(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 105659213 B
(45)授权公告日 2018.12.14

(21)申请号 201380080291.3
(22)申请日 2013.10.18
(65)同一申请的已公布的文献号
申请公布号 CN 105659213 A
(43)申请公布日 2016.06.08
(85)PCT国际申请进入国家阶段日
2016.04.15
(86)PCT国际申请的申请数据
PCT/US2013/065623 2013.10.18
(87)PCT国际申请的公布数据
WO2015/057240 EN 2015.04.23
(73)专利权人 株式会社日立制作所
地址 日本东京都
(72)发明人 O·基塞勒夫 G·保罗
C·杨沃斯
(74)专利代理机构 北京市金杜律师事务所 11256
代理人 陈伟 王娟娟
(51)Int.Cl.
*G06F 11/14*(2006.01)
*G06F 11/20*(2006.01)
(56)对比文件
CN 101005369 A,2007.07.25,
CN 101329691 A,2008.12.24,
CN 101616184 A,2009.12.30,
US 8401997 B1,2013.03.19,
审查员 赵天奇

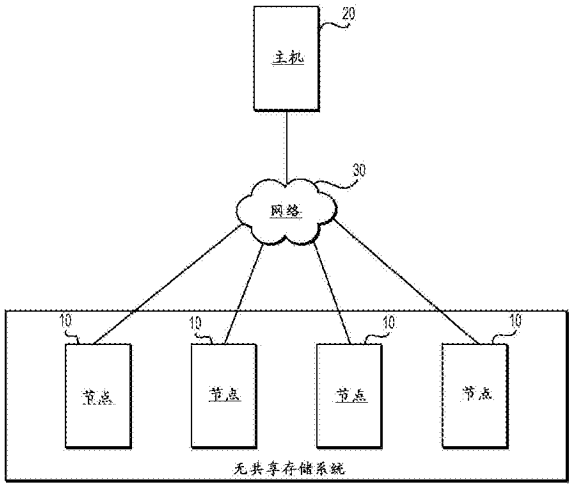
权利要求书5页 说明书20页 附图21页

(54)发明名称

无共享分布式存储系统中的目标驱动独立数据完整性和冗余恢复

(57)摘要

一种分布式无共享存储系统包括与网络和向在该存储系统上存储的数据发送I/O处理的主机相连的独立节点。每一个节点能够执行其自己的对部分写入数据的恢复和维持在存储系统上存储的数据的一致性。节点独立地计算跨节点的所有数据的位置并且独立地均衡数据，基于存储系统的冗余策略来维持一致性并且根据位置变化来迁移数据。如果节点判定其存储不完整或者损坏的数据，则该节点根据其他节点上的副本数据来重构其存储的数据。在节点之间的数据迁移期间，来自主机的I/O处理不被中断。




CN 105659213 B

1.一种存储系统，包括与一个或多个网络相连的集群中的多个节点(10)，每一个节点(10)包括用于存储数据和数据副本的一个或多个存储设备(67)；

其中，每一个节点(10)被配置为发送和接收输入/输出操作即I/O操作，并且每一个节点(10)被配置为基于指示所述数据的位置和数据副本的位置的位置信息(70;71)而意识到在所述多个节点(10)的所述存储设备(67)上存储的数据的位置和数据副本的位置；

其中，状态信息(100)被提供给所述多个节点(10)中的每一个节点，所述状态信息(100)指示每一个节点(10)的状态和每一个节点(10)上的每一个存储设备(67)的状态，在节点或节点的存储设备的激活或失活之后所述状态信息(100)基于对应的激活或失活而被更新和改变；

其中，基于所述状态信息(100)的变化，在激活包括新激活的节点或者具有新激活的存储设备的节点的情况下，每一个节点(10)被配置为判定是否需要恢复操作来维持所述多个节点(10)的所述存储设备(67)上存储的数据或数据副本的一致性，并且

其中，每一个节点(10)在需要时通过以下来恢复在其一个或多个存储设备上存储的数据：独立于其他节点而执行所述恢复操作以通过对存储受影响数据的其他节点发送和接收消息。

2.如权利要求1所述的存储系统，其中

数据作为数据副本或者纠删码段而被冗余地存储在所述集群中的多个节点(10)上，

数据副本或者纠删码段的数目取决于数据的冗余策略，并且

数据副本或者纠删码段的编码取决于数据的所述冗余策略和纠删码算法。

3.如权利要求2所述的存储系统，其中

每一个节点(10)被配置为接收事件的通知，其中所述事件是节点或是节点的存储设备的激活或失活。

4.如权利要求3所述的存储系统，其中

每一个事件被按照严格次序递送到每一个节点(10)，所述事件按照单调递增次序而被唯一且一致地编号，

保证每一个节点(10)接收所述事件的每一个事件，并且

保证在失活之后已激活的每一个节点(10)在其被激活时接收所述事件的每一个事件。

5.如权利要求3或4所述的存储系统，其中

所述状态信息(100)包括关于以下各项的信息：在每一个节点的一个或多个存储设备(67)上存储的数据的状态、在每一个节点的一个或多个存储设备(67)上的数据副本或纠删码段的状态以及其他节点的每一个节点的所述存储设备(67)上的数据和数据副本或纠删码段的状态，

其中，所述数据的状态、数据副本的状态或纠删码段的状态包括：

DIRTY标志，其在修改或写入所述数据之前针对数据、数据副本或纠删码段被设置，

DEGRADED标志，其在所述节点意识到对数据的写入或修改已经在少于所有所述副本或纠删码段上成功时针对数据、数据副本或纠删码段被设置，

指示数据在其之后被写入的事件编号的信息；以及

指示DEGRADED标志在其之后被设置的事件编号的信息，DIRTY标志是持久的并且DEGRADED标志是持久的，

当对数据的数据副本或纠删码段的至少预定法定数的写入已经成功完成时，DIRTY标志被清除，

数据和数据副本或纠删码段的位置可以被持久地存储或者根据所述数据的所述位置信息和元数据而被按需计算，并且

所述元数据包括唯一ID、数据的名称空间中的唯一名称属性或者对数据的内容的描述。

6.如权利要求5所述的存储系统，其中

每一个节点(10)所接收到的事件是在失活之后已被激活的第一节点的激活或者该第一节点上的存储设备(67)的激活，

所述判定基于节点之一是否基于所述位置信息在所述第一节点上存储了数据、数据副本或纠删码段，

所述第一节点被配置为接收包括DEGRADED标志、DIRTY标志、指示数据的最后一次修改或写入的事件编号的信息和指示DEGRADED标志在其之后被设置的事件编号的信息在内的信息的列表，

基于所述信息的列表，所述第一节点判定哪一个数据、副本或者纠删码段需要被重构，

基于所述状态信息(100)，所述第一节点判定哪些节点存储需要被重构的所述第一节点上的数据的所述副本或纠删码段，以在所述第一节点的一个或多个存储设备上重构所述数据、数据副本或纠删码段，

所述第一节点被配置为向被判定为存储需要被重构的数据的节点(10)请求将被重构的数据、数据副本或纠删码段的内容，写入要重构的数据的内容并且在完成之后向曾发送所述信息的列表的所述多个节点(10)的每一个节点发送消息，并且

如果判定在每一个节点(10)上恢复了所述数据的所述数据副本或纠删码段，则所述多个节点的每一个节点被配置为清除所述数据、数据副本或纠删码段的DEGRADED标志。

7.一种节点装置，其用于如权利要求1至6中至少任一项所述的存储系统中，所述节点装置(10)包括用于存储数据和数据副本的一个或多个存储设备(67)；并且所述节点装置(10)被配置为与一个或多个网络相连，以发送和接收输入/输出操作即I/O操作，并且以基于指示所述数据的位置和数据副本的位置的位置信息(70;71)而意识到在所述存储系统的所述多个节点(10)的所述存储设备(67)上存储的数据的位置和数据副本的位置，

其中，状态信息(100)被提供给所述存储系统的所述多个节点(10)中的每一个节点，所述状态信息(100)指示所述存储系统的每一个节点(10)的状态和所述存储系统的每一个节点(10)上的每一个存储设备(67)的状态；在节点或节点的存储设备的激活或失活之后所述状态信息(100)基于对应的激活或失活而被更新和改变；

其中，基于所述状态信息(100)的变化包括所述节点装置是被新激活的节点或具有新激活的存储设备的情况，所述节点装置(10)被配置为判定是否需要恢复操作来维持所述多个节点(10)的所述存储设备(67)上存储的数据或数据副本的一致性；并且

所述节点装置(10)被配置为在需要时通过以下来恢复在其一个或多个存储设备(67)上存储的数据：独立于所述存储系统的其他节点而执行所述恢复操作以通过对存储受影响数据的其他节点发送和接收消息。

8.一种计算机存储介质，存储有计算机程序，所述计算机程序被设置为使存储系统中

的一个计算机或多个计算机执行以下步骤，其中所述存储系统包括与一个或多个网络相连的集群中的多个节点(10)，并且每一个节点(10)包括用于存储数据和数据副本的一个或多个存储设备(67)，其中每一个节点(10)被配置为发送和接收输入/输出操作即I/O操作，并且每一个节点(10)被配置为基于指示所述数据的位置和数据副本的位置的位置信息(70；71)而意识到在所述多个节点(10)的所述存储设备(67)上存储的数据的位置和数据副本的位置，所述以下步骤为：

向所述多个节点(10)的每一个节点提供状态信息(100)，所述状态信息(100)指示每一个节点(10)的状态和每一个节点(10)上的每一个存储设备(67)的状态；

在节点或节点的存储设备的激活或失活之后基于对应的激活或失活来更新和改变所述状态信息(100)；

在每一个节点(10)处并且在激活包括新激活的节点或者具有新激活的存储设备的节点的情况下，基于所述状态信息(100)的变化判定是否需要恢复操作来维持所述多个节点(10)的所述存储设备(67)上存储的数据或数据副本的一致性；以及

在需要时在对应节点(10)处通过以下来恢复在其一个或多个存储设备上存储的数据：独立于其他节点而执行所述恢复操作以通过对存储受影响数据的其他节点发送和接收消息。

9.如权利要求8所述的计算机存储介质，其中

数据作为数据副本或者纠删码段而被冗余地存储在所述集群中的多个节点(10)上，

数据副本或者纠删码段的数目取决于数据的冗余策略，并且

数据副本或者纠删码段的编码取决于数据的所述冗余策略和纠删码算法。

10.如权利要求9所述的计算机存储介质，其中

每一个节点(10)接收事件的通知，其中所述事件是节点或是节点的存储设备的激活或失活。

11.如权利要求10所述的计算机存储介质，其中

每一个事件被按照严格次序递送到每一个节点(10)，所述事件按照单调递增次序而被唯一且一致地编号，

保证每一个节点(10)接收所述事件的每一个事件，并且

保证在失活之后已激活的每一个节点(10)在其被激活时接收所述事件的每一个事件。

12.如权利要求10或11所述的计算机存储介质，其中

所述状态信息(100)包括关于以下各项的信息：在每一个节点的一个或多个存储设备(67)上存储的数据的状态、在每一个节点的一个或多个存储设备上的数据副本或纠删码段的状态以及其他节点(10)的每一个节点的所述存储设备(67)上的数据和数据副本或纠删码段的状态，

其中，所述数据的状态、数据副本的状态或纠删码段的状态包括：

DIRTY标志，其在修改或写入所述数据之前针对数据、数据副本或纠删码段被设置，

DEGRADED标志，其在所述节点意识到对数据的写入或修改已经在少于所有所述副本或纠删码段上成功时针对数据、数据副本或纠删码段被设置，

指示数据在其之后被写入的事件编号的信息；以及

指示DEGRADED标志在其之后被设置的事件编号的信息，DIRTY标志是持久的并且

DEGRADED标志是持久的，

当对数据的数据副本或纠删码段的至少预定法定数的写入已经成功完成时，DIRTY标志被清除，

数据和数据副本或纠删码段的位置可以被持久地存储或者根据所述数据的所述位置信息和元数据而被按需计算，并且

所述元数据包括唯一ID、数据的名称空间中的唯一名称属性或者对数据的内容的描述。

13.如权利要求12所述的计算机存储介质，其中

每一个节点(10)所接收到的事件是在失活之后已被激活的第一节点的激活或者该第一节点上的存储设备(67)的激活，

所述判定基于节点之一是否基于所述位置信息在所述第一节点上存储了数据、数据副本或纠删码段，

所述第一节点接收包括DEGRADED标志、DIRTY标志、指示数据的最后一次修改或写入的事件编号的信息和指示DEGRADED标志在其之后被设置的事件编号的信息在内的信息的列表，

基于所述信息的列表，所述第一节点判定哪一个数据、副本或者纠删码段需要被重构，

基于所述状态信息(100)，所述第一节点判定哪些节点存储需要被重构的所述第一节点上的数据的所述副本或纠删码段，以在所述第一节点的一个或多个存储设备上重构所述数据、数据副本或纠删码段，

所述第一节点向被判定为存储需要被重构的数据的节点(10)请求将被重构的数据、数据副本或纠删码段的内容，写入要重构的数据的内容并且在完成之后向曾发送所述信息的列表的所述多个节点(10)的每一个节点发送消息，并且

如果判定在每一个节点(10)上恢复了所述数据的所述数据副本或纠删码段，则所述多个节点的每一个节点清除所述数据、数据副本或纠删码段的DEGRADED标志。

14.一种用于控制存储系统的方法，所述存储系统包括与一个或多个网络相连的集群中的多个节点(10)，并且每一个节点(10)包括用于存储数据和数据副本的一个或多个存储设备(67)，

其中，每一个节点(10)被配置为发送和接收输入/输出操作即I/O操作，并且每一个节点(10)被配置为基于指示所述数据的位置和数据副本的位置的位置信息(70;71)而意识到在所述多个节点(10)的所述存储设备(67)上存储的数据的位置和数据副本的位置，所述方法包括：

向所述多个节点(10)的每一个节点提供状态信息(100)，所述状态信息(100)指示每一个节点(10)的状态和每一个节点(10)上的每一个存储设备(67)的状态；

在节点或节点的存储设备的激活或失活之后基于对应的激活或失活来更新和改变所述状态信息(100)；

在每一个节点(10)处并且在激活包括新激活的节点或者具有新激活的存储设备的节点的情况下，基于所述状态信息(100)的变化判定是否需要恢复操作来维持所述多个节点(10)的所述存储设备(67)上存储的数据或数据副本的一致性；以及

在需要时在对应节点(10)处通过以下来恢复在其一个或多个存储设备上存储的数据：

独立于其他节点而执行所述恢复操作以通过对存储受影响数据的其他节点发送和接收消息。

15.如权利要求14所述的方法，其基于如权利要求8至13中至少任一项所述的计算机存储介质在所述存储系统中的一个计算机或多个计算机上被执行。

# 无共享分布式存储系统中的目标驱动独立数据完整性和冗余恢复

## 技术领域

[0001] 本发明涉及由通过网络相连的独立计算机系统(节点)的集群组成的无共享分布式存储系统。

## 背景技术

[0002] 在存储系统中的数据的迁移期间,存储系统中存储的数据必须被可靠地维持并且输入/输出(I/O)处理必须不被中断。对于写操作,例如,在迁移期间,存储系统必须可靠地跟踪数据对象状态。

[0003] 一种已知方法使用写标记,其中将被修改的数据区域在写入数据之前在公用/共享的"计分板"上被标记为例如"脏"标志。在该方法中,需要若干步骤,其包括记录写入数据的请求、向存储数据的每一个目标发送消息、等待写和响应、然后发送实际写操作。前述方法导致写操作的网络延迟的增加。

[0004] 另一种已知存储方法将整个高级数据存储区域标记为脏。然而,这种方法对于大量数据是不可行的,因为其需要数据的整个大聚合的恢复。已知的存储系统也可以在文件系统级别将文件标记为脏以指示修改。然而,文件系统级别处的标记导致标记数据具有过粗的粒度而无法对极大数据文件有效,这导致恢复需要过长时间段来完成。另外,在集中式数据库中将数据块标记为脏在本领域中-诸如在Parascale公司的向外扩展存储平台软件中-也是已知的。

[0005] 已知存储系统中的类似功能还包括例如在通过引用而被结合于此的美国专利第6,907,507、6,910,111、7,089,385和6,978,354号中描述的VERITAS卷管理器(VxVM)的快速镜像重新同步(FMR)特征。美国专利第6,907,507、6,910,111、7,089,385和6,978,354号使用多列位图、累积器地图和按照镜像的地图。针对从I/O错误的恢复,现有技术的存储系统(卷管理器和多拷贝文件系统)需要中央管理器要么通过直接读或者写数据来执行恢复,要么需要协调器来管理恢复过程。这种配置的缺点是中央管理的恢复在协调器经历故障时暂停,这导致恢复过程的进一步复杂化。此外,为了应对协调器故障的可能性,需要在共享存储装置中可靠地保持大量元数据。

[0006] 在部分写数据恢复的情况下,现有技术由许多使用中央数据库或者某种卷级位图的卷管理器实现方案所采用的镜像重连和镜像"重新同步(resilvering)"方法组成。其他实现方案使用从一个中央位置(所有卷管理器)进行直接读取和写入的中央恢复管理器或者具有用来驱动恢复的中央协调器(例如如在Parascale公司的向外扩展存储平台软件中一样)。

[0007] 当涉及在存储系统中添加或者移除节点或其盘的数据迁移的情况下,现有技术包括CEPH文件系统重布局特征,其基于可靠的散列和图生成。PICASSO和CEPH系统都使用通常称为"CRUSH"算法的放置算法来基于跨整个存储集群的存储配置的版本信息来确定性地计算数据块的正确放置。见Sage A.Weil;Scott A.Brandt;Ethan L.Miller;Carlos

Maltzahn;,"CRUSH:Controlled,Scalable,Decentralized Placement of Replicated Data,"Proceedings of the 2006 ACM/IEEE Conference on Supercomputing,p.31,2006,其通过引用而被结合于此。在CEPH系统中,重布局由中央元数据引擎执行。另外,在Parascale系统中,数据重布局由中央数据库驱动并且放置以自组织按块的方式完成。当Parascale系统中的重布局被中断时,数据布局被留在过渡但是一致的状态下,并且在恢复重布局过程之后,数据放置被重新计算。

[0008] 在对数据冗余做出策略改变的情况下,数据移动是中央管理的并且是从中央管理节点执行的。Parascale公司的系统具有对迁移的中央管理,其中被确定为新数据位置的位置被要求从现有存储位置"拉出"数据以满足冗余的变化。

## 发明内容

[0009] 本发明的实施例总地涉及计算机系统和控制所述计算机系统的对应方法,并且更具体地涉及用于无共享分布式存储系统中的数据管理和迁移的技术。下面描述的以下实施例是示例性的并且是这样一种性质以使得本领域技术人员将认识到本发明可以利用落入本发明的精神和范围的其他修改、替代构造和等价物来实现,并且还将认识到本发明的一个或多个实施例可以被组合在一起。

[0010] 在一个实施例中,一种在无共享分布式存储系统中维持数据完整性和冗余性和恢复数据完整性和冗余性的方法被提供。在无共享分布式存储系统中,存储系统包括多个独立系统(在这里称为节点),其被互连以形成存储系统(在这里称作集群)。集群中的每一个节点处理对文件的名称空间、元数据和位置信息的管理。另外,每一个节点被提供以存储区域,其独立于集群中的其他节点并且负责物理存储以及对在集群中的一个或多个文件系统中存储的文件的访问。因为每一个存储节点被提供以-无论是本地还是通过存储区域网络(SAN)等附接的-一个或多个持久存储设备,并且在正常情形下每一个节点仅可以访问其自己的存储设备,并且一般而言节点无法访问其他节点的存储设备,并且整个系统的存储因而分布在系统的各节点之间,因此该集群架构被称作无共享分布式存储系统。

[0011] 使用将存储状态和集群范围的活动信息考虑在内的算法,或者使用某一其他映射函数,数据可以跨集群中的存储设备而被分发。在示例性实施例中,伪随机、一致的散列(CRUSH算法)被使用,其使用集群中的存储的拓扑和负载因素并且使用与数据相关联的唯一"键"来确定集群内的数据的放置,以计算数据将被放置的位置。在示例性实施例中,数据被映射到离散、有限尺寸的容器(在这里称作"块")。通过向块或者块的分组的某一唯一属性施加CRUSH算法来放置这些块。

[0012] 为了提供对节点和存储故障的弹性,数据被冗余地存储。以当节点或者存储设备的子集发生故障时数据可以保持可访问和可恢复的方式将数据的拷贝或者数据的纠删码片段分发到集群中的多个位置。对于数据的可恢复冗余放置存在各种技术。示例性实施例利用了CRUSH算法的特征,其中CRUSH算法将计算跨集群伪随机分布的一组位置,受约束以避免在同一存储设备或节点上放置数据的多个拷贝。基于每一个对象来跟踪对象放置和对象元数据对集群施加大处理负荷。例如,具有数十亿对象的存储系统在基于每一个对象跟踪放置时遇到不合理的负荷。在示例性实施例中,块被聚集为更大的逻辑分组(在这里称作"一致性分组"或者"CG"),其中CG中的所有块的放置位置是使用相同的唯一属性来计算的。

在示例性实施例中，通过向CG的“名称”施加CRUSH算法来放置CG。CG通过跟踪作为多个对象的聚合的数据来解决前述问题并且进一步提供了聚合数据被冗余地存储在集群内的额外益处。另外，一致性分组具有减小计算量以及在存储和取回数据时系统必须跟踪的每对象元数据的数量的益处。例如，元数据可以包括与文件属性有关的一条或多条信息，诸如文件名、ID、权限、尺寸、类型、与数据何时被创建有关的信息(创建时间)、之后数据被写入的事件编号、之后用来表示数据状态的标志被设置的事件编号、数据的访问时间等。

[0013] 在一些其他实施例中，存储设备在本地被附接到每一个节点并且不可被集群中的其他节点访问。另外，数据处理在本地对每一个节点可用并且不可被集群中的其他节点访问。在这种存储系统中存储的数据在节点之间被复制以在节点中的一个或多个发生故障或者在本地附接到节点之一的存储发生故障的情况下提供数据可用性和弹性。在集群中的任何地方存储的数据可以经由节点间通信连接而从集群中的任何其他地方访问并且数据位置和数据访问结构对于创建、修改和读取在集群上存储的数据的客户端是不可见的。

[0014] 在一些实施例中，每一个节点的本地存储区域被配置为智能对象存储(在这里称作基于对象的存储设备或者OSD，但是不符合SNIA和T10OSD标准)。OSD类似于盘存储但是提供存储空间的更抽象视图。代替处理作为固定尺寸的数据块的读取和写入，OSD将数据组织为对象数据容器。每一个对象具有数据和元数据两者，诸如描述对象的属性信息。

[0015] 在一些实施例中，当诸如节点或盘故障之类的事故在集群中发生时，系统必须修复受影响的数据项。例如，在恢复的情况下，经受恢复处理的每一个节点负责对其自己恢复的管理。通过对数据修改操作的可靠跟踪、向所有节点通知存储集群及其中的数据状态和独立地确定是否必须对本地数据执行恢复操作以及这种操作需要什么，来完成恢复。在一些其他实施例中，如果集群的配置因为盘或节点已被添加或删除而改变，则数据有可能需要被再平衡或者重定位以利用新存储容量或者适应被减小的存储容量。在其他实施例中，如果冗余策略例如通过增加必须被存储的数据拷贝的数目而改变，则新数据拷贝必须被创建。这种发生被分组在这里称作“恢复”的类别下。本发明的一个目的是提供快速、高效和可靠的恢复，因为集群内的附加故障可能导致数据丢失，并且通过执行及时和高效的数据恢复，由附加故障引起的数据丢失的概率能够被大大降低。

[0016] 在其他实施例中，数据对象状态根据对作为缺失拷贝的数据对象的进行中写入和写入而被可靠地跟踪；通过使用从受影响节点的“拉出”方法而非“良好”节点向“坏掉”节点写入的“推送”方法，从由对同步复制数据的写入引起的I/O错误恢复；从由写入器的故障造成的对同步复制数据的部分写入恢复，而不产生全局状态依赖链；在存储和/或节点被添加或移除的情况下迁移数据；以及在数据冗余策略改变时添加和/或移除同步复制数据的冗余拷贝。

[0017] 在其他实施例中，分布式一致算法被实现在集群中以在节点之间协调事件和处理。一致算法的一个示例是在本领域中已知的Paxos算法家族。在Paxos算法中，节点中的任何节点都可以充当“领导者”并且试图提出用于供系统中的每一个节点执行的命令。每一个这种建议可以被与对应的建议编号一起发送以更容易地跟踪建议。这种建议编号无需和以下特定步骤有任何关系，该特定步骤中节点正试图对执行的命令取得一致意见。最初，领导者可以为领导者旨在提交的建议推荐建议编号。剩余设备中的每一个然后可以用对它们投票赞成的上个建议的指示或者它们尚未为任何建议投票的指示来回应领导者对建议编号

的推荐。如果，通过各种响应，领导者未了解到曾被设备投票赞成的任何其他建议，则领导者可以使用在早先消息中推荐的建议编号来建议设备执行给定客户端命令。每一个设备在该阶段可以决定是投票赞成该动作还是拒绝它。设备应当仅在其已经回应另一领导者对更高建议编号的推荐的情况下拒绝动作。如果称为法定数(quorum)的足够数目的设备投票赞成建议，则建议的动作被称为已经被达成一致意见，并且每一个设备执行该动作并且可以发送结果。以这种方式，Paxos算法允许每一个设备以相同次序执行动作，从而在所有节点当中维持相同状态。

[0018] 一般而言，Paxos算法可被认为按照两个阶段-如上所述的允许领导者了解曾被设备投票表决的先前建议的初始阶段，以及其中领导者可以建议供执行的客户端命令的第二阶段。一旦领导者已经了解先前建议，其无需重复第一阶段。相反，领导者可以不断地重复第二阶段，从而建议可以被分布式计算系统在多个步骤中执行的一系列客户端命令。以这种方式，虽然被分布式计算系统按每一个步骤执行的每一个客户端命令可以被看作Paxos算法的一个实例，但是领导者在为下一步骤建议另一客户端命令之前无需等待设备对给定步骤的所建议的客户端命令进行投票表决。

[0019] 在实施例中的一些中，Paxos算法被用来保证集群中的每一个节点以正确次序处理数据，并且保证每一个节点维持相同状态。以这种方式，遵循Paxos算法的单独节点可能发生故障，但是在故障期间在集群内将不会丢失操作。在实现Paxos算法的实施例中，单独节点中的任何节点都可以担当领导者并且试图向集群中的每个其他节点建议供执行的操作。通过要求每个节点以相同次序执行相同命令，Paxos算法在集群的节点当中实现同步。此外，如果集群中的节点发生故障或者以其他方式崩溃，则集群被使用Paxos消息通知。见Leslie Lamport的“Paxos Made Simple”ACM SIGACT News (Distributed Computing Column)32,4(Whole Number 121,2001 年12月)51-58，其通过引用而被结合于此。还见Tushar Deepak Chandra,Robert Griesemer和Joshua Redstone的“Paxos Made Live-An Engineering Perspective(2006Invited Talk)”Proceedings of 26th Annual ACM Symposium on Principles of Distributed Computing,ACM press(2007)，其通过引用而被结合于此。

## 附图说明

[0020] 图1是根据本发明一个实施例的集群无共享存储系统的例示。

[0021] 图2是根据本发明一个实施例的节点的框图。

[0022] 图3以框图例示出根据本发明一个实施例的一致性分组、块和片之间的关系。

[0023] 图4例示出根据本发明一个实施例的用于确定无共享存储系统上的数据对象的放置的存储配置地图的一个示例。

[0024] 图5例示出根据本发明一个实施例的由CRUSH生成的盘列表的一个示例。

[0025] 图6例示出根据本发明一个实施例的集群的所有节点的状态信息的一个示例。

[0026] 图7例示出根据本发明一个实施例的地图改变过程的高级流程图。

[0027] 图8例示出根据本发明一个实施例的每一个节点存储的事件列表的一个示例。

[0028] 图9例示出根据本发明一个实施例的图7中的地图改变过程的阶段一的过程的流程图。

[0029] 图10例示出根据本发明一个实施例的在阶段一期间接收迁移请求的节点的过程的流程图。

[0030] 图11例示出根据本发明一个实施例的图7中的地图改变过程的阶段二的过程的流程图。

[0031] 图12例示出根据本发明一个实施例的标志表的一个示例。

[0032] 图13例示出根据本发明一个实施例的块存根管理信息的一个示例。

[0033] 图14例示出根据本发明一个实施例的在节点的存储器中存储的位图的一个示例。

[0034] 图15和16例示出根据本发明一个实施例的迁移期间的WRITE请求的过程的流程图。

[0035] 图17例示出根据本发明一个实施例的节点在正常条件期间接收I/O操作的过程的流程图。

[0036] 图18例示出根据本发明一个实施例的在迁移期间的I/O操作完成之后将数据块设置为干净的过程的流程图。

[0037] 图19例示出根据本发明一个实施例的当在故障之后恢复在线时目标节点的过程的流程图。

[0038] 图20例示出根据本发明一个实施例的迁移期间的源节点故障的恢复的过程的流程图。

[0039] 图21例示出根据本发明一个实施例的图7中的地图改变过程的阶段三的过程的流程图。

[0040] 图22例示出根据本发明一个实施例的一致性分组的策略表。

[0041] 图23例示出根据本发明一个实施例的冗余级别策略的变化的过程的流程图。

[0042] 图24例示出根据本发明一个实施例的部分写入数据的节点在重新加入集群之后的恢复过程的流程图。

[0043] 图25例示出根据本发明一个实施例的部分写入数据的节点在重新加入集群之后的恢复过程的消息序列。

[0044] 图26例示出根据本发明一个实施例的用于处理(I/O)操作的接收的节点与主机之间的通信流程。

[0045] 图27例示出根据本发明一个实施例的用于处理I/O操作的节点与主机之间的替代通信流程。

## 具体实施方式

[0046] 在这里讨论的实施例例示出本发明的一个或多个示例。因为本发明的这些实施例是参考例示来描述的，因此对描述的方法和/或具体结构的各种修改或适应性修改对本领域技术人员可以变得显而易见。依赖于本发明的教导并且这些教导由此已经使本领域前进的所有这种修改、适应性修改或者变更都被认为在本发明的范围内。因此，本说明书和附图不应当以限制性的意义考虑，因为明白本发明绝非仅限于在这里例示出的实施例。除非另有指示，否则用相似的标号来引用相似的部分和方法步骤。

[0047] 介绍

[0048] 本发明可以被实现为网络附加存储(NAS)系统的架构基础并且可以被在任何分布

式数据平台中使用。一般而言，提供了一种无共享分布式存储系统，其如在图1中示出那样在与网络30相连的节点10上分发和存储文件系统中的数据对象。

[0049] 如在图1中示出，集群中的每一个节点10通过网络30相连。一个或多个客户端或主机20被连接到无共享存储系统。主机20一般是计算机，但是也可以是经由网络相连的另一存储系统或服务器。主机20例如向存储系统发送I/O操作。I/O操作例如是写操作(WRITE)、读操作(READ)、移除操作(REMOVE)或者截断(TRUCATE)操作。修改块数据的I/O操作(诸如WRITE)向存储该块的节点提供全局唯一的键。如在图2中示出，每一个节点10在存储器50中保持I/O键列表95，其保持其存储的每一个块的I/O键。此外，如果向宕机的节点10请求I/O操作，则该节点10在节点10恢复在线时将通过查询集群中的其他节点以寻找它们的最新I/O键列表来接收I/O操作。活动的节点10或盘65在这里被称作在线，而不活动的节点10或盘65在这里被称作离线。节点10中的任一个可以根据需要而被手动地使得离线或者在线，或者它们可以由于软件或硬件故障而离线。

[0050] 通常，主机20是计算机系统，诸如个人计算机(PC)、工作站、膝上型计算机、个人数字助理(PDA)、服务器、大型机等。主机20被配置为使用诸如NFS、CIFS、HTTP、FTP等文件访问协议来访问远程文件和文件系统。

[0051] 节点10可以是PC、工作站、服务器、大型机等；或者它们可以是具有充足的嵌入式计算能力来实现在这里描述的系统的存储设备，诸如磁盘阵列或者存储部件。节点可以存储与本地文件系统、网络附加存储(NAS)、存储区域网络(SAN)、数据库等上的文件系统中的文件相关联的信息。另外，节点可以由任何用来存储文件系统中的文件的硬件和/或软件元素构成，并且可以实现一个或多个用来存储文件的文件系统，诸如NTFS、EXT、XFS、GFS等。

[0052] 在集群中，文件系统本身可以横跨一个或多个节点或者整个集群，因为其未被物理绑定到特定节点。数据被存储在节点10上并且经由节点间通信连接可以被集群中的其他节点10访问。网络接口40和底层网络通信的技术方面不在本发明的范围内并且因而那些方面不被详细说明。在集群中的节点10之间的数据迁移期间，来自主机20的I/O处理不被中断。

[0053] 在存储系统中的节点10上存储的数据在节点10之间被复制以在节点10中的一个或多个发生故障或者附接到节点10的盘65发生故障的情况下提供数据可用性和弹性。所有节点10经由状态信息100而被通知其他节点10的状态以及其盘65和其他节点10的盘65上的数据的状态。纠删码方法可被在存储系统中使用。例如，在Reed-Solomon方案中，关于数据块的计算产生纠删码的块。许多其他合适的冗余或者纠删码的存储方案对本领域普通技术人员将是显而易见的。每一个节点10可以具有处理和存储能力两者。集群中的节点10可以存储与本地文件系统、NAS、SAN、数据库等上的文件系统中的文件相关联的信息。节点10可以被添加或者从集群中移除并且节点10上的盘65可以被添加或者移除以缩放用于服务在集群中存储的文件的容量和带宽。

[0054] 存储配置地图70、71如在图4中被所有节点10用来确定跨节点10的集群的存储系统上的数据的分布式存储的布局。当节点10或盘65被添加时，例如，地图改变建议处理被发起以提交基于集群中的节点或者盘的添加的新地图71。基于旧的和新的地图70、71，每一个节点10意识到每一个数据对象的放置或者位置。旧地图70和新地图71都可用于I/O处理并且可用于节点10中的布局参考。

[0055] 如在图4中示出的地图70、71是节点10和盘65或者卷的集合，CRUSH算法根据该集合并基于一致性分组ID(CGID 77)来确定数据块的放置。存储系统提供使用在图2中示出的逻辑器件(LDEV)66构成的一个或多个逻辑存储区域或者逻辑卷。存储设备67管理逻辑卷与LDEV 66之间的对应关系(关系)。盘65或者逻辑驱动器上的卷可被提供作为地图70、71上的CG的块的位置。相应地，地图70、71中的给定CG块的放置或位置也可以使用卷或者逻辑卷而非盘65来定义。

[0056] 每一个CG具有标识CG的唯一标识符CGID 77。在创建CG之后，创建节点10创建所创建的CG的CGID 77。地图70、71在确定数据存储系统上的数据对象的分布式存储的布局时被利用。CRUSH基于地图70、71来选择在节点10和盘65的集群内包含的CG和存储对象的放置。CG是一组诸如文件之类的数据对象，其由共享CGID 77的数据块组成并且因而具有节点10和盘65上的相同放置。数据对象是可与其他数据对象单独识别的有限长度的数据并且可被传送到存储系统并且可以是文本文件、图像文件或者程序文件等。CG及其对应复制品被跨集群中的多个节点10存储。图3示出了CG、块和片之间的关系的例示。如在图3中示出，CG由多个块组成并且每一个块是有限最大尺寸(例如，64MB)的数据区域。块为了可靠性而被存储在多个拷贝中或者是跨节点10纠删码的。每一个块可以被进一步划分为称为片的数据区域。每一个片可以是固定尺寸的。

[0057] CRUSH算法可以被用来根据地图70、71产生存储有数据的盘65的列表。任何节点10都能够执行CRUSH算法。节点10向CRUSH算法传递包括CGID 77和若干所需唯一放置位置在内的信息。CRUSH例如用盘65的显式有序列表来回应。CRUSH保证针对相同地图70、71内的相同CGID 77每一次产生相同序列的盘65。就是说，CRUSH可以返回3个盘{A,B,C}或者产生盘列表76{A,B,C,D}的4个盘的盘列表76。列表76中的盘65可能是可访问的或者它们可能由于软件或者硬件问题-例如，特定盘65所被物理附接于的节点10的崩溃-而是不可访问的。崩溃产生的列表76中的第一个可用盘65可以被称作“第一可用”盘(卷)。

[0058] 图4示出了旧地图70和新地图71的一个示例。旧地图70是包括先前地图70版本的CRUSH地图的一组节点10和盘65(卷)。而新地图71可以被定义为包括建议的新地图71版本的CRUSH地图的一组节点10和卷65。旧地图70在新地图71被节点10提交之前被用作数据对象的布局配置。换言之，地图70对应于用于集群中的数据的放置的当前地图，而地图71是集群中的数据的新建议放置。如在图4中的键中表示，节点10使用节点标识符-例如，编号1、2、3、4-来表示，并且盘65使用大写字母A、B、C、D、E..来表示。每一个盘65通过对(节点编号，盘字母)来标识。例如，1A表示节点1上的盘A。

[0059] 图5示出了CRUSH使用CGID 77“1”的旧地图70和新地图71生成的示例性盘列表76。如在图4和5中示出，根据CGID 77“1”的旧地图70的盘列表76包括1A、3C和4F。然而，根据新地图71，CGID 77“1”被存储在盘2B、3C和5A上。根据建议(新)的地图71，CRUSH为CGID 77“1”生成的盘列表76包括2B、3C和5A。

[0060] 节点10存储与文件系统、网络附加存储(NAS)、存储区域网络(SAN)、数据库等中的数据相关联的元数据。元数据包括文件属性，诸如文件名、ID、权限、尺寸、类型、与数据何时被创建有关的信息(创建时间)、之后数据被写入的事件编号、之后用于表示数据状态的标记被设置的事件编号、数据的访问时间等。如在图2中示出，每一个节点10具有对象存储设备(OSD)60、网络接口40、处理器45、管理工具(在这里被称作SA)55、存储器50和多个盘65。

网络接口40与主机和网络上的其他节点通信并且负责接收和回应来自主机20的I/O处理。此外，Paxos和CRUSH算法、地图70、71、元数据和位图75可以在存储器50中被配置和管理。

[0061] 每一个节点上的存储器被用来执行CRUSH和Paxos算法以及存储和管理与每一个CG及其块的元数据有关的信息、标志表120、I/O键列表95、状态信息100、位图75、块存根管理信息105、事件列表125和块拷贝列表115。

[0062] 每一个节点10具有独立的存储设备67。存储设备67包括多个存储驱动器或者盘65。每一个盘65例如是诸如SAS(串行附接SCSI)、SATA(串行ATA)、FC(光纤通道)、PATA(并行ATA)和SCSI之类类型的硬盘驱动器、半导体存储设备(SSD)等；或是呈现为通过光纤通道网络经由存储区域网络(SAN)远程访问的SCSI逻辑单元(LUN)或者通过TCP/IP网络访问的iSCSI LUN的存储设备；或是其他永久存储设备和机制。

[0063] 每一个节点10具有提供存储访问层(SA 55)的管理工具，存储访问层是主机20与之通信以执行I/O的层。SA处理存储的镜像和一致性。SA是分布式I/O系统的“发起者”部分。此外，SA 55能够向每一个节点10的OSD请求数据。每一个节点10的OSD 60处理节点10上的本地I/O并且对存储设备67执行操作。OSD 60例如在从存储设备67读取数据或者向存储设备67写入数据时执行与存储设备67的通信。OSD 60能够为CG确定其将该CG的块存储在哪一个盘65上。地图70、71是在每一个节点10的SA 55中创建的。每一个节点10也意识到地图迁移状态，其指示迁移当前是否正被执行。CG被用状态指示标记，状态指示向系统通知是否CG本身在正被恢复的过程中或者是否CG处于迁移状态。此外，SA 55管理地图70、71。SA 55层使用地图70、71，无论地图70、71是旧地图70还是新地图(建议地图)71。

[0064] 每一个节点10通过网络提供状态信息100，状态信息100被其他节点10接收到并被用来构建和维护节点10的集群的单独视图。图6是节点及其盘的状态(状况)信息100的一个示例，其包括关于集群10中的节点是否是活动的以及其盘是否可用于数据的存储的信息。此外，状态信息100包括每一个本地附接存储设备67中的单个盘65的可用性。本领域普通技术人员认识到状态信息100可以包括不同的属性或者可以被不同地定义，诸如更新状态或错误状态。在图6中示出的示例中，节点1是在线的并且盘A-C是在线的，而节点2的盘B、D和E是离线的。状态信息100是动态的并且被存储在每一个节点的存储器中。

[0065] 当附加的存储容量-诸如节点10上的盘65或者新的节点10-被添加到集群时，当存储容量被减小时-诸如盘65的移除或者节点10被移除(使得附接到给定节点10的所有盘65离线)，或者当盘权重因数被改变以使分配从重负载的节点10或盘65转移到未充分利用的节点时，地图改变必将发生以使得集群的I/O处理根据新地图71而被执行以反映集群配置的变化。地图改变将使得一些CG被映射到新位置(节点10/盘65)并且要求这些CG(其是CG中的所有块)的拷贝在新地图71被提交之前被重定位到其新位置。在根据地图改变的迁移期间，来自主机20的I/O处理不被中断。

[0066] 地图改变过程

[0067] 在图1中的存储系统的分布式无共享架构中，每一个节点10负责在没有集中协调或者监督的情况下执行其自己数据的恢复处理。另外，通过使用Paxos实现地图改变事件分发算法，每一个恢复事件被依次分发到每一个节点10并且每一个节点10接收已经发生的所有事件。

[0068] 图8例示出每一个节点10在存储器50中管理的事件列表125的一个示例。当每一个

节点10中的SA 55通过Paxos接收事件时，其按照时间被以顺序次序存储在事件列表125中。Paxos管理事件被递送以被处理的次序。当节点10恢复在线时，其查询其在集群中的对等体以获得当其离线时其可能已经错过的事件和任务的列表。节点接收事件并将它们存储在事件列表125中。如在图8中示出，节点1接收MAP_CHANGE_PROPOSAL(地图_改变_建议)事件，接着是MAP_MIGRATE(地图_迁移)事件和后续时间处的MAP_COMMIT(地图_提交)事件。相应地，节点10必须以顺序次序执行对应于事件的处理。节点10在当前阶段完整结束之前无法继续到如在图7中示出的下一阶段。图7中示出的三个阶段中的每一个将被详细说明如下。

[0069] 如在上面提到，当集群的存储容量例如因为向集群添加或者移除节点或者向节点添加或者移除存储设备而被改变时，CRUSH算法被用来确定用于跨集群放置数据的新地图。从现有地图(例如，旧地图)到建议地图(例如，新地图)的转变被称作地图改变过程。在地图改变过程期间，数据可以在一个或多个节点上被重定位以符合新地图。当数据被重定位时，重要的是数据仍然通过集群而对主机20可用并且I/O处理在转变期间不被中断。地图改变过程以三个阶段发生。

[0070] 图7示出了地图改变过程的抽象视图。一般而言，阶段一100由地图改变建议组成。在地图改变建议之后，每一个节点10为地图改变做准备并且每一个节点10回应建议。地图改变是由节点建议的或是由管理命令触发的。地图改变建议根据Paxos而被分发到集群中在线的所有节点10。每一个节点10必须通过结束任何正在进行的处理等来经历使自己为地图改变做准备的第一步骤。因为每一个节点10认为其自己结束了其阶段一的相应处理，因此其根据Paxos报告其已经接受了地图改变的建议。在所有“可用”节点(例如，在线并且响应的节点10)已经回应地图改变建议并且已经向建议节点表明地图改变是可接受的之前，地图改变的建议者无法继续进行到阶段二。如果任何节点10拒绝建议，则建议失败。响应于建议未能被集群中的所有可用节点接受，在进入地图改变过程的阶段二200之前，另一地图建议将需要被生成和接受。

[0071] 在另一实现方式中，宕机达最大允许时间以上的集群节点10可以被从配置中删除。最大时间量可以是预定时间量。节点10删除可能触发地图改变。在这种情况下，删除可能被地图改变建议器解释为使地图改变失败的原因。在地图改变过程的阶段二200中，根据Paxos发送一消息，该消息指示所有节点10继续进行要使地图改变发生所需的处理。就是说，阶段二触发集群中的块迁移和写代理。当所有可用的节点10根据Paxos发回报告它们已经完成实现新地图所需的其相应迁移处理时，阶段二结束。

[0072] 阶段三300由通过向所有节点10通知新地图71生效来“提交”地图改变组成。这使得所有节点10上的SA 55停止使用旧地图70并且开始使用新地图71。在第三阶段中，每一个节点10的OSD 60移除根据新地图71不再被存储的根据旧地图存储的数据。

[0073] 阶段一

[0074] 阶段一参考图9中示出的流程图来说明(PHASE 1)。在阶段一的开始，每一个节点10根据Paxos从地图改变的建议节点接收MAP_CHANGE_PROPOSAL事件(101)。这是阶段一消息。在图9中，建议的地图为了说明目的而指代新地图71。在每一个节点10接收到事件(101)之后，在每一个节点10中生成新地图并将其存储在节点的存储器50中(步骤105)。

[0075] 在步骤110处，基于新地图71，每一个节点10判定其是否存储任何受对新地图71的地图改变影响的CG。该判定是针对每一个CG做出的。如果当与根据旧地图70的位置相比时

根据新地图71对于CG存在位置变化，则节点存储的CG受新地图71影响。旧地图70与新地图71之间的比较是通过检查每一个CG的CGID 77并且比较对应于CGID 77的(旧地图70和新地图71的)盘列表76来完成的。对于受新地图71影响的每一个CG，每一个节点10遵循图9中的步骤。要么节点10没有受新地图71影响的CG要么节点10具有受地图改变影响的CG。如果节点10没有任何受影响的CG，则节点在步骤115处根据Paxos张贴其接受，即其已经接受的地图改变建议。如果地图改变影响已经在节点上的CG，则处理继续到步骤140。

[0076] 如果节点10将拷贝存储在旧地图70中的节点10的本地卷65上，但是根据新地图71不在本地存储它，则节点10张贴其对新地图建议的接受并且在新地图71被提交之前继续接受I/O操作。如果节点10根据旧地图70和新地图71两者都在其本地卷65中具有CG的拷贝，则节点10张贴其对新地图71的接受。节点10同样在(建议的)新地图71生效-其稍后将在阶段二和三中得到描述-之前继续接受I/O请求。

[0077] 在阶段一期间，节点10还判定其是否包含旧映射序列70中的第一“可用”卷。可用卷被称为节点10中的在线、响应并且能够将数据迁移到新位置的卷65。可用卷可以是数据被迁移到新位置的卷。可能存在多个可用的卷，但是一个被指定为第一可用卷。可用卷可以被节点确定为CRUSH基于CGID 77为CG产生的盘列表76中的第一盘65。

[0078] 在步骤140处，如果节点10判定其具有第一可用卷，则其是负责向根据新地图71的一致性分组的其他成员通知它们具有将根据新地图71存储的CG的节点10。具有第一可用卷的节点使用新地图71来识别CG块的新位置。节点10在步骤145处向作为根据新地图的CG的成员的节点发送CG_MIGRATION_REQUEST(CG_迁移_请求)消息，表明它们将从第一可用卷节点接收到CG。根据阶段一，对于将根据新地图存储CG的根据新地图71的节点成员，除非第一可用卷在步骤145处通知新成员节点10，没有其他方式知道它们是CG块的新位置。CG_MIGRATION_REQUEST由将被迁移的CGID 77的列表组成。此外，包括标志表120的对应于CG的元数据可以被与CG_MIGRATION_REQUEST一起发送。接收CG_MIGRATION_REQUEST的节点的OSD 60可以接收来自一个或多个节点10中的多个可用卷65的多个这种请求，如果该节点根据新地图将存储多个CG的话。OSD 60无需跟踪哪一个节点10发送每一个请求，因为节点10足以生成存储CG的位置的列表。

[0079] 图10例示出一流程图，其示出了在图9中的步骤145处接收CG_MIGRATION_REQUEST的节点10的处理。在步骤160处接收CG_MIGRATION_REQUEST的节点的OSD 60在步骤161处判定CG是否已经存在并且是否是活动的。在判定CG是否存在时，节点判定CG是否实际存在或者被标记为BEING_CONSTRUCTED(正被构建)的CG是否已经在OSD 60的存储设备内存在。如果CG没有与之相关联的错误并且其当前能够由主机写入和/或由主机读取，则CG是活动的。在步骤161处，如果没有已经创建的CG和/或其具有错误，则节点10继续进行到步骤165，其将创建根据新地图将被存储的每一个CG的存根和CG路径层级结构。如果步骤161处的判定是“是”，则处理继续进行到步骤175，这是因为节点10无需创建该CGID的CG存根，因为其已经在节点上存在。

[0080] CG层级结构还包括其必需存储的CG的CGID 77。在步骤170处，CG元数据被创建并被OSD 60持久标记为BEING_CONSTRUCTED。一旦CG存根层级结构被创建，节点就在步骤175处通过向曾向其发送CG_MIGRATION_REQUEST的节点发送ACK消息来确认迁移请求。ACK消息被第一可用卷节点(步骤150)接收到并且一旦节点接收到来自它已经向其发送迁移请求的

所有节点10的ACK消息，它就可以通过在步骤155处根据Paxos张贴对建议的接受来回应地图改变建议事件。当集群中的所有节点10已经接受地图改变建议时，阶段一成功地完成。如果任何节点10例如由于错误而未能接受建议，则建议失败并且建议者必须发出MAP_CHANGE_CANCEL(地图改变取消)事件。

[0081] 地图改变取消

[0082] 如果建议失败，则每一个节点10根据Paxos接收MAP_CHANGE_CANCEL事件。在MAP_CHANGE_CANCEL事件之后，地图迁移状态被清除并且每一个OSD 60检查其是否具有曾针对建议的新地图71构建的CG。当节点10接收到MAP_CHANGE_CANCEL事件时，曾构建的新地图71被无效。另外，预期根据新地图71接收CG而已经创建CG层级结构的节点10移除CG层级结构。在MAP_CHANGE_CANCEL事件被发送到所有节点10并且因为节点10崩溃或者以其他方式离线而并非每个节点10都接收到事件的情况下，未接收到事件的节点10在其恢复在线于集群时将根据集群的在线节点所维护的事件列表125而接收到消息。此外，在地图改变取消之后，SA 55停止向新地图71成员发送I/O操作请求。

[0083] 阶段二

[0084] 阶段二以根据Paxos而被发送到集群的节点10的地图迁移指示开始。在图11中的步骤201处，每个节点10接收到作为阶段二消息的MAP_MIGRATE事件消息。在步骤202处，节点10将地图迁移状态设置为“1”。地图迁移状态被设置以使得每一个节点10的OSD 60和SA 55意识到地图迁移已被发起。

[0085] 在步骤205处，每一个节点的OSD 60判定其是否具有任何被标记为BEING_CONSTRUCTED的CG。如果节点10没有，则在步骤210处节点立即根据Paxos张贴ACK消息以确认地图迁移事件。具有标记为BEING_CONSTRUCTED的CG的节点10中的OSD通过从根据旧地图70存储有属于标记为BEING_CONSTRUCTED的CG的块的可用卷(节点10)拷贝这些块来开始拷贝处理。在步骤220处，接收节点的OSD 60通过参考在阶段一中与CG_MIGRATION_REQUEST一起发送的CGID 77的列表而向具有标记为BEING_CONSTRUCTED的CG的CG的旧地图70的节点成员请求块的列表。旧地图70成员(例如，存储有这些块的节点)将块的列表(即，拷贝列表)115发送到新地图成员。

[0086] 可替代地，每一个新地图节点10(接收请求的节点)请求相同的旧地图节点10发送块的列表115和每一个块的片以利用底层文件系统的缓存效果。在一种情况下，将接收块(片)的每一个新地图70节点独立地决定其从根据旧地图的哪一个可用卷接收块。不同的可用卷可以被将接收相同块的每一个新地图节点选择。在另一种情况下，将接收块的每一个新地图节点选择从根据旧地图的相同可用卷节点接收块。在后一种情况下，由于将被发送到新地图节点的块(片)(根据前一盘读取)将位于选择的可用卷节点的存储器中并且因而如果将被传送的数据根据前一传送而已经在节点的存储器中则不需要盘读取，缓存效率可以被利用。例如，如果旧地图70节点1和3存储将被传送到新地图71节点2和5的相同数据，则节点2和5两者仅向节点1发送RECOVERY_READ(恢复_读)请求来传送数据。如果节点2向节点1请求数据，然后节点5向节点1请求相同数据，则节点1将无须执行盘读取来向节点5发送数据，这是因为数据将在节点1的存储器中。

[0087] 在步骤225处，接收节点10接收拷贝列表115并且OSD 60根据拷贝列表115将其必须处理的块的列表保持在存储器50中。节点10以顺序次序拷贝拷贝列表115中的块。优选的

是拷贝列表115大约对每一个CG在10K条目以下以避免用过大的拷贝列表来加负担于节点。在步骤226处，节点10为每一个块创建标志表120。

[0088] 所有节点10为存储在其中的每一个块管理作为存储在存储器50中的数据结构的标志表120。集群中的每一个节点能够访问在集群上存储的每一个CG的每一个块的标志表。图12例示出标志表120的一个示例。标志表120包含每一个块是否设置了CLEAN(干净)标志、DIRTY(脏)标志或DEGRADED(退化)标志的信息。此外，标志表120存储DEGRADED标志被设置之后的事件编号以及之后数据被写入的事件编号。被设置的标志具有值“1”，而值“0”表明标志尚未被设置。在步骤227处，节点10针对将被拷贝的每一个块设置DIRTY标志。如果节点10崩溃，则将通过扫描标记为DIRTY的块的CG来重新生成需要恢复的块的列表，其在下面将被更详细地说明。

[0089] 在步骤228处，节点10为其已经接收到的块的列表中的每一个块创建存根。图13是在存根中管理的块存根管理信息105的一个示例。每一个OSD 60针对标记为BEING_CONSTRUCTED的每一个CG的每一个块将块存根管理信息105存储在存储器50中。块存根管理信息105包括指示存根对应于的块编号的块字段106，以及指示块对应于的CG的CGID字段107。块存根管理信息105还包括块的旧地图位置108，以及指示该块的拷贝处理已经开始与否的RECOVERY_READ请求发送字段109。如果拷贝处理已经开始，则RECOVERY_READ请求发送字段被用“1”标记，否则其被用“0”标记。块存根管理信息105还在所有片拷贝字段110中指示是否块的所有片已被拷贝并且在键列表空字段111中指示I/O键列表是否为空。OSD 60检查位图75以确认是否所有片已被拷贝。如果所有片已被拷贝，则值被设置为“1”，否则值是“0”。OSD判定块的I/O键列表95中是否存在任何I/O键以指示I/O键列表95是否为空。如果I/O键列表95为空，则字段被用“1”标记。

[0090] 在图11中的步骤229处，节点10针对每一个块构建片的位图75。图14是位图75的一个示例。位图75是易失性的并且被存储在存储器50中。例如，对于32K片，描述64MB块的位图将是2048位或者说256字节。在步骤230处，新位置从旧地图70的成员节点10中选择源位置(可用卷)。新位置从CRUSH为CGID 77创建的盘76的列表中选择源卷(第一可用卷)65。例如，CRUSH在选择来自旧地图70的源OSD 60时提供一定程度的伪随机性并且可以被实现为使得所有新地图71目标将其RECOVERY_READ请求发送到相同的源。该处理利用底层文件系统的缓存来降低恢复对集群的处理负荷的总体影响。

[0091] 如在图3中示出，每一个CG由块组成，并且每一个块进一步由数据片组成。在数据迁移期间，通过传送块的组成片来迁移每一个块。在步骤235处，OSD 60将RECOVERY_READ请求发送到在步骤230处确定的可用卷以传送片。RECOVERY_READ请求是针对位于在步骤225中接收到并存储在存储器中的块的列表115中的每一个块的每一个片发送的。在步骤236处，块存根管理信息105中的RECOVERY_READ请求发送字段109被改变为“1”以表明该块的拷贝已经开始。

[0092] 在步骤245处，当块中的每一个片被传送并写入到新节点中时，对应于该片的位在位图75中被改变以表明其已被写入。位图75指示每一个块中的哪些片已被写入。在步骤240处，块的每一个片可以根据位图75按照顺序次序使用RECOVERY_READ请求而被拷贝并且被写到接收节点10的新位置盘65上。在图14中示出的示例性位图75中，在迁移期间，在片1被成功传送之后，片2被接着传送，通常遵循剩余片的顺序次序。图14中的位图75示出了对应

于每一个片的多个位。至少一个位表示片是否已被拷贝并写入到目标节点中。

[0093] 节点10检查位图75以判定相应块的所有内容是否已被拷贝(250)。节点判定块拷贝列表115中的每一个块的位图75是否表明所有片已被拷贝。在步骤251中,OSD将块存根管理信息105中的所有片拷贝字段110设置为“1”以表明该块的所有片已被拷贝。如果任何片尚未被拷贝,则在步骤235处发送对那些片的RECOVERY_READ请求。在步骤252处,块被从目标节点上的块拷贝列表115中移除。节点继续拷贝块拷贝列表115上的每一个块的片。在步骤253处,如果不是所有块已被拷贝,则处理返回到步骤228。当所有块的所有内容已被拷贝时,新位置(即,新节点)在255处发送确认阶段二事件的ACK消息。当所有节点根据Paxos发回报告它们已经完成它们的迁移工作时,阶段二结束。

[0094] 如在步骤227处示出,块被用标志表120中的DIRTY标志标记以表明该块正在例如被WRITE操作修改,或者该块将被拷贝。DIRTY标志是在块被写入或者以其他方式修改之前设置的。在SA 55已经向其发送WRITE操作的所有OSD 60已经回应WRITE之后,DIRTY标志例如被SA 55异步地清除。如果块被标记为DIRTY,则该块要么在块拷贝列表115上、在位图75、I/O键列表95上,要么在位图75和I/O键列表95两者上。当每一个块恢复完成时,其DIRTY标志被清除,这在下面被说明。

[0095] 在迁移期间,如果SA 55接收到I/O操作,则SA 55必须向根据旧地图70存储CG的节点10和根据新地图71存储CG的节点10两者发送针对CG中的特定数据片的I/O操作。该处理确保如果迁移块的内容例如被来自主机20的WRITE请求重写,并且通过SA 55而被处理到根据旧地图70的节点10,则块的完整性被保持。

[0096] 图15和16例示出根据本发明一个实施例的迁移期间的WRITE请求的处理的流程图。在图15中的步骤400处,节点10接收WRITE请求。在步骤405处,接收节点10的SA 55检查WRITE是否影响根据当前(旧)地图70存储的CG。如果SA 55判定WRITE影响其存储的CG,那么在步骤410处,SA 55检查地图迁移状态。如果SA55判定WRITE不影响其存储的CG,则处理返回到开始。如果节点10未处于地图迁移状态,则节点10向OSD 60发送WRITE请求并且WRITE被正常施加(425)。如果节点10处于地图迁移状态并且WRITE请求的CG受影响,则SA 55必须向根据旧地图70和新地图71两者存储CG的节点10发送WRITE请求。节点各自能够识别哪些CG处于迁移状态。在步骤415处,节点判定其是否具有根据旧地图70和新地图71两者存储在其本地存储设备67中的CG拷贝。如果其有,则只有一个WRITE请求被发送到OSD 60并且WRITE被施加(425)。WRITE被正常地施加并且处理继续进行到图17中的步骤800。否则,如果节点没有根据旧地图70和新地图71两者存储的CG拷贝,则WRITE请求在步骤420处被发送到根据新地图存储CG的节点并且处理如在图16中示出继续进行,这在下面被描述。

[0097] 图26例示出主机20与用于处理I/O操作的接收的节点10a和10b之间的通信流程。在图26中,主机20将对在集群中存储的数据的I/O操作的I/O请求2501发送到集群中的节点10a。在一些情况下,接收来自主机20的I/O请求的节点10a可以不存储与I/O请求有关的数据。如在图26中示出,在这些情况下,接收节点10a可以通过向主机20发送指定I/O操作的正确节点10b的重定向2502来将I/O请求重定向到正确节点10b。响应于重定向2502,主机20可以向作为用来处理I/O请求的正确节点的节点10b发送I/O请求2503。在完成I/O操作之后,节点10b可以向主机20发送响应2504。

[0098] 图27例示出对集群中的节点10a进行在集群中存储的数据的I/O操作的替代通信

流程。在图27中，主机20将对在集群中存储的数据的I/O操作的I/O请求2601发送到集群中的节点10a。在一些情况下，接收节点10a可以通过向I/O操作的正确节点10b发送重定向2602来将I/O请求直接重定向到正确节点10b。在完成I/O操作之后，节点10b可以向主机20发送响应2603。

[0099] 图17例示出在正常条件期间接收I/O操作的节点10的处理的流程图。正常条件可被定义为当地图迁移状态为0时，从而表明根据地图改变过程的迁移当前没有在节点中被处理。在步骤800处，节点10的SA 55接收I/O操作。

[0100] 在接收到I/O操作之后，接收节点然后将I/O操作发送到节点的OSD 60(801)并且OSD 60将操作添加到与I/O操作相对应的块的I/O键列表95(802)。在步骤805处，节点将块的标志表120中的DIRTY标志设置为“1”。在步骤810处由块上的OSD 60执行I/O操作。在步骤815处，操作被从对应块的I/O键列表95中移除。处理继续到步骤820，此时节点10判定块的I/O键列表95中是否存在任何其他操作。如果对于块存在其他操作，则处理继续进行到步骤810并且操作被执行。如果不存在任何其他操作，则对应于块的标志表120中的DIRTY标志被设置为“0”(825)。在替代方案中，处理可以从步骤815继续进行到步骤825，而没有在继续进行到步骤825之前在步骤820处检查是否存在任何其他操作。

[0101] 如果新地图71成员中的任何成员未能完成WRITE请求，则SA55利用标志表120中的DEGRADED标志将新地图71中的块标记为DEGRADED。此外，如果旧地图70成员中的任何成员未能通过WRITE，则旧地图70中的节点的SA 55将块标记为DEGRADED。块被标记为DEGRADED以表明在节点10知道并非块的所有拷贝(不是所有CG拷贝)在线并且响应时块正被写入。块被标记为DEGRADED，要么因为SA 55将I/O请求标记为产生DEGRADED状态，要么因为SA 55请求刚被写入的DIRTY块的DEGRADED标记并且SA 55知道一些镜像(例如，存储CG拷贝的节点)已经返回错误或者在WRITE期间崩溃，但是仍然存在足够的CG拷贝来形成用来恢复数据可靠性的法定数。将块标记为DEGRADED允许移除DIRTY标志并且不丢失关于在不可访问的拷贝再次变得可访问时-例如，当节点在故障之后恢复在线时-恢复块的需要的信息。此外，如果旧地图70成员中的任何成员未能通过WRITE，则对应于旧地图的节点的SA 55将块标记为DEGRADED。对应于旧地图的节点中的所有节点10在其节点10判定CG的所有拷贝都存在并且被恢复的情况下清除DEGRADED标志。

[0102] 在整个迁移处理中，旧地图70中的所有节点10-包括根据新地图71将不会存储数据的节点10-继续接受I/O请求。根据旧地图70节点的每一个节点10的SA 55将WRITE、TRUNCATE和REMOVE请求转发到新位置，如在上面参考图15说明的。SA 55与对旧位置的请求并行地将操作发送到相应新位置。

[0103] 图16例示出目标节点接收来自根据旧地图存储CG的拷贝的SA55的WRITE请求的过程的流程图。在步骤435处，新位置接收WRITE。在步骤440处，节点10检查块存根管理信息105以通过参考所有片拷贝字段110来判定块的所有片是否已被拷贝。如果块已经被从源拷贝(即，其是CLEAN)，则其在步骤445处被标记为DIRTY并且WRITE被立即施加到块(步骤450)。如果块被标记为DIRTY，则OSD 60在步骤455处检查位图75以判定WRITE所施加于的片是否已被拷贝。如果在步骤455中位图75表明片已被拷贝，则在步骤490处WRITE被施加于对应的片。如果写所施加于的片尚未被拷贝，则节点在步骤460处检查将被WRITE操作写入的数据是否小于片尺寸。如果将被WRITE写入的数据不小于或者等于片尺寸，则WRITE被施加

在适当片位置中(480)。位图75中针对该片的位被改变,以使得片在迁移期间将不会经历被拷贝到新地图位置(步骤485)。如果将被WRITE写入的数据小于片尺寸,则该片的RECOVERY_READ处理不按照位图75中的次序(即,不按照来自片次序的顺序)被发送,以使得该片的数据可以被迁移(470)。就是说,该片的RECOVERY_READ在按照根据位图75的顺序次序正常情况下将被下一个传送的片之前被发送。一旦该片的数据被迁移和拷贝(475),WRITE操作就被施加并且在步骤485处位图75中针对该片的位被改变以表明其已被拷贝。

[0104] 诸如TRUNCATE(截断)或REMOVE(移除)的其他修改操作被发送到根据旧地图70和新地图71两者的节点10。一般而言,WRITE请求的相同处理适用于TRUNCATE和REMOVE操作。截断处理取决于拷贝过程已经进行多远而不同。如果块尚未被拷贝,则TRUNCATE操作不是接着发生的操作。如果块已经被完全拷贝(其是CLEAN),则其在标志表120中被标记为DIRTY并且TRUNCATE操作被正常施加。如果块在存在TRUNCATE操作时是DIRTY(正被拷贝)并且位图75表明截断点过了当前拷贝偏移(尚未被拷贝),则TRUNCATE操作不是接着发生的操作。如果块在存在TRUNCATE操作时是DIRTY并且位表明块的该片已被拷贝,则块被锁定以防止从源的进一步拷贝。块然后根据TRUNCATE操作而被截断并且块不被从源进一步拷贝。

[0105] 如果在迁移期间,I/O操作是REMOVE,则块被从新位置移除并且被从存储器内结构中清除。在这种情况下,在块被移除之前OSD60等待直到块的待定拷贝(如有的话)被拷贝为止。如果块正被从CG中移除,则新地图成员也将其移除。

[0106] 在迁移期间的I/O处理期间,新位置的OSD以在正常WRITE操作中OSD 60将会做的相同方式保持跟踪请求SA的键。图18例示出根据本发明的在迁移期间的I/O操作完成之后将CHUNK设置为干净的过程的流程图。在步骤505处,当操作已被完成之后,OSD 60用ACK消息对SA做出回应(步骤510)。此后,OSD 60向旧地图70和新地图71的OSD两者中的节点10发送CLEAN指示(520)。在步骤525处,针对每一个块,每一个新位置从该块的I/O键列表95中移除I/O键。新位置节点10在将块设置为CLEAN之前检查以下:块的未决I/O键95的列表是否为空(步骤530),位图75中的位是否表明所有片已被拷贝(步骤535),以及块是否在块拷贝列表115上(步骤540),在替代方案中,在步骤535处,节点10可以通过参考块存根管理信息105来判定所有片拷贝字段是“1”还是“0”,以检查是否所有片已被拷贝。如果不是所有这些因素都得到满足,则块在步骤550处在标志表120中留为DIRTY。如果所有这些因素都得到满足,则该块的DIRTY标志被清除并且该块在步骤545处在对应于该块的标志表120中被标记为CLEAN。在步骤555处,节点执行使块被标记为干净所必需的处理。例如,如果在步骤530中判定块的I/O键列表不为空,则节点执行未决I/O操作。在处理完成之后,块被再次检查以将块设置为CLEAN(即,步骤530-540)。在判定是否要将块设置为CLEAN时可能考虑的因素还可以包括判定对指示法定数的至少预定值的CG的WRITE操作是否已被成功完成。块被留为DIRTY并且其在存在法定数时可以被恢复。在这种情况下,所有的CG拷贝及其块可以被检查,包括将Paxos事件编号考虑在内。换言之,系统验证当足够拷贝可用时块的所有拷贝是相同的。可替代地,当至少对CG的拷贝的法定数的WRITE操作已被成功完成时,DIRTY标志被清除。法定数是具有块/CG的CLEAN拷贝的最小数目的节点。

[0107] 一旦法定数被建立,则所有节点判定块中的任何拷贝是否具有DIRTY标志。如果存在具有DIRTY标志的块,则存储有DIRTY拷贝的节点选择可用卷并且发送RECOVERY_READ请求以恢复块。节点选择根据新地图71的可用卷成员。

[0108] 节点崩溃

[0109] 在迁移期间，节点10或者节点10上的盘65可能崩溃，这使迁移中断，并且在这种情况下，在迁移期间正在进行的WRITE操作可能是块在节点崩溃之前被处理的唯一部分。由于这种块在标志表120上是DIRTY并且CG在迁移期间被标记为BEING_CONSTRUCTED，这指示对应的数据内容是不可靠的，该情形被检测到并且是可恢复的。

[0110] 图19例示出节点在崩溃之后恢复在线时遵循的过程的流程图。当节点10或者盘65恢复在线时，在步骤560处节点判定哪些块被标记为DIRTY以及哪些CG被标记为BEING_CONSTRUCTED。在替代方案中，节点10可以参考块拷贝列表115来判定哪些块需要被拷贝，如果该列表在节点崩溃或故障之后可用的话。根据该判定，节点10在步骤565处创建将被拷贝的块的列表。节点从位图75中清除针对将被拷贝的块的位，即使某些片被指示为在崩溃之前已被拷贝(570)。此外，在步骤570处，块存根管理信息105被清除。节点10在步骤575处选择要从中拷贝块的合法源节点(可用卷)，并且包括新写入位在内的整个块将使用RECOVERY_READ请求而被从合法源拷贝。在步骤580处，处理继续进行到图11中的步骤235以向根据步骤575确定的可用卷发送RECOVERY_READ请求，并且根据遵循阶段二的步骤235的处理继续进行以拷贝数据。

[0111] 如果节点在正常操作期间(即，不在地图改变或者迁移期间)崩溃，则节点除了不如在图11中的步骤255中一样根据Paxos张贴ACK消息之外遵循上面的例程来恢复。如果节点在修改数据的I/O操作-诸如WRITE操作-期间崩溃，则节点将从确定的可用卷恢复新写入的数据。因此数据完整性被保持。

[0112] 合法源可以是来自恢复迁移的旧地图70的节点或是自身已经恢复到足以完成曾被中断的地图改变的新地图71成员。每一个发生故障的节点可以随后经历将块设置为CLEAN的处理，如上所述。

[0113] 地图迁移期间的源节点故障

[0114] 如果源节点10或者节点10上的盘65在迁移期间崩溃，则如在图20中示出的以下处理流程被发起。对于在崩溃之前未被完全拷贝的每一个CG，曾从发生故障的源节点接收块的新地图71的成员在步骤601处使用CRUSH来针对需要被迁移的CG生成存储有CG的根据旧地图70的盘列表76。节点然后在步骤605处选择列表中的第一可用卷。被确定为第一可用卷的卷65被称为“有效”拷贝。被确定为保持有效拷贝的节点根据旧地图70(包括已经崩溃并且现在恢复在线的相同节点)和新地图71两者来恢复对等节点。有效拷贝现在接收来自目标节点的RECOVERY_READ(例如，步骤235)。换言之，目标节点在旧地图成员当中选择将是数据传送的源的新可用卷。在其他方式中，如上所述的正常恢复处理是相同的。当在迁移期间崩溃的一个或多个节点10恢复在线时有效OSD恢复这一个或多个节点10。有效OSD通过检查对所有节点可用的状态信息100而意识到崩溃的位置恢复在线。

[0115] 根据本发明的该方面，源节点10要从故障恢复仅需要单个例程。该例程在生成对等列表时并且在确定哪一个OSD是有效OSD时检查单元的地图迁移状态以确定它们是否处于地图迁移的阶段二。

[0116] 判定为不可写入的CG

[0117] 如果目标节点10崩溃，或者其盘65崩溃，则存储CG的拷贝的节点10判定法定数是否不论目标节点的故障而被维持。当存储CG的拷贝的节点10或盘65的数目在法定数值之上

时，法定数被维持。法定数值是每一个节点10意识到的预定值。在替代方案中，如果包含CG的块的OSD 60的数目在预定法定数值之上则法定数也可以得到满足。如果根据新地图71的节点10维持法定数，则恢复继续。如果新地图71成员的数目落到法定数之下，则剩余的OSD 60完成恢复，但是CG可以被判定为不可写入的。如果CG被判定为不可写入的，则该条件在下一地图改变时被改变。如果法定数当前不可用，则CG被判定为不可写入的。

[0118] 如果源节点10崩溃或者其盘65发生故障，则只要存在可用的其他源OSD 60(如在上面说明)，迁移就继续。如果所有源拷贝都宕机，则目标节点10(接收节点)放弃迁移。在这种情况下，在目标上构建的CG拷贝将仍然标记为BEING_CONSTRUCTED并且在源拷贝恢复在线并被变得可用时如果可能则将被恢复。在这种情况下，被标记为BEING_CONSTRUCTED的CG拷贝除了恢复目的之外是不可读或者不可写的。

[0119] 阶段三

[0120] 图21例示出根据本发明一个实施例的阶段三的过程的流程图。在步骤301处，建议者检查所有节点10是否已经如在步骤255处所示那样通过生成确认来确认阶段二。当所有节点10已经确认阶段二时，阶段二完成。建议者发起地图改变的阶段三，其是建议地图71的提交。在步骤305处，地图改变的MAP_COMMIT事件消息根据Paxos而被发送到集群中的所有节点。在步骤310处，所有已经接收到提交消息的OSD评估它们存储的CG。在步骤315处，每一个OSD对照新地图71来检查其存储的每一个CG，以针对其存储的每个CG来检查节点10是否是根据新地图的CG的恰当持有者。

[0121] 如果节点10判定其存储的CG根据新地图71未被指示为由该节点存储，则节点10在步骤320处用永久CG_INVALID_PLACEMENT标志来标记CG并且在步骤330处发起对该CG中的所有块的异步移除和空间回收。为该CG存储的所有块都被从节点10中移除。在块的移除之后，在步骤335处，CG层级结构和块存根管理信息105被移除并且CG被从对应于OSD 60的缓存中进一步删除。如果OSD发现其根据新地图71正确存储CG，则其从CG中清除BEING_CONSTRUCTED标志，如果该标志在步骤325处被设置的话。在步骤340处，新地图被提交并且每个SA现在使用新地图并且每个节点现在根据新地图71的数据放置来接受I/O操作。

[0122] 冗余策略变化

[0123] 本领域技术人员将认识到冗余级别在存储系统中可被改变。例如，存储系统的冗余策略可以被设置在RAID1-RAID6之间并且此后可被改变。冗余级别可以在集群中被反映如下并且使用RAID1的不同镜像计数针对策略变化来描述。

[0124] 镜像(拷贝)策略可以例如通过管理命令来改变。地图改变建议也可以基于冗余级别的变化而发生。不论策略变化是增加还是减少，新地图71被创建。在增加镜像数目的情况下，因为冗余CG拷贝的数目必须根据新地图的布局而被分布在集群中，因此地图改变发生。在减少镜像数目的情况下，地图改变发生以从集群中的节点中移除CG的拷贝和在集群内重新分布剩余数据的放置。

[0125] 图22是示出与每一个CG相关联的镜像计数的策略表85的一个示例。在该示例中，针对每一个CG指定镜像计数，但是也可以针对整个集群或者其子部分来指定镜像计数。镜像计数例如是整数3，并且可以由管理员指定。此外，可以由存储系统为集群中的所有CG设置镜像计数缺省值。

[0126] 根据CRUSH序列，在序列的结尾处添加和移除盘65(卷)。就是说，如果CRUSH被要求

3个盘，则其产生盘列表76{A,B,C}。如果CRUSH随后被要求4个盘，其将产生盘列表76{A,B,C,D}，而要求2个盘将产生盘列表76{A,B}。无论改变的性质如何，存储系统被保证要么列表的第一可用成员曾在先前盘列表76中，要么在来自先前CG列表的盘65变得可用之前数据是完全不可访问的并且对块不能做任何操作。

[0127] 图23中的流程图示出了存储系统中的策略变化的过程。管理员增加或者减小CG的镜像计数。新镜像计数被存储在策略表85中。为了使策略变化发生，POLICY_CHANGE(策略_变化)事件消息根据Paxos被发送到集群中的节点10。POLICY_CHANGE事件消息包括策略变化是CG的镜像计数的增加还是减小。该消息在步骤701处被集群中的OSD 60接收到。在步骤705处，节点判定其存储的哪些CG受策略变化影响。在步骤710处，对于受影响的每一个CG，节点基于变化构建新地图71。如果策略变化是增加，则包括相应CG的副本拷贝的新位置的新地图71被构建。如果策略变化例如使镜像的数目增加2，则新地图71将具有CG的副本拷贝的两个新位置。根据新地图，为了符合策略变化而生成的CG的新拷贝将被存储在集群中在策略变化之前未曾存储CG的位置处。如果策略变化是减小，则新地图71(根据旧策略与新策略之间的镜像计数的差异)将不会包括根据旧地图70曾被包括的CG中的一个或多个副本拷贝。

[0128] 在步骤715处，节点判定策略改变是减小了CG的拷贝的数目还是增加了CG的拷贝的数目。如果策略减小了拷贝的数目并且OSD60在步骤720处发现根据新地图不再需要存储CG的拷贝，则其在步骤725处发起从其本地存储中对CG的异步移除。如果在步骤720期间节点判定其根据新地图71存储CG的拷贝，虽然其他拷贝可以被从集群上的其他地方移除，其继续进行到图9中的步骤135(即，在阶段一中根据Paxos张贴接受)。

[0129] 然而，如果策略增加拷贝的数目，则来自当前(旧)地图70的节点10的OSD 60取决于盘列表76根据旧策略是否可用而向根据新地图71存储CG的所有成员或者仅向根据新地图71为新的节点10通知那些节点现在必须存储CG的拷贝。在步骤740处，CRUSH所产生的盘列表中的第一可用卷被确定为如下节点10，该节点10将向根据新地图71的所有目标节点成员通知它们具有要存储的CG。通过向根据新地图71的每一个目标节点10发送CG_MIGRATION_REQUEST而在步骤745处随后通知新成员。在步骤750处，过程继续进行到地图改变过程的阶段一的步骤150。

[0130] 当特定CG被存储于在针对策略变化产生列表时不可用的盘65上的情况下，这种CG将不会被可用的任何节点发现并且将不会经历策略变化恢复。然而，当目标节点10恢复在线并且该节点此时将经历恢复处理时，策略变化事件将被记录并且根据Paxos而对存储CG的OSD 60可用。

[0131] 本发明的存储系统跟踪盘65和节点10故障。如果在预定时间段之后，曾离线的一个或多个盘65或者一个或多个节点10未恢复在线时，它们将被从地图配置中删除并且新地图71将被建议。在这种情况下，新地图71触发根据上面描述的地图改变过程的到新地图71中的数据迁移和数据复制。结果，冗余策略所指定的冗余级别将被恢复。预定时间可以基于使后续故障-其将导致某些数据的所有拷贝的全部丢失-的概率最小化的考虑。系统管理员也可以发出地图改变命令。

[0132] 部分写入数据的恢复

[0133] 以下例程的一些步骤的细节已被先前描述并且在下面不被重复。每一个节点接收

到对集群中的事件的通知。事件可以是盘或者节点的故障，或是盘或节点重新加入或者被添加。集群中的每一个节点在事件发生的通知之后评估其存储的CG以判定其是否存储受事件影响的CG的任何拷贝。节点然后基于事件发生来判定其是否需要采取动作来重构或者拷贝受影响的CG的块。图24例示出根据本发明一个实施例的在重新加入集群之后部分写入数据的节点的恢复过程的流程图。图25例示出根据本发明一个实施例的在重新加入集群之后部分写入数据的节点的恢复过程的消息序列。

[0134] 在步骤900处，节点或者盘65(逻辑卷)在其已经离线之后重新加入集群。如果盘未能通过I/O操作并且重试或恢复I/O操作也已经失败，则该盘可被声明离线。节点/盘(卷)恢复在线并且处于活动状态，该节点10被称作重新加入的节点。所有节点通过通知和对状态信息100的参考而意识到节点10重新加入集群901。在步骤905处，包括重新加入的节点在内的所有节点根据当前地图判定其是否具有任何存储在重新加入的节点上的CG。对于其判定具有重新加入节点上的拷贝的每一个CG，节点通过检查标志表120来判定该CG的任何块是否设置了DEGRADED或者DIRTY标志。

[0135] 在步骤910中的“是”判定之后，在步骤915处，节点10计算设置了DEGRADED或DIRTY标志的块的列表115并且将该列表发送到重新加入的节点。重新加入的节点在步骤915处接收来自多个节点的列表。每一个节点在步骤915处发送的列表包括与每一个块有关的信息。该信息是根据标志表120来确定的。列表中的信息例如包括块是否设置了DEGRADED标志或者设置了DIRTY标志、之后块被对块的WRITE操作写入的事件编号以及之后DEGRADED标志被设置的事件编号。事件是来自Paxos的消息并且事件编号是与向集群通知设备故障的消息相关联的顺序号。因此，如果事件是盘故障，则消息是以比WRITE被执行的编号更高的顺序号经由Paxos来发送的。因此，与DEGRADED状态相关联的Paxos顺序号允许系统明确地判定哪些拷贝是陈旧的。

[0136] 在915中的列表的接收之后，重新加入的节点针对每一个具有需要被恢复的块的CG判定根据当前地图的第一可用卷。节点基于列表中的信息来判定第一可用卷。根据列表中的信息，节点可以判定哪一个节点存储块的最新拷贝以及哪些拷贝足以重构CG的块。在判定可用卷之后，重新加入的节点向需要被恢复或重写的每一个节点的可用卷发送RECOVERY_READ请求。在步骤930处，重新加入的节点接收数据并且基于其他节点上的块的其他拷贝来重写块或者重构块。在步骤935处，重新加入的节点针对其根据当前地图存储的CG来判定是否所有数据已被恢复。如果重新加入的节点尚未恢复所有数据，则其发送RECOVERY_READ请求直到所有块被重构为止。

[0137] 当重新加入的节点在步骤935处判定所有块都被重写或者重构时，重新加入的节点在步骤940处向曾发送信息列表914的所有节点发送消息以向它们通知恢复完成。在步骤945处，所有节点判定是否块的所有拷贝都存在并被恢复。如果存储块拷贝的每一个节点和块是在线的并且如果所有块拷贝被确定为是相同的并且与具有与其DEGRADED状态相关联的最高事件编号的块相同，则所有块都存在并被恢复。如果所有块都存在并被恢复，则包括重新加入的节点在内的所有节点清除标志表120中的每一个块的DEGRADED标志(步骤955)。在步骤945中，如果节点在其判定期间发现其所有的块不存在并且未被恢复，则该节点计算需要被恢复的块的可用卷并且将RECOVERY_READ请求发送到可用卷以恢复数据。

[0138] 在替代方法中，在所有节点在步骤905中判定它们在重新加入的节点上存储CG拷

贝之后，每一个节点将受影响CG的列表和关联元数据发送到重新加入的节点。重新加入的节点然后计算CG的列表并且将其自己的CG列表和元数据发送到所接收的位置列表中的所有节点。在接收到CG列表和元数据之后，包括重新加入的节点在内的每一个节点检查接收到的CG元数据并且判定其存储的CG拷贝是否是在其他拷贝上存在的缺失数据，或者该节点的拷贝是否需要来自其他拷贝的恢复。如果其需要来自其他拷贝的恢复，则节点遵循在图24中描述的从步骤920起处理前进的例程以恢复数据。

[0139] 如根据在此提供的前述公开和教导可以认识到的，本发明的实施例可以以软件或硬件或其组合以控制逻辑的形式来实现。控制逻辑可以作为适用于指示信息处理设备执行如在本发明的实施例中公开的一组步骤的多个指令而被存储在信息存储介质中。基于在此提供的公开和教导，本领域普通技术人员将认识到用来实现本发明的其他方式和/或方法。

[0140] 以上描述是例示性的而非限制性的。在审阅本公开之后，本发明的许多变形对本领域技术人员将变得显而易见。本发明的范围因而不应当参考以上描述来确定，而是应当参考未决的权利要求连同它们的完整范围及其等价物来确定。

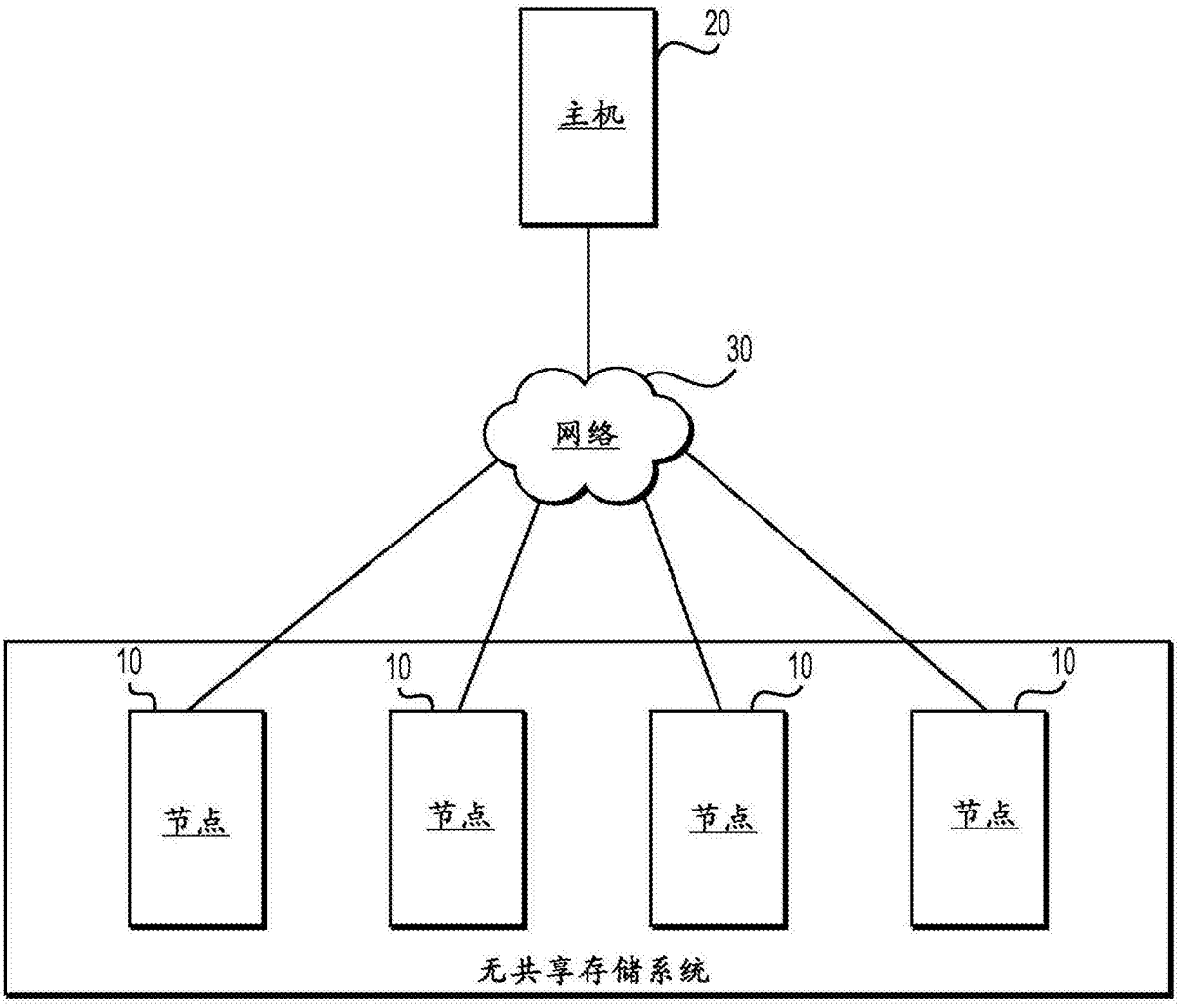



图1

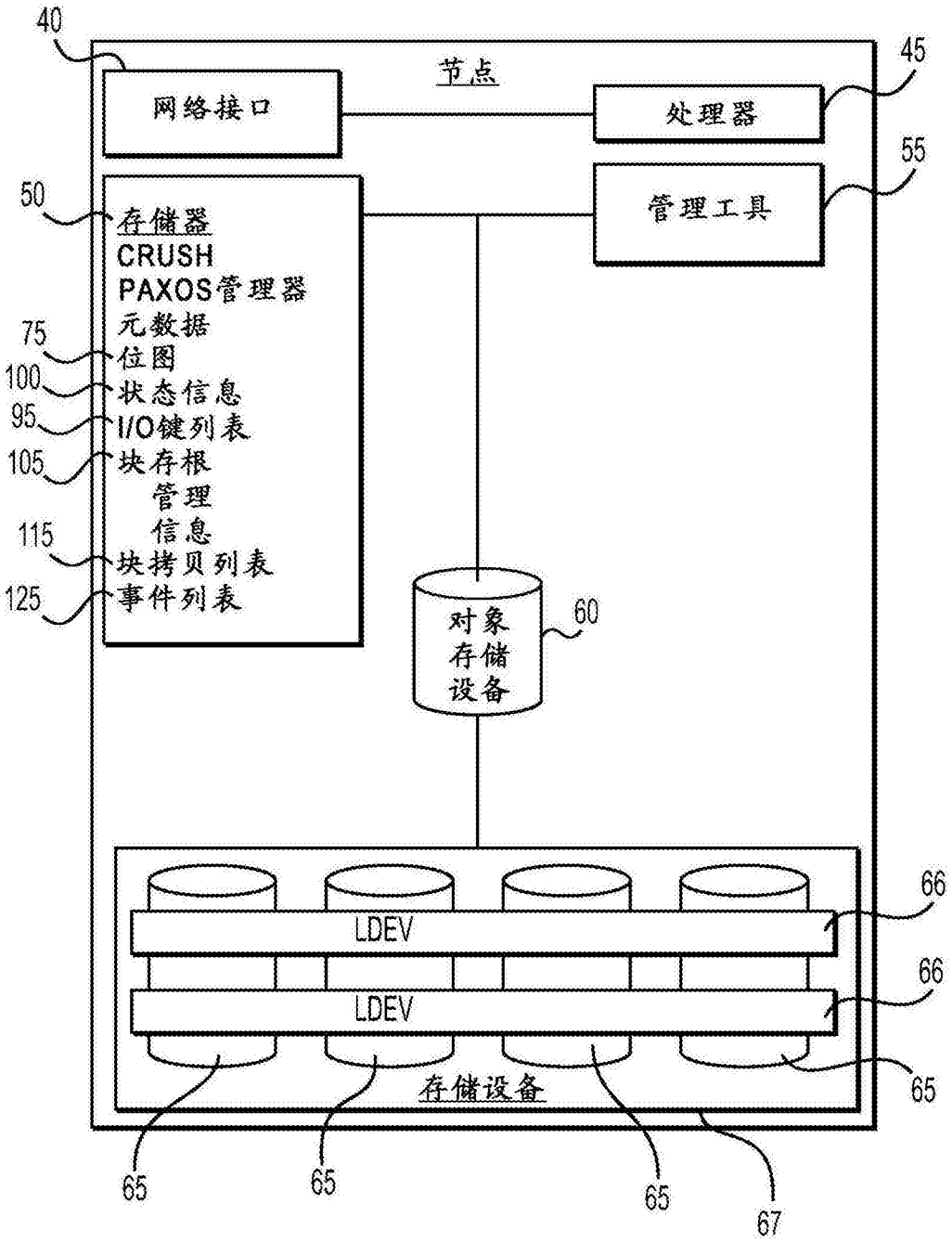

40
节点
网络接口
45
处理器
55
管理工具
50
存储器
CRUSH
PAXOS管理器
元数据
75
位图
100
状态信息
95
I/O键列表
105
块存根
管理
信息
115
块拷贝列表
125
事件列表
对象
存储
设备
60
66
LDEV
66
LDEV
65
存储设备
65
65
65
67

图2

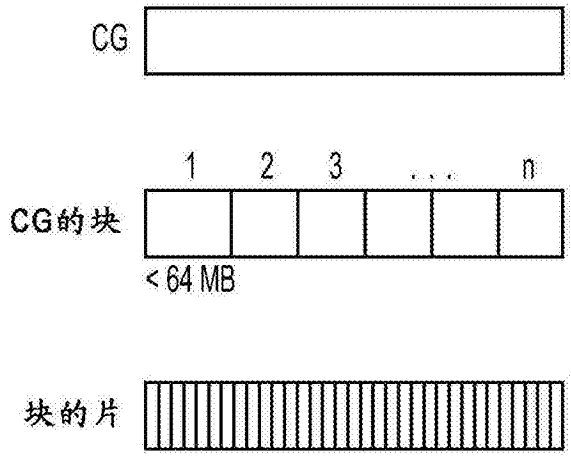

CG
1 2 3 . . . n
CG的块
< 64 MB
块的片

图3

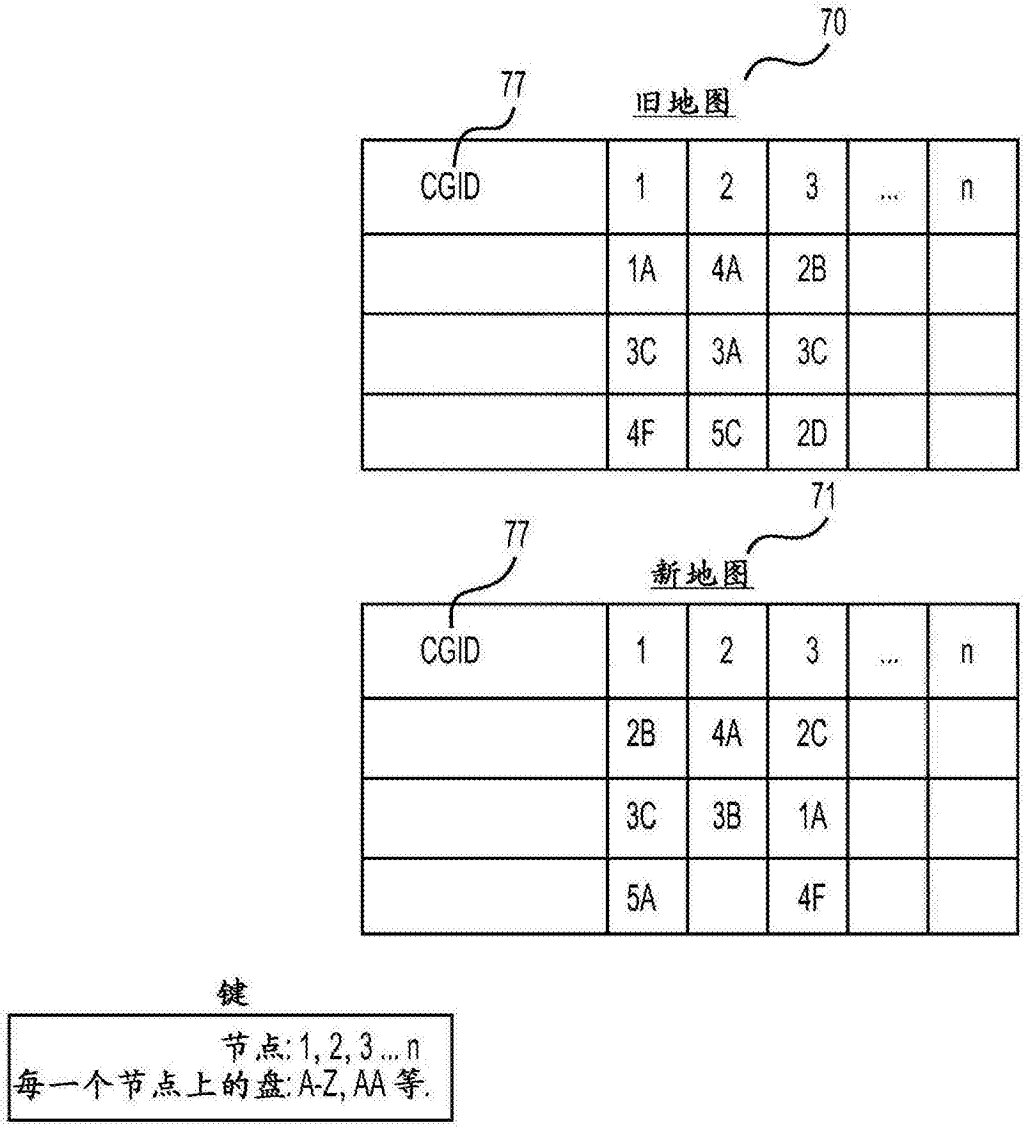

70
77
旧地图
CGID 1 2 3 ... n
1A 4A 2B
3C 3A 3C
4F 5C 2D
71
77
新地图
CGID 1 2 3 ... n
2B 4A 2C
3C 3B 1A
5A 4F
键
节点: 1, 2, 3 ... n
每一个节点上的盘: A-Z, AA 等.

图4

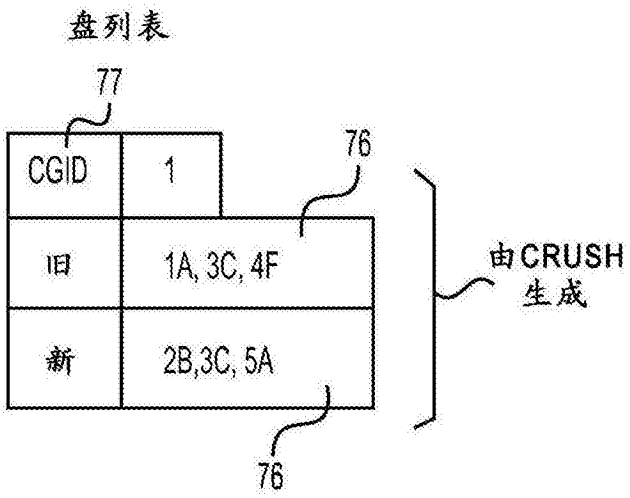



图5

100

| 节点# | 节点状态 | 盘 | 盘状态 |
|---|---|---|---|
| 1 | 1 | A | 1 |
| | | B | 1 |
| | | C | 1 |
| 2 | 1 | A | 1 |
| | | B | 0 |
| | | C | 1 |
| | | D | 0 |
| | | E | 0 |
| | | F | 1 |
| | | G | 1 |

图6

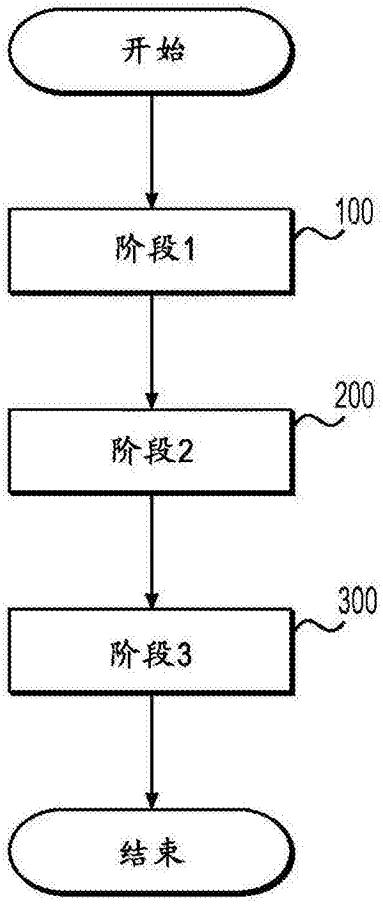



图7

节点1

| 事件# | 事件 |
|---|---|
| 1 | MAP_CHANGE_PROPOSAL |
| 2 | MAP_MIGRATE |
| . . . | |
| n | MAP_COMMIT |

125

图8

阶段一

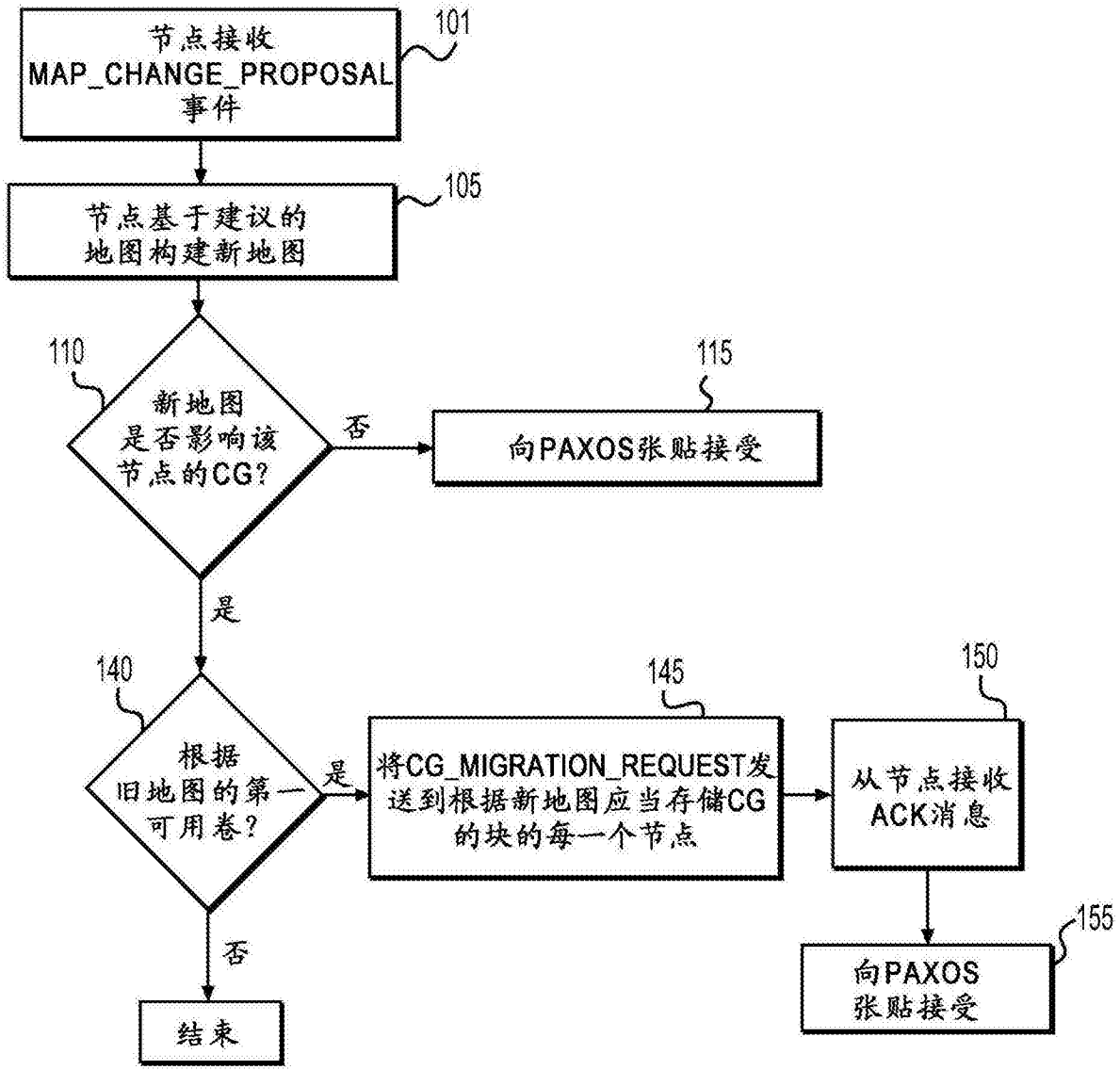

节点接收
MAP_CHANGE_PROPOSAL
事件
101
节点基于建议的
地图构建新地图
105
新地图
是否影响该
节点的CG?
110
否
向PAXOS张贴接受
115
是
根据
旧地图的第一
可用卷?
140
是
将CG_MIGRATION_REQUEST发
送到根据新地图应当存储CG
的块的每一个节点
145
从节点接收
ACK消息
150
向PAXOS
张贴接受
155
否
结束

图9

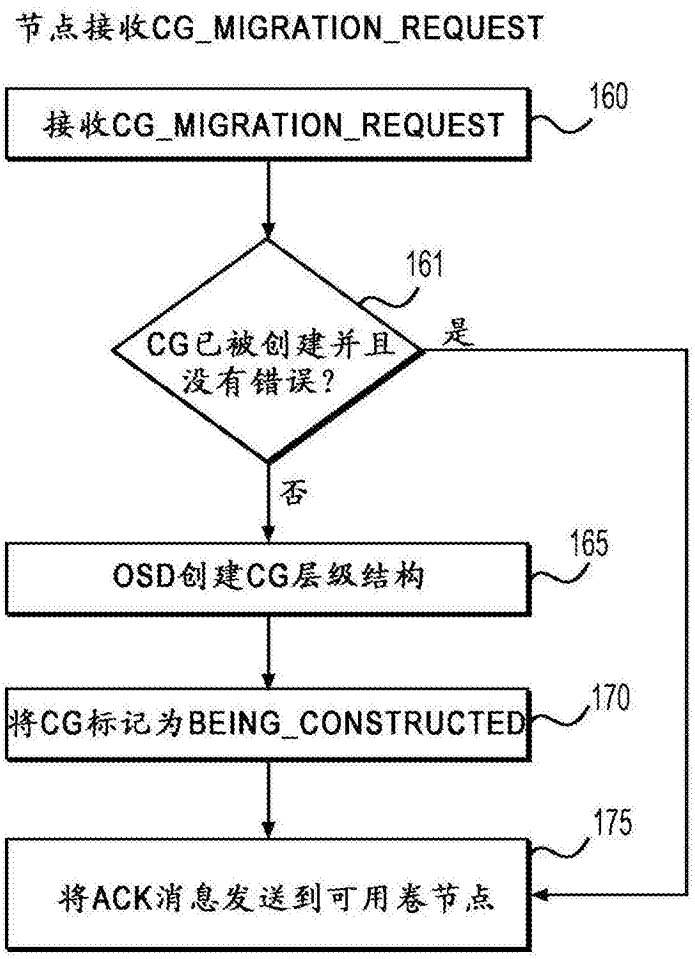

节点接收CG_MIGRATION_REQUEST
接收CG_MIGRATION_REQUEST
160
161
CG已被创建并且没有错误？
是
否
OSD创建CG层级结构
165
将CG标记为BEING_CONSTRUCTED
170
将ACK消息发送到可用卷节点
175

图10

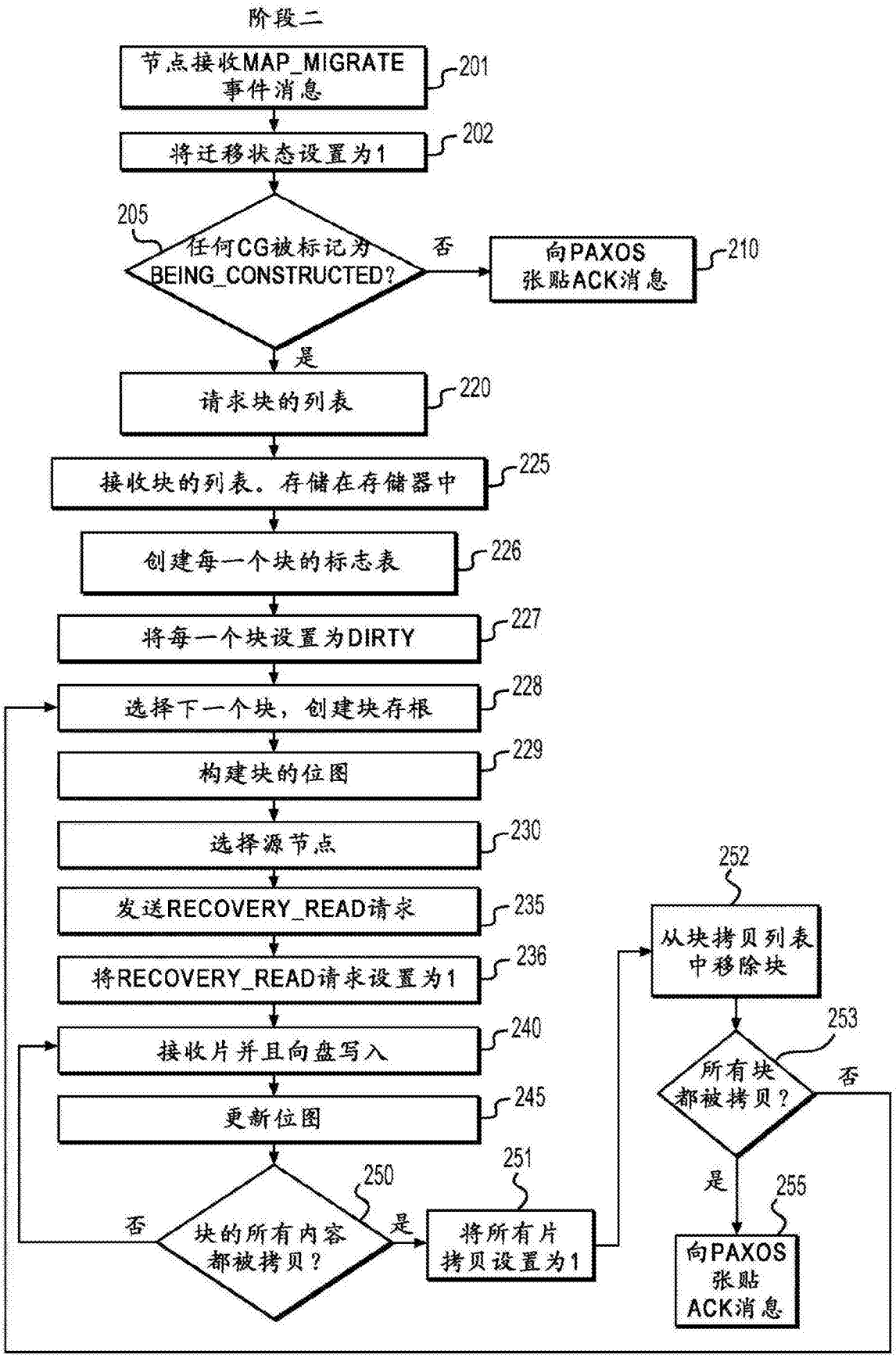

阶段二
节点接收MAP_MIGRATE事件消息
201
将迁移状态设置为1
202
205
任何CG被标记为BEING_CONSTRUCTED?
否
向PAXOS张贴ACK消息
210
是
请求块的列表
220
接收块的列表。存储在存储器中
225
创建每一个块的标志表
226
将每一个块设置为DIRTY
227
选择下一个块，创建块存根
228
构建块的位图
229
选择源节点
230
发送RECOVERY_READ请求
235
将RECOVERY_READ请求设置为1
236
接收片并且向盘写入
240
更新位图
245
250
块的所有内容都被拷贝？
否
是
251
将所有片拷贝设置为1
252
从块拷贝列表中移除块
253
所有块都被拷贝？
否
是
255
向PAXOS张贴ACK消息

图11

标志表

| | 块1 | | 块2 | | 120 |
|---|---|---|---|---|---|
| | 状态 | DEGRADED 标志或WRITE 之后的事件编号 | 状态 | DEGRADED 标志或WRITE 之后的事件编号 | |
| CLEAN | 0 | | 1 | | |
| DIRTY | 1 | | 0 | | |
| DEGRADED | 1 | 3 | 1 | 2 | |
| WRITE | | 2 | | 1 | |

图12

块存根管理信息

| | | | |
|---|---|---|---|
| 106 | 块编号 | 8 | 105 |
| 107 | CG ID | 4 | |
| 108 | 旧地图位置 | 2A | |
| 109 | RECOVERY_READ 请求发送 | 0 | |
| 110 | 所有片拷贝 | 0 | |
| 111 | I/O键列表空 | 0 | |

图13

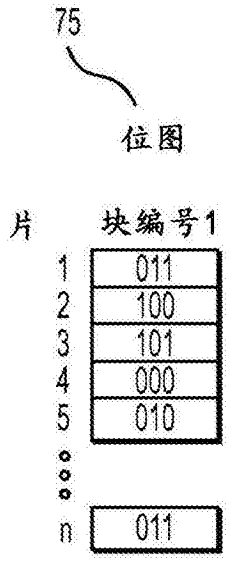



图14

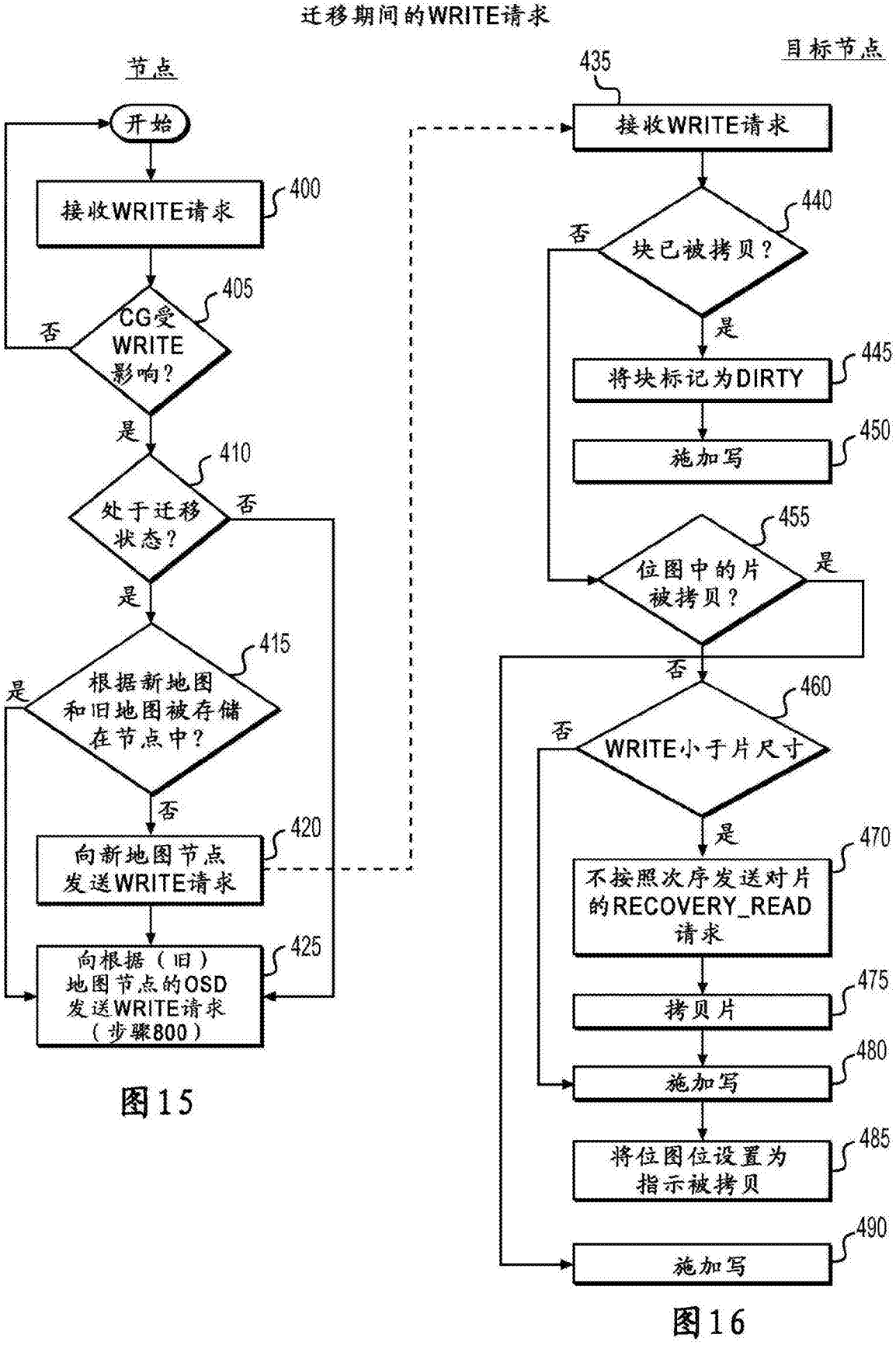

迁移期间的WRITE请求
节点
目标节点
开始
接收WRITE请求
400
CG受WRITE影响？
405
否
是
处于迁移状态？
410
否
是
根据新地图和旧地图被存储在节点中？
415
是
否
向新地图节点发送WRITE请求
420
向根据（旧）地图节点的OSD发送WRITE请求（步骤800）
425
接收WRITE请求
435
块已被拷贝？
440
否
是
将块标记为DIRTY
445
施加写
450
位图中的片被拷贝？
455
是
否
WRITE小于片尺寸
460
否
是
不按照次序发送对片的RECOVERY_READ请求
470
拷贝片
475
施加写
480
将位图位设置为指示被拷贝
485
施加写
490

图15

图16

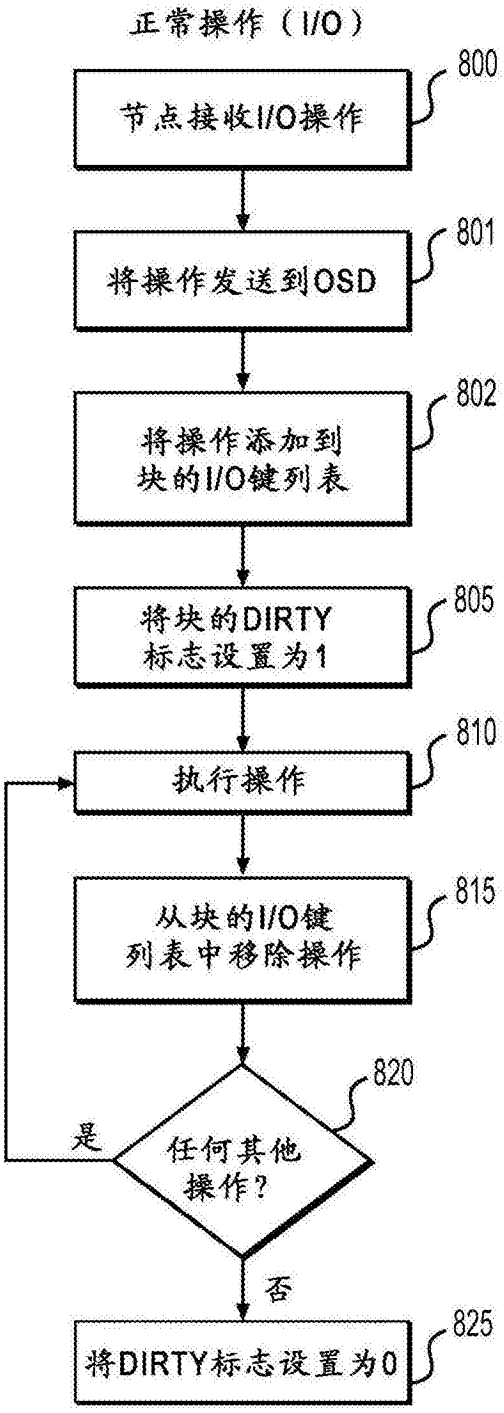

正常操作（I/O）
节点接收I/O操作
800
将操作发送到OSD
801
将操作添加到
块的I/O键列表
802
将块的DIRTY
标志设置为1
805
执行操作
810
从块的I/O键
列表中移除操作
815
任何其他
操作？
820
是
否
将DIRTY标志设置为0
825

图17

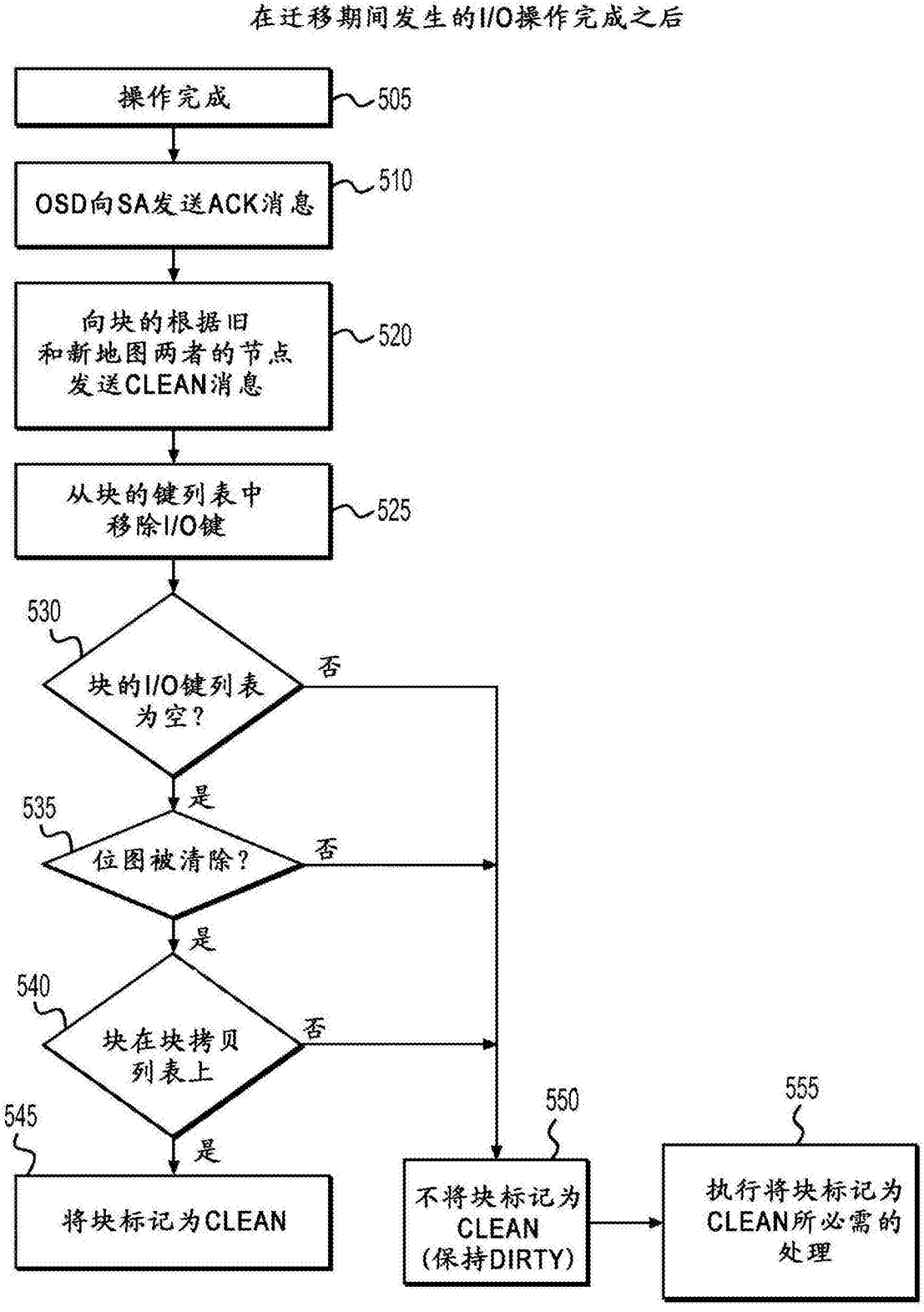

在迁移期间发生的I/O操作完成之后
操作完成
505
OSD向SA发送ACK消息
510
向块的根据旧和新地图两者的节点发送CLEAN消息
520
从块的键列表中移除I/O键
525
530
块的I/O键列表为空？
否
是
535
位图被清除？
否
是
540
块在块拷贝列表上
否
是
545
将块标记为CLEAN
550
不将块标记为CLEAN(保持DIRTY)
555
执行将块标记为CLEAN所必需的处理

图18

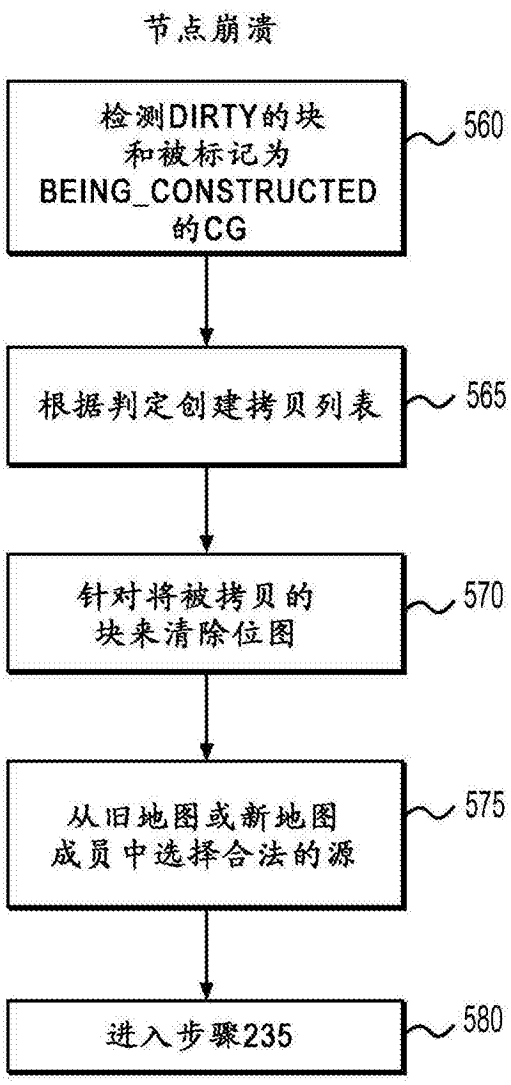

节点崩溃
检测DIRTY的块和被标记为BEING_CONSTRUCTED的CG
560
根据判定创建拷贝列表
565
针对将被拷贝的块来清除位图
570
从旧地图或新地图成员中选择合法的源
575
进入步骤235
580

图19

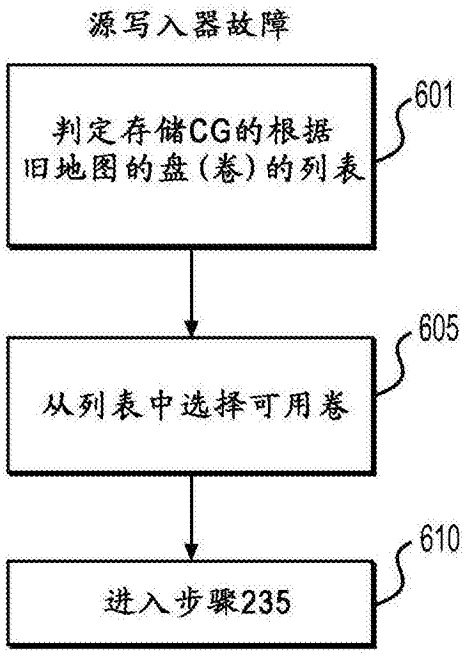



图20

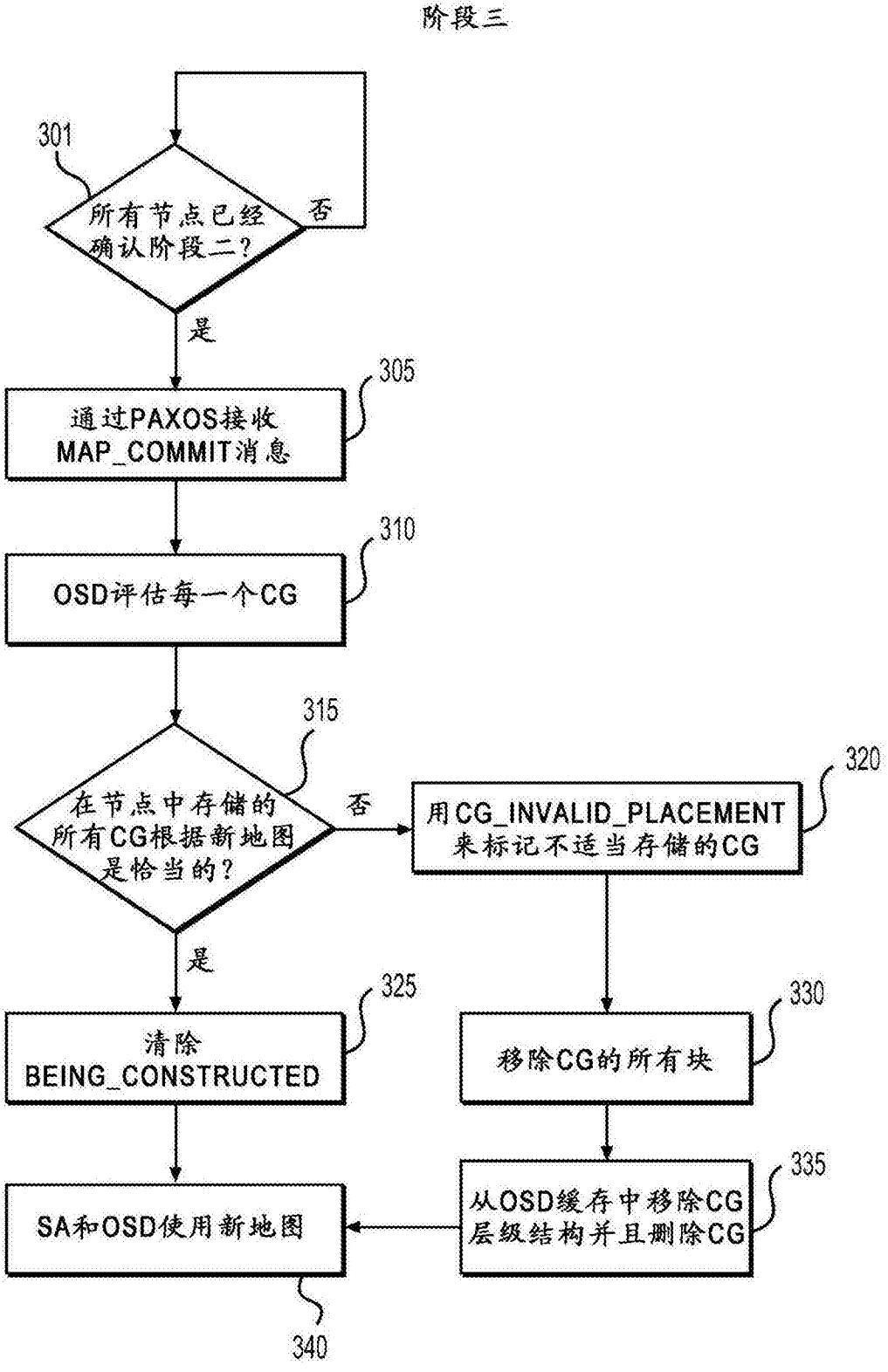

阶段三
301
所有节点已经确认阶段二?
否
是
305
通过PAXOS接收MAP_COMMIT消息
310
OSD评估每一个CG
315
在节点中存储的所有CG根据新地图是恰当的?
否
320
用CG_INVALID_PLACEMENT来标记不适当存储的CG
是
325
清除BEING_CONSTRUCTED
330
移除CG的所有块
335
从OSD缓存中移除CG层级结构并且删除CG
SA和OSD使用新地图
340

图21


图22

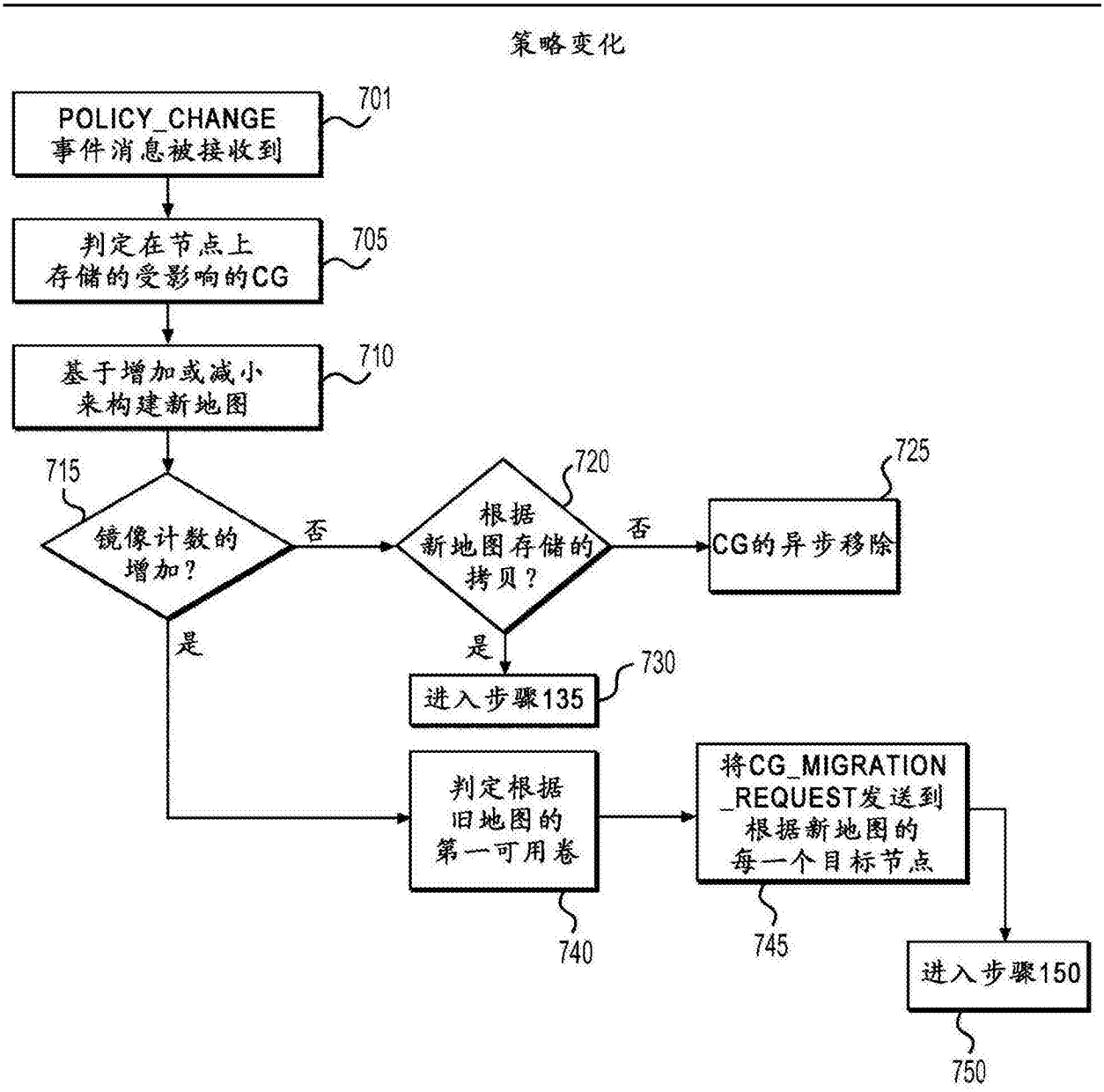

策略变化
POLICY_CHANGE
事件消息被接收到
701
判定在节点上
存储的受影响的CG
705
基于增加或减小
来构建新地图
710
715
镜像计数的
增加？
否
720
根据
新地图存储的
拷贝？
否
725
CG的异步移除
是
是
进入步骤135
730
判定根据
旧地图的
第一可用卷
740
将CG_MIGRATION
_REQUEST发送到
根据新地图的
每一个目标节点
745
进入步骤150
750

图23

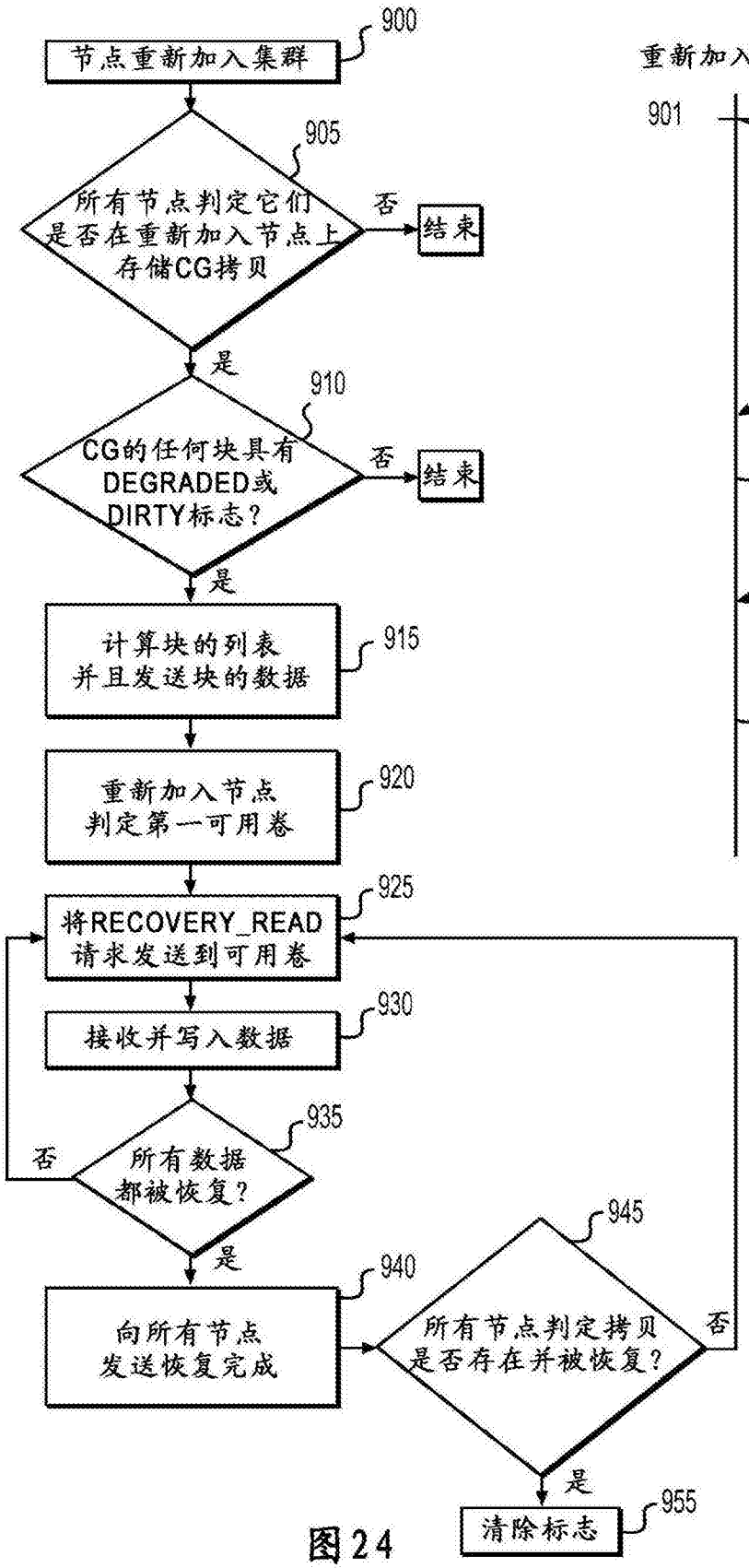

节点重新加入集群
900
所有节点判定它们是否在重新加入节点上存储CG拷贝
905
否
结束
是
CG的任何块具有DEGRADED或DIRTY标志？
910
否
结束
是
计算块的列表并且发送块的数据
915
重新加入节点判定第一可用卷
920
将RECOVERY_READ请求发送到可用卷
925
接收并写入数据
930
所有数据都被恢复？
935
否
是
向所有节点发送恢复完成
940
所有节点判定拷贝是否存在并被恢复？
945
否
是
清除标志
955

图24

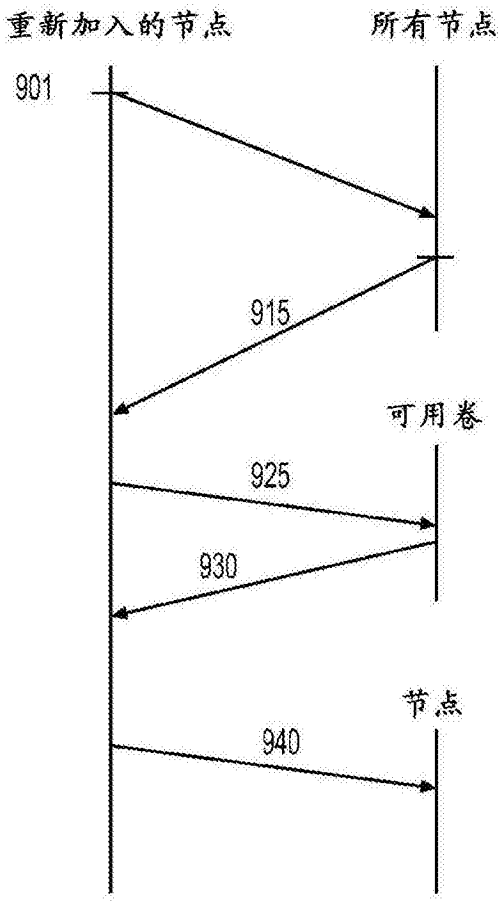

重新加入的节点
所有节点
901
915
可用卷
925
930
节点
940

图25

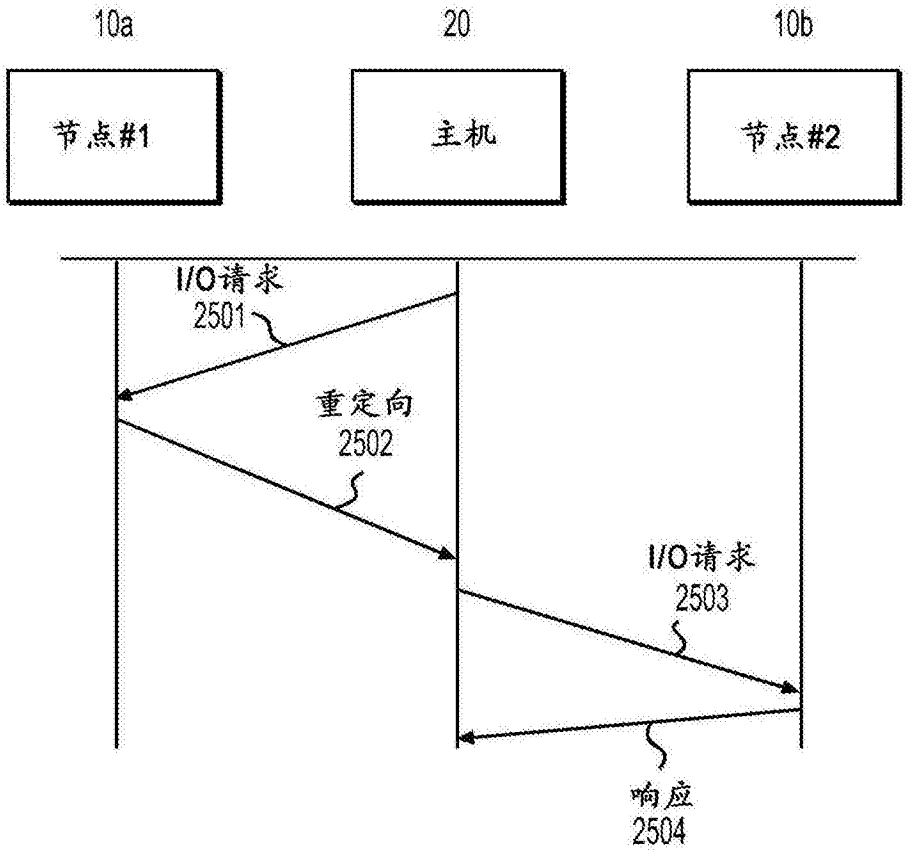



图26

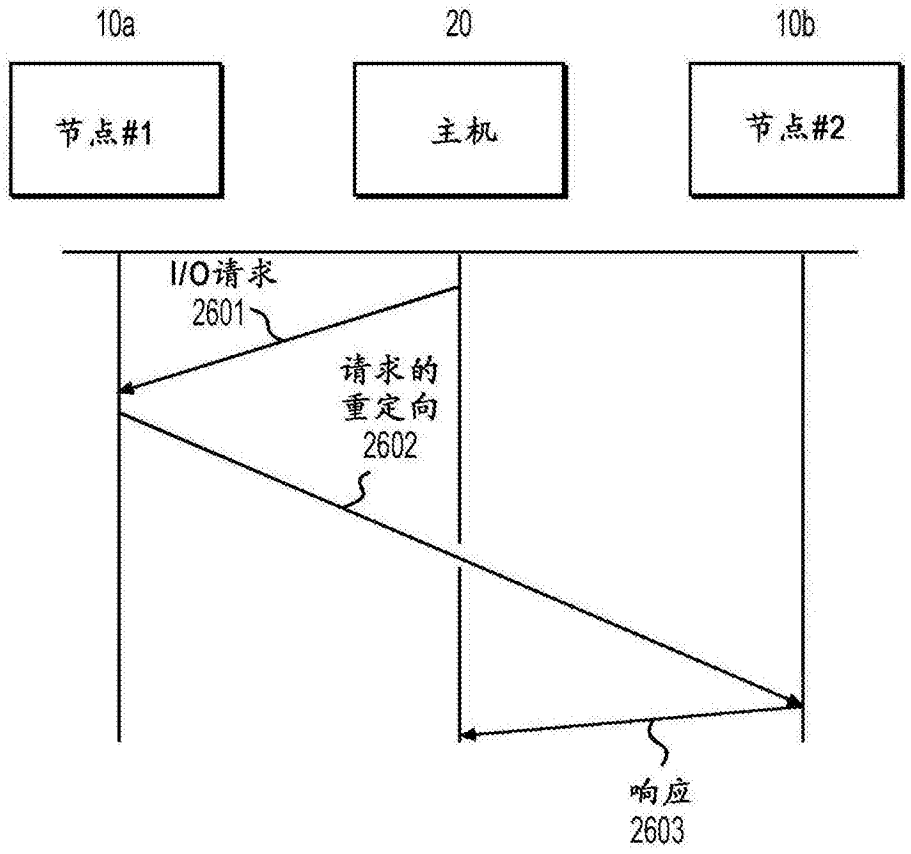



图27