最近的考古发现表明，在Paxos小岛上，尽管兼职议会成员都有逍遥癖，但议会模式仍然起作用。他们依旧保持了一致的会议记录，尽管他们频繁的进出会议室并且他们的信使还很健忘。Paxon议会协议提供了一种新方法去实现设计分布式系统的状态机。

---

1 问题

1.1 Paxos小岛

公元十世纪初，爱情海上的Paxos小岛是一个繁荣的商业中心。财富导致了政治的复杂化，Paxon的公民采用了议会形式的政府代替了古代的神权政治。但是商业在公民义务之上，在Paxon，没有人愿意将其一生投入到议会当中。Paxon议会需要在议员们不断出入议会的情况下保持工作。

兼职议会面对的问题和当今的容错分布式系统是非常相似的，议员对应于进程，议员离开会议室则对应于分布式系统中的失败（故障）。因此，Paxon的解决方案也许对计算机科学会有借鉴意义。这里简短的介绍Paxon议会协议，然后更简短的介绍它和分布式系统的相关性。

Paxon文明被外国文明入侵破坏，最近考古学家才发掘它的历史。因此，我们对Paxon议会的了解是支离破碎的。尽管我们了解基础的协议，但是并不清楚细节。出于兴趣，我将冒昧的推测Paxons可能做些什么。

1.2 要求（Requirements）

议会的核心任务是决定岛上的法律，这是由它通过的一系列法案组成的。现代议会会雇佣一个专职秘书来做会议记录，但在Paxon，没有人愿意把自己所有时间投入到议会当中作为一个议会秘书。作为替代方案，每个Paxon的议员会维护一个律簿，用来记录已经通过的法令，每个法令都有一个序列号。例如议员A的律簿上会有这样一条法案：

> 155：橄榄油的税率是3个银币每吨

如果他相应155法令被议会表决通过。议员们采用不会被擦掉的墨水来记录法令，法案内容不会发生变更。

议会协议的第一个要求是***律簿之间的一致性***，这意味着任何两个律簿之间不能有相互冲突的内容。比如议员B在他的律簿上记录了：

> 132：油灯必须使用橄榄油

那么其他议员的律簿上的132法令不能出现不同的内容。但是另一个议员的律簿上的132号法令可以是空白的，因为他可能缺席了议会，还不知道议会已经通过了132号法令。

律簿之间的一致性还是不够的，因为保持所有律簿空白就可以轻易的保持律簿间的一致性。需要一些其他要求来保证法令最终被通过并记录在律簿中。在现代议会中，议员们之间的分歧阻止了法令的通过。在Paxon，情况并非如此，议员会同意所有提出的法令。但是他们的“逍遥癖”会成为一个问题。一致性会丢失，如果一组议员通过了：

> 37：禁止在圣殿墙上图画

之后离开了会议，而另一组议员进入了会议，并且对之前的通过的法令一无所知，然后通过了：

> 37：允许自由的艺术表达

那么一致性就失去了保证，除非足够多的议员长时间的呆在会议室中。因为Paxon议员不愿意取消他们的外出活动，那么不可能保证所有的法令都会被通过。但是无论如何，议员和他们的助理保证，只要他们在会议中，他们会快速的处理议会相关的议会事务。这保证了Paxon公民能设计出一个保证***进展性***的议会协议，只要满足：

> 如果大多数的议员在会议室中，并且在一段足够长的时间内没有议员进入或者离开会议室，那么任何一个议员提出的法案都会在会议室中被通过，并且被会议室中的所有议员记录在他们的律簿中。

1.3 假设（Assumptions）

通过给议员提供必要的资源，议会协议的要求（requirements）是可以达成的。每个议员需要一本坚固的律簿来记录法令，一支笔以及不可擦拭的墨水。议员们在离开会议室，在回来后可能会忘记之前在做什么，所以他们会把一些重要的备注记录在律簿的背面。律簿上的法令永远不会改变，但是背面的备注（notes）可能会被划掉。为了满足要求，议员们需要能知道时间的流逝，所以他们都拥有一个简单的计时器。

议员们无时无刻不带着他们的律簿，并且总是能够读取律簿上的法令和未被划掉的备注。律簿是用最好的羊皮纸做的，只用来记录最重要的备注。议员用其他纸条来记录其他的备注，这些纸条可能会在他们离开会议室的时候丢失。

会议室的环境是吵杂的，议员们只能通过信使（Messenger）来通信。信使们不会修改信息，但是可能会忘记他们已经传递过信息而重复的传递。和议员一样，信使也只是投入他们的一部分时间到议会的工作中（兼职的）。信使们可能会在传递信息前离开会议室——休一段时间的假期。也许他们也可能一去不复返，那么消息可能永远不被传递。

尽管议员和信使都可以在任何时刻离开和进入会议室，但是只要他们身处会议室，他们就会进到他们在议会中的职责。当他们在会议中，信使及时的传递信息，议员们对他们收到的信息立即做出响应。

官方声称信使和议员都是非常非常诚实的执行议会协议，但是很多学者认为这是一种宣传，为了突出Paxos在道德上由于东方的领国。但是因为官方文档中没有描述如何这种情况，我们也不知道议会会如何处理不诚实的信使和议员的情况。

2 单一法令圣会（The Single-Decree Synod）

Paxon议会是由早期的每19年执行一次的，为了选出一项具有象征意义的法令的圣会演化而来的。几个世纪以来，圣会通过要求牧师出席的传统程序来进行法令的选择。但是随着经济的繁荣，牧师们开始在圣会进行时进出会议室。最终，原来的协议失败了，圣会没有选择出任何法令。为了防止这种神学上的灾难，虔诚的人们请求数学家设计一套新的协议来选择圣会的法令。该协议（Synod-Protocol）的要求（requirements）和假设（assumptions）与后来的议会协议（Parliament-Protocol）是一样的，除了律簿只能记录一条单一的法令，而不是一系列的法令。议会协议（Parliament-Protocol）将在第3章中说明。

数学家们分多个步骤来推演Synod协议。他们首先证明了满足一定约束的协议能保证一致性，且同时可以推进。一个初步协议直接从约束条件中被推演出来，一个受限制的初步协议保证一致性，但是没有进展性。最终的Synod协议通过限制初始协议来达到可进展的目的。

数学家们的结论在2.1节中，协议的非正式描述在2.2-2.4节中。在附录中有更正式的描述和正确性证明。

2.1 数学结果

Synod法令通过一系列的有编号的选票来表决，一张选票是对单一的一个法令的表决。在一轮投票表决中，牧师只能选择投票或者不投票。和表决相关的是一群牧师的集合，称为quorum（法定人数）。一轮表决是否成功只取决于法定人数的牧师给该法令投票。正式的，一轮表决*B*包含下面四个部分：

B_{dec} 被表决的一条法令

B_{qrm} 非空的牧师集合，表决的法定人数集，一般指多数派

B_{vot} 投票给该法令的牧师集合

B_{bal} 选票序号

一轮表决B被认为是成功的，当且仅当$B_{qrm} \subseteq B_{vot}$（多数派赞成）。

表决的序号取于无界有序的数字。如果$B'_{bal} > B_{bal}$，那么表决B'在B之后。然而这并没有标明表决之间的执行顺序，后面的表决可能实际上比之前的表决更早进行。

Paxon的数学家们在一个由多轮表决构成的集合上定义了3个条件，证明如果这个集合β的表决满足这些条件，那么一致性将会被满足并且是可进展的。前两个条件非常简单，他们的非正式表述如下：

B1(β) 每一轮表决都有唯一的编号

B2(β) 任意两轮表决中的多数派都至少有一个相同的牧师。

第三个条件更为复杂，Paxon手稿中包含令人费解的描述：

B3(β) 对于 β 中每一轮表决B，如果B的法定人数集中的任何一个牧师在β中一个更小轮的表决中投过(赞成)票，那么B的法令与所有这些更小轮表决中最大那次表决法令相同。

为了对上面隐晦的文字加以说明，手稿中补充了图1，它通过一个拥有A、B、Γ、Δ、E五个牧师的五轮表决对B3(β)进行说明。

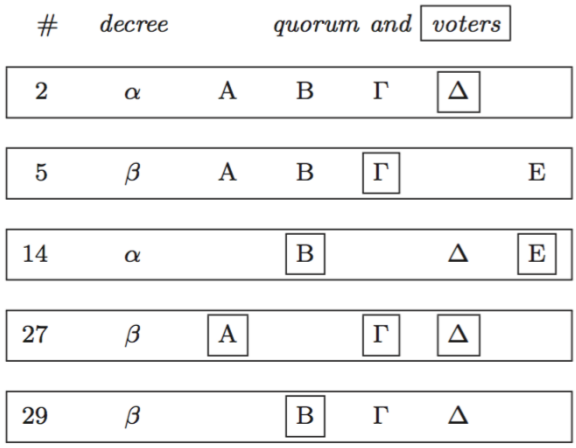

Fig. 1. Paxon manuscript showing a set $\mathcal{B}$, consisting of five ballots, that satisfies conditions $B1(\mathcal{B})$–$B3(\mathcal{B})$. (Explanatory column headings have been added.)

举个例子，标号为14的表决法令为α，多数派需要三个牧师，只有两个赞成的牧师。图1中对条件B3(β)的描述如下：

- 编号为2的表决是最早的表决，所以条件B3(β)是显然满足的
- 编号为5的表决中，四个参与者中没有任何一个在编号为2的表决中投过赞成票，所以也满足条件B3(β)
- 编号为14的表决中，只有Δ在之前的表决中投了赞成票，所以表决14和表决2的法令必须是相同的，图1中2和14的表决的法令都必须是α
- （这是一轮成功的表决）编号为27的表决中，之前投过赞成票的是Γ、Δ，Δ在表决2中投了赞成票，Γ在表决5中投了赞成票，所以表决27的法令必须和表决5的法令一致，即必须为β
- 编号为29的表决中，参与者是B、Γ、Δ，B在表决14中投了赞成票，Γ在表决5、27中投了赞成票，Δ在表决2、27中投了赞成票，所以表决29的法令必须和表决27的法令相同，即必须为β

B1(β) -B3(β) 正式的表述需要更多的符号。我们定义v为一次投票，它包含3个部分：v_{pst}表示牧师，v_{bal}表示表决编号，已经表决的法令v_{dec}。Paxon同样定义了v_{bal}=-∞并且v_{dec}=BLANK的投票为null。对于任何-∞<b<∞的表决编号b，不会以BLANK作为表决的法令。对于任意牧师p，他们也定义了$null_p$作为牧师p的null投票。

Paxon的数学家定义了投票的全局顺序，对于任意的投票v和v'，如果$v_{bal}<v_{bal}'$，那么v<v'。但是没有定义$v_{bal}=v_{bal}'$的情况下v和v'的顺序关系。

对于任意的表决集合β，定义集合Votes(β)为所有满足如下条件的投票v的集合：对于任意的B∈β，v满足$v_{pst} \in B_{vot}$，$v_{bal} = B_{bal}$，并且$v_{dec} = B_{dec}$。如果p是一个牧师，b是一个表决编号或者±∞，那么Max(b,p,β)定义为β中由p投出的表决编号小于b的最大的投票v或者$null_p$。 因为$null_p$比所有p实际投出的票都要小，这就意味着MaxVote(b,p,β)是下面集合中的最大投票：

$$\{v \in Votes(\mathcal{B}) : (v_{pst} = p) \wedge (v_{bal} < b)\} \cup \{null_p\}$$

对于任意非空的牧师集合Q，MaxVote(b,Q,β) 定义为Q中编号小于b的最大的投票。 于是，条件B1(β) - B3(β) 正式表述如下：

$$B1(\mathcal{B}) \triangleq \forall B, B' \in \mathcal{B} : (B \neq B') \Rightarrow (B_{bal} \neq B'_{bal})$$
$$B2(\mathcal{B}) \triangleq \forall B, B' \in \mathcal{B} : B_{qrm} \cap B'_{qrm} \neq \emptyset$$
$$B3(\mathcal{B}) \triangleq \forall B \in \mathcal{B} : (MaxVote(B_{bal}, B_{qrm}, \mathcal{B})_{bal} \neq -\infty) \Rightarrow (B_{dec} = MaxVote(B_{bal}, B_{qrm}, \mathcal{B})_{dec})$$

定理1： 如果β 中的表决 B 是成功的，那么β 中更大编号的表决和 B 的法令相同

$$\textbf{Lemma}\ \ If\ B1(\mathcal{B}),\ B2(\mathcal{B}),\ and\ B3(\mathcal{B})\ hold,\ then$$
$$((B_{qrm} \subseteq B_{vot}) \wedge (B'_{bal} > B_{bal})) \Rightarrow (B'_{dec} = B_{dec})$$
$$for\ any\ B,\ B'\ in\ \mathcal{B}.$$

定理1的证明

对于β中的任意表决B，有：

$$\Psi(B, \mathcal{B}) \triangleq \{B' \in \mathcal{B} : (B'_{bal} > B_{bal}) \wedge (B'_{dec} \neq B_{dec})\}$$

表示β中编号比B大并且法令不同的表决的集合。为了证明引理，只需要证明如果$B_{qrm} \subseteq B_{vot}$那么Ψ(B, β) 是一个空集合。

Paxos通过矛盾法证明引理。他们假设存在$B_{qrm} \subseteq B_{vot}$，并且Ψ(B, β) 不是一个空集合，那么显而易见的会得到如下的矛盾：

1. C是Ψ(B, β) 中编号最小的那轮表决
2. 由C的定义可知，C的编号一定大于B的编号，（因为Ψ(B, β)表示的是编号比B大的法令不同的表决集合）
3. 根据B2(β)条件，B表决的投票牧师和C的表决的多数派一定会有交集

4. MaxVote(C_{bal},C_{qrm},β)表示C的多数派中编号小于C_{bal}的最大的投票，因为C_{qrm}和B的投票者一定有交集，所以MaxVote(C_{bal},C_{qrm},β)的编号一定大于等于B的编号
5. 由4可知MaxVote(C_{bal},C_{qrm},β)一定产生了一次投票，所以MaxVote(C_{bal},C_{qrm},β)∈Votes(β)
6. 根据B3(β)条件和上面的5可以得出MaxVote(C_{bal},C_{qrm},β)$_{dec}$=C_{dec}，即MaxVote(C_{bal},C_{qrm},β)的法令和C的是相同的
7. 因为MaxVote(C_{bal},C_{qrm},β)的法令和C的是相同的，而C的法令和B不同，那么MaxVote(C_{bal},C_{qrm},β)的法令和B的法令不同
8. 已知4，即MaxVote(C_{bal},C_{qrm},β)的编号一定大于等于B的编号，并且他们的法令是不同的（7的证明），假设他们的编号相等，那么法令一定是相同的（B1(β)的条件），所以为了满足7，4中的大于等于可以推导为一定是大于的，即MaxVote(C_{bal},C_{qrm},β)$_{bal}$>B_{bal}
9. 由7、8和Ψ(B, β)的定义可以得出MaxVote(C_{bal},C_{qrm},β)∈Votes(Ψ(B, β))
10. 由定义可知MaxVote(C_{bal},C_{qrm},β)一定是小于C_{bal}的
11. 这里产生了矛盾：
12.
    1. 9中得出了MaxVote(C_{bal},C_{qrm},β)∈Votes(Ψ(B, β))，那么Votes(Ψ(B, β))不能为空
    2. 10中得出了MaxVote(C_{bal},C_{qrm},β)一定是小于小于C_{bal}的
    3. 而C_{bal}又是Ψ(B, β)中的最小值，这是矛盾的

根据上面的引理，显而易见的，如果B1(β)-B3(β)成立，那么任何两个成功的表决，法令一定是相同的。

$$\textsc{Theorem}\ 1.\ \textit{If}\ B1(\mathcal{B}),\ B2(\mathcal{B}),\ \textit{and}\ B3(\mathcal{B})\ \textit{hold, then}$$

$$((B_{qrm} \subseteq B_{vot}) \wedge (B'_{qrm} \subseteq B'_{vot})) \Rightarrow (B'_{dec} = B_{dec})$$

$$\textit{for any}\ B,\ B'\ \textit{in}\ \mathcal{B}.$$

更直观的表述是，如果满足B1(β)-B3(β)，那么任意两轮成功的表决，都是对相同的法令的表决。

定理2：令b为一个表决标号，Q为牧师的集合，对于β中的任意表决B，如果b和Q满足b>B_{bal}并且Q∩B_{qrm}不为空，如果条件B1(β)、B2(β)、B3(β)都成立，那么存在一轮表决B'，B'$_{bal}$=b同时B'$_{qrm}$=B'$_{vot}$=Q使得B1(β∪{B'})、B2(β∪{B'})、B3(β∪{B'})成立。

定理2的证明

首先翻译一下定理2：一定存在编号更大的表决满足三个条件使协议运行下去。对于投票集合β，如果B1(β)、B2(β)、B3(β)都成立，那么在β之后一定能找到一次成功的投票B'使B1(β∪{B'})、B2(β∪{B'})、B3(β∪{B'})成立。

证明：

1. 因为B'$_{bal}$=b，b>B_{bal}所以B'$_{bal}$>B_{bal}成立，B1(β∪{B'})成立（条件1：每轮表决由唯一的编号）
2. Q∩B_{qrm}不为空，B_{qrm}=Q，所以B2(β∪{B'})成立（条件2：成功的两轮表决中多数派由交集）
3. 如果MaxVote(b,Q,β)=-∞，那么B'$_{dec}$可以是任意法令，否则令B'$_{dec}$为MaxVote(b,Q,β)，B3(β∪{B'})成立。（条件3：对于 β 中每一轮表决B，如果B的法定人数集中的任何一个牧师在β中一个更小轮的表决中投过(赞成)票，那么B的法令与所有这些更小轮表决中最大那次表决法令相同。）

（如果β只包含一轮投票，显然三个条件都能满足；如果β不知包含一轮投票，那么根据上面的理论，协议是可以进行下去的）

2.2 The Preliminary Protocol（初步协议）

Paxons根据条件B1(β)-B3(β)为true的要求得出了初步协议，其中β是已经完成或者正在进行的表决的集合。协议的定义指明了β集合如何变更，但是从未说明如何计算出β。Paxons认为β只为神灵所理解，永远不被凡人所知。

每次表决由一个牧师发起，他选择表决的编号、法令以及参与的法定人数。每一个法定人数中的牧师之后决定是否为表决投票。确定发起人如何选择编号、法令、和法定人数，以及牧师如何决定是否给表决投票的规则来源于保持B1(β)-B3(β)的需求。

为了保持条件B1能满足，每个表决都需要一个唯一的编号。通过记录之前的编号，牧师能非常容易的避免给两次表决使用相同的编号。为了避免不同的牧师采用相同的表决编号，表决编号需要被分区。显而易见的一个办法是将编号设置成一个二元组，由一个数字和一个牧师组成，按照字典序进行排序，如：(13, A) < (13, B) < (15, A)。

为了保持条件B2能满足，只需要保证每次表决的投票牧师能形成多数派（n/2 +1）。

条件B3要求，如果MaxVote(b,Q,β)$_{dec}$不是BLANK，那么表决编号为b并且法定人数为Q的那次表决的法令必须为MaxVote(b,Q,β)$_{dec}$。如果MaxVote(b,Q,β)$_{dec}$是BLANK，则表决可以采用任何的法令。为了保持条件B3，在初始化表决（编号为b，法定人数为Q）前，牧师p必须知道MaxVote(b,Q,β)$_{dec}$。为了完成这个操作，p需要知道Q中的每个牧师q的MaxVote(b,q,β)$_{dec}$。

回想一下，MaxVote(b, q, β)表示的是牧师q投出的编号小于b的最大的选票，或者null，如果q没有给任何编号小于b的表决投过票。牧师p通过消息通信从牧师q那获取MaxVote(b, q, β)。因此，协议中发起单次表决的前两步是：

（1）牧师p选择一个表决编号b，并且发送NextBallot(b)请求给其他的牧师。

（2）牧师q将LastVote(b,v)响应给NextBallot(b)请求，v是q投出过的比b小的最大的表决的编号，或者是nullq，如果q没有给比b编号小的表决投过票

牧师q必须在他备忘录中记录他之前投出过的选票。

当q发出LastVote(b,v)的时候，v和MaxVote(b,q,β)是相等的，β会在新表决被初始化和投票时发生改变。一旦q采用v作为MaxVote(b,q,β)的值来选择法令，为了保持条件B3为真，在q发出LastVote(b,v)之后MaxVote(b,q,β)需要保持不变。为了避免MaxVote(b,q,β)发生改变，q不应该再给表决编号再v_{bal}和b之间的表决投票。

协议的下两步是：

（3）在收到Q（Q是一个多数派集合）中的所有牧师的LastVote(b,v)之后，牧师p初始化一个编号为b，多数派为Q，法令为d的表决（d的选择满足条件B3），之后他在他律簿的背面记录这个表决并发送BeginBallot(b,d)消息给Q中的每个牧师。

（4）在收到BeginBallot(b,d)消息时，q决定是否在编号为b的表决中投票给p。如果q决定投票，那么他发送Voted(b,q)给p并在律簿背面记录这次投票。

步骤（3）可以理解为添加B到β集合，B满足B_{bal}=b，B_{qrm}=Q，Bvot=∅，B_{dec}=d。在步骤（4）中，如果q决定投票，那么他的执行过程可以认为是将q添加到B_{vot}中。

即使投票也不会违反任何先前的承诺，牧师也可以在步骤（4）中选择不投票。实际上协议中的所有步骤都是可选的（不是必须要执行），举个例子，牧师q可以在步骤（2）中无视NextBallot(b)请求。失败的情况可以阻止表决进行下去，但是并会引起任何不一致的问题。协议保证牧师离开或者信使丢失了消息的情况下，仍然不会出现不一致的情况。

收到重复的消息可以导致行为重复的执行。除了步骤（3）之外，其他步骤重复执行并不会有什么影响，比如在步骤（4）中，重复发送Voted(b,d)和发送一次的效果是一样的。步骤（3）的重复可以通过律簿背面记录的信息来解决。那么一致性条件会得到满足，即使信使重复的传递消息。

步骤（1）-（4）描述了初始化表决和为其投票的完整协议。剩下的就是决定投票结果和颁布法令。回忆一下，当一轮表决都到多数派的投票时，它是成功的。成功的表决的法令即是圣会选择的法令。协议剩余的内容是：

（5）如果p收到Q中所有的牧师的投票Vote(b,d)，那么他将法令d记录到律簿中并发送Success(d)消息给所有的牧师。

（6）当收到Success(d)消息时，牧师将其记录到自己的律簿中。

步骤（1）-（6）描述一轮独立的表决如何执行。初步协议允许任何牧师在任何时候发起一轮表决。每个步骤满足B1B2B3的条件，那么整个协议也满足这三个条件。一个牧师只会将成功的表决的法令记录到律簿中，这保证了律簿的一致性，但是并没有解决进展性的问题。

在步骤（3）中，法令d是要满足条件B3的，那么这个法令可能已经被记录到一些牧师的律簿中。那个牧师可能不在多数派中，他可能已经离开会议室。那么可能发生不一致的情况，如果在步骤（3）选择法令时给予更多的自由。

2.3 The Basic Protocol（基础协议）

在初始协议中，一个牧师必须记录（1）他所发起的所有表决的编号，（2）他所投出的所有选票，（3）每一个他发出的LastVote。对于繁忙的牧师们来说，记录所有这些信息是非常困难的。因此，Paxons对初步协议进行了限制，以获得更加实际的基础协议。在基础协议中，牧师只需要在他的律簿中记录：

lastTried[p] 牧师p尝试启动的最后一轮表决的编号，或者-∞，如果他没有发起过表决

prevVote[p] 牧师p所有投票的表决中，编号最大的投票，如果p没有进行过任何投票，那么为$null_p$。

nextBal[p] 由p发出的所有LastVote(b,v)中，最大的表决编号

初始协议的步骤（1）-（6）描述了如何通过牧师p进行一轮独立的表决。初始协议允许牧师p并发的发起任何数量的表决。而在基础协议中，他一次只能发起一个表决，编号为lastTried[p]。在发起这个表决后，他忽略所有和他之前发起的表决相关的信息。牧师p在纸片行记录lastTried[p]相关的所有信息，如果小纸片丢失，那么这轮表决就终止。

在初始协议中，牧师q发出LastVote(b,v)意味着不再为任何编号在v_{bal}和b之间的表决进行投票。在基础协议中，他有了更强的承诺，不再给任何编号小于b的表决投票。更强的承诺可能让他在基础协议的步骤（4）中不进行投票，而相同的情况下他可能在初始协议中会进行投票。但是初始协议本身就允许q不进行投票，基础协议并没有要求他做任何违反初始协议的事情。

初始协议的步骤（1）-（6）在基础协议中演化成了如下六个步骤。

（1）牧师p选择一个大于lastTried[p]的表决编号b，之后将lastTried[p]设置为b，并发送NextBallot(b)请求给所有剩余的牧师。

（2）在收到p的NextBallot(b)请求时（b > nextBal[q]），牧师q将nextBal[q]设置为b，并发送LastVote(b,v)给p，v等于prevVote[q]。（如果b <= nextBal[q]，请求会被忽略）

（3）p在收到多数派Q的LastVote(b,v)，并且b=lastTried[p]，p启动一轮编号为b，多数派为Q，法令为d（d的选择满足条件B3）的表决。之后发送BeginBallot(b,d)消息给Q中所有的牧师。

（4）收到BeginBallot(b,d)，b=nextBal[q]的消息时，牧师q在编号b的表决给出他的投票，将prevVote[q]设置为他的投票，并发送Voted(b,q)给牧师p。（如果b≠nextBal[q]，BeginBallot(b,d)会被牧师q忽略）

（5）如果p从Q中的每个牧师q那里收到Voted(b,q)，b=lastTried[p]的消息，那么他将d记录到他的律簿中并发送Success(d)的消息给所有的牧师。

（6）收到Success(d)的消息后，牧师q将法令d记录到自己的律簿中。

基础协议是初始协议的限制版本，这意味着任何基础协议中被允许的行为，在初始协议中也是被允许的。因为初始协议满足一致性要求，那么基础协议一定满足一致性要求。和初始协议一样，基础协议也不要求任何操作必须要被执行（比如一定要进行投票，牧师可以在满足要求的情况下放弃投票），那么他也没有解决协议的可进展性问题。

基础协议满足条件B1B2B3显而易见的能推导出满足一致性要求，但是一些更“明显”的古代智慧被证明是错误的。持怀疑态度的公明要求更加严格的证明。Paxons的数学家更严谨的对协议的证明在附录中。

2.4 The Complete Synod Protocol（完整的圣会协议）

基础协议保持一致性，但是它并没有保证任何的可进展性，因为它只是允许牧师做一些操作，但是并没有要求牧师一定要做什么操作。完整协议和基础协议一样包含六个步骤。为了达到可进展的目的，完整协议包含更加显而易见的要求：牧师需要尽可能快的执行（2）-（6）的步骤。为了保持进展性的条件，必须有牧师牧师来执行步骤（1），即发起表决。完整协议的关键在与确定一个牧师上面时候发起一轮表决。

永远不发起表决显然会阻碍进展性。但是发起太多表决同样会阻碍进展性。如果b比任何其他的表决的编号都大，那么q在步骤（2）中接收NextBallot(b)消息将会使他在步骤（4）拒绝给之前的表决投票。因此，新启动的表决会阻碍之前的表决完成。如果新的表决在之前的表决成功之前被持续的发起，那么协议将不具备可进展性。

为了保证进展性，新的表决需要在其他表决成功后发起，但是也不能发起的太频繁。为了开发完整的协议，Paxons公民首先需要知道信使需要多少时间传递信息，牧师需要多长时间做出响应。他们确定信使在没有离开会议室的情况下需要4分钟的时间来传递消息，牧师在没有离开会议室的情况下都能在7分钟内完成需要的操作。因此，如果p和q始终没有离开会议室，p发起了一个事件发送消息给q，并且q做出一个响应给p，那么在没有信使离开会议室的情况下，p将在22分钟内收到q的响应（p发起事件需要7分钟，消息传递给q需要4分钟，q处理消息需要7分钟，消息再返回给p需要4分钟，总共22分钟）。

假设只有一个牧师p发起表决，并且他在步骤（1）中将消息发送给所有的其他牧师。如果p发起表决时，多数派牧师都在会议室中，那么他可以期望在22分钟内执行步骤（3），并且在下一个22分钟内执行（5）。如果他没能在这个时间内执行这些操作，那么一定是有牧师或者信使在他发起表决后离开了会议室，或者一个更大编号的表决被其他牧师提出（在p成为唯一的发起表决的牧师之前）。为了解决后面这个问题，p必须学习其他牧师使用的任何编号比lastTried[p]大的表决。可以通过拓展协议，要求：如果牧师q收到牧师p的b < nextBal[q]的NextBallot(b)或者BeginBallot(b)消息，他响应给p一个包含nextBal[q]的消息。然后牧师p可以发起一个编号更大的表决。

依旧假设p是唯一一个可以发起表决的牧师，假设他想发起一轮表决，他需要（a）在之前的22分钟之内没有执行步骤或者（b）他知晓别的牧师发起了一个编号更大的表决。如果会议室的门被锁上（没有人会离开会议室），牧师p和大多数牧师都在会议室内，那么在99分钟内，一条法令会被通过并被记录在会议室内所有牧师的律簿上（22分钟用于p发起一次表决，22分钟用于发现一个更大编号的表决被其他牧师提出过，55分钟用于完成一轮成功的表决）。因此，进展性会得到满足，如果只有一个不会离开会议室的牧师发起表决。

完成的协议包含了一个选举唯一的发起表决的牧师的过程，这个牧师称为president（总统）。在多数形式的政府中，选举一个总统是困难的，因为大多数政府都要求在任何时刻只能有唯一的一个总统。比如在美国，如果一些人认为Bush已经被选为总统，而另一些人认为Dukakis才是总统就会造成混乱，因为其中某个总统可能会签署某个法令而另一个总统不赞成该法令。在Paxon圣会，拥有多个总统只是会阻碍可进展性，并不会引起不一致的情况。为了满足可进展性，选举唯一总统的方法只需要满足presidential selection requirement：

“如果没有人进出会议室，那么在T分钟之后，会议室中的牧师可以认为自己是总统。”

如果总统的选举条件被满足，那么完整的协议将保证，如果多数派牧师在会议室中且T+99分钟内没有人进入或离开会议室，在这样一个周期结束时，会议室中的每个牧师的律簿上都会增加一条法令。

Paxon公明按照牧师名字的字典序来选择成为总统的牧师。presidential selection requirement会被满足，如果牧师没T-11分钟发送一次包含他名字的消息给其他的牧师，如果他T分钟内没有收到更高级别的名字的消息，那么他认为自己是总统。

通过要求基础协议中的牧师快速的执行步骤（2）-（6），并且增加一个选举总统用于发起表决的方法，要求总统在合适的时机发起表决来得到完整的协议。更多的协议细节还不清楚，以上只是简单的描述了选举总统的办法和总统发起表决的时机。但显然这不是Paxons公民所使用的办法。我给出的规则要求在一个法令被选出后总统需要继续发起表决，以让刚进入会议室的牧师能知道被选中的法令。显然还有更好的办法让刚进入会议室的牧师学习被选中的法令。另外，在挑选总统时，每个牧师可以发送lastTried[p]给其他牧师，以便于总统在第一次发起表决时能选择一个足够大的表决编号。

Paxons的公民意识到，任何涉及到进展性的条件都需要测量时间的推移。在拥有精准计时器的前提下，上面描述的总统选举和表决的发起都可以通过超时的方式被描述为准确的算法。更详细的分析表明，这种算法可以在时间有精准性边界的情况下工作。Paxons上熟练的工匠可以造出满足精度要求的沙漏。

3 TheMulti-Decree Parliament（多法令议会）

当议会成立时，一个满足一致性和进展性要求的协议从圣会协议中演进而来。原始议会协议的演进和性质在3.1和3.2节中说明。3.3节讨论了协议的进一步发展。

3.1 The Protocol (协议)

代替只通过一条法令，Paxon议会一次通过一系列的法令。在圣会协议（Synod protocol）中，会选出一个总统。任何人想要通过一条法令，都需要通知总统，由它分配法令编号并发起表决。逻辑上讲，议会协议针对多个不同编号的法令使用多个独立的圣会协议实例。但是，会为所有的实例选择一个全局的总统，由他负责执行协议中的前两步（只执行一次）。

得出议会协议的关键是观察到在圣会协议中，总统直到步骤（3）才直到法令或者法令牧师集合。一个新选出的总统p可以给一些议员发送一条消息：所有圣会协议的实例都采用NextBallot(b)提供服务。一个议员q可以响应LastVote作为所有圣会协议的步骤（2）的结果。这个响应只包含有限的信息，q只能给有限的圣会协议实例投票。

当新总统收到多数派中所有议员的响应，他可以为每个圣会协议实例执行步骤（3）。对这些实例来说，他们的法令由条件B3来决定。总统快速的为每个实例执行步骤（3）来通过这些法令。无论何时他收到通过法令的请求，他都选择可以选择的最小的编号的法令并为它执行步骤（3）来尝试通过这条法令。

对这个协议进行简单的修改得到真正的Paxon议会协议：

没有理由为已经知道结果的法令在执行一次圣会协议。因此，如果一个新总统p被选出，在他的律簿上已经记录了所有编号小于等于n的法令，那么他给所有法令编号大于n的实例发送NextBallot(b,n)消息作为原先的NextBallot(b)消息。在这个消息的响应中，议员q将自己律簿中编号大于n的法令通知给p，并要求p将所有编号小于等于n的不再他自己律簿中的法令发送给他。

假设法令125和126在周五下午晚些时候引入，法令126被写入了一个或两个律簿，之后议员们都回家过周末了。假设下周一，A被选为新的总统并学习了126法令，但是他并不知道125法令，因为之前的总统和议员们都还没有回到会议室。他将举行一个表决使126通过，这使得在律簿上留下一个空洞。给新的法令分配125的编号会使得它出现在126法令之前，而126法令是上周就得到通过的。乱序的通过法令会产生混乱，比如，125法令是在议员知道126法令的内容对的情况下提出的。做为替代方案，总统一般尝试通过（125：二月是国家橄榄日）一个对每个Paxon公民都没有差别的法令（通过一个无关紧要的法令）。一般来说新总统都通过这样的方式来填充他律簿上的空洞。

一致性和可进展性直接从圣会协议继承而来。据我所知，Paxon公民可以写出一个更精准的协议描述，因为它只是非常简单的从圣会协议演进过来。

3.2 Properties of the Protocol（协议的属性）

3.2.1 *The Ordering of Decrees* 不同编号的表决可以并发的发生，由不同的议员发起——每个议员发起表决的时候都认为自己是总统。我们无法确定法令的顺序，在他们通过之前，特别在不知道总统如何选举出来的时候。但是，由一个顺序相关的重要属性是可以推导出来的。

当总统在圣会协议的步骤（3）中选择了一个法令，这个法令被称为是已提议的（proposed）。当一个法令第一次被写入到一个律簿中后，这个法令被认为是通过的（passed）。一个总统在提出任何法令之前，他需要从多数派的议员那里学习他们已经投过票的法令。任何被通过的法令都至少有一个属于多数派的议员为其投过票。因此，总统必然能在提出新的法令之前学习到所有之前通过的法令。总统不会使用非"二月是国家橄榄日"的法令来填补空洞，并且不会没有顺序的提出法令。因此协议满足以下的顺序属性：

如果法令A和B是重要的法令（非"二月是国家橄榄日"这样的法令），并且A在B被提议之前已经被通过，那么A比B拥有一个更小的编号。

3.3.2 *Behind Closed Doors* 尽管我们不知道选出一个总统的细节，但我们确切的知道当总统选出之后并且没有人进出会议室的情况下协议的进行。在收到一个通过某个法令的请求后——通过公民或者其他议员转发而来——总统为法令设定一个编号，然后通过下面的消息交换来通过它（序号和圣会协议的序号对应）：

（3）总统给多数派中的每一个议员发送BeginBallot消息

（4）议员发送Voted消息给总统

（5）总统发送Success消息给每个议员

假设议会有N个议员，多数派为N/2，那么这里有3个消息延迟和3N个消息。另外，如果议会繁忙，总统可以将BeginBallot和上一个法令的Success消息合并，这样每个法令就只有2N个消息了。

3.3 Further Developments

事实证明管理这个岛比Paxons公民想象的要复杂的多。出现了许多需要修改协议去解决的问题。重要的修改如下。

3.3.1 *Picking a President*

议会总统的选举依旧采用了圣会协议中按照议员名字排序的方式。因此名字排序靠前的议员可能从一段6个月的休假中一回到会议室即成为了总统——他可能对之前发生的事情一无所知。碰巧他又是一个写字很慢的人，这样在他费力的补齐之前6个月的法令之前，议会所有活动都会停下来。

这个故事导致了总统选举的最佳方法的争论。一些人认为一旦一个医院成为总统，那么他需要一直是总统，直到他离开会议室。一些有影响力的人认为应该选择最富有的议员来当总统，因为他有能力雇佣更多的帮手来履行总统的职责。他们主张一旦一个富有的议员的律簿更新到了最新的内容，他没有理由不承担总统的职责。其他一些人主张让正直的人担任总统，而不是最富有的议员。正直意味着不会不诚实，尽管Paxon公开承认没有官方渎职的可能。不幸的是，这些争论的最终产出还不知道；没有最终关于总统选举方法的记录。

3.3.2 *Long Ledgers*

随着时间一年年的过去，议会通过了越来越多的法令，为了找到当前的橄榄油税率或者什么颜色的山羊可以出售，Paxons需要翻阅长长的律簿。长途旅行回来后的议员需要复制一系列的数据来使他的律簿更新到最新的法令。最终，议员们不得不将他们的律簿转换为法律书籍，这些书籍只包含当前法律状态和最后一条通过的，反映当前状态的法令的编号。

需要知道橄榄油的税率，只需要翻阅书籍中税率相关的内容；需要知道可以售卖的山羊的颜色，只需要查看商品相关的法律。如果一个议员的律簿包含了知道1298的法令，然后他学习到1299法令将橄榄油的税率调整为6银币，他只修改书籍中橄榄油相关的条目，并记录他的律簿已经包含到1299的法令。如果他之后学习到1302法令，他将其记录在律簿背面，直到他学习到1300和1301法令，然后将他们合并记录到律簿中。

为了让短暂离开的议员不用拷贝整本书的内容而保持最新的状态，议员们在书的背面记录过去几周的法令。他们可以在纸片中记录这些内容，但是在律簿的背面记录会更加方便一些，因为他们两三个星期才通过一条法令并更新律簿。

3.3.3 *Bureaucrats*

随着Paxon的兴盛，议员们变得非常繁忙。议会无法再处理所有政府的细节，因此官僚制度被建立。议会不再通过法令决定每块奶酪是否符合销售要求，而是通过议会通过一条法令指定给一个奶酪质检员来做这个事情。

显而易见，选举官员并不像起初看起来的那么简单。议会通过了小王成为第一个奶酪质检员。几个月后商家抱怨小王太严格了，拒绝了完美的奶酪。议会接着通过了：

1375：小李成为新的奶酪质检员

但是小王并没有关注议会的工作，他并不知道这条法令。那么有一段时间，在奶酪超时，小王和小李都对奶酪进行检查，并可能给出冲突的结论。

为了避免这种混乱的情况，Paxon需要保证任何时刻最多只能有一个官员。为了达到这个目的，总统在提议法令时，为每个法令加入了日期和时间。小王成为奶酪检查员的法令看起来是这样的：

2716：8:30 15 Jan 72-未来三个月，由小王作为奶酪检查员

这就定义了小王的任职从72年1月15日早上8点30开始，前一个检查员的任期到这个时间结束。小王的任期将到72年3月15日早上8点30结束，除非总统通过一条法令：

2834：9:15 3 Mar 72-小王辞去奶酪检查员的职务

一个官员采用较短的任期，这样可以被快速的替换，例如他离开了小岛。一会可以通过一个法令来延长官员的任期，如果他非常的称职。

一个官员需要知道当前的时间，以确定他是否还在某个职位上。Paxon上还没有机械钟表，但是他们可以通过太阳和星星的位置确定误差在15分钟内的时间。如果小王的任期从8:30开始，那么在8:45之后他将不再负责检查奶酪。

很容易将这种方法在官员任命的法令中使用，只需要使更大的编号的法令推迟15分钟。但如果有以下两条法令：

2854：9:45 9 Apr 78-小王作为白酒测试员两个月

2855：9:20 9 Apr 78-小李作为白酒测试员1个月

如果这两条法令在9:30和9:35被不同的，认为自己是总统的议员提出，会发生什么？这种提议时间顺序颠倒的情况可以很容易避免，因为议会协议满足如下特性：

如果两个法令由不同的总统提出，那么其中一个总统必须先学习另一个法令后才会提出他的法令

我们来看看这个特性是否满足要求。假设表决 b 通过了法令 D，表决 b1 通过了法令 D1，b<b1，假设议员 q 在两个表决中都投了票。法令 D1 的表决以一个 NextBallot(b1,n)消息开始。如果这个消息的发送者还不知道法令 D，那么 n 比法令 D 的编号要小，从而 q 对 NextBallot 消息的答复必须说明他已经对 D 投了票。

3.3.4 *Learning the Law*

除了请求通过法令之外，Paxon上的公民还需要获知岛上的法律。Paxon上起初认为可以让公民简单的查阅任何议员的律簿，但是下面的故事显示出需要更严格的方法来查询法律。几个世纪以来，只有出售白色的山羊是合法的。农民老杨让议会通过了一条法令：

77：售卖黑色山羊是被允许的

之后老杨将他的一些黑色山羊卖给了商人老吕。作为守法的公民，老吕向议员老秦寻味这样是否是合法的。但是老秦这段时间并没有在议会，他律簿上没有76号之后的法令。他告知老吕这比买卖在当前法律之下是非法的，然后老吕决绝了老杨的这比交易。

这个故事导致法律查询上的如下单调一致性的条件建立：

如果一个查询先于第二个查询，那么第二个查询的法律状态不能比第一个查询的法律状态更旧

如果一个公民得知一个特定的法令通过了，那么得知这一信息的过程被认为是遵循这个条件的隐含查询。我们将会看到，对单调性条件的解释在若干年后发生了变化。

刚开始，单调性查询由为每一个查询通过一条法令来达成。如果老吕想知道当前的橄榄油税率，他将让议会通过这样一条法令：

87：公民老吕在阅读法律

然后他阅读任何一个包含到86法令的律簿，来得知当时的橄榄油税率。如果之后老蒋想查询橄榄油税率，这个查询法令在87法令之后被提议，那么他会接收一个比87更大的法令编号（3.2.1节中说明过），因此老蒋不会比老吕使用更早的橄榄油税率法令。

如果"先于"的含义是必须保证A查询在B查询开始之前已经完成，那么这种阅读法律的方法满足单调性条件。

为每一次查询法律通过一个法令被证明太笨重了。Paxon的人们意识到，如果改变"先于"的含义而弱化单调性条件，那么可以通过更简单的方法来查询法律。他们定义一个事务先于另一个事务，不仅是一个事务在发生时间上更早，而且是影响第二个事务的条件。弱化的单调性条件可以组织农民老杨和商人老吕遇到的问题，因为老杨的查询结束和老吕的查询开始之间有一个因果链的关系。

弱化的单调条件可以通过在所有的事务和查询中使用法令编号来满足。例如农民老杨有一些非白色的山羊，他到议会去通过法令：

277：允许售卖棕色的山羊

当他把山羊卖给商人老吕的时候，他告知老吕交易在法令277下是合法的。之后老吕向议员老秦咨询在277法令下这比交易是否是合法的。老秦因为律簿上还没有277法令，那么他等待学习277法令或者告知老吕让他去问其他议员。如果议员老秦的律簿上法令已经到298号，那么他将告诉老吕直到298号法令，这比交易都是合法的。老吕会记录这个编号，以便在后续的交易和查询中使用。

这个方式满足了单调性条件，但是普通的公民并不喜欢去记法令编号。Paxons再一次通过解释单调性条件来解决这个问题——这一次通过改变"法令的状态"的含义。他们将法律区分为不同的领域，并未每个领域指定议员作为专家。每个领域的法律的状态有专家的律簿决定。 例如，假设1517号法令改变了关税法，第1518号法令改变了赋税法。 如果赋税法专家在关税法专家了解到这两个法令之前就已经了解了这两个法令，那么税法将首先发生变化，从而产生一种法律状态，这种法律是通过以数字顺序颁布法令而无法获得的。

为了避免对当前状态的定义的冲突，Paxons要求每一个领域都只能有一个专家，可以通过使用和3.3.3节选择官员一样的方法来选择专家。如果每次查询只涉及到一个领域，单调性条件可以通过将查询重定向到该领域的专家来保证，由该领域的专家根据自己的律簿来给出响应。由于了解法律已经过去构成了隐含的查询结果，因此Paxons要求法令最多改变一个法律领域，并且法令通过的通知只能来自该领域的专家。

查询多个领域的法律并不是难以解决的。当商人老吕查询进口羊毛的关税是否比本地羊毛的交易税高时，赋税专家和关税专家需要协同来给出回答。例如，赋税专家可以先询问关税专家关于羊毛的关税，直到他收到回复前不作出响应。

这种方法被证明是令人满意的，直到有必要一次修改多个领域的法律。然后Paxons意识到，维护单调性的必要条件不是一条法令只影响一个领域，而是他影响的所有领域拥有相同的专家。议会可以通过指定一个议员为所有领域的法律的专家，然后通过一条法令来改变多个领域的法律。更进一步，同一个领域可以有多个专家，只要这个领域的法律不允许修改。

3.3.5 *Dishonest Legislators and Honest Mistakes*

和官方断言相反，Paxon历史上有一些不诚实的议员。被抓时他们可能被流放。通过发送矛盾的消息，一个恶意的议员可以使其他的议员的律簿发生不一致的情况。诚实的议员和信使也可能因为记忆的问题导致律簿的不一致。

当不一致被发现时，可以很容易的通过新的法令来修复。例如橄榄油税赋不一致，可以通过一个新的法令定义橄榄油的税赋为一个确定的值来消除。修复不一致的困难点在于可能没人发现律簿不一致的情况。

立法者存在不诚实或错误，可以从议会成立几年后开始出现在律簿中的多余法令推断出来。 例如，法令：

2605：橄榄油的赋税为9个银币

即使已经通过2155法令通过了，并且之后并没有法令去修改它。议会看起来没六个月循环的过一遍他们的法律，这样即使开始时有一些律簿不一致，议员们也会在6个月内对岛上的法律达成一致。Paxos 人相信通过这些冗余 法令的使用，可以使他们的议会自稳定（self-stabilizing）。（self-stabilizing 是 Dijkstra 于 1974 年提出的一个现代术语）

在议会允许随意进出的情况下，尚不清楚自稳定的确切含义。Paxons不会满足于一个要求所有议员在某个时间都坐到会议室来保证一致性的定义。无论如何，一致性要求当一个议员的律簿中有确定编号的某个法令，而另一个议员没有时，后者最终会填入这个条目。

很不幸的，我们没有确切地知道Paxos议会运转的自稳定性是哪一种，或者它是怎样达到的。 Paxos 的数学家毫无疑问解决了这个问题，但是他们的工作还没有被发现。我希望 Paxos 的 进一步考古发掘会在寻找自稳定性的手稿方面给予更高的优先级。

3.3.6 *Choosing New Legislators*

早期，议会成员是世袭的，由上一代传给下一代。当最早的政治家老吴退休后，他把律簿交给了他的儿子。其他和老吴共事的议员也没有什么不同。

随着旧家族的移民和新家族的加入，这种情况发生了改变。Paxons决定通过议会的法令来添加和移除议员。这暴露了一个循环的问题：议会的成员由法令的通过决定，而法令的通过又需要确定一个多数的议员集合由哪些成员组成，而这又反过来依赖于谁是议会的成员。让通过法令 n 的议员由法令 n-3 时法律确定的成员来组成可以打破这个循环 。总统 在知道直到 3252 的所有法令之前，不会去通过法令 3255。事实上，通过法令：

3252：小陈现在是一个议员

之后，总统可以立即通过 3253 和 3254 这样的“橄榄节”法令。

以这种方式改变议会的组成是危险的，必须小心谨慎，虽然一致性和进展性可以保持。但是进展性需要在多数派议员在会议室内才能得到保证，而它并不保证多数派的议员都在会议室中。事实上，这种选择议员的机制可能使Paxon议会机制崩溃。由于抄写员的错误，一项应该尊重沉没在沉船中的水手的法令宣布他们是议会的唯一成员。它的通过阻止了任何新的法令通过——包括提出的纠正错误的法令。Paxos 政府就这样停止运转了。一个叫 Λαμπσων 的将军趁着这个混乱时期发动了兵变，建立了一个军事独裁政权，结束了发展了几个世纪的议会政权。Paxos在几任腐败堕落的独裁者统治下变得衰弱。终于不能抵挡一次来自东方的入侵，他们的文明也在这次异族入侵下毁灭了。