[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 11/00 (2006.01)

[12] 发明专利说明书

专利号 ZL 200510092247.1

[45] 授权公告日 2009 年 3 月 25 日

[11] 授权公告号 CN 100472461C

[22] 申请日 2005.6.17
[21] 申请号 200510092247.1
[30] 优先权
[32] 2004.6.18 [33] US [31] 10/871,134
[73] 专利权人 微软公司
地址 美国华盛顿州
[72] 发明人 J·R·洛齐 J·R·豪威尔
J·R·道苏尔
[56] 参考文献
US6401120B1 2002.6.4
Paxos Made SimpleURL:http://research.microsoft.com/users/lamport/pubs/paxos-simple.pdf. Leslie, Lamport.. 2001
分布式系统的不确定性及其对 Uni —CRM 测试的影响. 张川, 王柏, 艾波. 北京邮电大学学报, 第 26 增刊卷. 2003
审查员 孙泽竑
[74] 专利代理机构 上海专利商标事务所有限公司
代理人 沈昭坤

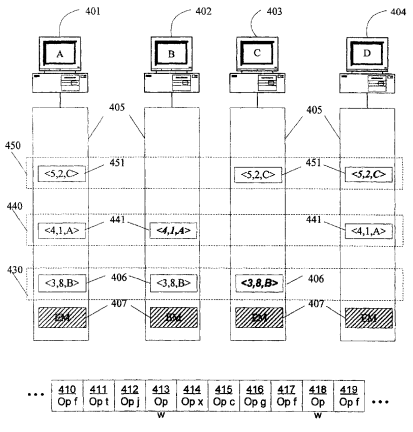
权利要求书 3 页 说明书 20 页 附图 8 页

[54] 发明名称

分布式容错计算系统中复制品集合的有效改变

[57] 摘要

分布式计算系统使用计算装置集合能够以容错方式操作。 通过实行状态机的独立复制品和选择建议，计算装置集合能够容许许多错误。 通过使复制品成为主机，参与在分布式计算系统中的计算装置能够通过从集合中增加或删除计算装置、或者通过为参与者指定特殊计算装置来调节。 通过用可操作的装置、或通过增加系统中冗余的数目来代替有缺陷的装置，改变装置中参与的计算装置，提高了容错。



1、一种在容错分布式计算系统中允许第二计算装置集合确定步骤序列中操作的方法，所述容错分布式计算系统包括第一计算装置集合，在第一集合中的计算装置确定将由状态机执行的的操作以作为步骤序列，此方法包括：

由所述第一集合中的一个计算装置指定所述第二计算装置集合，所述第一和第二集合的计算装置是复制的计算机装置；

指定一个步骤作为最后的步骤，其中第一集合为该步骤确定将要由状态机执行的一个操作；并且

由第二集合中的计算装置确定在所述指定的最后步骤的后续步骤中，将要由状态机执行的操作，其中，对于计算装置是其内部活动成员的各个集合，在计算装置上分别运行一个独立的，由世代值所标示的协议模块 AM。

2、如权利要求 1 所述的方法：进一步包括：

通过第二集合中至少一个计算装置，验证第一集合中的至少一个计算装置的可靠性。

3、如权利要求 1 所述的方法，其中此方法响应一决策而被自动初始化。

4、如权利要求 1 所述的方法，其中此方法响应明确的用户请求而被初始化。

5、一种在容错分布式计算系统中产生计算装置第二集合的方法，所述容错分布式计算系统包括计算装置的第一集合，第一集合中的每一计算装置运行状态机的复制品，将要在状态机上运行的操作由第一集合的引导机提议，在第二集合中的每一计算装置运行状态机的复制品，此方法包括：

由第一集合中的一个计算装置请求生成所述第二集合；

在第一集合装置的法定数目中，同意组成第二集合的计算装置将在给定步骤后运行状态机；

第一集合的引导机提出状态机无效操作的充分次数以达到给定步骤；

其中，对于计算装置是其内部活动成员的各个集合，在计算装置上分别运行一个独立的，由世代值所标示的协议模块 AM。

6、如权利要求 5 述的方法，其中此方法自动初始化以响应决策。

7、如权利要求 5 所述的方法，其中此方法初始化以响应明确的用户请求。

8、如权利要求 5 所述的方法，进一步包括：

通过第二集合中的至少一个计算装置，验证第一集合中至少一个计算装置的真实性。

9、如权利要求5所述的方法，其中只有在第一集合中计算装置的法定数目同意此建议时，状态机执行引导机提议的操作。

10、一种在第一复制计算装置上运行的方法，所述方法用于容错分布式计算系统中，所述容错分布式计算系统包括多个复制的计算装置，复制计算装置第一集合确定将由状态机执行的第一步骤序列中的操作，复制计算装置第二集合确定后来将由状态机执行的第二步骤序列中的操作，其中第一序列和第二序列互斥，所述方法包括：

用运行模块运行状态机各步骤中的操作；

用第一协议模块协调第一集合中其它复制计算装置以确定第一序列中的操作；和

用第二协议模块协调第二集合中其它复制计算装置以确定第二序列中操作。

11、如权利要求10所述的方法，其中在运行模块运行完第一序列的步骤后，第一协议模块被撤消。

12、如权利要求11所述的方法，其中，在撤消前，第一协议模块轮询至少一个第二集合中的复制计算装置，以确保复制计算装置的第二集合确定将要由状态机执行的操作。

13、如权利要求12所述的方法，其中，在撤消前，紧跟着第一步骤序列中的最后操作的性能，第一协议模块确保在第二集合中法定数目的复制计算装置已经将状态机状态存储在稳定存储器中，。

14、如权利要求11所述的方法，其中已撤消的第一协议模块保持有关复制计算装置第二集合的信息。

15、如权利要求11所述的方法，其中如果没有协议模块保留在复制计算装置上，就撤消运行模块。

16、如权利要求10所述的方法，其中复制计算装置的第一集合包括引导机装置，并且只有在第一集合中法定数目的复制计算装置对引导机装置的操作的建议一致同意时，才由状态机执行第一序列中的操作。

17、一种容错分布式计算系统中的复制计算装置，所述容错分布式计算系

统包括多个复制计算装置、复制计算装置第一集合确定将要由状态机在第一步骤序列中执行的操作，复制计算装置第二集合确定将要由状态机在第二步骤序列中执行的操作，其中第一序列和第二序列互斥，并且第一步骤序列在第二步骤序列之前，包括：

运行状态机操作的运行模块；

与第一集合中其它复制计算装置协调的第一协议模块，以确定第一序列中的操作；和

与第二集合中其它复制计算装置协调的第二协议模块，以确定第二序列中的操作。

18、如权利要求17所述的计算装置，其中所述运行模块被配置成执行由第一集合的引导机所提议的充分数量的空操作以到达第二步骤序列。

19、如权利要求17所述的计算装置，其中在运行模块运行完第一序列的步骤后，就撤消第一协议模块。

20、如权利要求19所述的计算装置，其中在被撤消前，第一协议模块轮询至少一个第二集合中的复制计算装置，以确保复制计算装置的第二集合确定将要由状态机执行的操作。

21、如权利要求20所述的计算装置，其中在被撤消前，紧跟着第一步骤序列中的最后操作的执行，第一协议模块确保第二集合中法定数目的复制计算装置已经将状态机的状态存储在稳定存储器中。

22、如权利要求19所述的计算装置，其中已撤消的第一协议模块保持有关复制计算装置第二集合的信息。

23、如权利要求19所述的计算装置，其中如果没有协议模块保留在复制计算装置上，就撤消运行模块。

24、如权利要求18所述的计算装置，其中运行模块执行无效操作，直到第一集合确定的步骤，并且在确定的步骤后执行第二序列的操作。

分布式容错计算系统中复制品集合的有效改变

技术领域

本发明主要涉及分布式计算系统，尤其是涉及容错分布式计算。

背景技术

分布式系统的优点是在面临削弱单片集成电路计算装置的物理困难时能继续运行。这些困难包括：持续的电源储运损耗、恶劣天气、洪水、恐怖活动等诸如此类。

为了弥补独立计算装置可能出现断网、关闭、遭受系统故障或者不能工作等增加的危险，可使用冗余以使分布式计算系统保持运行。于是，在任意计算装置上存储的信息或运行的程序可以冗余地存储在其它计算装置上，即使当计算装置之一出现故障时，仍可存取信息。

分布式计算系统可实行完全冗余，其中系统中的每一装置执行同一任务并存储同一信息。即使大半装置出现故障时，这种系统仍允许用户继续执行有用操作。可选的是，这种装置能够允许同一信息的多个副本遍布地理区分布。例如，跨国公司可建立全球性的分布式计算系统。

然而，由于包括以相同次序来执行同一操作的系统的独立装置的完全保证的复杂性，因此分布式计算系统很难维护。为了便于处理这种经常出现的困难任务，经常采用状态机方法在独立装置中调整行动。状态机由状态集合、命令集合、响应集合及联系响应/状态对到每一命令/状态对的客户端命令来描述。通过改变其状态并发出响应，状态机能够运行命令。因此，能够通过其当前状态和将要执行的行动来完整地描述状态机。

因此，状态机的当前状态依靠其前状态、由此以后执行的命令、和执行这些命令的顺序。为了使两个或更多状态机之间达到同步，可以建立通用初始状态，每一状态机可以用该初始状态启动并以相同顺序运行相同命令。因此，为了使状态机与另一个达到同步，需要确定由其它状态机执行的命令。因此同步问题，成为确定执行命令顺序的问题，或尤其是确定为给定步骤执行特殊命令

的问题。

一种为确定给定步骤将要执行的命令的机制为 Paxos 算法。在 Paxos 算法中，任何独立装置能够作为引导机(leader)，并试图建议一个由系统中的每一装置所执行的给定客户端命令。每个这样的建议能够用建议数目来发送以更加容易地跟踪该建议。这些建议数目不需要负担任何与尝试同意一命令的装置所要执行的特定步骤有关的关系。最初，引导机能够为引导机将要提交的建议提议建议数目。然后，每一剩余的装置能够使用他们表决赞成的最后建议的指示，或者他们没有表决赞成任何建议的指示，来响应建议数目的引导机意见。如果通过不同的响应，引导机没有知晓任何其它由所述诸装置表决赞成的建议，那么通过使用在早先消息中提议的建议数目，引导机能够提议由诸装置运行的给定客户端命令。在那种状态下，每一装置能够确定是否表决赞成或否决行动。仅当装置已经对更大建议数目的另一引导机建议进行了响应时，装置会否决其行动。如果装置的充分数量（称之为法定数目）表决赞成此建议，那么所建议的行动被称为已经达成一致，并且每一装置执行此行动，并能够传输此结果。通过这种方式，每一装置能够以相同顺序执行行动，以在所有装置中达到相同的状态。

通常，Paxos 算法在两个阶段中考虑，一个是如前所述的，允许引导机知晓由诸装置表决的在先建议的初始阶段，以及引导机能够提议供执行的客户端命令的第二阶段。一旦引导机学会了在先的建议，它就不需要连续地重复第一阶段。取而代之，引导机能连续地重复第二阶段，提议由分布式计算系统在多个布骤中运行的一系列客户端命令。于是，尽管由分布式计算系统为每一步骤运行所建议的每一客户端命令能够作为 Paxos 算法的一个实例提议，但是在为下一步骤提议另一客户端命令之前，引导机不需要等待装置来为对给定步骤的提议客户端命令进行表决。

分布式计算系统，作为一个整体，能够模拟为状态机。于是，执行完全冗余的分布式计算系统能够使每一装置复制整个系统的状态，于是每一装置是同一状态机本身的“复制品”。这种系统要求每一复制品保持相同的状态。如果某些复制品认为一个客户端命令被运行，而第二组复制品认为不同的客户端命令被运行，那么整个系统就不再作为单一的状态机来操作。为了避免这种情况，通常多数复制品被要求选择一个提议的供系统运行的客户端命令。由于复制品

的任意两个组中（每一组都有多数复制品）共享至少一个复制品，诸如 Paxos 算法的机制可以被实现为依靠所述至少一个共同的复制品来阻止两组复制品（每一组有多数复制品）选择不同的所提议的客户端命令，每一组里都包括多数复制品。

然而，预期运行延长的一段时间的系统必须应对永远退役或永远有故障的装置。因此系统应该提供用于交换出某些装置并用其它（如改变状态机复制品集合）装置加以代替的手段，从而使得系统的未来容错恢复到它的最初水平。另外，一旦产生法定数目的错误，系统不再容错。替换出错装置恢复了系统的容错以容许新的错误。另外，系统应该允许增加新装置以提高容错装置的数目，以提供更大的容错。对于系统如何确定改变在协议中参与的状态机复制品的集合，最初的 Paxos 算法给予了简单的描述。但是由最初的 Paxos 算法提议的方法不能应对可在复制品集合中的改变中出现的多种特殊环境。

发明内容

使用计算装置集合，分布式计算系统能够以容错模式操作。通过实施状态机的相同复制品和利用 Paxos 算法选择提议，计算装置集合能够容许多个错误。本发明的实施例允许修改参与在分布式计算系统中的计算装置集合，所述修改是通过从集合中增加或删除计算装置、或者通过指定用于参与的特殊计算装置来进行的。通过用可操作的装置来代替有缺陷的装置或者通过增加系统中的冗余数目来改变集合中的参与计算装置，容错得到了提高。

因此，在本发明的一个实施例中，计算装置的第一集合负责确定由在诸装置中被复制的状态机所执行的步骤的序列。第二装置集合可以被定义为从第一集合中删除责任，并承担确定将要被状态机执行的步骤序列中步骤的责任。第一集合负责的步骤范围与第二集合负责的步骤范围互斥。

在本发明的另一实施例中，负责确定由状态机执行的步骤序列的计算装置第一集合的引导机知晓已经选出了计算装置的新集合以继任该集合。新集合承担确定起始于特定步骤的步骤序列的责任，并且当在先于所述特定的步骤被执行时，第一集合的引导机使得无效操作被确定。

在本发明的其它实施例中，运行相同状态机的复制品的计算装置主管运行模块和一个或多个协议模块。每一装置的运行模块以连续顺序运行复制的状态

机的步骤。每一装置有对应于每一该装置为其中一员的复制品集合的协议模块。在同一状态机上的不同复制品集合的协议模块均分别与在该状态机上的一个运行模块相通信，以向它提供用于步骤序列的选中操作。

尽管这里的描述主要集中在分布式计算系统中计算装置的操作上，但是应认识到，这种描述与用于运行单独计算装置的程序是相同的，如在独立处理器上或在独立的存储空间中。于是，无论多处理器在物理上是位于一个或更多计算装置中，并且在多虚拟机环境中，无论多处理机正在由一个或更多计算装置运行，其它实施例包括多处理器环境中改进 Paxos 算法的操作。参考对应的附图，本发明的其它特征和优点可以从下面将要详细描述的说明性实施例中看出。

附图说明

附具的权利要求详细地说明了本发明的特征，而本发明及其目的和优点可以结合附图从下面的详细描述中得到最好的理解：

图 1 是表示本发明一个实施例可执行的典型分布式计算系统的方框图；

图 2 是表示本发明一个实施例可执行的典型计算装置的方框图；

图 3 是表示本发明一个实施例的可执行已复制状态机典型装置的方框图；

图 4a 是表示本发明一个实施例的可执行已复制状态机典型装置的方框图；

图 4b 是表示本发明一个实施例的可执行已复制状态机典型装置的方框图；

图 5 是表示根据本发明的一个实施例，已复制状态机操作槽的典型装置的方框图；

图 6 是表示本发明一个实施例的可执行已复制状态机典型装置的方框图；及

图 7 是表示根据本发明的一个实施例的示例性的复制品集合证明链的方框图。

具体实施方式

分布式计算系统可包括许多独立个人计算装置、服务器计算装置、或其它具有高效处理器和存储能力以在此系统中使用的装置。分布式计算系统能够综合其子计算装置的能力，以提供快速增长的处理能力和存储空间，或者实施冗余以允许多个装置对同一信息进行存取。于是，分布式计算系统的一个普通应

用是综合公共网络上的许多不同的个人计算装置的处理能力和存储空间。这种分布式计算系统可以保持有关系统的信息，例如哪个装置当前为系统的一部分、在哪个装置上存储了给定的信息集合。这些信息对于装置综合它们的能力和存储空间是必需的，并且作为结果，每一装置可包含一副本。如下面将要描述的，系统中装置的信息同步能够通过状态机方法得到促进。

可选的是，分布式计算系统的日益常见的用途是作为不同信息类型中央存储仓库的网络服务器。这种分布式计算系统尝试在所有子装置上复制中央存储器，以使得试图与中央存储器通信的每个客户端能够找到方便高效的通信装置。更进一步，由于系统的分布属性，本地事件如能量损耗、洪水、社会动荡等诸如此类仅可影响一些计算装置，而允许整个系统继续正常操作，并给客户端提供信息存取及其它服务。

这种分布式计算系统可被看作为一个状态机，状态机的未来状态由当前状态和将要采取的行动来定义。然后，分布式计算系统的每一子装置能够独立运行整个系统的状态机。状态机方法能够异步执行；所以，不需保持子装置间的精确同步，并且通过为所有装置设置初始值，然后以相同顺序运行相同函数，就能够得到装置间的同步。保持同步的普通方法是允许分布式计算系统的子装置在运行下一个函数前都对该函数达成一致意见，并且可保持已经运行的函数列表。于是，每个装置都有相同的状态，并且如果一个装置出现故障，它只需确定其最后运行的函数，从列表中识别自该最后函数后被同意的函数，并运行这些函数。

作为服务器的分布式计算系统尤其可用于给不同的客户端装置提供大量的信息，如跨国公司的中央数据库，或者流行的万维网站点。在此情况下，大量客户端可向作为服务器的分布式计算系统请求信息。通过在多装置间执行服务器功能，由于增加的冗余，服务器作为一个整体更不易于发生故障。

子计算装置同意运行的下一函数的一种机制是 Paxos 算法。在 Paxos 算法中，正如下面将要进一步描述的，任何装置可以作为引导机(leader)并向分布式计算系统中的其它装置发送建议数目。其它装置可以用与具有装置已经表决的最大建议数目的建议的指示互相响应，或者与装置未对任何以前的建议进行表决的指示相响应。一旦引导机从其它装置接收到响应，它就可以确定提议哪个函数并对已提议的函数要求表决。每个装置将对此建议进行表决，除了在要求

表决前的某时，它已经对更大建议数目的建议有了响应。如果有法定数目的装置表决同意此建议，那么此建议就被接受，并且引导机能够把消息发送到所有的装置，请求它们运行已同意的函数。

然而与其它任何合意算法一样，Paxos 算法要求大量的计算装置容错。尤其是，为了容许 F 个错误，算法就要求分布式计算系统包括至少 2F+1 个计算装置。只需要这些装置的简单多数来选择命令并继续系统的正常操作。其它装置可以保持不用，直到其中一个选择命令和操作此系统的装置出现故障。

然而，当一个计算装置出现永久故障时，将它从 2F＋1 个装置中删除并用新装置来代替它是所期望的，为此 Paxos 算法将容许 F 个附加的故障。

分布式计算环境

现在转向附图，其中相同的数字表示相同的元件，由分布式计算系统的执行来说明本发明，如图 1 中所示的典型分布式计算系统 10。只是为了容易说明，本发明将参考如系统 10 的分布式计算系统进行描述，系统 10 包括如图 1 中所示的互连计算装置 11 到 15。本领域的技术人员可以理解的是，本发明可用于所有的分布式计算环境，并不只限于图 1 中所示的为了说明目的已经简化的典型分布式计算系统的任何形式。

尽管本发明倾向于在具有任意数目的客户端计算装置的环境下使用，图 1 中表示了单个客户端计算装置 20。图中客户端计算装置 20 表示为与分布式计算系统 10 有普通的通信连接。如本领域的技术人员所知道的，这种通信连接可使用任何的通信介质和协议，并允许客户端计算装置 20 与在分布式计算系统 10 中的至少一个计算装置通信。

另外，图 1 中表示了计算装置 30 和 31，它们都没有作为分布式计算系统 10 的一部分表示出来，但是也与系统 10 保持有通用的通信连接。如上所述，这种通信连接能够使用任何的通信介质和协议，并允许计算装置 30 和 31 与在分布式计算系统 10 中的至少一个计算装置通信。如下面将要进一步详细描述的，计算装置 30 和 31 能够学习由系统 10 执行的运行结果，而不作为系统 10 的一部分。

尽管未被请求，本发明将在计算机可执行指令的一般上下文中详细描述，如由计算装置运行的程序模块。通常，程序模块包括执行特殊任务或执行特殊抽象数据类型的例程、程序、对象、构件、数据库等。此外，本领域的技术人

员应认识到本发明可用许多不同的计算装置来实施，包括手持装置、多处理器系统、微处理器或可编程消费品电子装置、网络 PCs、微型计算机、大型计算机等诸如此类。如上所述，本发明也可在分布式计算环境中实施，如分布式计算系统 10，其中由远程处理装置执行任务，这些装置通过通信网络相连。在分布式计算环境中，程序模块可位于本地和远程存储装置上。

转向图 2，表示出了将在其上执行本发明的示例性计算装置 100。计算装置 100 只是合适的计算装置的一个例子，并不意味表示本发明功能或使用范围的任何限制。例如，示例性计算装置 100 并不试图仅仅表示图 1 中所示的计算装置 11-15、20 或 30-31。示例性计算装置 100 可执行这些计算装置中的至少一个，如通过存储器分区、虚拟机、多处理器或类似的可编程技术，这些技术允许一个物理计算结构执行行动，这些行动在下面作为多计算装置的特征来描述。更进一步，计算装置 100 不应理解为对于图 2 中所示的任一外设或外设的结合有任何依靠或请求。

本发明可以在计算机可执行指令的一般上下文中进行描述，如由计算机运行的程序模块。通常，程序模块包括执行特殊任务或执行特殊抽象数据类型的例程、程序、对象、构件、数据库等。在分布式计算环境中，任务可通过远程处理装置来执行，这些装置通过通信网络互连。在分布式计算环境中，程序模块可位于包括记忆存储装置的本地或远程存储介质上。

计算装置 100 的组件包括但不限于，处理单元 120，系统存储器 130 和系统总线 121，此系统总线将包括系统存储器在内的不同系统组件与处理单元 120 连接起来。系统总线 121 可为多种类型总线结构的任意一种，包括存储器总线或存储控制器、外设总线和使用任意总线结构的本地总线。顺便举例来说，但并不限于，这种结构包括工业标准结构（ISA）总线、微通道结构（MCA）总线、扩展工业标准结构（EISA）总线、视频电子标准协会（VESA）本地总线和也称作 Mezzanine 总线的外设组件互连（PCI）组件。更进一步，处理单元 120 包括至少一个物理处理器。

计算装置 100 典型地包括多种计算机可读介质。计算机可读介质可为任意可通过计算装置 100 存取的可用介质，并包括可变和不可变介质，可移动和不可移动介质。顺便举例来说，但并不限于，计算机可读介质包括计算机存储介质和通信介质。计算机存储介质包括可变和不可变、可移动和不同移动介质，

为了如计算机可读指令、数据结构、程序模块或其它数据等信息的存储，它们以任意方法或技术实施。计算机存储介质包括但不局限于 RAM、ROM、EEPROM、闪存或其它存储技术，CD-ROM，数字通用光盘（DVD），或其它光盘存储器、磁盒、磁带、磁盘存储器或其它磁性存储装置，或任何可用于存储所需信息并可由计算装置 100 读取的其它介质。典型地，通信介质在模块化数据信号如载波或其它传输机制中包含计算机可读指令、数据结构、程序模块或其它数据，也包括任意的信息传输介质。术语“模块化数据信号”表示一个信号，它具有至少一个特征集合或以这样一种形式变化，以在信号中为信息编码。顺便举例来说但不限于，通信介质包括有线介质如有线网络或直线连接，和无线介质如声音、RF、红外线和其它无线介质。上述的任意组合也包括在计算机可读介质的范围内。

系统存储器 130 包括可变和/或不可变形式的计算机存储介质，如只读存储器（ROM）131 和随机存取存储器（RAM）132。基本输入/输出系统 133（BIOS）典型地存储在 ROM131 中，包括如在启动过程中，帮助在计算机 110 中的组件间传输信息的基本例程。RAM132 典型地包括数据和/或程序模块，这些程序模块可立即存取和/或稍后由处理单元 120 操作。顺便举例来说但并不局限于，图 2 表示出了操作系统 134、应用程序 135、其它程序模块 136 和程序数据 137。

计算装置 100 也可包括其它可移动/不可移动、可变/不可变计算机存储介质。只是顺便举例来说，图 2 表示出了读出或写入不可移动、不可变磁性介质的硬盘驱动器 141，读出或写入可移动、不可变磁盘 152 的磁盘驱动器 151，和读出或写入可移动、不可变光盘 156 如 CD ROM 或其它光介质的光盘驱动器 155。可用于典型操作环境中的其它可移动/不可移动、可变/不可变计算机存储介质包括但不局限于盒式磁带、闪存卡、数字通用光盘、数字视频磁带、固态 RAM、固态 ROM 等诸如此类。硬盘驱动器 141 通过不可移动存储器接口如接口 140 典型地与系统总线 121 相连，并且，磁盘驱动器 151 和光盘驱动器 155 典型地通过可移动存储器接口，如接口 150 与系统总线 121 相连。

上面所讨论的和在图 2 中所示的驱动器及与它们相关的计算机存储介质提供了计算机可读指令、数据结构、程序模块和计算装置 100 的其它数据的存储器。在图 2 中，例如，如图中所示硬盘驱动器 141 存储了操作系统 144、应用程序 145、其它程序模块 146 和程序数据 147。请注意这些组件能够与操作系统 134、

应用程序 135、其它程序模块 136 和程序数据 137 相同或不同。这里给予操作系统 144、应用程序 145、其它程序模块 146 和程序数据 147 不同的数目，以在图中表示出最小值时，它们都是不同的副本。用户可以通过输入装置如键盘 162 和一般为鼠标、轨迹球或触摸板的定点装置 161，来输入命令和信息到计算装置 100 中。其它输入装置（未示出）也可包括麦克风、操纵杆、游戏操纵杆、圆盘式卫星电视天线、扫描器等等。这些和其它的输入装置通常通过与系统总线连接的用户输入接口 160 与处理单元 120 相连，但也可通过其它接口和总线结构，如并行接口、游戏端口或通用串行总线（USB）。监视器 191 或其它类型的显示器也可通过如视频接口 190 的接口与系统总线 121 相连。除了监视器，计算机也可包括其它外围输出装置如扬声器 197 和打印机 196，它们也可通过输出外围接口 195 连接。

计算装置 100 可在如图 1 所示的网络环境中工作，其使用与一个或更多远程计算机的逻辑连接。图 2 中表示出了与远程计算装置 180 的通用网络连接 171。通用网络连接 171 和图 1 中所示的网络连接，能够为任意多种不同类型的网络和网络连接，包括局域网（LAN）、广域网（WAN）、无线网络、遵照以太网协议的网络、令牌环协议或其它逻辑、物理或包括因特网或万维网的无线网络。

当在网络环境中使用时，计算装置 100 通过网络接口或适配器 170 与通用网络连接 171 相连，这些适配器可为有线或无线网络接口卡、调制解调器或类似的网络装置。在网络环境中，有关计算装置 100 描述的程序模块或它的一部分可存储在远程存储装置中。应认识到，图示出的网络连接都是典型的，还可以使用在计算机间建立的其它类型的通信连接。

在下面的描述中，本发明将参考由至少一个计算装置执行的操作的行动和符号表示来描述，除了它表示其它意思。同样，可以理解的是，有时参考计算机可运行的这些行动和操作，包括由表示结构类型数据的电子信号的计算装置的处理单元的处理。这个处理转换数据或使其保持在计算装置的存储系统中，它们以本领域技术人员容易理解的形式，重新分配或另外改变计算装置中的操作。保持数据的数据结构在物理上位于存储器上，此存储器具有由数据类型定义的特殊性质。然而，当在前文中描述本发明时，并不意味着对本发明的限制，本领域的技术人员应理解，此后描述的不同行动和操作也可在硬件中实施。

总述

依据本发明，当允许从系统中增加或删除装置时，分布式计算系统可执行容错算法。将在下面更加详细描述的这种 Paxos 算法，能够提供一种分布式计算系统实施的机制，此系统能够容许一定数量的故障（假定超过该数量两倍的计算装置被使用）。在算术上，如果变量“F”表示能够容许的故障数量，那么有至少 2F＋1 个计算装置的分布式计算系统能够通过尝试运行状态机的相似复制品来实施 Paxos 算法。在这些 2F＋1 个计算装置中，至少含它们大部分的任意集合，换句话说，至少 F＋1 个装置的任意集合可为法定数目，并能够以容错形式选择建议。

如果没有计算装置出现故障，2F＋1 个计算装置的集合能够依据 Paxos 算法执行状态机。如果一个或多个计算装置出现故障，那么一个或多个剩余的计算装置 F 能够被用于使系统继续使用 Paxos 算法，并以容错形式来操作。然而，如果一个计算装置出现永久故障，就希望，要么将它从 2F＋1 个计算装置的集合中删除，或者增加新的计算装置到 2F＋1 个计算装置的集合中。

根据本发明预期的一个实施例，计算装置的初始集合被用于提供容错分布式计算系统。当一个或多个计算装置出现故障时，可从初始集合中删除此装置，并且也可给此集合增加新的计算装置。因此，容错系统的操作可以更改其上有容错分布式计算系统操作的计算装置的集合。

状态机

在分布式计算环境中，如图 1 中所示的分布式系统 10 中，异步操作装置间的协调是一个困难的任务。解决这些困难的一种机制是按照状态机模拟分布式计算系统，其中，函数的执行使状态机从一个状态移到另一个状态。状态机能够参考状态集合、命令集合、响应集合和将响应/状态对和每一命令/状态对连接起来的函数来描述。状态机的客户端能够发出请求状态机运行函数的命令。然后此函数改变状态机的状态并产生响应。

包括分布式计算系统的独立装置能够分别运行系统的状态机。因此这些装置能够通过确定初始状态，然后自此在其上以相同顺序运行相同函数来来进行协调。通过简单确定装置运行的最后一个函数、在函数有序列表中定位由其它装置运行的函数，然后命令该装置执行来自有序列表的尚未执行过的函数，就能够使装置同步。这种状态机方法最初在 1978 年 7 月出版的《ACM 通信》第 21 卷第 7 号中 Leslie Lamport 的论文“时间、时钟和分布式系统的时间顺序”中

提出，其全部内容在这里被参考引用。

Paxos 算法

通过使用状态机方法，图 1 中示出的分布式计算系统 10 的子装置 11－15 间的同步，能够通过同意将要执行的函数和执行它们的顺序来得到。一个同意将被执行函数的方法是 Paxos 算法。即使面临出错，Paxos 算法也允许系统 10 适当操作，其中装置不经过提前警告也能够停止操作。Paxos 算法要求在系统作为一个整体执行一个函数前至少有法定数目的装置同意该函数。在 Paxos 算法中，法定数目可为一个简单多数，或者更大的数字，这取决于系统的特殊需要。然而，法定数目可定义为充分地大，以至任意两个法定数目具有至少一个共同装置。Paxos 算法的详细描述在 1998 年 5 月出版的《计算机系统上 ACM 事务》第 16 卷第 2 号第 133－169 页的 Leslie Lamport 的题为“部分时间会议”的论文，和 2001 年 12 月出版的《ACM SIGACT 新闻》第 32 卷第 4 号第 18－25 页 Leslie Lamport 的题为“简化的 Paxos”文章中，其中它们的全部内容没有排除任一部分在这里被参考引用。

为了清楚起见，当在本发明的实施例中描述 Paxos 算法时，提出一些新的或可选术语和概念。参考图 3，图中表示出了四台装置 301－304。装置 301－303 的每一个运行普通状态机的复制品 305，一起形成“复制品集合”。装置 304 不在复制品集合中。每一复制品 305 运行两个逻辑模块，协议模块 306（AM）和运行模块 307（EM）。在下面描述的方式中，AM 彼此协调以为状态机中的每一槽（或“步”）选择操作，其中槽“n”为用于每一状态机复制品应当运行的第 n 步操作的逻辑容器。例如，槽 310 包括将被执行的操作“f”，槽 311 包括操作“t”等。EM307 从配置的 AM306 中获知操作选择。

通过使一个复制品为“引导机”并只允许引导机提议可为槽选择的操作，Paxos 算法确保没有两个 AM 为相同的槽选择不同的操作引导机引导机。然而，引导机可能会出错，于是为了选出新引导机，就要有一个更合适的协议。通过选择协议，新引导机充分地学会了以前的引导机的行动，不会做出矛盾的选择。

引导机的“任期”称作“视图”，并且每一视图有唯一的视图标识（ID），它与用在经典 Paxos 描述的“建议数目”相等。也就是说，对于每个特殊的视图，单独的引导机提议所有将由状态机在视图中执行的操作。每一复制品跟踪其当前的视图，并只从那个视图的引导机接收操作建议。视图 ID 由视图计数器

和初始复制品标识组成；于是，如果多个复制品在同一时间初始化视图，那么它们将启动不同的视图。在图 3 的例子中，计算装置 C 303 为所显示视图的引导机，装置 B 302 初始化此引导机，并且视图 ID 为数字 8。

引导机记忆其最后提议操作的槽。当客户端对其发送新请求时，通过对所有复制品发送提议（proposed）的消息，它提议此请求成为在最后槽之后的槽中的操作。当复制品从它当前视图的引导机中接收到提议的消息时，它通过在本地日志中记录提议的消息的方式来记录它。然后，它给引导机发送已准备(prepared)的消息。当引导机从法定数目（一般包括它本身）中接收到已准备的消息时，它选择操作，并给所有复制品发送已选择(chosen)的消息。换句话讲，它保证没有选择矛盾的操作，因此任何 EM 都可运行它。

在引导机提议一个操作后，在运行该操作前，存在网络和处理延迟。在此期间，引导机可提议进一步的操作，从而通过流水线操作来使处理的多个阶段的工作重叠。

在本发明的实施例中，每一状态机在稳定的存储器上生成它状态的周期复制品，于是算法能够阻止复制品日志不确定的增加。这种副本称作“校验点”，并且校验点的槽定义为在生成此副本前执行的最后操作的槽。复制品在磁盘上的最后校验点的槽称作“本地稳定点”。

通过周期性消息，引导机跟踪复制品的本地稳定点，并告知它们集体稳定点，此点为第 q 个最高本地稳定点，其中 q 为法定数目（如 F＋1）的大小。此值的重要性是由于最后法定数目的复制品将是起作用的，最后总有一个反映第一 c 操作的可得到校验点，其中 c 为集体稳定点。由于某些复制品能够在小于或等于综合稳定点的槽中提供反映所有的操作结果的状态，那就不需要记忆这些槽的操作选择。于是，在本发明的实施例中，在称作“日志切断”的处理中，复制品丢弃这些槽的准备(prepare)入口，避免它们的日志不确定增长。为了阻止任意复制品的日志超过固定或可变的最大日志长度，直到综合稳定点在槽的最大日志长度内，引导机才为槽提议一个操作。

由新引导机选出并学习足够的以前视图来确保它不提议矛盾操作的处理被称作“视图变化”。作为总结，如果任何复制品以本总结范围外的方式学习，那么引导机就可能出错，它可能初始化视图变化。它选择比目前视图 ID 更大的新视图 ID，并给所有复制品发送视图－变化的消息。接收到视图－变化的复制品

输入此新视图，直到它已经在更大数目的视图中。它用描述其日志中准备输入的 VC－REPLY 消息回复给初始化器。一旦视图改变初始化器有来自法定数目的 VC－REPLY 消息，它为新视图选择引导机并将这些消息提供给它。新引导机使用这些消息来学习足够多的有关以前的视图，以确保它生成没有矛盾的建议。视图改变处理在以前引用参考的 Lamport 有更加全面的描述。在图 3 的例子中，装置 B 302 为视图数目 8 初始化视图改变。

在本发明的实施例中，通过周期性消息，引导机告诉每一复制品有多少已经选出的连续操作。这样 AM 就可听到它的 EM 需要的下一个操作已经被选择了。但是，当引导机为那个操作发送提议或选择的消息时，也许由于断线， AM 可能不知道为这个槽选择了什么操作。在这种情况下，AM 就询问它的对等 AM，是否它们中的任意一个在最近选择操作的高速缓存中有这个操作。AM 保存最近选择操作的高速缓存，这些操作包括来自其日志的操作，但是另外也可包括从日志中本地切断的操作。如果 AM 在其高速缓存有这个操作，它就将其提供给请求的对等 AM，而避免请求的 AM 执行全状态传递的需要。如果没有对等的 AM 能够提供操作，并且 AM 直到由于日志切断引导机丢弃了操作，复制品的 EM 就不能进行。这样，它就从更新的复制品中取得状态。

使用复制品集合的改变来修改 Paxos 算法

从上面的描述可以看出，通过掌管使用中的公共状态机的复制品，就不需使每一计算装置参与算法的实例。换句话讲，某些装置可能起先不是使用中的状态机复制品集合的成员，如在图 3 中的例子中的装置 D304。例如，为了代替永远出现故障或已经从分布式系统中删除的装置，可能需要给复制品集合增加这种装置。通过改变复制品集合，可以通过 Paxos 算法增加计算装置和从事项中删除装置。于是，被删除的装置没有算在 2F＋1 个装置中（法定数目可从中算出）。本发明的实施例帮助从复制品集合中的增加和删除计算装置（更准确的说是它们的状态机）。

下面给出改变复制品集合的方法的一般描述：作为其状态的一部分，状态机包括，负责服务的复制品集合。存在一个常数管道深度“α”，其用于保持待定操作的数目，从而使得复制品集合的改变在α个操作后生效。换句话讲，如果复制品集合在第 n 个槽改变，通过只在槽 n＋α或更大的槽中选择操作，新的复制品集合允许提高到α个待定操作。α限制了操作管线的深度，所以期望有

大α值：引导机无法提议第n+α个操作，直到它知道负责那个操作的复制品集合，引导机而直到运行操作n，它才可能知道这一点。

本发明的实施例改变了经典的Paxos模型，从而使得每一视图针对复制品集合特定化(replica-set-specific)，即仅仅涉及一个固定的复制品集合。以这种方式，由于它们涉及固定的复制品集合，复制品集合不互相干涉，并且视图都很容易管理。为了确保视图是针对复制品集合特定化的，本发明的实施例扩展了视图ID的描述以包括下面描述的“世代(epoch)”值。对于其在内当前为活动成员的每一个复制品集合而言，每一装置运行单独的由世代标示的协议模块（AM）。如果一个装置在多个复制品集合中，其多个AM独立且并行地操作，每一个AM都在其自身的世代中，并且因此在其自身的视图中。

参考图4a，示出了扩展的视图ID的例子。图示的复制品集合430包括来自计算装置A401、B402和C403的复制品405。作为图3中的例子，计算装置B402初始化视图，此视图具有装置C403作为其引导机并有为8的视图ID。然而，为该视图的每个装置的AM406都存储了一个额外字段，其指示复制品集合的世代值。在此例子中，复制品430的世代值为数字3。

作为在本发明实施例中的应用，用针对复制品集合特定化的视图改变复制装置的使用的例子现在参考图4a和4b来描述。复制品集合被连续生成并标签为R_i，其中指数i为复制品集合世代，并且R_0为初始化的复制品集合。对于给定的复制品集合，集合中的每一装置有对应的AM。在图4a中，复制品集合中的每一状态机有AM，它对应其复制品集合R_3 430中的成员。每一AM有一世代3。图4a中的每一AM406在视图<世代→3,视图→0,初始化器→nil>中开始。当时间进行时，发生视图改变并提议视图ID。例如，如果第一视图改变由复制品C初始化，新视图ID就为<3,1,C>。在图4a中，已经产生了若干视图变化，所以视图ID为<3,8,B>。当EM在图4b中运行产生新复制品集合R_4 440的操作时，它通知新复制品集合的成员各自启动一个有适当世代的AM。如果一些状态机在R_3 430和R_4 440中，这就使其运行两个独立的AM406和441。新AM将在视图<4,0,nil>中启动。不管视图ID R_3是否在其中，这个视图ID是独立的，并将独立于该视图而推进。复制品集合R_4 440的AM将R_4 440中的某些AM选为引导机；引导机这种引导机无需和复制品集合R_3 340的引导机在相同的状态机上。在图4b中，视图变化已经使AM440推进到视图ID<4,1,A>。

通过引用固定复制品集合，在本发明实施例中的诸视图都非常容易管理。例如，视图改变，涉及一个明确定义的参与者集合：复制品集合中的所有 AM。这些 AM 是视图改变初始化器向其发送视图－改变消息的 AM，并且来自这些 AM 的大部分的 VC－REPLY 消息构成了足以让下一引导机进行的法定数量的复制品。更进一步，很清楚地是，当一个 AM 对视图改变初始化负责时，并且当其是适格的引导机时：其被创建时具有这些职责，并保有这些职责直至其被撤销。另外，引导机准确地知道它需要和哪些 AM 交换的世代消息。

现在转向图 5，为了帮助针对复制品集合特定化的视图，同一状态机上的不同复制品的 AM 各自与那个状态机上的一个 EM 通信，以给出它所选出的操作。在未改变的 Paxos 算法中，EM 以槽顺序运行选出的操作。由于没有两个 AM 曾经为相同的槽负责，每一操作来自于单一的定义明确的 AM。在图 5 中，槽 500 为包括生成复制品集合 R_4 的操作的槽，而槽 515 为包括生成复制品集合 R_5 的操作的槽。固定管道深度为 $\alpha=10$。所以复制品集合 R_4 负责从槽 510（r_4 ＝复制品集合生成槽 500＋α）到槽 524（r_5-1）中选出操作。因为引导机不提议一个它不负责的操作，复制品集合就不选择它不负责的操作。引导机并不提议操作 n，直到其配置的 EM 执行了操作 n－α，而该操作的结果表明引导机的复制品集合负责槽 n 引导机引导机。因此，如果 EM 运行产生复制品集合 R_m 的操作，使该集合为槽 r_m 和以后的槽负责，复制品集合 R_{m-1} 和更早的集合的 AM 将不为槽 r_m 或更大的槽提议建议。

为了进一步促进特殊针对复制品集合特定化的视图，如果已经选择了新复制品集合，本发明的实施例允许引导机清空它的提议管道。这种引导机为“投机 (lame duck)”引导机。清空提议管道在下面的情形是有用的：复制品集合 R_{m-1} 的引导机提议它所负责的的最后操作 r_m-1，并开始将客户端引导向新的复制品集合。然后它崩溃并且随后的视图改变丢弃了所有老引导机的近期建议，于是新引导机提议的最后槽小于 r_m-1。由于所有活动的客户端已经被重新定位到新的复制品集合，它们将不向老复制品集合的新主要部分提交请求，并且没有操作得到为操作 r_m-1 的建议。这就意味着状态机不会进展，因为只有在槽 r_m 或更大槽中的操作可以被选中，并且直到运行操作 r_m-1 才能够执行它们。为了避免这种情况，本发明的实施例预想以下内容，当引导机得知另一个复制品集合将在操作 r_m 时接管，但是其提议操作的最后槽小于 r_m-1，那么它就为最后建

议槽和 r_m 之间的每一槽提议无效(null)操作。这就要求最多有 α －1 个无效操作，因为最后提议的槽至少为状态机已经运行的操作数目，即至少为 r_m－α，。这种清空管道的处理确保了状态机的执行最后达到槽 r_m。

为了进一步促进特殊针对复制品集合特定化的视图，本发明的实施例丢弃了旧的、不需要的 AM。一旦 R_m 的综合稳定点成为至少 r_m－1，那么 R_m 就“已经确定”。也就是说，其成员的法定数目已经启动，并且通过综合稳定点的定义，最后它们中的一个能够至少为 r_m－1 的槽提供校验点。这就避免了有关小于 r_m 操作（即，以前的复制品集合（如，R_0，R_1，…，R_{m-1}）所负责的供选择的所有操作）的任何信息的需要。于是，当 R_m 建立时，所有以前的复制品集合都被认为“停止(defunct)”，即不再需要。任何得知其具有停止 AM 的状态机可撤消其停止的 AM，包括它们的日志。如果状态机撤消了它仅有的正在运行的 AM，它也可撤消它的 EM；在服务中它就不再是活动的参与者。

为了进一步促进特殊针对复制品集合特定化的视图，在本发明的实施例中，状态机可只为已撤消 AM 保留少量的的不变状态：在已撤消本地 AM（“MaxDefunct”）中的最大世代，和对应于随后世代的复制品集合（$R_{MaxDefunct+1}$）。其将它们提供给任意尝试初始化与已撤消 AM 的通信的客户端或远程 AM。接到已撤消 AM 消息的客户端开始与随后的复制品集合通信，而知晓其对等停止的 AM 将知道撤消它自己。

本发明的实施例为复制品集合的 AM 提供机制以确保它的后继复制品集合得以建立，并使得 AM 最终知晓所述的建立，因此它们可以被撤销。当复制品集合 R_{m-1} 的 AM 知晓它的 EM 已经运行了操作 r_{m-1}（即该 AM 的复制品集合所负责的最后一个操作）时，它就生成一个线程，该线程周期地给新复制品集合中的每一 AM 发送 IS－SUCCESSOR－UP－TO－DATE 消息。此消息询问接收者的本地稳定点是否至少为 r_{m-1}，并且如果不是，就提议在那个槽或随后槽提供校验点。它没有把此消息传送给已经肯定地响应的 AM。最后，它从新复制品集合接收到法定数目的肯定响应，并因此知道 R_m 已经被建立。在此点上，它知道它已经停止，并撤消了自己。

由于在 R_{m-1} 中的每一 AM 发送 IS－SUCCESSOR－UP－TO－DATE 消息，最后它们之一建立了 R_m。发送 IS－SUCCESSOR－UP－TO－DATE 消息的责任可以只指派给 R_{m-1} 的引导机。可选的，发送 IS－SUCCESSOR－UP－TO－

DATE 消息的责任分派给所有 AM，而不是只给 R_{m-1} 的引导机。如果旧复制品集合引导机的 EM 在新复制品上时，就允许新复制品更快地得到它的初始校验点。同样，它扩散了在旧复制品集合的 AM 中提供校验点的负荷，而不是把全部负荷放置在引导机上。把责任分配给所有 AM 的另一个原因和如下面所描述的使用证书链来最优化有关。

在以下场景中， EM 使用建议(offer)以提供一个隐含在 IS－SUCCESSOR－UP－TO－DATE 消息中的校验点。假设EM需要从并置的AM获得下一操作，但是没有已并置的 AM 为此操作负责。例如，在图 6 中，如果在装置 C430 上的 EM 需要，来自由 R_4 管理的一个槽的操作，由于其状态机不是 R_4 的一部分，那么就没有 AM 可供询问。在这种情况下，它询问除了它需要的那个数字外的负责最小操作数目的 AM，在此例子中是对应 R_5 的 AM。此 AM 意识到它不能够提供 EM 所需的操作，并且事实上不能提供任何在 r_5-1 前的操作。于是，它就告诉 EM 在哪儿得到校验点，此校验点包括它的复制品集合未负责的所有操作的结果，如在 r_5-1 或更后的校验点。如果它在同一时期内的对等体 AM 之一局部地具有这种校验点，它就告诉 EM 和在那个对等体上的状态机联系。如果不是，但是它已经从先前的复制品集合中的某些复制品收到最近的 IS－SUCCESSOR－UP－TO－DATE 消息，它就指引 EM 从那个复制品得到校验点。

如果只有 IS－SUCCESSOR－UP－TO－DATE 消息能够生成新的 AM，那么直到 EM 运行操作 r_m-1，新 AM 才能够开始。作为让 AM 尽快开始的可选的最优化，本发明的实施例允许当 EM 运行生成了集合的操作 $r_m-\alpha$ 时，EM 给每个新复制品发送 JOIN 消息。收到 JOIN 消息的状态机将为新复制品集合启动 AM。JOIN 消息没有被承认或重新传送，因为它们只是优选方案。

本发明的实施例生成并使用了关于随后建立的复制品集合的“后继者信息(successor information)”。当 EM 运行生成复制品集合 R_m 的操作时，它就生成由复制品集合 R_{m-1} 的本地 AM 使用的后继信息。这种信息包括它的后继者负责的第一操作数目，r_m，和随后复制品的特性(identify)。AM 需要 r_m，所以如果它曾是引导机，引导机它不会提出标号比 r_m 大或相等的操作。它需要随后复制品的特性，从而使它稍后能够发送 IS－SUCCESSOR－UP－TO－DATE 消息。

有可能 EM 不运行生成复制品集合 R_m 的操作 $r_m-\alpha$，而是接收到使它跳过该操作的状态传递。于是，在完成状态获取后，EM 进行检验，看它现在是否

知道任何与其并置的 AM 的后继者信息，如果是，就把它分配给它们。为了达到这一点，所述状态包括 r_m 和 R_m，甚至可以是老复制品集合的 r_m 和 R_m。状态机能够不确定地保留此信息。可选的是，状态机可以下面的方式最终丢弃这种信息。请回忆直到它的复制品集合的综合稳定点在那个槽的最大日志长度中，引导机才为槽提议操作。所以，当状态机运行一个槽中的操作，该槽和 r_m-1 相距了最大日志长度，它能够推断复制品集合 R_m 的综合稳定点至少为 r_m-1，因此具有小于或等于 m－1 的世代的复制品集合都是停止的。于是，它可能丢弃世代数为 m－1 或更小的复制品集合的信息，只记着它知道将要停止的最近复制品集合的数目。于是，它不能给某些 AM 提供后继者信息，但是对于那些 AM，EM 能够告诉它们来撤消它们自身。

本发明的实施例使用“证书链”来表示复制品集合中的装置为合法的。收到告知它为新复制品集合的一部分的信息的状态机可能需要验证发送器不是在说谎。通常，在故障-中止的共识(fail-stop consensus)下，信任不是问题。然而，当复制品集合改变时，交换这种信息的状态机也可能不在同一个故障-中止共识组中。在最初复制品集合中的状态机通过服务证书（如由指定最初复制品集合的可信的证明机构（CA）签发的证书）来知道它属于那儿。由于在复制品集合改变期间，CA 可能断线，本发明的实施例就使用复制品来证明新的复制品集合。

证书链为一系列证书。每一证书证明一个特定的复制品集合。在链中的第一证书为服务证书，它证明最初复制品集合 R_0。每个其它证书由在前的复制品集合的成员签发。参考图 7，表示出了证书链 700 的例子。认证机构（如可信的第三方）发布由装置 A、B 和 C 组成的复制品集合 R_0 的证书 710。复制品集合 R_0 的成员装置 A，在证书 711 中证明由装置 A、B 和 D 组成了复制品集合 R_1。装置 D 发布一个证书 712 以证明复制品集合 R_2 由装置 A、D 和 E 组成。

如果一个状态机接收到表示它在特定复制品集合中的证书链，它就可确认这一点。新的 AM 保存该证书链此作为它的原始链。对于 R_0 中的 AM，服务证书自身是一个原始链。当 AM 需要告诉它的后继复制品集合的成员以生成 AM 时，它就签发描述新复制品集合的证书、将其附加到它的原始链中、并发送此证书作为新 AM 的原始链。

本发明的实施例利用两个可选的优化方法来保持原始链较短。首先，如果一个复制品集合由一些证书证明，并且复制品集合包含一些链中后续证书的签

名者，那么链中二者之间的所有中间证书能够被删除。第二，如果 AM 从另一在前者处接收到更短的链，AM 可以替换它的原始链。例如，再次参考图 7，如果 E 接收到来自 D 的链 700，那么稍后 A 发送相同的链，此外 A 前述其最后的证书 712， E 可以删除不必需的中间证书 711 以生成更短的链 720，然后用这个更短的链 720 代替它的原始链 700。收到来自在前者的更短证书链的机会是可能的，因为复制品集合中的每一 AM，而不仅仅是引导机，会发出如上面所讨论的 IS－SUCCESSOR－UP－TO－DATE 消息。

本发明的实施例提供接口，通过此接口状态机能够改变复制品集合。状态机，当运行操作时，可以调用 change_replicas()函数，此函数指定将要包括在新复制品集合中的状态机排列。

本发明的实施例使用决策来确定复制品集合应当何时改变。在某些实施例中，通过指导客户端提交复制品集合改变请求，管理员明确地管理复制品集合。状态机忽略缺少管理权限证据的请求。例如，管理员可以使用明确的(explicit)请求来永久性的撤下一个状态机或为了其它的容错而配置新的状态机。

在一些实施例中，为了不请示管理员就应对持久的状态机错误，复制品集合改变被自动地执行。按照定期间隔内，如五分钟，每一 EM 给服务发出特别的客户端请求来报告它自身还在工作。状态机跟踪额外状态，这由每一 EM 最后一次报告自身工作组成。此状态完全由操作确定，而不是由每一复制品上的本地时钟所确定，所以每一操作包括引导机分配的时间戳，并且每一操作要在特定“时间”运行。在给定时间运行操作的附加细节在 Adya，et al.,FARSITE：“完全可信环境的联合、可用且可读的存储器”,在 2002 年 12 月出版的 Proc,5thOSDI pp.1-14,Boston,MA 中讨论，其全部内容没有任何部分的扩展在这里被引用。按照给定的间隔，如每小时，状态机检验复制品集合中的最久的最后工作时间(last-alive time)是否大于一个小时。如果是，它就假定对应的装置永久出错，并且调用 change_replicas()函数以在复制品集合中替换它。

状态机也可确定地选择新装置成为替代者。期望成为复制品的装置周期性地给服务发出特别的客户端请求。这种请求包括来自可接受的权限的授权发送器参与故障-中止服务的证明。状态包括从潜在复制品到状态机最后作出这样的请求的时间戳的映射。为了选择新状态机，状态机在那些时间戳充分近的装置中生成伪随机但确定的选择。

由于有许多可能应用本发明原理的实施例，应该认识到这里参考附图所描述的实施例只是为了说明，并不是对本发明的范围的限制。例如，本领域的技术人员将意识到图示实施例的软件中某些元件可以在硬件中实施，并且反之亦然，或者图示的实施例能够在排列和细节中改变，而不脱离本发明的精神。因此，这里描述的本发明预期的所有这种实施例同样可以在包含在下面的权利要求和其同等范围内。

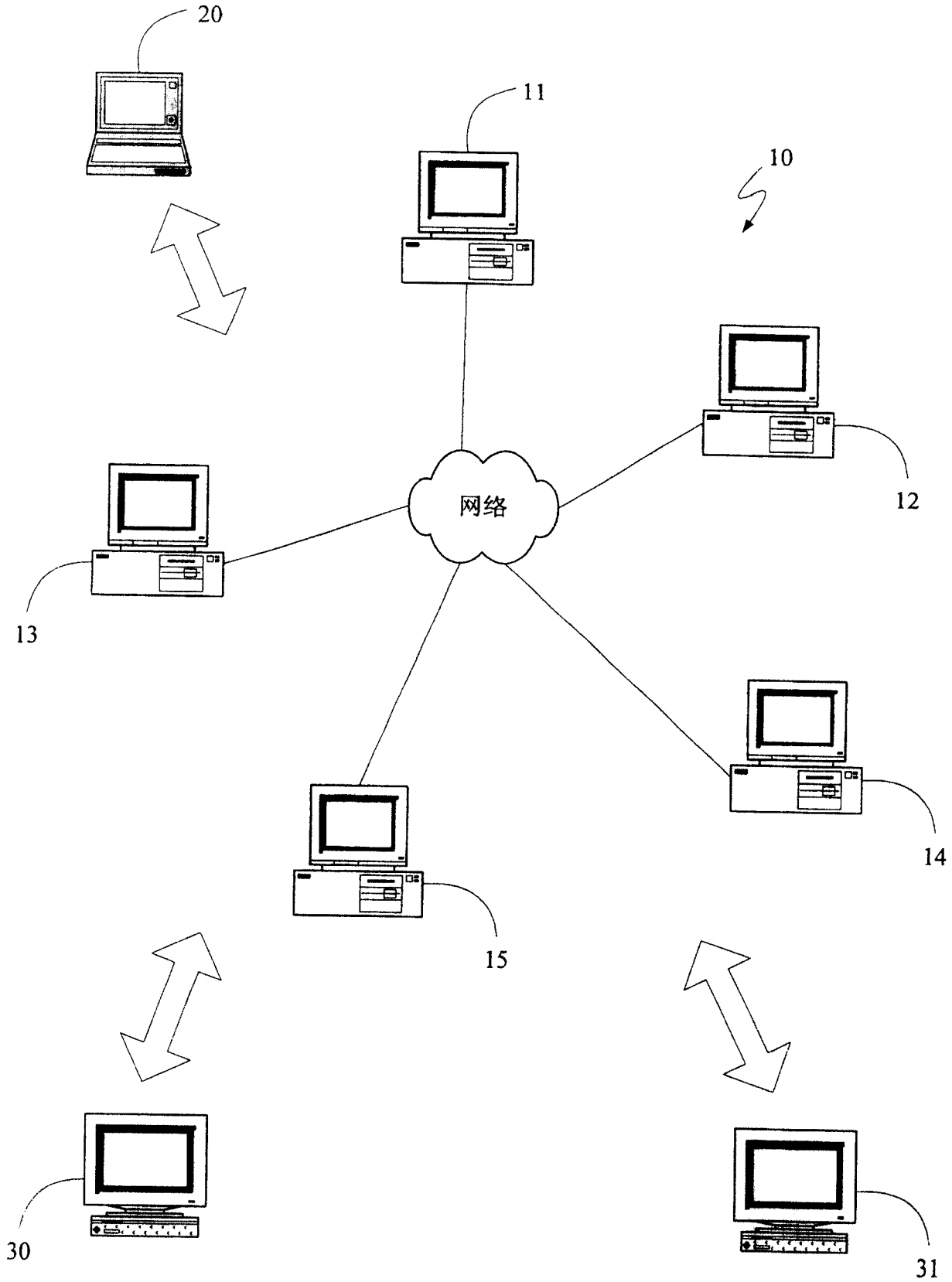


图 1

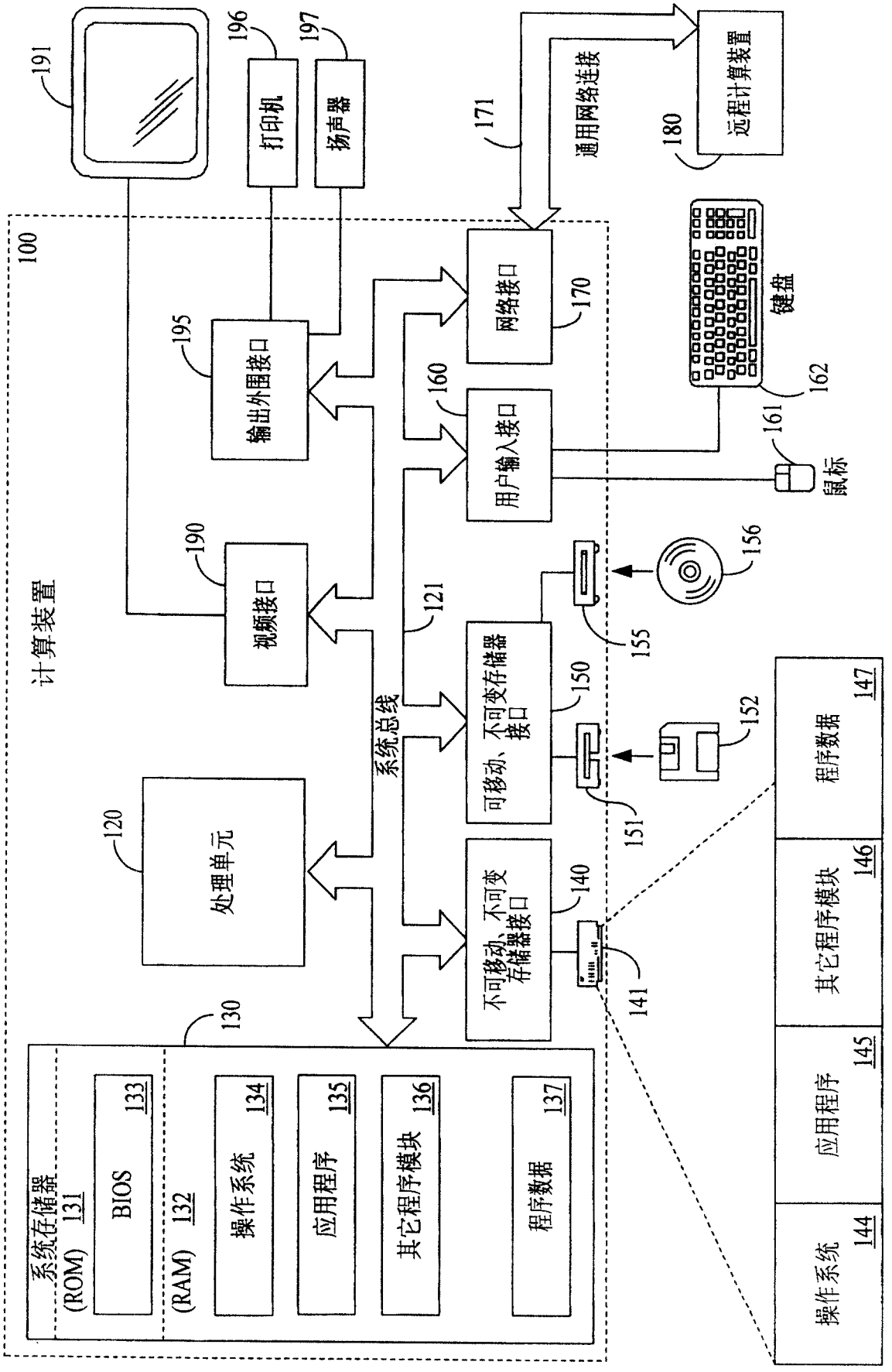

计算装置
100
191
196
打印机
197
扬声器
171
通用网络连接
180
远程计算装置
195
输出外围接口
网络接口
170
160
用户输入接口
161
鼠标
162
键盘
190
视频接口
121
系统总线
156
155
150
可移动、不可变存储器接口
120
处理单元
151
152
140
不可移动、不可变存储器接口
141
130
系统存储器
(ROM) 131
BIOS 133
(RAM) 132
操作系统 134
应用程序 135
其它程序模块 136
程序数据 137
操作系统 144
应用程序 145
其它程序模块 146
程序数据 147

图 2

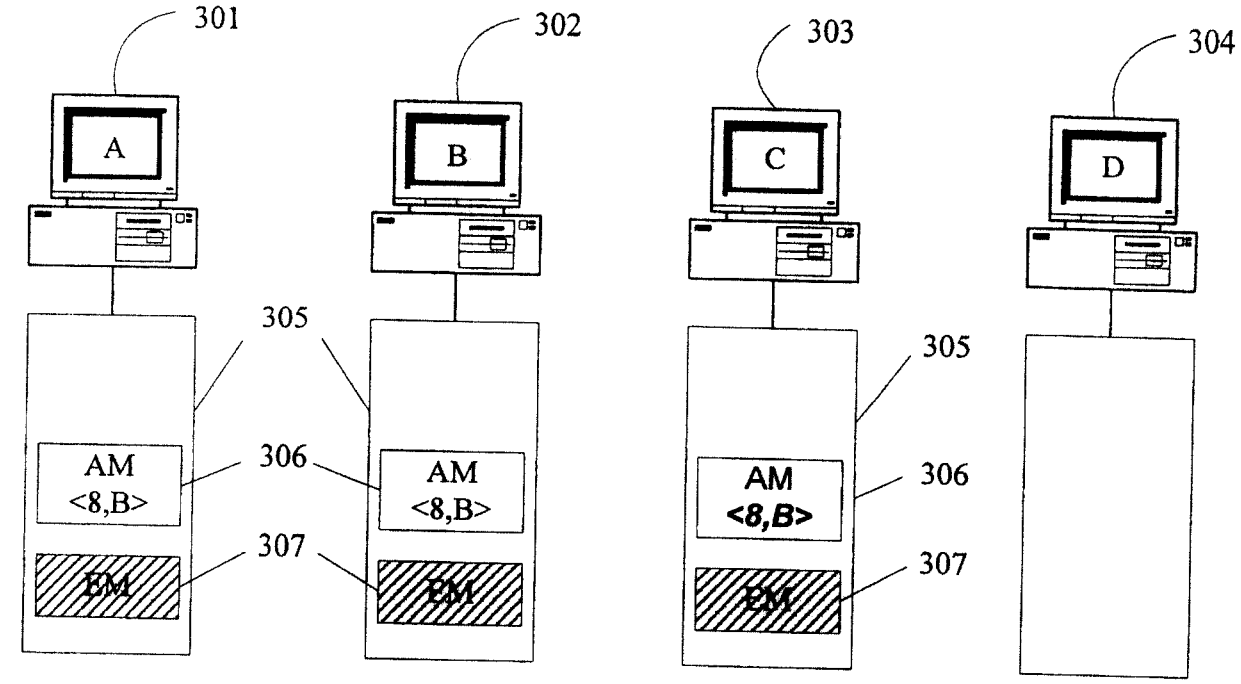
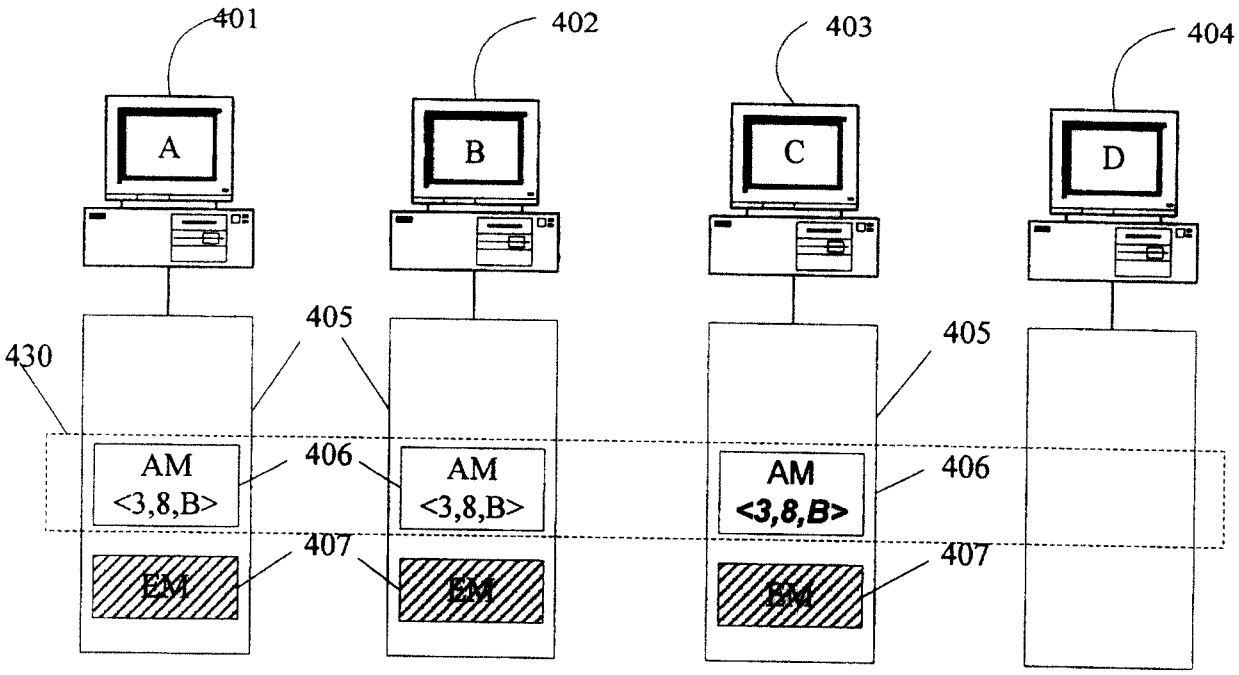

301
302
303
304
A
B
C
D
305
305
305
AM
<8,B>
306
AM
<8,B>
306
AM
<8,B>
306
EM
307
EM
307
EM
307
310
Op f
311
Op t
312
Op j
313
Op
W
314
Op x
315
Op c
316
Op g
317
Op f
318
Op
W
319
Op f

图 3


401
402
403
404
A
B
C
D
430
405
405
AM
<3,8,B>
406
AM
<3,8,B>
406
AM
<3,8,B>
407
EM
407
EM
407
EM

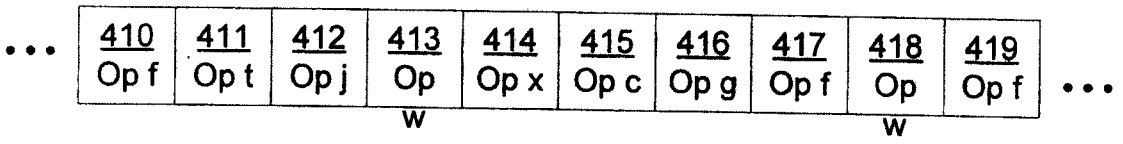

410
Op f
411
Op t
412
Op j
413
Op
W
414
Op x
415
Op c
416
Op g
417
Op f
418
Op
W
419
Op f

图 4a

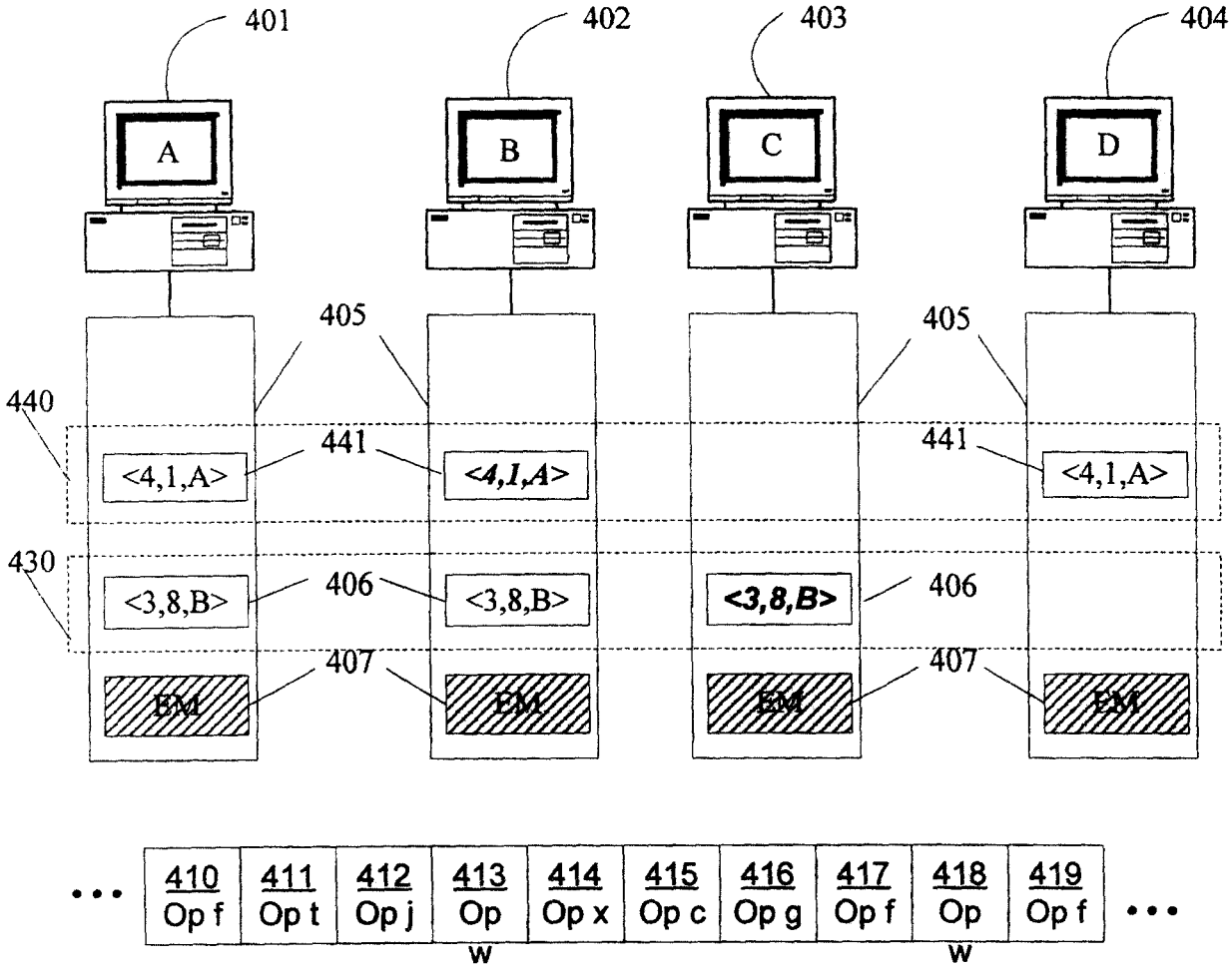

401
402
403
404
A
B
C
D
405
405
440
441
441
<4,1,A>
<4,1,A>
<4,1,A>
430
406
406
<3,8,B>
<3,8,B>
<3,8,B>
407
407
EM
EM
EM
EM
410 Op f
411 Op t
412 Op j
413 Op
414 Op x
415 Op c
416 Op g
417 Op f
418 Op
419 Op f
W
W

图 4b

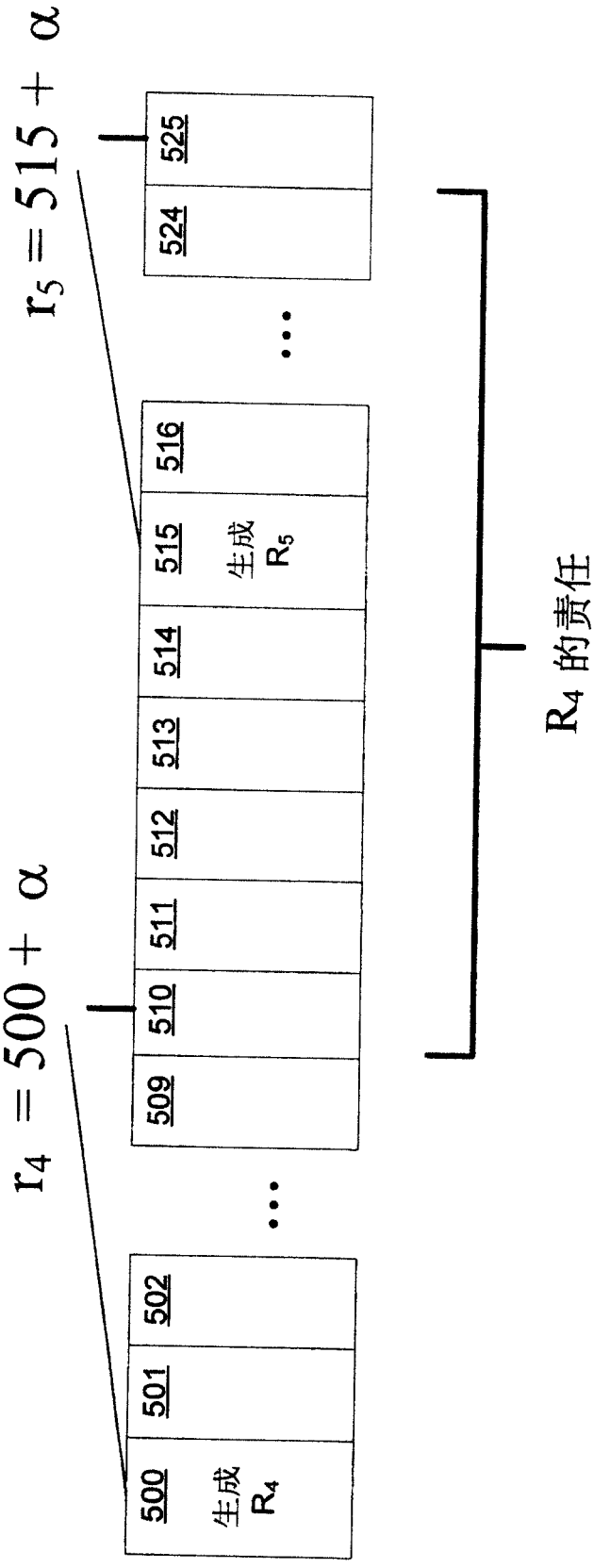
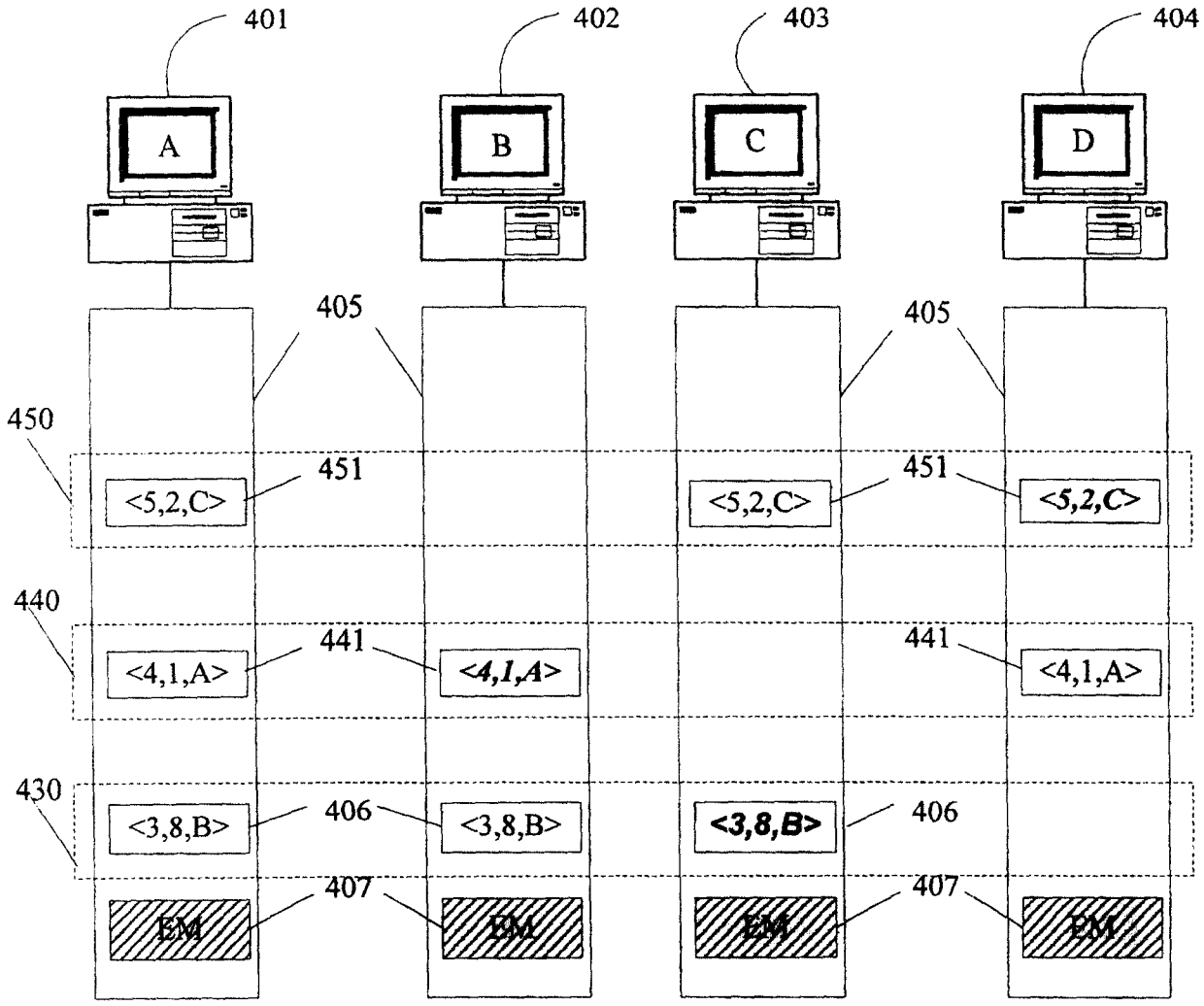

$r_4 = 500 + \alpha$
$r_5 = 515 + \alpha$
500
生成 R_4
501
502
509
510
511
512
513
514
515
生成 R_5
516
524
525
R_4 的责任

图 5


401
402
403
404
A
B
C
D
405
405
450
451
451
<5,2,C>
<5,2,C>
<5,2,C>
440
441
441
<4,1,A>
<4,1,A>
<4,1,A>
430
406
406
<3,8,B>
<3,8,B>
<3,8,B>
407
407
EM
EM
EM
EM

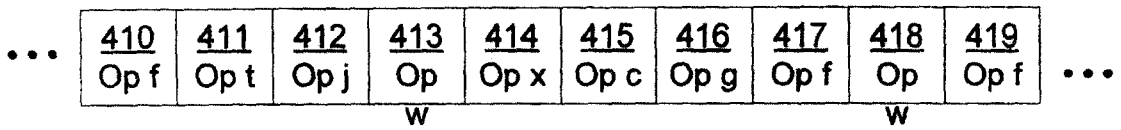

410 Op f
411 Op t
412 Op j
413 Op
414 Op x
415 Op c
416 Op g
417 Op f
418 Op
419 Op f
W
W

图 6

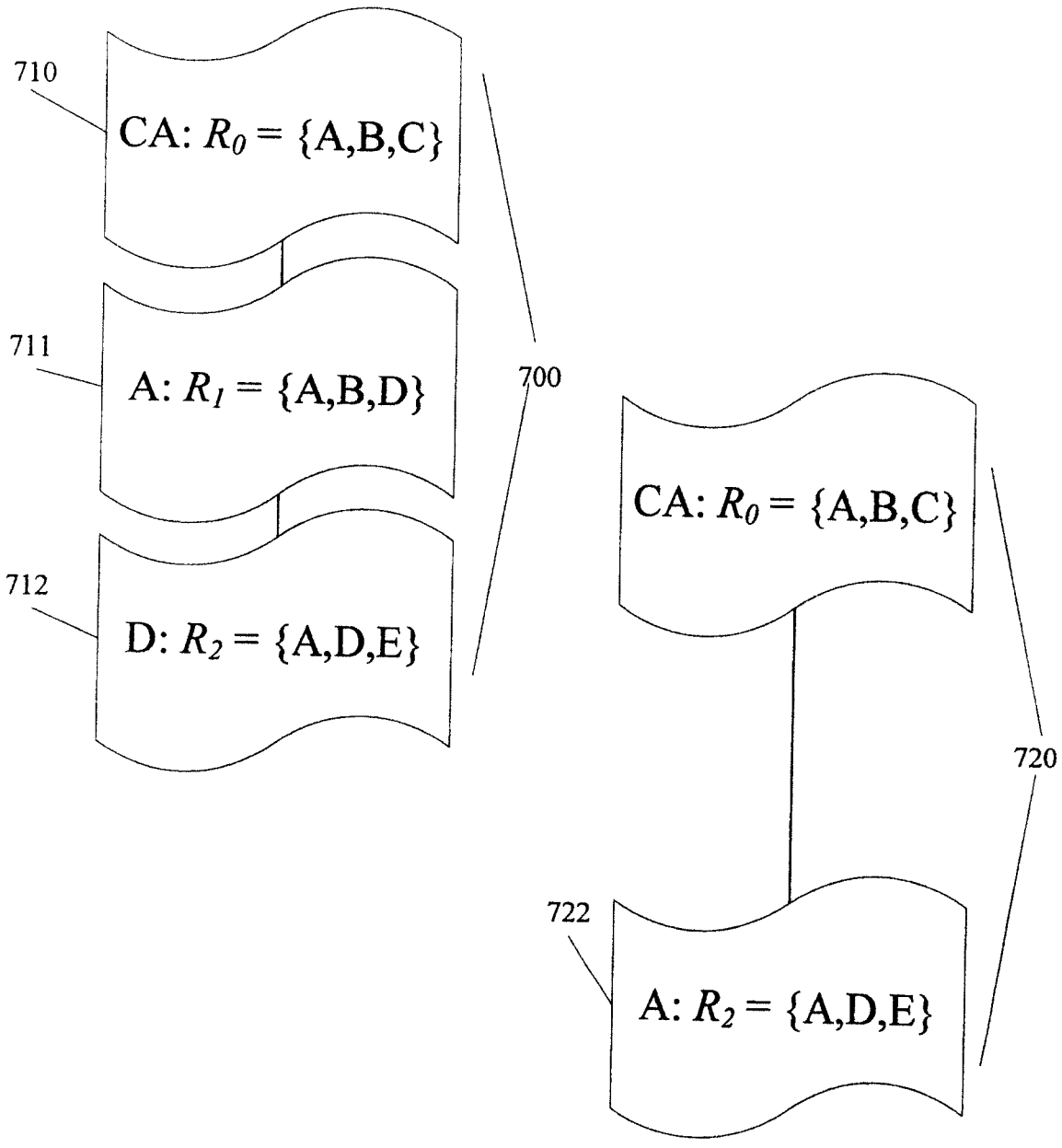

710
CA: R_0 = {A,B,C}
711
A: R_1 = {A,B,D}
712
D: R_2 = {A,D,E}
700
CA: R_0 = {A,B,C}
720
722
A: R_2 = {A,D,E}

图 7