统决定成为领导者，因为其计算出的持续故障概率非常低 $p_{cf} \approx 0$。他必须发出强烈的信号。他的信号强度有多强，需要投多少轮，这取决于 p_{sf} 在一轮中出现错误的概率。首先，本轮有一个 “信号” 从其他出错结果的神谕系统。每一个试图发现领导者的神谕系统者都会考虑概率 p_{sf}，神谕系统的数量 $|O_{n-1}|$ ，包括上一轮的错误投票者。假设 w 为的期望值，我们有：$E[w] = \frac{p_{sf}}{2}|O_{n-1}|$ 因为平均值一半有缺陷的神谕系统会给出错误值。

值 w 越大，就越有可能至少有一位神谕系统试图成为领导者。让

$\Delta w = w - E[w] > 0$ Δw 表示一些神谕系统试图组成联盟。

我们将信号强度定义为信号强度，并用 ss 表示神谕系统故意投错票的概率。仅此一项指标就足以让一个神谕系统判断是否另一个神谕系统发起联盟。如果对于一个神谕系统 $ss \approx 1$，这意味着这个神谕系统试图成为领导者。一个领导者面临的严重问题是，如果他的信号在一轮中出现故障，它的信号就会中断。在这一轮，可能投正确的值，削弱其信号。当本轮出现错误时，它可以通过投票选出正确的值 $\frac{p_{sf}}{2}$。我们必须计算信号强度 ss。对于第一轮，$r = 1$，故障概率为 p_{sf} 和持续故障 p_{cf}。所以错误地选择错误的值，总的概率是

$$\frac{p_{sf}}{2} + \frac{p_{cf}}{2} - \frac{p_{sf}}{2}\frac{p_{cf}}{2} = \frac{p_{sf} + p_{cf}}{2} - \frac{p_{sf}p_{sf}}{4} = \frac{2p_{sf} + 2p_{cf} - p_{sf}p_{cf}}{4}$$

我们可以用 ss_1 来表示第一轮探测的信号强度。
$ss_1 = 1 - (\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4}) \Leftrightarrow ss_1 = \frac{4-2p_{sf}-2p_{cf}+p_{sf}p_{cf}}{4}$
第二，第三，。。。第 i 轮每轮信号强度都是相同：
$ss_i = \frac{4-2p_{sf}-2p_{cf}+p_{sf}p_{cf}}{4}$
由于轮数是线性独立的，神谕系统在 r 轮后发送给其他人的故意错误投票的信号强度为：

$\overline{ss_{1,r}} = \overline{ss_1}\ \overline{ss_2}\ \overline{ss_3}...\overline{ss_r} = \prod_{i=1}^{r} \overline{ss_i} = (\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4})^r$
$ss(p_{sf}, p_{cf}, r) = 1 - \overline{ss_{1,r}} = 1 - (\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4})^r$
$ss(p_{sf}, p_{cf}, r) = 1 - (\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4})^r$

(8.1.1)

方程（8.1.1）对我们的方案是最重要的。它将用于形成支付函数 $B()$ 它是值得多分析一下。$ss()$ 依赖于 r, p_{sf} 和 p_{cf}。让我们根据输入来检验函数 $ss(p_{sf}, p_{cf}, r)$ 的单调性。为了证明单调性，我们必须找到每个参数的导数函数。

显然，$\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4} < 1$
因此：

$$0 < p_{sf} < 1,\ 0 < p_{cf} < 1$$
$$p_{sf} < 1 \wedge p_{cf} < 1$$
$$\Rightarrow p_{sf} + p_{cf} < 2$$
$$\Leftrightarrow 2p_{sf} + 2p_{cf} < 4$$

和：

$$p_{sf} > 0 \wedge p_{cf} > 0$$
$$\Rightarrow p_{sf}p_{cf} > 0$$
$$\Leftrightarrow -p_{sf}p_{cf} < 0$$

我们有：

$$2p_{sf} + 2p_{cf} < 4 \wedge -p_{sf}p_{cf} < 0$$
$$\Rightarrow 2p_{sf} - 2p_{cf} - p_{sf}p_{cf} < 4$$
$$\Rightarrow \frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4} < 1$$

证明完毕
我们将研究 $ss(p_{sf}, p_{cf}, r)$ 为所有变元的单调性

$$\frac{dss}{dp_{sf}} = \frac{(1-(\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4})^r)'}{dp_{sf}} = 0 - (\frac{2*1+2*0-0*1}{4})^r = -(\frac{1}{2})^r < 0$$

所以：

$$\frac{dss}{dp_{sf}} < 0$$
(8.1.2)

导数函数总是负的，因此 $ss()$ 是单调的，因为它严格地随着 p_{sf} 的增加而减少。
同样：

$$\frac{dss}{dp_{cf}} = \frac{(1-(\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4})^r)'}{dp_{sf}} = 0 - (\frac{2*0+2*1-0*1}{4})^r$$
$$-(\frac{1}{2})^r < 0$$
$$\frac{dss}{dp_{cf}} < 0$$
(8.1.3)

$ss()$ 也随着 p_{cf} 的增加而单调下降。
结果是，神谕系统出错的概率越接近于零，投票错误的信号就越强。在边缘情况下，$p_{sf} = p_{cf} = 0$（8.1.1）给出 $ss(p_{sf}, p_{cf}, r) = 1.$ 的结果。这意味着即使从第一次投票反对实际价值开始，神谕系统可以确定这个神谕系统想要成为领导者，并以 100% 的概率建立联盟。这是合乎逻辑的，因为一个没有缺陷的理性神谕系统只有在有理由（向他人发出信号）的情况下才会愿意接受惩罚。

最后，让我们检查 $ss()$ 在 r 变化时的单调性。我们预计，随着 r 的增加，信号强度 $ss()$ 也会增加。的确：

$$\frac{dss}{dr} = (\frac{(1-(\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4})^r)}{dr})' = (\frac{-(\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4})^r}{dr})'$$

但 $\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4}$ 不依赖于 r，所以我们可以替换它。
让：

$$\omega = \frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4}$$

我们有：
$\frac{dss}{dr} = (\frac{-\omega^r}{dr})' = -(\frac{e^{r\ln\omega}}{dr})' = -e^{r\ln\omega}(\frac{r\ln\omega}{dr})' = -e^{r\ln\omega}\ln\omega(\frac{r}{dr})' = -e^{r\ln\omega}\ln\omega * 1 = -e^{r\ln\omega}\ln\omega$
另外，$e > 0 \Rightarrow e^{r\ln\omega} > 0$。而，我们已经证明了 $\frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4} < 1$
因此：

$$\omega = \frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4} < 1 \Rightarrow \omega < 1 \Leftrightarrow \ln\omega < \ln 1 \Leftrightarrow \ln\omega < 0$$
$$e^{r\ln\omega} > 0 \wedge \ln\omega < 0 \Rightarrow e^{r\ln\omega}\ln\omega < 0 \Leftrightarrow -e^{r\ln\omega}\ln\omega > 0 \Rightarrow \frac{dss}{dr} > 0 \tag{8.1.4}$$

(8.1.4) 表明 ss 对 r 的导数总是正的，因此它是单调递增的。当然，这正是我们所期望的。每一次错误的投票都会加强信号的强度。

让我们看一个例子。假设每一轮神谕系统都有 15% 的错误概率和 3% 的持续错误概率。另外，对神谕系统投三次 $r = 3$ 反对票。这个神谕系统信号期望形成联盟的可能性有多大？从（8.1.1）我们得到：

$$
\begin{aligned}
& ss(p_{sf}, p_{cf}, r) = 1 - (\tfrac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4})^r \\
\Rightarrow\ & ss(0.15, 0.03, 3) = 1 - (\tfrac{2*0.15+2*0.03-0.15*0.03}{4})^3 \\
\Leftrightarrow\ & ss(0.15, 0.03, 3) = 1 - (\tfrac{0.3+0.06-0.0045}{4})^3 \\
\Leftrightarrow\ & ss(0.15, 0.03, 3) = 1 - (0.21375)^3 \\
\Leftrightarrow\ & ss(0.15, 0.03, 3) = 1 - 0.009766037109375 \\
\Leftrightarrow\ & ss(0.15, 0.03, 3) = 0.990233962890625
\end{aligned}
$$

这意味着神谕系统有 99% 的概率试图建立联盟。只有 1% 的人怀疑它有问题。

我们可以改进更多（8.1.1）。到目前为止，我们假设每轮投票都是独立的，但这并不完全正确。如果神谕系统经常出错，那么线性独立性假设就不太可靠。在我们的例子中，连续出现错误的神谕系统的概率为：

$$
ss(p_{cf}, r) = p_{cf}(\tfrac{1}{2})^r = 0.03 * 0.5^3 = 0.03 * 0.125 = 0.00375 = 0.375\%
$$

在我们的例子（8.1.1）中可以这么认为。不过，也有另一种情况也的可能：$p_{cf}(\frac{1}{2})^r > \overline{ss_{1,r}} = \frac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4}{}^r$ 更准确地说，我们必须找这两个表达式的最低值，正如我们所期待的理性神谕系统所做的。最后，我们将（8.1.1）替换为（8.1.5）

$$
ss(p_{sf}, p_{cf}, r) = min\{1 - (\tfrac{2p_{sf}+2p_{cf}-p_{sf}p_{cf}}{4})^r, 1 - p_{cf}(\tfrac{1}{2})^r\} \tag{8.1.5}
$$

领导者将在前 k 轮接受惩罚，直到看到大多数人随他（他们复制了他的价值）。在这一点上，他将切换到下一个阶段——过渡阶段。至于追随者，对他们的最初惩罚将比领导者轻，因为他们会在几轮之后加入（当他们几乎确定了领导者）。他们最好的策略——他们的主导策略——就是跟随领导者。

对于领导者来说，协议中的负回报必须很高，以防止他发起联合。也就是说，

$$
\sum_{i=1}^{k} B_{O_l} - \sum_{i=k+1}^{\infty} EB_{C_{O_l}} < 0 \tag{8.1.6}
$$

B_{O_l} 为初始惩罚 $EB_{C_{O_l}}$ 为领导者预期联盟的分红。

理论上，上述不等式没有解决方案。这是因为轮次是无限的，这意味着联盟盈余的红利总额是无限的，所以它总能覆盖最初的惩罚。幸运的是，有一个聪明的策略，即 NBP （人无完美），可以解决这个问题。

8.2 过渡阶段

领导者可以很容易地发现其追随者的数量，应用（8.1.5）的等式来计算其他每个神谕系统。当它确信有多数票时，它将从反对实际价值转到投真正的价值。这只会发生在一轮。这是一个广播信息，意思是“每个人现在都切换到这个常量值”。领导者不能投票或非法投票以达到同样的结果。在下一轮，所有其他成员将看到转变（除了有问题的成员）。他们可以立即跟踪或等待几轮。对于将要等待的神谕系统，领导者不是故意切换而是发生故障的概率由等式（8.1.5），在 $r = 1$ 和 $p_{cf} = 0$ 神谕系统可以安全地得出结论，领导者并不是出现故障的:

$$ss(p_{sf}, p_{cf}, 1) = min\{1 - \frac{2p_{sf}}{4}, 1\} = 1 - \frac{p_{sf}}{2} \tag{8.2.1}$$

在每轮 c , r，增加领导者的投票与实际值相同时。换言之，忽略领导投票反对实际值的几轮，可以使用（8.1.5）计算领导者转换的概率，在 $p_(cf) = 0$:

$$ss(p_{sf}r) = min\{1 - (\frac{2p_{sf}}{4})^r, 1\} = 1 - (\frac{p_{sf}}{2})^r \tag{8.2.2}$$

在领导者不改变出价值方式情况下，r 是不计算领导投票反对实际值的轮数。在几个轮内 $ss(p_{sf}r) \approx 1$，此时所有神谕系统都将跟随领导者。这确定终觉了过渡阶段，并建立联盟完成。

8.3 维持阶段

在建立联盟之后，成员们总是投出同一个值，而不管实际值是多少（总是对还是总是错）。实际上，他们甚至不需要从现实世界中获取价

值。其他属于小规模的联盟和未加入联盟的神谕系统，每次正确的数值与占主导地位的联盟投票不同时，都会受到经济损失。所以他们因为诚实而受到惩罚。他们的主要策略是加入联盟。事实上，因为神谕系统的行为是理性的，他们将在过渡期结束后加入第二轮。对于大量的神谕系统 $|O_n|$ 来说，即使在联盟进入最后阶段之前，他们也能以很高的概率发现联盟。使用 (8.1.5) 对彼此的神谕系统，他们可以得出一个非常接近 $p_a = 1$ 的结论，即联盟已经发生：

$$p_a = 1 - \prod_{i=1}^{c}(1 - ss(p_{sf}, p_{cf}, r_c)) \approx 1$$
(8.3.1)

其中 $c = |C|$ 是可能的联盟成员（经常投票反对实际价值的神谕系统）。

因此，所有的神谕系统-除了那些经常犯错的，无法察觉其他神谕系统的行为-都将加入联盟，组成格兰特联盟。此后，每次的输出值都是一样的。这是理性神谕系统的平衡点，在这个平衡点上，输出毫无价值，使系统完全无用。

很明显，这个系统决不能达到这种状态。系统架构必须能够防止这种情况发生。

9 系统架构

9.1 简介

我们系统的目标是为消费者生产最优质的结果（§3），必须成功地应对这两种威胁。一是搭便车的信息，可以降低、二神谕系统的联盟输出的结果（§4）。

搭便车信息可以很容易地被提交-揭示策略所阻止。一对“提交”和“揭示”是一轮的定义。提交值是对所有神谕系统都通用的某个算法的投票的哈希结果。预哈希的消息不仅包含投票值（因为只预先计算两个可能的组合，在隐藏之前将显示投票结果）。预哈希值包含：

- • 神谕系统的 ID。这是为了防止其他神谕系统复制哈希值。如果没有 ID，任何神谕系统都可以复制并提交哈希值（因为在现实中，系统不是同步的），从而复制另一个神谕系统的投票。包含神谕系统的 ID 将使所有其他神谕系统的哈希无效。对于区块链，公共地址可以用作 ID。
- • 一个固定长为度、并使一次性随机值。这是为了防止其他神谕系统成功蛮力的预哈希值。协议必须将第二次使用相同随机值的预哈希值视为无效。原因很明显，使用相同的值实际上揭示了投票前的隐藏步骤。
- • 单位布尔值。这是投票（1 代表“正确”，0 代表“错误”）。

从上面我们可以确定，没有搭便车的情况发生，因为本轮投票的复制是不可能的，在我们假设的，神谕系统不可能直接沟通的情况下（§2.3）。

对付正在形成的联盟或解散现有的联盟更具挑战性。乍一看不等式（8.1.6）令人沮丧。神谕系统的主导策略，即使是经常犯错的神谕系统，都是加入联盟。均衡点是大联盟的建立，此时产出值完全是无用。由于在无限轮的假设，任何开始的惩罚都不能阻止他们达到这一点。

有人可能会争辩说，当大多数票的神谕系统输出值决不改变时，这整体体系很有可能成立一个联盟。如果实际值（不管是真值还是假值）都不接近 100%，这将是真的。不幸的是，在现实上，消费者可以设置一长串事件进行验证，而相同值的概率接近 $p = 1$。例如，要求保证产品质量。产品可能只有 $\frac{1}{1000}$ 甚至 $\frac{1}{10000}$ 的变化会导致功能紊乱。所以，记录相同的顺序值一千或更多回合是有效的，但用这个逻辑检测是无效的。系统不能仅仅从这个事实来判断，因为它不知道每个事件的真正值。只有没有错误的神谕系统才能读到真正的值。

幸运的是，有一个应对这种情况的策略，*人无完美*的策略，我们将其简称为 *NBP*。

9.2 人无完美（NBP）策略

NBP 是一种非正统的方法，至少乍一看，它可以阻止参与者作弊。不等式（8.1.6）看起来不可能被击败，因为当我们建立它的时候，我们假设神谕系统每次投票“正确”时都会得到奖励（这与大多数人一致）。惩罚（或至少不奖励）赢得投票的神谕系统，有时看起来不合理，但仔细看是非常合理的。因为没有人是十全十美的 $p_{sf} \neq 0$，一个神谕系统不可能在任何时候都 100% 正确地投票。在大量样本的轮数，它偏离预期值太多，即 $E[X] = k(1 - p_{sf})$，其中 k 是样本量（轮数），正如我们已经看到的。

以现实生活中的一个例子，弱智学生作弊时并没有正确回答所有的考试，而是故意犯错。这是为了让他们的老师相信他们没有使用其他帮助。他们希望自己的表现看起来真实，因为它更接近预期。

我们将更正式地研究上述问题。我们假设神谕系统每轮都可能出现故障，概率 $p_{sf} > 0$。假设神谕系统是诚实的，它有可能投票给真实值 $p_c = \overline{p_{sf}} = 1 - p_{sf}$，并对所有 r 轮投票正确：

$$p_{c=r} = \prod_{i=1}^{r} (1 - p_{sf}) = (1 - p_{sf})^r \tag{9.2.1}$$

例如，如果随机抽样 $r = 100$，$p_{sf} = 5\%$ ，则神谕系统在没有加入联盟的情况下正确投票的概率为：

$$p_{c=100,r=100} = (1 - p_{sf})^r = (1 - 0.05)^{100} = 0.95^{100} = 0.00592$$

我们可以得出结论：神谕系统是幸运的，在概率为 0.592% 的情况下，得到了所有正确的答案，并且概率为 $100\% - 0.592\% = 99.408\%$ 神谕系统在作弊（作为联盟的一员）。

我们要惩罚每一个“太幸运”的神谕系统。惩罚将取决于 r 和 p_{sf} 值以及基于预期性能: $E[c] = r(1 - p_{sf})$。在上述示例中，$E[c] = r(1 - p_{sf}) = 100(1 - 0.95) = 95$ 因此，在 r 轮之后，协议必须移除神谕系

统从预期奖励中获得的任何超额奖励。更好的是，每轮奖励将被计算，但直到样本 r 结束时才给出。我们将这批连续轮回称为会期。

我们将为系统引入一个常数，阈值 th，它是方程（9.2.1）概率的极限。根据这个参数和 p_{sf} 代表了每一个神谕系统在一轮中出现故障的概率（不能得到实际值），系统可以计算 r。每个会期由连续的轮组成，因此 r 可以从（9.2.1）开始计算，用阈值 th 代替 $p_{c=r}$:

$$\begin{gathered} p_{c=r} = (1-p_{sf})^r \\ \Rightarrow th = (1-p_{sf})^r \\ \Leftrightarrow ln_r th^r = ln_r (1-p_{sf})^r \\ \Leftrightarrow rlnth = ln(1-p_{sf}) \\ \Leftrightarrow r = \frac{ln(1-p_{sf})}{lnth} \end{gathered}$$

例：我们认为神谕系统幸运的概率是 $1/500$。所以 $th = 1/500 = 0.002$。我们认为 $p_{sf} = 7\% = 0.07$。因此，每轮数 r 的会期必须是：

$$r = \frac{ln(1-p_{sf})}{lnth} = \frac{ln(1-0.07)}{ln0.002} = \frac{ln(0.93)}{ln0.002} \approx \frac{-0.072570692834835}{-6.21460809842219} \approx 85.635$$

r 必须是一个整数，所以 $r = \lceil 85.635 \rceil = 86$，这意味着每节课必须有 86 轮。因此：

$$r = \lceil \frac{ln(1-p_{sf})}{lnth} \rceil$$
$$(9.2.2)$$

每会期回报也必须计算出来（没有最终的超额移除，没有 NBP 策略）。让我们把它象征为 $E[r_s]$。对每一个不经常出错的神谕系统者，的预期奖励（红利）的随机一轮 i 是 $E[r_i] = (1-p_{sf})\frac{R}{|O|(1-sp_f)} = \frac{R}{|O|}$。这里，$R$ 是每一轮的外部奖励（服务的消费者给予这个奖励），而 $|O|$ 神谕系统的数量。我们期望 $|O|(1-sp_f)$ 神谕系统是无缺陷的，并且从 R 中得到红利。神谕系统没有缺陷的概率为 $(1-p_{sf})$，因此它也代表在第 i 轮中获得红利的概率。对于一个会期，一个没有行为不端、并且不常出错的神谕系统的预期回报是：

$$E[r_s] = rE[r_i]$$
$$\Rightarrow E[r_s] = \frac{rR}{|O|}$$
$$(9.2.3)$$

（9.2.3）为我们提供了整个会期中每个神谕系统者的预期奖励。我们的 NBP 策略必须从实际奖励中移除任何超额奖励。在会期的最后一轮，协议将给出

$$min\{r_s, \frac{rR}{|O|}\}$$
$$(9.2.4)$$

其中 r_s 是如果没有 NBP 策略的情况下神谕系统将获得的总会期奖励。可能每轮的奖励 R 是不同的，因为系统是接受不同的客户愿意支付不同的金额。在这种情况下，我们可以简单地用平均会期奖励 $\overline{R}$ 代替 R。

这种策略的好处是什么？这个系统策略的好处是，当它属于一个联盟时，它降低了对未来轮回的回报预期，当神谕系统诚实时，也把奖励预期宽松到预期限制上。这具有使联盟盈余归零的性质。也就是说，（9.2.4）带来了（5.2）的期望结果：

$$\lim_{r->\infty} Cs, i = 0 \ \ \forall i \ \in C$$

研究（8.1.6）在 NBP 策略下我们可以很容易地看到现在它每次都适用。现在 $\sum_{i=k+1}^{\infty} EB_{C_{O_l}} = 0$ ，因为预期的联盟盈余为零。而且，$\sum_{i=1}^{k} B_{O_l}$ 是负数的，因为领导者或追随者在创建阶段和切换阶段。这就带来了一个事实，即（8.1.6）不等式每次都是对的。任何理性的神谕系统，以及我们的假设（§2.1）所有的神谕系统，都不会试图组成或加入联盟。有了 NBP 战略，联盟的威胁就被成功地反击了。

9.3 奖惩

在形成支付函数 $B()$ 之前，需要对奖惩进行总结。收益函数必须是：

1. 防止经常出错的神谕系统参与（§7.4b）。它必须严惩或完全禁止这些神谕系统。我们更要篇处罚而不是永久禁止，因为总有可能误报的小错误，特别是在阈值 th 很高的情况下。选择权（和风险）由每个神谕系统决定。无论如何，不断错误的神谕系统会受到重罚，迫使他们最终停止。
2. 阻止神谕系统加入联盟，即成为追随者（§7.4d）。必须在创建阶段采用惩罚和 NBP 策略相结合的方法，这样将预期联盟盈余归零。
3. 阻止神谕系统建立联盟（领导者，§7.4f）。与上述类似，在创造阶段的惩罚和 NBP 策略相结合是必要的。
4. 对于所有非经常性错误诚实节点，总期望利润必须为正。换句话说，他们必须有利润。不然，神谕系统就会离开系统。

现在我们来了解支付函数 $B()$.

9.4 形式支付功能

支付功能必须实现上述所有 §9.3.1-§9.3.4 的几点。$B()$ 的参数是系统的输入 th, R_{min},p_{sf}。R_{min} 是系统从客户那里接受的最低回报，因为它必须涵盖神谕系统的运营成本。如果一个客户很匆忙，而有一个队列，也就是说，他需要优先于其他客户，那么他可能愿意提供比 R_{min} 更高的 R 奖励。但在任何情况下，对于任何一轮 i，$R_i \geq R_{min}$。我们将在系统核心架构研究了解 R_{min} 更详细。

另外，r 是根据（9.2.2）计算的输入：$r = \lceil \frac{ln(1-p_{sf})}{lnth} \rceil$ 因为不断犯错的或潜在的领导者必须被紧急反击，惩罚必须具有足够的严重。我们将遵循指数法。每一次神谕系统不断地出现错误投票，惩罚就会成倍增加。开始时，不需要任何押金。在第一个错误出现之后，必须有押金作为担保，这取决于概率 p_{sf}。系统的高 p_{sf} 必须更容易忍受，而低 p_{sf} 必须施加更重的惩罚。我们不能忘记我们所设计的 §9.3.4 这些限制，它规定诚实节点的报酬预期必须是正的，$E[s_i] > 0$，否则它们将放弃系统。

对于一个诚实的节点，一次不走运的概率是 p_{sf}，两次 p_{sf}^2 甚至连续 n 次 p_{sf}^n。另外，如果没有惩罚，它期望会期平均奖励：$E[r_s] = \frac{r\overline{R}}{|O|}$(9.2.3) $E[r_s] > 0$ 为正。但是，对于诚实的神谕系统，即使在减去惩罚之后，预期也必须是正的。如果 PU_s 是我们的全部惩罚：

$$\begin{gathered} E[r_s] - PU_s > 0 \\ \Rightarrow \frac{r\overline{R}}{|O|} - PU_s > 0 \\ \Leftrightarrow PU_s < \frac{r\overline{R}}{|O|} \end{gathered} \tag{9.4.1}$$

这应该是我们的极限。我们设置 $PU_1 = 0$，这意味着对于第一个错误，我们只是容忍神谕系统。但是连续两次投错票我们将：$PU_2 = \frac{R_2}{|O_c|p_{sf}}$ 对 e 序列的误差，我们设置 $PU_e = \frac{R}{|O_c|p_{sf}^{(e-1)}}$。每次新一轮开始前，担保金额必须始终覆盖该值。在每一轮连续的错误投票中，都会有一笔新的存款来覆盖 PU_e。正确的投票会将超出的保证释放回给神谕系统，并将 PU 总计为零。显然，对于一个诚实的节点来说，最坏的情况是所有轮都出现故障（一种不同的情况）。预期的惩罚是：

$$E[PU_r] = p_{sf}^r \frac{\overline{R}}{|O_c|p_{sf}^{(r-1)}} + \sum_{i=2}^{r} ED_i \tag{9.4.2}$$

p_{sf}^r 为诚实的神谕系统每轮都不幸（错误）的概率，$\overline{R}$ 是平均奖励或轮次，而 $|O_c|$ 是投票正确并在每轮奖励获得红利的神谕系统。使用 (9.4.2)，我们必须证明 $maxE[PU_e] < \frac{r\overline{R}}{|O|}$ 我们的协议不会对诚实的节点造成太大的破坏，迫使他们离开系统。

此外，惩罚还包括在第一个错误的答案后，不会给任何奖励，所以这也增加了惩罚。我们将预期奖励象征为 $\sum_{i=2}^{r} ED_i$ 如（9.4.2）最后，$|O_c| = \overline{p_{sf}}|O|$ 我们必须证明 $maxE[PU_e] < \frac{r\overline{R}}{|O|}$ (9.4.3)。我们有：

$$\begin{aligned} maxE[PU_e] = E[PU_r] = p_{sf}^r \frac{\overline{R}}{|O_c|p_{sf}^{(r-1)}} + \sum_{i=2}^{r} ED_i \\ = p_{sf}\frac{\overline{R}}{|O_c|} + \sum_{i=2}^{r} p_{sf}\frac{\overline{R}}{|O_c|} = p_{sf}\frac{\overline{R}}{|O_c|} + (r-1)p_{sf}\frac{\overline{R}}{|O_c|} \end{aligned}$$

$$= rp_{sf}\frac{\overline{R}}{|O_c|} = p_{sf}\frac{r\overline{R}}{\overline{p_{sf}}|O|} = \frac{p_{sf}}{1-p_{sf}}\frac{r\overline{R}}{|O|}$$
(9.4.4)

我们假设概率 p_{sf} 低于 50%（§7.2.6）。因此，

$$\begin{gathered} p_{sf} < \tfrac{1}{2} \\ \Leftrightarrow 2p_{sf} < 1 \\ \Leftrightarrow p_{sf} + p_{sf} < 1 \\ \Leftrightarrow p_{sf} < 1 - p_{sf} \\ \Leftrightarrow \tfrac{p_{sf}}{1-p_{sf}} < 1 \\ \Leftrightarrow \tfrac{p_{sf}}{1-p_{sf}}\tfrac{r\overline{R}}{|O|} < \tfrac{r\overline{R}}{|O|} \\ \Rightarrow maxE[PU_e] < \tfrac{r\overline{R}}{|O|} \end{gathered}$$

证明完毕

我们证明我们的支付函数符合（§9.3.4）。诚实的节点可以留在系统中。但是攻击者（潜在的领导者）和经常出错的神谕系统必须受到严厉的惩罚，迫使他们离开我们的系统。事实上，（9.4.3）保证了这一点。要理解这一点的关键是 p_{sf}^r。让我们分析三种情况的限制，这些包括对于诚实的节点、攻击者和经常出错神谕系统的节点：

- 诚实的神谕系统：$\lim_{r->\infty} p_{sf}^r = 0$ 因为 $p_{sf} < 1$
- 攻击者（领导者）：1，因为他自愿选择错误地投票 $p_{sf} = 1$
- 功能失调的神谕系统：因为它无法获得真正的价值，它平均有一半的投票是正确的。长期而言，大数的 r，它的财务预期是负的，所以理性的行为它会离开系统。

作为攻击者的一个例子，如果 $p_{sf} = 5\%$，连续 4 次错误投票 $r = 4$ 将导致他失去这几轮的 4 次红利加上：$PU_4 = 1^4\frac{\overline{R}}{|O_c|0.05^{(4-1)}} = 20^3\frac{\overline{R}}{|O_c|} = 8000\frac{\overline{R}}{|O_c|} = 8000\overline{D}$。这意味着它必须支付 8000 奖励！这样的话，显然攻击者是在经济上自杀的。我们可以将（9.2.4）和（9.4.2）结合起来，将我们的支付函数形式化。

$$B(th, R_{min}, p_{sf})$$
(9.4.4):

- 每会期的轮数：$r = \lceil \frac{ln(1-p_{sf})}{lnth} \rceil$ (9.2.2)

- NBP 战略：$min\{r_s, \frac{rR}{|O|}\}$(9.2.4)
- 处罚 $PU_e = p_{sf}^e \frac{\overline{R}}{|O_c| p_{sf}^{(e-1)}} + \sum_{i=2}^{e} ED_i$ （根据 9.4.2）

9.5 核心架构

我们终于准备好了解整个系统的描述。

9.5.1 输入

我们的系统需要一些必要的输入来实现上述所有功能。这些输入包括：

1. th，阈值是方程（9.2.1）概率的极限。
2. R_{min} 系统从消费者接受的最低奖励
3. p_{sf_0}，神谕系统在每一轮都可能出错的初始已知概率。这些信息非常重要，且必须非常接近实际。
4. z 和 e，其中 z 是置信水平，e 是最大误差可接受性。参见确定样本量。该值将用于设置一个间隔，该间隔将重新计算每个纪元的概率 p_{sf} 。纪元是一个轮数，可使 r 整除（包含会期值）

9.5.2 初始计算

最初，我们的系统计算 r 和 ep 值。

- r 是每会期的轮数，根据计算公式：(9.2.2)
- ep 使用公式：$ep = \frac{p_{sf_0}(1-p_{sf_0})z^2}{e^2}$ 计算
- ep 是纪元的整数，并且必须可以被 r 整除。所以 ep 必须向上取整，$ep \equiv 0(modr)$

ep 表示足以重新计算 p_{sf} 的样本规模。有两个原因，它必须重新计算：

- • 最初估计的概率 p_{sf_0} 可能与实际概率相差很大。在这种情况下，可能会出现严重的问题，使用上一个纪元的数据重新计算 p_{sf} 可以解决该问题。
- • 实际运行中，情况可能会发生变化。概率可能因外部事件而改变。重新计算有助于系统更新更接近真实值。

9.5.3 过程

我们的系统运行过程如下：

- 用户设定一个句子，支付奖励，并等待系统的输出，这是一个布尔值数组（选票）。用户还可以访问账本，读取每个神谕系统的性能历史记录。从这两个方面，他们可以了解（或说服他人）可疑事件是真是假。
- 系统检查他们的奖励是否低于最低值。如果是，则返回并拒绝请求。
- 对于许多请求，系统优先考虑具有较高奖励的用户。
- 根据（§9.1），神谕系统同时投票，遵循提交-揭示计划
- 如果神谕系统的投票无效或属于少数部分，则必须向系统提交担保。根据 $B()$ 和惩罚函数（§9.4），连续错误投票增加每轮存款的金额。一次正确的投票会重置惩罚计数器。
- 来自惩罚的收入保留在系统中，它们不会作为奖励，不会分给其他神谕系统。这避免了对诚实节点的多种类型的攻击，因为它消除了任何动机。
- 奖励在每轮计算，并在占多数的神谕系统之间平均分配。奖励不是每轮都给予，而是在会期的最后一轮后累积和释放。
- 在每个纪元结束时，简单地通过获得上一个纪元的错误与总票数的比率来重新计算 p_{sf} 。纪元持续时间可能只有一个会期。
- 如果神谕系统的奖励超过了会期的预期值，则超额部分将被系统削减并保留：人无完美的策略（§9.2）
- 根据支付函数（9.4.4）始终分配奖励
- 在系统下包括神谕系统之间支付都完全禁止。在任何这种情况下，所有相关的神谕系统立即被禁止。所有的担保都被没收了。
- 新神谕系统只能在第一轮加入。

以上内容完成了协议。我们希望通过数学方法能够证明，在（§2）的假设下，该协议实现了所有目标（§3）。

10 结论

我们提供了一个系统和一个协议，强制理性的神谕系统诚实。诚实的神谕系统与真相 p_{sf} 不高的事实相结合，产生了一个高质量的输出值，消费者可以以非常高的准确率得出事件的结论。

为了完整看待一个系统，我们需要了解我们系统的币端。我们的系统在以下情况失效：

- 实际错误概率与初始概率大不相同。这可以在下一个纪元时纠正
- 实际错误概率非常接近 50%
- 神谕系统可以在系统外部进行通信（侧信道）。这样，系统就不再是 NTU 了，还有纳什均衡点使得输出无用
- 许多神谕系统在系统中没有理性行为，他们行为恶意（他们愿意承受巨大的经济损失来降低系统输出）。神谕系统可能看起来不合理，如果它们是由系统之外的人支付的（贿赂）。这有影响神谕系统共识传递错误的价值，如果行贿者愿意付出高昂的代价。在这种情况，神谕系统的行为在系统之外是理性的，如果这只发生在一个特定的轮回。无论如何，我们所说的是神谕系统行贿，而系统却无法察觉。

我们的系统设计完美，在以下情况下可产生最佳质量输出（在 p_{sf} 的范围内允许）：

- 我们所有的假设（2.1-2.7）成立。
- 系统已知高精度的真实 $p_{sf_{real}}$，即 $p_{sf_{real}} \approx p_{sf_0}$。如果每一个神谕系统在一轮中出现故障的真实概率与设定的值相差甚远，那么 $B()$ 支付函数将是不正确的。要么过度惩罚神谕系统，使他们的操作无利可图，因为预期收益将为负，并迫使他们离开系统，或者对他们进行惩罚，允许他们成立和维持联盟。在上述假设下，我们证明了我们的系统可以产生最佳的输出，p_{sf} 的限制下。p_{sf} 越近 $\frac{1}{2}$，输出的有用性就越低。

- On Harsanyi Dividends and Asymmetric Values, https://www.researchgate.net/publication/336722902_On_Harsanyi_Dividends_and_Asymmetric_Values
- Determining sample size; how to calculate survey sample size https://www.researchgate.net/publication/322887480_Determining_Sample_Size_How_to_Calculate_Survey_Sample_Size
- Binomial proportion confidence interval https://en.wikipedia.org/wiki/Binomial_proportion_confidence_interval
- The free-rider problem https://en.wikipedia.org/wiki/Free-rider_problem
- Entropy - Information Theory https://en.wikipedia.org/wiki/Entropy_(information_theory)
- Essentials of Game , Theory Kevin Leyton-Brown and Yoav Shoham , 2008
- Cooperative games: core and Shapley value , Roberto Serrano , 2007
- On non-transferable utility games with coalition structure https://www.academia.edu/29499635/On_non-transferable_utility_games_with_coalition_structure
- Cooperative Game Theory and Its Application in Localization Algorithms https://www.intechopen.com/books/game-theory-relaunched/cooperative-game-theory-an
- An analytical study of the N-person prisoners' dilemma https://www.researchgate.net/publication/266710753_An_analytical_study_of_the_N-person_prisoners'_dilemma