(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113366424 B
(45)授权公告日 2024.03.08

(21)申请号 201980090951.3
(22)申请日 2019.10.03
(65)同一申请的已公布的文献号
申请公布号 CN 113366424 A
(43)申请公布日 2021.09.07
(30)优先权数据
16/265,491 2019.02.01 US
(85)PCT国际申请进入国家阶段日
2021.07.30
(86)PCT国际申请的申请数据
PCT/US2019/054575 2019.10.03
(87)PCT国际申请的公布数据
WO2020/159585 EN 2020.08.06
(73)专利权人 EMC IP控股有限公司
地址 美国马萨诸塞州
(72)发明人 P·石兰 K·卢 J·布兰特
N·能登 T·特隆 M·阿雷瓦洛
(74)专利代理机构 北京同达信恒知识产权代理有限公司 11291
专利代理师 黄志华 何月华
(51)Int.Cl.
G06F 3/06(2006.01)
G06F 11/14(2006.01)
(56)对比文件
US 9411815 B1,2016.08.09
US 8825720 B1,2014.09.02
US 2019018742 A1,2019.01.17
WO 2018191709 A1,2018.10.18
US 2018373615 A1,2018.12.27
CN 101727355 A,2010.06.09
CN 101996233 A,2011.03.30
CN 1509435 A,2004.06.30
US 2011167096 A1,2011.07.07
US 2011225214 A1,2011.09.15
US 9141301 B1,2015.09.22
US 9367397 B1,2016.06.14
审查员 韩艳

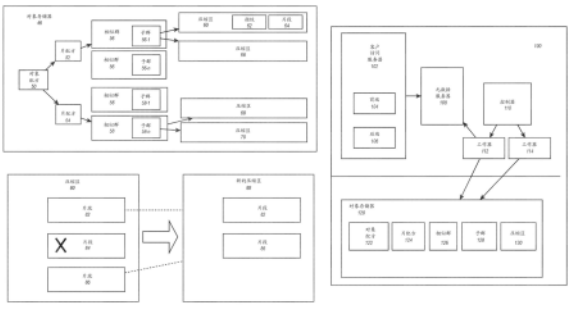
权利要求书2页 说明书14页 附图13页

(54)发明名称

用于重复数据删除存储器的可扩展垃圾收集

(57)摘要

用于清理存储系统的系统和方法。通过识别包含死片段或未引用片段的结构来清理重复数据删除存储系统。这包括处理配方以识别不再是活对象配方的一部分的片段。然后,移除死片段。这是通过拷贝转发活片段、然后作为整体删除包含死片段的结构来实现的。



CN 113366424 B

1.一种用于从存储系统中清理被删除对象的方法,所述方法包括:

识别受垃圾收集操作影响的受影响的相似群,其中,所述受影响的相似群包括与所述被删除对象相关联的片段和与活对象相关联的片段;

确定单个受影响的相似群的大小,并确定所有受影响的相似群的大小;

基于所述单个受影响的相似群的大小和所述所有受影响的相似群的大小,确定用于执行所述垃圾收集操作的工作器的数量,所述垃圾收集操作从所述存储系统存储的对象中清理所述存储系统存储的所述被删除对象;

基于所述单个受影响的相似群的大小和所述所有受影响的相似群的大小,为每个工作器分配一系列的受影响的相似群;

识别所述受影响的相似群中的与所述活对象相关联的活片段;

移除与所述被删除对象相关联的片段;

更新所述受影响的相似群以反映与所述被删除对象相关联的片段已被移除。

2.如权利要求1所述的方法,还包括启动控制器以控制所述垃圾收集操作,其中,所述控制器控制所述工作器。

3.如权利要求1所述的方法,其中,移除与所述被删除对象相关联的片段包括将与所述活对象相关联的片段拷贝到新结构,使得所述新结构不存储不再被活对象引用的片段。

4.如权利要求3所述的方法,还包括在所述新结构完成之后删除存储与被删除对象相关联的片段和与所述活对象相关联的片段的结构。

5.如权利要求1所述的方法,还通过考虑一个或多个因素来估计工作器的数量,所述一个或多个因素包括设置所述工作器的数量的环境变量、每个工作器的内存分配、受垃圾收集过程影响的相似群的数量和/或大小、输入/输出操作、或吞吐量。

6.如权利要求1所述的方法,其中,对象配方包括片配方并且每个片配方与相似群相关联。

7.如权利要求1所述的方法,还包括识别所述受影响的相似群和相似子群。

8.如权利要求7所述的方法,还包括检索对应于所述被删除对象的记录,其中,所述记录至少标识用于所述受影响的相似群和相似子群的对象配方,并且,从与所述对象配方相关联的片配方中识别所述受影响的相似群。

9.如权利要求7所述的方法,还包括记录所述受影响的相似群的大小。

10.如权利要求1所述的方法,还包括至少写锁定所述受影响的相似群的受影响的子群,以便同时支持正常操作和所述垃圾收集操作。

11.一种非暂时性计算机可读介质,包括在被执行时执行权利要求1所述的方法的计算机可执行指令。

12.一种用于从重复数据删除存储系统中清理被删除对象的方法,其中,所述重复数据删除存储系统使用引用片配方的对象配方来存储对象,所述片配方引用相似群并且所述相似群引用压缩区,其中,所述重复数据删除存储系统中的对象的片段被存储在所述压缩区中,所述方法包括:

执行垃圾收集操作的启动阶段以实例化控制器,其中,所述垃圾收集操作从所述重复数据删除存储系统中清理未引用片段;

确定所述重复数据删除存储系统中与所述未引用片段相关联的受影响的相似群,其

中,所述受影响的相似群与所述未引用片段和活片段相关联;

由所述控制器基于所述受影响的相似群的数量、工作器容量、每个受影响的相似群的大小和所有受影响的相似群的大小来估计执行所述垃圾收集操作所需的工作器的数量;

基于各个受影响的相似群的大小和所述所有受影响的相似群的大小,为每个工作器分配一系列的受影响的相似群;

标记所述受影响的相似群中的所述活片段;

由所述工作器移除所述未引用片段;以及

更新所述受影响的相似群以反映与所述被删除对象相关联的所述未引用片段已被删除。

13.如权利要求12所述的方法,还包括通过将所述活片段从现有压缩区拷贝转发到新的压缩区来移除所述未引用片段,其中,在所述现有压缩区中的所述未引用片段未被拷贝到所述新的压缩区中。

14.如权利要求12所述的方法,还包括基于删除记录来确定所述受影响的相似群。

15.如权利要求14所述的方法,其中,所述删除记录被存储在所述重复数据删除存储系统的删除桶中,并且,所述删除记录至少包括对象配方。

16.如权利要求15所述的方法,还包括基于包括在所述对象配方中的或由所述对象配方引用的片配方来确定所述受影响的相似群。

17.如权利要求12所述的方法,还包括将所述受影响的相似群划分给所述工作器,其中,每个工作器至少部分地基于所述受影响的相似群的预期大小和标识符来接收一系列的所述受影响的相似群。

18.如权利要求12所述的方法,还包括写锁定所述受影响的相似群,使得所述垃圾收集操作和正常操作能够同时发生。

19.一种非暂时性计算机可读介质,包括用于执行如权利要求12所述的方法的计算机可执行指令。

用于重复数据删除存储器的可扩展垃圾收集

技术领域

[0001] 本发明的实施方式涉及用于执行数据保护操作(例如收集垃圾)的系统、装置和方法。更具体地,本发明的实施方式涉及用于在重复数据删除存储系统(例如基于重复数据删除云的存储系统)中收集垃圾的系统和方法。

背景技术

[0002] 保护数据是当今计算机技术的基本方面。如果数据不受保护,则数据更有可能丢失,并且数据丢失可能对实体造成重大损害。因此,许多实体将它们的数据或它们数据的备份存储在存储系统(例如基于云的存储系统)中。然而,由于相关的成本以及由于强加于数据或与数据相关的要求和策略,保护数据比简单地将数据的副本存储在云中要复杂得多。例如,备份通常受制于备份策略(例如,每天、每周、每月创建备份)和保留策略。这导致大量数据在存储要求和计算要求方面具有相应的成本,即使对数据进行了重复数据删除时也是如此。

[0003] 由于各种原因,备份通常随着时间推移而被删除。例如,系统可能在保留期到期时删除备份。删除备份不是简单的任务,特别是在重复数据删除存储系统中。在重复数据删除系统中,数据通常被分成块或片段并以重复数据删除形式存储。这通过允许将相同的块或片段用于多个备份或多个对象来降低存储要求(和成本)。

[0004] 不可避免地,存储在数据保护系统中的一些数据或对象是死的。客户端或存储系统不引用或不再需要死的对象或数据。由于备份过期和出于其他原因,备份系统执行垃圾收集操作以删除或移除不再由任何有效备份引用的对象。然而,这不能通过简单地删除死对象的片段来实现,因为那些片段可能对应于活对象。此外,传统方法(例如引用计数)是不灵便的,因为它们可能需要保护系统来维护数十亿的计数。因此引用计数消耗大量存储空间并且它们非常难以管理,特别是在分布式系统和基于云的系统中。

附图说明

[0005] 为了描述可以获得本发明的至少一些方面的方式,将通过参考在附图中示出的本发明的特定实施方式来呈现更具体的描述。应理解这些附图仅描绘了本发明的示例性实施方式并且因此不被认为是对本发明范围的限制,将通过使用附图以附加的特征和细节来描述和解释本发明的实施方式,其中:

[0006] 图1A示出用于在存储系统(例如基于云的存储系统)中存储重复数据删除数据的方式的示例;

[0007] 图1B示出在存储系统(例如基于云的存储系统)中已存储数据在清理之前和清理之后的示例;

[0008] 图1C示出保护系统的示例,该保护系统被配置为在存储系统(例如基于云的存储系统)中执行包括垃圾收集的数据保护操作;

[0009] 图2示出保护系统用来清理对象存储器的对象存储桶和删除桶的示例;

[0010] 图3示出将对象从对象存储桶移动到删除桶以准备执行垃圾收集操作的过程；

[0011] 图4示出用于执行数据保护操作(例如垃圾收集)的方法的示例；

[0012] 图5示出垃圾收集操作的启动阶段的示例,其中估计了工作器和工作器的容量；

[0013] 图6示出基于垃圾收集过程所影响的相似群来估计工作器的数量的方法的示例；

[0014] 图7示出在估计垃圾收集过程所需的工作器的数量时可以考虑的因素的示例；

[0015] 图8A和图8B示出处理相似群的示例,包括在执行垃圾收集过程时标记受影响的相似群；

[0016] 图9示出相似群和与该相似群具有相同标识符的相关联子群的示例;以及

[0017] 图10示出在准备垃圾收集过程的拷贝转发阶段中标记活片段的示例。

具体实施方式

[0018] 本发明的实施方式涉及用于提供或执行数据保护操作的系统、装置和方法。示例性数据保护操作包括但不限于备份操作、恢复操作、重复数据删除操作、复制操作和/或垃圾收集操作。举例来说,执行垃圾收集操作是为了清理存储系统的死对象或未引用对象。换句话说,执行垃圾收集操作是为了从存储系统中移除客户端不再需要或者不再被活对象引用或不再为活对象的一部分的对象。

[0019] 在重复数据删除存储系统中删除对象是复杂的,因为与被删除对象相关联的片段不能立即从存储系统中移除,这是因为被删除对象的一些片段可能与其他活对象相关联。非限制性地,活对象可以是应该保存在存储系统中的对象。死对象是可以从存储系统中丢弃或移除的对象。对象可以表示数据、文件、数据集(例如备份数据集)、单一文件等或其组合。本文讨论的数据保护操作可以在例如包括云实现的DELL EMC Data Domain(数据域)的系统中执行。

[0020] 本发明的实施方式涉及确保数据完整性、在不运行时不产生货币成本并且可扩展以满足性能和/或时间限制的垃圾收集操作。此外,本发明的实施方式在执行垃圾收集的同时支持同时发生的读取/写入。本发明的实施方式还简化了与编码和调试活动相关联的困难。

[0021] 在一个示例中,基于云的数据保护系统(保护系统)可以实现为微服务或基于容器的应用程序,并且可以被配置为在云环境中操作。更具体地,垃圾收集可以被实现为微服务,其通过从存储系统中移除未引用片段(或其他数据表示或结构)而不影响活片段来清理存储系统的由客户端删除或根据保留策略删除的对象。保护系统可以在容器中运行,并且保护系统可以根据需要放大和缩小。保护系统的组件可以实现为微服务。

[0022] 本发明的实施方式通过确保未引用数据不消耗不必要的存储空间并且通过确保未引用数据不消耗计算资源来改进数据保护系统的操作,包括由数据保护系统执行的操作。更具体地,通过移除死对象,数据保护系统不必负担来处理未被引用的数据。这消除了一些处理,从而改进了数据保护系统的操作。此外,存储(例如基于云的存储)成本通常基于存储的数据的量。通过执行垃圾收集,可以移除死对象或死片段。

[0023] 在一些云系统中,也存在消耗计算资源的成本。本发明的实施方式节省了计算资源,至少因为用于垃圾收集操作的计算资源仅在垃圾收集操作正在执行时才被分配和使用。当垃圾收集操作未运行时,可以释放计算资源。

[0024] 在一个示例中,保护系统可以通过将对象分为或分块为片和片段(非常大的对象可以被分为部分,这些部分被分为片,这些片被分为片段)来对对象进行重复数据删除。

[0025] 图1A示出可以如何将对象存储在基于云的存储系统中的示例。图1A示出对象存储器48,其可以是对象存储桶。存储器可以以其他方式表示或配置。与对象配方(recipe)50相关联的对象的实际数据(片段)被存储在压缩区中,例如压缩区60、66、68和/或70。因此可以将与对象配方50相关联的对象的片段存储在压缩区60、66、68和70的一者或多者中。

[0026] 压缩区60可以包括片段64和这些片段64的指纹62。其他压缩区66、68和70被类似地配置。在该示例中,相似群56与包括压缩区60和66的多个压缩区相关联。类似地,相似群58与压缩区68和70相关联。

[0027] 相似群56可以具有如示出为子群56-1到子群56-n的一个或多个子群或与该一个或多个子群相关联。如图所示,类似地,相似群58包括子群58-1到58-n。在该示例中,相似群56和子群56-1被存储为对象。其他子群被类似地存储。每个子群可以具有与相应相似群相同的相似群标识符(ID),并且子群可以被不同地编号(例如,按升序)。相似群56标识与对象相关联的压缩区名称和相关联指纹。更具体地,压缩区可以与特定的子群相关联。片配方52标识相关联对象的片的相似群和子群。在一个示例中,每个片与单个相似群相关联。对象配方50标识对象的片。

[0028] 在一个示例中,相似群56包括子群。换句话说,对象与相似群和子群相关联。例如,相似群可以存储为具有群ID和子群ID、但绝不会没有子群ID的对象。图1A示出相似群56和相关联的子群可以被存储,如被存储为单独的对象。然而,对象的配方通常标识相似群、子群和压缩区中的每一者。

[0029] 图1A基本上示出了单个对象并且该对象已被分成两个片。在该示例中,每个片大约为8MB。对象配方50标识对象的所有片。因此,对象配方50在将相关联的对象存储到存储系统时生成并且用于在从存储系统读取对象时重新组装对象。对象存储器48可以包括多个以这种方式的对象。此外,对象配方50、片配方52、54、相似群56、58和压缩区都可以是对象存储器48中的独立对象。在该示例中,对象的片配方52标识相似群56:子群56-1,以及片配方54标识相似群58:子群58-n。每个子群可以与多个压缩区相关联或标识多个压缩区(例如,压缩区60中的片段3、4和5以及压缩区66中的片段1和2(取决于配方))。

[0030] 在重复数据删除期间,对象可以被分为片,并且可以将片进一步分为或分块为片段。片和块的大小是可配置的。仅举例来说,对象可以被分成8MB的片。每个片可以被分成8KB的片段。为了执行重复数据删除,每个片都被映射到相似群。可以基于片的内容或基于应用于片的内容的函数来确定相似群。因为片被映射到相似群,所以被重复数据删除的片的片段或内容可以已经存在于相似群中。仅举例来说,片可以仅关于相似群和关于相似群的特定子群进行重复数据删除。

[0031] 例如,对象配方可以标识片。片配方可以标识相似群和子群。压缩区包括标识的相似群和子群或与标识的相似群和子群相关联。

[0032] 在重复数据删除期间,可以将压缩区中的唯一指纹附到子群1。一旦子群1达到阈值大小,就创建子群2。然后将相似群的新指纹和压缩区添加到子群2,因为子群1是满的。根据需要添加附加子群。

[0033] 通常,通过将要写入存储器的片段的指纹与保护系统已经存储的片段的指纹进行

比较来执行重复数据删除。指纹是片段的标识符(例如,片段的散列)并且可以在数据为明文或被加密的系统中实现。

[0034] 在相似群的上下文中,如果传入片段的指纹与已经存储在相似群中的片段的任何指纹匹配,则来自传入片的片段的指纹被标记为重复。如果来自传入片的片段的指纹与相似群的任何现有指纹都不匹配,则这些片段被认为是新的并存储在相似群中。

[0035] 图1B示出垃圾收集操作的结果。在垃圾收集期间,可以确定被删除对象包括片段82、84和86并且这些片段被包括在压缩区80中。还可以确定片段82和片段86与另一活对象相关联。因此,片段82和86是活片段,而片段84是死的并且未被任何其他配方引用。在识别活片段和死片段之后,通过仅将活片段写入(例如,拷贝转发)到新的压缩区88来清理压缩区80。然后可以删除压缩区80。因此,与压缩区80相关联的相似群被清理了死片段并且新的压缩区88仅包含压缩区80的与活对象相关联的片段。在一些对象存储系统中,可能无法修改现有对象,因此不能直接改变压缩区80。相反,在垃圾收集期间创建具有活片段的新的压缩区88。新的压缩区88可以具有与压缩区80相同的名称,在这种情况下,压缩区80在被替换时不需要被删除。在一个示例中,基于压缩区的内容为压缩区生成唯一名称。

[0036] 在一个示例中,通常可以使用元数据服务器来管理相似群或管理指纹。例如,元数据服务器可以存储指纹、片段和/或相似群之间的关系。元数据服务器可以存储由保护系统管理的所有片段的所有指纹。在重复数据删除和/或垃圾收集期间,可以查询元数据服务器以确定片段或一组片段是否为唯一的或是否为重复的或是否为活的。例如,当将片添加到相似群时,可以使用正被添加的片中的片段的指纹来对元数据服务器进行查询以确定是否有任何片段是重复的。唯一片段被添加到相似群中,并记录重复片段。通常,仅关于与相似群和相似群的特定子群相关联的指纹来执行重复数据删除。

[0037] 图1C示出执行包括垃圾收集的数据保护操作的计算环境或保护系统的示例。图1C示出保护系统100(或重复数据删除对象存储系统的示例)和对象存储120。保护系统100可以是数据保护系统的容器化实现并且可以包括微服务。在一个示例中,保护系统100可以在Kubernetes环境中实现。对象存储器120可以是基于云的存储系统,例如托管在数据中心中的存储系统。对象存储器120可以存在于私有云内的本地环境中。系统100和/或对象存储器120本质上可以是分布式的并且可以是可扩展的。

[0038] 对象存储器120被配置为存储对象或数据。系统100被配置为以重复数据删除的形式存储对象或数据,尽管在一些情况下重复数据删除可能不是100%。如前文所述,对象可以存储在对象存储器120中。因此,对象存储器120可以包括与片配方124、相似群126(和子群128)和压缩区130相关联的对象配方122。

[0039] 保护系统100可以包括客户访问服务器102。客户访问服务器102可以包括前端104和后端106。前端104和后端106可以是使用分配的计算资源(处理器、内存和其他需要的硬件组件)运行的微服务。前端104可以向客户提供接口。可以通过前端104接收对象。使用前端104,用户或客户端可以能够查看对象、添加对象、删除对象、配置例如备份操作和还原操作的数据保护操作等或其组合。在一些示例中,保护系统100可以在客户端和实际数据之间设置逻辑构造。

[0040] 前端104还可以负责将对象分为片。后端106可以被配置为对数据或对象执行各种操作。一旦对象已被分为片(这也可以由后端106执行),则后端106可以负责计算散列并形

成对象配方、片配方等,这些可以被存储在对象存储器120中。后端106可以访问元数据服务器108以便识别片的相似群。后端106可以生成或确定对象配方并与元数据服务器108通信。

[0041] 在垃圾收集的上下文中,垃圾收集可以被配置为周期性地或根据计划表运行的作业。当垃圾收集完成时,用于执行垃圾收集的资源可以被释放。这允许仅在需要时获取资源并在不需要时释放资源。与不释放计算资源的解决方案相比,这有利地降低了成本。

[0042] 垃圾收集概述

[0043] 垃圾收集由控制器110(例如,服务器或计算环境中的节点)管理或控制。控制器110可以控制例如工作器112和工作器114的一个或多个节点来执行垃圾收集。

[0044] 数据保护系统100通过从对象存储器120移除或删除未引用片段或死片段来执行垃圾收集。虽然提及的是片段,但是应当理解,本发明的实施方式可以使用其他数据表示来操作。

[0045] 通过从对象存储器120中移除片段,有利地降低了存储成本和计算成本。本发明的实施方式还考虑到在云存储系统中保护系统100将不耗尽存储空间的事实。这允许垃圾收集操作在收集垃圾时有一些余地并且允许保护系统100等待部分是活的对象(例如,与一些片段相关联的对象,这些片段与另一对象相关联)。此外,可以通过继续存储部分是活的对象(partially-live object)一段时间来降低或减少通常高于存储成本的计算成本。例如,压缩区可以包括活片段和死片段。如果死片段的数量较低,那么在清理压缩区之前等待直到死片段的百分比超过阈值可能是更有利的。

[0046] 本发明的实施方式还基于要执行的工作的量和基于执行垃圾收集操作所需的资源和/或基于例如内存约束、IO约束、吞吐量约束等的约束来分配计算资源(例如,工作器节点)。

[0047] 在一个示例中,使用删除桶来执行垃圾收集。删除桶存储与已删除或从未完成(例如,部分写入)的对象相对应的记录。当执行垃圾收集过程时,处理删除桶中的记录。

[0048] 以下讨论涉及桶。桶是存储系统的至少一部分的一般表示。对象可以存储在对象存储桶中。当删除客户端写入的对象或出于其他原因删除对象时,将被删除对象的删除记录添加到删除桶中。删除记录可以以某种方式识别客户端写入的对象,例如通过对象的配方来识别。在一个示例中,删除记录可仅包括对象的名称(名称可足以识别配方)。因此,删除记录可包括关于对象的一些信息。这允许在垃圾收集操作期间识别相关片段。配方可以识别与对象相关联的相似群和压缩区。因此存储在删除记录中的数据的量或种类可以变化。使用删除记录,控制器110和工作器112、114能够使用删除记录来识别所有受影响的相似群(可能包括死片段的那些相似群)并清理对象存储器120。

[0049] 受影响的相似群(或特定子群)可以被写锁定(write-locked),使得传入的对象不影响正在被清理的相似群/子群。为确保在垃圾收集操作期间保留写入访问,必要时可将新子群添加到受影响的相似群中。在垃圾收集操作期间写入对象存储器120的任何对象可以关于新的和/或未受影响的相似群或关于未写锁定的相似群或子群进行重复数据删除。

[0050] 在一个示例中,垃圾收集过程工作负荷被分成多个部分(通常基于相似群)并分配给工作器。在识别哪些相似群和子群受垃圾收集过程影响之后,控制器110和/或工作器112、114可以识别并标记它们各自的相似群中的活指纹。例如,已经删除的对象可由片段1、2、3、4和5组成。另一个未被删除的对象可包括片段1、2、3、6和7。在这种情况下,片段1、2、3、

6和7的指纹(或例如散列的标识符)可以被标记为活的。将片段4和5删除。

[0051] 然后将相似群/子群中(并且更具体地压缩区中)的活片段结转到新的压缩区中。例如,如果压缩区包括片段1、2、3、4和5,则新的压缩区将包括片段1、2和3。包括片段4和5的旧压缩区被移除或删除。因此,从对象存储器120中清理掉未使用或未引用的片段4和5。然后可以释放写锁定。

[0052] 垃圾收集

[0053] 图2至图4示出垃圾收集操作和执行垃圾收集的保护系统的各方面。在以下讨论中,术语“桶”用于描述对象和记录如何被存储。然而,对象和记录可以存储在其他结构(例如容器、数据库等)中。在对象存储器中,客户端可以在称为桶的结构中组织它们的对象。可以创建和删除存储桶,以及可以基于(可能为空的)前缀字符串或以其他方式列出桶内的对象。在一个示例中,可以在客户端260和桶230之间实现例如逻辑桶262的间接层。客户端260可以与逻辑存储桶262交互。在查询底层对象存储桶230之前,可对各种桶名称进行字符串操作。

[0054] 在图2中并且举例来说,存储系统可以包括对象存储桶230(或多个桶)和删除桶250。在垃圾收集操作期间,控制器210(控制器可以基于计划表或按需实例化)可以创建一个或多个工作器(例如,工作器212和工作器214)并将要执行的工作的一部分分配给工作器212、214中的每一者。在一个示例中,每个工作器可以被分配一系列相似群。在一个示例中,控制器210以及工作器212和214可以被实现为舱(pod)。

[0055] 控制器210(和/或工作器)可以确定哪些相似群受到垃圾收集操作的影响。该确定例如基于删除桶250中包括的记录。如前文所述,在客户端由于重复数据删除而删除对象时清理与该对象相关联的所有片段可能是不现实的。相反,垃圾收集操作聚焦于从被删除对象引用的结构。当客户端从对象存储器230中删除对象或当对象出于其他原因被删除时,对象的对象配方可以被移除(例如,移动到删除桶250)并且可以对客户端不可见。

[0056] 在该示例中,对象存储桶230包括对象X和对象Y。对象X具有对象配方232。为了简单起见,假设对象X(片246)和对象Y(片248)中的每一者均有单个片。对象X的片246与相似群234(具体地,相似群A,子群1)相关联。对象X的片段物理地存储在压缩区236(具体地,压缩区3和4)中。类似地,对象Y具有对象配方240并且与相似群242(具体地,相似群A,子群1)和压缩区244(压缩区3和4)相关联。这表明压缩区和相似群可以被多个对象配方引用以实现重复数据删除。

[0057] 客户端可以删除对象X。当对象X被客户端删除或根据保留策略或出于其他原因被删除时,对象配方232从对象存储桶230中移除并且可以被存储在删除桶250中。

[0058] 图3示出该过程并且示出对应于与对象配方232相关联的对象的删除记录252已经被放置在删除桶250中。删除记录252可以包括数据254和/或时间戳258。在一个示例中,数据254对应于对象配方232。因此,对象X不再对客户端260可见。对象X的名称可以作为数据254记录在删除桶250中而不是拷贝对象配方232。该信息(例如,对象配方232、对象名称等)是删除记录252的示例。然后在垃圾收集过程期间使用删除桶250来清理对象存储桶230。

[0059] 除了由客户端明确删除或由于例如保留策略的策略而删除的对象之外,本发明的实施方式还清理或删除仅部分写入(partially written)或在完成之前已被放弃的对象。部分写入的对象可以对客户端不可见,因此,客户端将不会删除该对象。然而,释放这些类

型的对象所消耗的空间是有用的。

[0060] 在一个示例中,删除桶252对应于进行中(in-progress)的写入。因此,删除记录252被输入到删除桶250中以用于进行中的写入。当写入完成时,删除记录252从删除桶250中移除。当写入正在进行中时,可以更新删除记录252中包括的时间戳258。特别地,时间戳258代表修改时间戳,其标识进行中的对象上次被修改的时间。时间戳258可以在对象被写入的同时以一定间隔(例如每十分钟)更新。

[0061] 当执行垃圾收集时,当时间戳258早于阈值时清理或移除进行中的对象。换句话说,如果进行中的对象在阈值时间段内没有被写入,则将该进行中的对象删除,因为它已被客户端放弃。这可以包括依赖于如本文所讨论的对象配方以从对象存储桶230中识别和移除进行中的对象的过程。

[0062] 图4示出垃圾收集操作的示例。在一个示例中,垃圾收集操作是分阶段执行的。该方法可以从启动阶段402开始,其中控制器被实例化并且工作器被分配或实例化。分配工作器的部分可以包括确定执行垃圾收集操作所需的工作器的数量。图5进一步示出启动阶段402。

[0063] 图5示出启动阶段的示例。启动阶段通常通过启动或实例化502控制器开始。例如,当达到执行垃圾收集操作的时间时,创建作业(例如,计时程序(cron)作业)并实例化控制器。可以在集群(例如Kubernetes集群)内创建控制器。

[0064] 一旦控制器被实例化,则控制器估计504执行垃圾收集操作所需的工作器的数量。估计504工作器的数量可以包括确定506或识别受垃圾收集操作影响的相似群。这可以通过评估删除记录来确定。也通过估计508每个工作器的容量而影响工作器的数量。此外,该估计可以考虑保护系统面临的各种约束,包括内存、输入/输出(IO)、IO操作或吞吐量。

[0065] 估计504工作器的数量可以通过各种方式实现。在一个示例中,工作器的数量可以通过环境变量设置。环境变量可以用于测试、调试和性能比较目的。使用环境变量还有助于评估使用更复杂的估计方法可能无法考虑或预期的情景。环境变量可以基于过去的性能或出于其他原因来更新。

[0066] 如前文所述,要执行的工作可以受到受影响的相似群的数量的影响。因此,当估计要使用的工作器的数量时,可以确定506受影响的相似群的数量。

[0067] 图6示出估计执行垃圾收集操作所需的工作器的数量的方法的示例。在一个示例中,创建602用于确定大小的映射。该映射帮助识别工作器执行垃圾收集可能需要的空间或内存。在一个示例中,垃圾收集操作基于相似群被分配给工作器——每个工作器负责一系列相似群。然而,每个系列可以不相同。例如,可以基于受影响的相似群的大小为每个工作器分配一系列相似群来进行清理。

[0068] 映射可以包括每个相似群的子群的相似群映射和用于跟踪所有受影响的相似群所需的内存的总体大小的总体相似映射。在创建映射之后,从删除桶中读取604被删除对象的记录。解析或评估该记录以便识别或列出606与删除桶中的被删除对象相关联的片。可以从通过删除记录识别的片列表中获得相似群标识符。在一种命名约定中,每个片配方的名称包括片配方引用的相似群和子群的名称。从删除记录获得的相似群标识符可以被插入到映射中并计算大小。在一种实现方式中,立即检查<相似群,子群>的大小。在另一实现方式中,相似群标识符存储在映射中,单独的列表取代了所有相似群和子群连同其大小,并且需

要的大小存储在映射中。因此,受影响的相似群及其大小被记录在相似群映射和整体相似映射中。例如,在将工作分配给工作器时,计算和使用映射的大小。

[0069] 在该示例中,大小是指内存中的表示受影响的相似群所需的位(bit)。因此,每个相似群和子群可以与大小相关联。将所有相似群的子群的大小一起合计到整体映射中,可以确定整体的总大小。使用大小,可以为每个工作器分配一系列可以有效处理的相似群。在一个实施方式中,相似群的所有子群被分配给相同的工作器以简化确定哪个工作器处理哪些相似群和子群。可以使用例如散列表来实现映射。每个相似群和子群的大小可以指对象存储器内的使用散列表、布隆过滤器或完美散列向量来表示相似群的指纹所需的大小或位(或字节)。

[0070] 在记录相似群及其大小之后,可以划分610相似群。例如,可以从最低相似群ID到最高相似群ID对总体相似映射排序。该过程可遍历映射并将相似群ID分配给工作器,直到当前分配的相似群的大小对于工作器来说太大。在这种情况下,划分结束并为下一工作器确定划分。当前分配将相似群的所有子群分配给工作器,并将连续的相似群标识符分配给工作器,但其他分配技术也是可能的。然后可以移除612删除记录。

[0071] 删除记录的内容可以影响确定受影响的相似群的数量的操作。例如,如果删除记录包含对象配方或对象名称,则这将允许识别受影响片的数量,并给出受影响的相似群数量的上限。这也可以减少对被删除记录所引用的独有相似群进行计数所需的时间,因为列出片涉及潜在的昂贵的对象存储操作。

[0072] 在估计用于垃圾收集操作的工作器的数量的上下文中,工作器的数量还可取决于工作器容量。容量可取决于内存、IO操作、吞吐量等。这些因素也可以被纳入确定用于垃圾收集过程的工作器的数量的过程中。

[0073] 分配给工作器在其上运行的一个或多个节点的内存可受到限制或约束,并且可以在估计用于垃圾收集操作的工作器的数量时考虑。在这种情况下,可以通过基于工作器的内存分配来估计工作器的工作容量而估计工作器的数量。

[0074] 例如,工作器可受到内存的限制。相似群引用一个或多个压缩区,每个压缩区可具有一个片段或超过1000个片段,每个片段的大小约为8KB。相似群用它引用的每个压缩区名称来记录与压缩区中的片段相对应的指纹和片段大小的列表。工作器为分配给工作器的每个相似群和子群维护每个指纹的记录,以便它可以确定从这些相似群中引用的每个片段的存活性。相似群子群当前的总大小被限为8MB。每个工作器的工作容量(或工作器可以处理的相似群的数量)可以如下地确定或估计508:

[0075] $$\frac{\text{工作器节点内存}}{8\,\text{MB}}$$

[0076] 在进一步的扩展中,不是将指纹记录在每个相似群和子群的散列表中,而是可以将指纹记录在布隆过滤器中。这将内存需求从每个相似群8MB减少到大约400KB,因为布隆过滤器是紧凑的集合成员结构。可以使用完美散列向量代替布隆过滤器,这将内存需求减少到大约130KB。

[0077] 一旦已经计算了每个工作器的工作容量和受影响的相似群的总数量两者,则可以如下地计算垃圾收集操作所需的工作器的数量:

[0078] $$\left\lceil \frac{\text{受影响的相似群的数量}}{\text{每个工作器可以处理的相似群的数量}} \right\rceil$$

[0079] 在进一步的扩展中,不是假设所有相似群的最大可能大小为8MB,而是相似群和子群的大小可以由控制器在计算表示相似群和子群的指纹在工作器内部所需的内存时确定。该大小基于所选的表示(例如散列表、布隆过滤器或完美散列向量)来修改。该大小是总计的并除以每个工作器可以具有的内存的量,以确定要分配的工作器的数量。

[0080] 在其他示例中,垃圾收集操作或保护系统的各方面可能受到IO(输入/输出操作)的约束,并且该约束也可能影响垃圾收集操作所需的工作器的数量。在该示例中,可以以有效或最佳使用分配给工作器节点的IO的方式来确定工作器的数量。

[0081] 在一个示例中,分配给工作器舱在其上运行的节点的IO可以与允许垃圾收集操作运行的时间长度组合。为了估计垃圾收集操作期间发生的IO操作的数量,可以区分保护系统中发生的IO的类型。例如,存在与打印日志、在服务之间发送RPC调用以及对对象存储器的调用相关联的IO操作。在这些类型的IO操作中,对象存储器的延迟占主导地位。这允许本发明的实施方式聚焦于对象存储器调用以获得对总IO操作的估计。

[0082] 在一个示例中,估计或确定清理单个相似群所需的IO操作的数量。可以有1个IO操作来读取相似群,1个IO操作来写入相似群,以及1个IO操作来读取每个压缩区。随着压缩区被清理,可以假设每写入1个压缩区而读取2个压缩区。这是2:1的压缩区读取对写入比率。接下来是对旧压缩区的删除调用,这与IO操作的数量和压缩区读取的数量大致相同。

[0083] 关于每个相似群引用的压缩区的数量可以做出假设。例如,相似群可以包括标识片标识符的大约8MB的值。~8MB的片包含大约1024个8KB的片段。假设在重复数据删除期间移除了这些片段中的50%,则每个压缩区中输入或写入约512个片段。相似群引用的每个压缩区都具有名称和一定数量的指纹引用(SHA1散列为20个字节,大小为4个字节)。因此,每个片段指纹需要24个字节。因此,压缩区在相似群中需要大约512*24=12288个字节。在~8MB时,相似群除以12288个字节意味着相似群可能引用~683个压缩区。可能还需要考虑在“标记活指纹”阶段期间读取的片。作为估计,假设每个压缩区有一个片读取是合理的。

[0084] 该信息允许如下地估计清理相似群所需的IO操作的数量:

[0085] 1(读取相似群)+683(用于压缩区读取)+

[0086] 683(用于片读取)+1(写入相似群)+

[0087] 342(用于压缩区写入)+683(用于压缩区删除)+

[0088] 1(删除旧相似群)=2394次IO操作

[0089] 在估计清理相似群所需的总IO操作之后,有必要对在运行时受影响的相似群计数以确定需要多少IO操作来清理所有受影响的相似群。IO操作次数的估计可以基于相似群和子群的大小来调整。小于整(full)8MB或小于定义的最大大小的相似群和子群将需要比给出的示例更少的IO操作。

[0090] 一旦确定或估计了IO操作的总数,就可以基于工作器节点的性能特征来决定工作器的数量,所述性能特征决定了该特定节点的潜在IOPS(每秒输入输出操作)以及完成垃圾收集运行所期望的时间。在一个示例中,IO操作通常受到网卡、CPU或处理器等的限制。在一个示例中使用离线分析,可以确定每个实例可以支持的IOPS的数量,并将其用作计算的输

入。

[0091] 利用该信息,可以如下地估计工作器的数量:

[0092] $$\left\lceil \frac{\text{总I/O操作}}{\text{每个节点的IOPS*以秒计的完成GC（垃圾收集）运行的时间}} \right\rceil$$

[0093] 在另一个示例中,可以通过改变一些假设来调整这些方法。在一个示例中,可以使用计数器来跟踪垃圾收集操作期间的IO操作。这允许在配置文件中更新IO操作计数,该配置文件可以在随后的垃圾收集操作期间使用。这是反馈环路的示例,该反馈环路允许基于先前数据来提高IO操作估计的准确性。

[0094] 利用该信息,可以估计工作器的数量并且控制器可以完成创建工作器。图7示出可用于确定或估计工作器的数量的因素的示例。估计工作器702可以依赖于包括环境变量704、工作器内存706、受影响的相似群708和/或IO操作710中的一者或多者的因素。

[0095] 应当理解,估计工作器的数量可以以任何组合使用这些属性中的一者或多者。在进一步的示例中,可以使用每个属性来计算工作器的数量,并且可以为每个属性分配所估计的工作器的最小、平均或最大数量。

[0096] 在一个示例中,垃圾收集操作可以聚焦于清理相似群和相似群引用的压缩区。因为相似群的ID在给定范围内(例如,0到40亿),所以多个相似群可以在工作器之间均匀拆分(基于数量和/或预期大小)。该拆分可以记录在控制器和工作器之间共享的表中。举例来说,控制器和工作器可以使用RPC(远程过程调用)调用来相互通信。

[0097] 回到图4,可以在启动阶段402之后标记404受影响的相似群。为了知道要从对象存储桶中清理哪些数据结构,分析删除记录以识别与删除记录相关联的片配方和受影响的相似群。因为相似群与片相关联,所以识别片允许标记或识别相关联的相似群。

[0098] 图8A示出处理相似群的示例。在一个示例中,在启动控制器之后(并且在实例化工作器之前),控制器可以评估删除记录以便识别受影响的相似群、确定受影响的相似群/子群的大小、以及准备工作器分配。

[0099] 图8A示出控制器访问802删除记录。删除记录允许控制器检索或访问804片配方。因为每个片配方都与特定的相似群相关联,所以可以识别受影响的子群并且可以确定806它们的大小。可以假设或实际确定每个受影响的子群的大小。在确定受影响的相似群以及相似群和受影响的子群的大小时,可以存储各种大小。例如,可以存储每个受影响的子群的大小、一受影响的相似群的大小、和所有受影响的相似群的大小。

[0100] 基于受影响的相似群的大小和/或数量,控制器可以准备工作器分配808。换句话说,相似群被划分并分配给工作器。这些分配或划分有效地将一组相似群分配给每个计划的工作器。换句话说,控制器可以估计需要的工作器的数量并为这些工作器中的每一者准备分配。相似群可以基于大小而均匀分布,使得分配给每个工作器的相似群的数量可以变化等。替选地,相似群可以分布成使得对于每个工作器它们的大小大致相等。

[0101] 接下来,工作器被实例化并且进行分配810。在该示例中,工作器可以与控制器通信以获得它们分配的相似群和子群的列表。基于相关联的大小,工作器可以创建映射结构来跟踪片段的指纹。这允许识别活片段,从而可以结转活片段。

[0102] 图8B示出识别或标记受影响的相似群的另一示例。例如,由控制器检索或接收820

删除记录(例如,作为列表)。然后控制器可以基于为工作器分配的相似群向工作器中的一者发出调用(例如,RPC调用),使得该工作器接收删除记录列表的至少一部分。

[0103] 从删除记录检索822或列出片配方。然后工作器检索822被删除对象和相关联相似群的片配方。工作器通常负责从对象存储器中清理删除记录中识别的片配方。更具体地,每个片配方的名称包括每个片配方所引用的相似群。

[0104] 这允许只要相似群落入分配给工作器的一系列相似群内,工作器就标记824相似群。如果相似群不在为工作器分配的一系列相似群内,则工作器可以对适当的工作器进行调用,以便被调用的工作器可以标记相似群。

[0105] 作为标记的一部分,可以映射相似群/子群并确定其大小826。换句话说,可以生成将受影响的相似群映射到大小的映射。更具体地,该操作或过程导致将受影响的相似群映射到保持在后续阶段中使用的活片段、活片的数量和活片段的数量的数据结构,后续阶段包括标记活指纹408和拷贝转发410阶段。

[0106] 垃圾收集可以是时间密集的过程。因此,在相似群被标记404之后,受影响的相似群被写锁定406,如图4所示。因此,为了允许客户端进行正常操作,受影响的相似群被写锁定。这允许垃圾收集操作与正常操作在同一时间或同时处理。

[0107] 例如,垃圾收集操作可能影响作为正常写入操作的主体的相似群。这可能导致移除写入操作引用的片段或其他问题。这可以通过写锁定受影响的相似群来防止。

[0108] 更具体地,为了防止或减少延迟以及为了允许同时发生的写入操作,相似群可以包括子群。通常,正常的写入操作被定向到编号最高的子群(例如,因为其他子群是满的)。如果相似群中编号最高的子群被标记为要清理,则将新的子群添加到相似群中以用于传入的写入。因此,没有传入的写入操作将引用受影响的相似群,并且这些操作可以同时执行。

[0109] 图9示出相似群的示例。相似群902与子群1到子群N相关联。如图9所示,每个子群与同一相似群902相关联,并且每个子群可以是不同的对象。相似群ID 904表示相似群的每个子群都与同一相似群ID相关联。在该示例中,子群N是编号最高的子群。如果子群N是受影响的子群,则添加新的子群N+1。每个子群作为单独的对象存储在对象存储器中。

[0110] 更具体地,对象的每个片基于片内的数据的函数被映射到相似群。在一个示例中,该函数通常产生在1和40亿之间的标识符(ID)。片通常仅针对具有同一相似群ID的相似群和最高子群进行重复数据删除。当子群达到阈值大小(例如,8MB)时,形成具有相同的相似群ID、但子群ID递增的空的相似群。映射到相似群的未来片将针对当前子群ID进行重复数据删除。这确保新的写入不会干扰垃圾收集操作正在清理的受影响的相似群或子群。

[0111] 这可能导致重复数据删除中的潜在损失,因为子群开始为空的。然而,可以移除空的子群,因为一旦被清理,针对相似群和/或子群进行重复数据删除是安全的。替选地,可以由元数据服务器执行重复数据删除任务,以便通过与垃圾收集工作器通信来标记适当的指纹。

[0112] 回到图4,在锁定受影响的相似群之后,在受影响的相似群中标记408活指纹。图10示出标记活指纹的示例。还使用图10来示出用于标记活指纹的方法。

[0113] 最初,控制器1002可以获得活片配方的列表。这可以通过收集存储系统中的所有重复数据删除域标识符来实现。在一个示例中,重复数据删除域标识符是与用户相关联的唯一标识符。该用户存储在对象存储器中的每个对象都包含对重复数据删除域标识符的引

用。出于租户隔离、隐私和安全的考虑,新对象仅针对与相同重复数据删除域标识符相关联的其他对象进行重复数据删除。例如,用户可以指实体或组织。该信息也可以从元数据服务器获得。然后确定重复数据删除域标识符的所有对象配方,并从每个对象配方列出活片配方。在获得活片配方的列表之后,控制器可以基于先前的分配将片配方分配给工作器(例如,工作器1004和工作器1006)。例如,分配给工作器1004的片是对应于控制器1002分配给工作器1004的相似群系列的那些片。

[0114] 更具体地,控制器1002可以解析或分析片配方名称以确定相似群ID和子群ID。利用相似群ID和子群ID,控制器1002查看其工作器表以识别其分配的相似群系列包含所确定的相似群ID的工作器。然后将片推入工作器的活片通道1036(例如,队列)。每个工作器都有自己的活片通道,并且该映射由控制器使用工作器的IP地址来管理。一旦控制器完成浏览所有活片配方并将活片配方推送到其各自的工作器通道,则控制器可以关闭所有的工作器通道。

[0115] 同时,工作器1004(和其他工作器)调用控制器1002并从控制器1002将活片配方放入的通道1036中获取一批片配方。工作器1004将继续从通道1036中批量拉出活片配方,直到通道为空。利用活片配方列表,工作器1004确定相似群ID。利用相似群ID,工作器1004检查相似群是否被标记为要清理或者是否是受影响的相似群。如果相似群被标记,则工作器1004读取相关联的片配方并在内部活片段结构1034(例如布隆过滤器)中记录活指纹的列表。该活片段结构1034可以被配置为包含信息,例如活片的数量、活片段的数量以及哪些片段是活的列表。为了减少内存需求,可以在散列表、布隆过滤器或完美散列向量中表示片段的列表。工作器1004可以为工作器负责的每个受影响的相似群维护片段结构的列表。在所有工作器已经浏览它们的活片配方的列表后,每个活片段结构都已完全更新。

[0116] 图10示出在标记活片段的阶段中的保护系统。例如,对象X可以由片段1、2、3、4和5形成。对象X可能如图3所示已经被删除。图10示出在对象存储器1020中,受影响的相似群1030包括相似群A,子群1。如果这是最高的子群,则将如前文所述在垃圾收集操作期间创建新的子群2。

[0117] 相似群A与包括CR 3和CR 4的压缩区(CR)相关联,CR 3包括指纹1、2、3、4、5和相应的片段,CR 4包括指纹6、7和相应的片段。

[0118] 对象Y没有被删除并且对象存储器1020包括对象配方1022和Y的片配方1024,其标识了相似群A、子群1和指纹1、2、5、6和7。

[0119] 因此,对象X和对象Y共享片段1和2。CR 3 1026包括片段1、2、3、4和5,并且CR 4 1028包括片段6、7。

[0120] 当工作器1004从控制器1002检索片配方时,工作器确定片配方是否引用受影响的相似群。如果否,则跳过该片。如果是,则读取片配方并在相似群中标记活指纹。

[0121] 因此,当接收到对象Y的配方时,CR 3中的指纹或片段1、2和5被标记并且CR 4中的片段6和7被标记。这反映在结构1034中,其中片段1、2、5、6和7被标记为活的。

[0122] 参考图4,可以在活片段被标记之后继续拷贝转发410阶段。拷贝转发是确保对象存储器中没有留下未引用的结构或片段的阶段。这有利地降低了存储成本。同时,可能存在某些结构未基于例如活片段与死片段的比率而被清理的情况。

[0123] 在一个示例中,工作器1004处理其片列表和相应的受影响的相似群。每个相似群

都与每个相似群的活片段的映射相关联。因此,结构1034是针对相似群A的映射。对于每个相似群,读取所引用的压缩区并且确定它们是否足够死以便清理或应该留在它们的当前状态。在读取片段指纹的压缩区时,可以创建从压缩区名称到活指纹数量的映射。通过基于活指纹的数量计算活的压缩区的百分比并将该百分比与被认为在压缩区内是足够活的预定义的阈值(例如,85%)比较来确定是否应该清理每个压缩区。如果压缩区中活指纹的百分比降到低于预定义的阈值,则认为该压缩区值得清理。可以调整阈值以优先考虑空间回收或最小化IO成本。

[0124] 对于每个被清理的压缩区,活片段被拷贝以形成新的压缩区。一旦所有新的压缩区形成并记录在新版本的相似群中,则存储该新版本的相似群。元数据服务被提醒驱逐旧的相似群并添加新的相似群。最后,删除旧的相似群和压缩区。这将死片段从对象存储器中移除。

[0125] 垃圾收集操作可以被实现为部分或延迟的标记和清除操作。垃圾收集操作包括从活对象中清理或移除被删除对象。当对象被删除时,记录被记录在删除桶1010(或其他结构)中。以后只要执行垃圾收集操作,就使用删除桶中的记录。垃圾收集操作可以分阶段或以连续的动作步骤操作。本发明的实施方式是聚焦标记和清除垃圾收集,其聚焦于可至少部分地包括死片段的相似群。

[0126] 应当理解的是,可以以很多方式实现本发明,这些方式包括过程、装置、系统、设备、方法、或者诸如计算机可读存储介质的计算机可读介质或计算机网络,在计算机网络中,计算机程序指令通过光通信链接或电通信链接来发送。应用程序可以采用在通用计算机上执行的软件的形式或者可以被硬连线或硬编码在硬件中。在本发明中,这些实现方式、或本发明可以采用的任何其它形式都可以被称为技术。通常,所公开的过程中步骤的顺序可以改变而仍在本发明的范围内。

[0127] 本文所公开的实施方式可以包括使用包括各种计算机硬件或软件模块的专用或通用计算机,如下面更详细讨论的。计算机可以包括处理器和承载指令的计算机存储介质,所述指令在被处理器执行时和/或使得所述指令被处理器执行时,执行本文所公开的方法中的任一者或多者。

[0128] 如上所述,在本发明的范围内的实施方式还包括计算机存储介质,所述计算机存储介质是用于承载或带有其上存储的计算机可执行指令或数据结构的物理介质。这种计算机存储介质可以是能够被通用或专用计算机访问的任何可用的物理介质。

[0129] 以示例而不是限制的方式,这种计算机存储介质可以包括硬件,所述硬件诸如固态硬盘(SSD)、RAM、ROM、EEPROM、CD-ROM、闪存、相变内存(“PCM”)、或者其他光盘存储器、磁盘存储器或其他磁存储设备、或者可用于以计算机可执行指令或数据结构的形式存储程序代码的任何其他硬件存储设备,所述计算机可执行指令或数据结构可以被通用或专用计算机系统访问和执行以实现本发明所公开的功能。以上的组合也应该被包括在计算机存储介质的范围内。这些介质也是非暂时性存储介质的示例,非暂时性存储介质也包含基于云的存储系统和结构,然而本发明的范围未被限定到非暂时性存储介质的这些示例。

[0130] 计算机可执行指令包括例如使得通用计算机、专用计算机、或专用处理设备执行某一功能或某一组功能的指令和数据。尽管已经用特定于结构特征和/或方法动作的语言描述了主题,应理解的是,所附权利要求中限定的主题不必限定到上述特定特征或动作。相

反,本文所公开的特定特征和动作被披露作为实现权利要求的示例性形式。

[0131] 如本文中所使用的,术语“模块”或“组件”可以指在计算系统上执行的软件对象或例程。本文描述的不同的组件、模块、引擎、和服务可以实现为例如作为单独的线程在计算系统上执行的对象或进程。尽管本文描述的系统和方法可以以软件实现,但是以硬件或者软件和硬件的组合实现也是可以的且可预期的。在本发明中,“计算实体”可以是如本文之前所限定的任何计算系统、或者在计算系统上运行的任何模块或模块组合。

[0132] 在至少一些实例中,提供硬件处理器,该硬件处理器可操作成执行用于执行方法或过程(诸如本文所公开的方法和过程)的可执行指令。该硬件处理器可以包括或不包括其他硬件、诸如本文所公开的计算设备和系统的元件。

[0133] 在计算环境方面,本发明的实施方式可以在客户端-服务器环境(无论是网络环境还是本地环境)、或任何其他合适的环境中执行。用于本发明的至少一些实施方式的合适的操作环境包括云计算环境,在云计算环境中,客户端、服务器、和目标虚拟机中的一者或多者可以驻留并操作在云环境中。

[0134] 本发明可以以其他特定形式实现而不脱离其精神或实质特性。所描述的实施方式在所有方面均仅作为说明性的而非限制性的被考虑。因此,本发明的范围由所附权利要求而不是由之前的描述来指示。落入权利要求的等价含义和等价范围内的所有改变均被包含在权利要求的范围内。

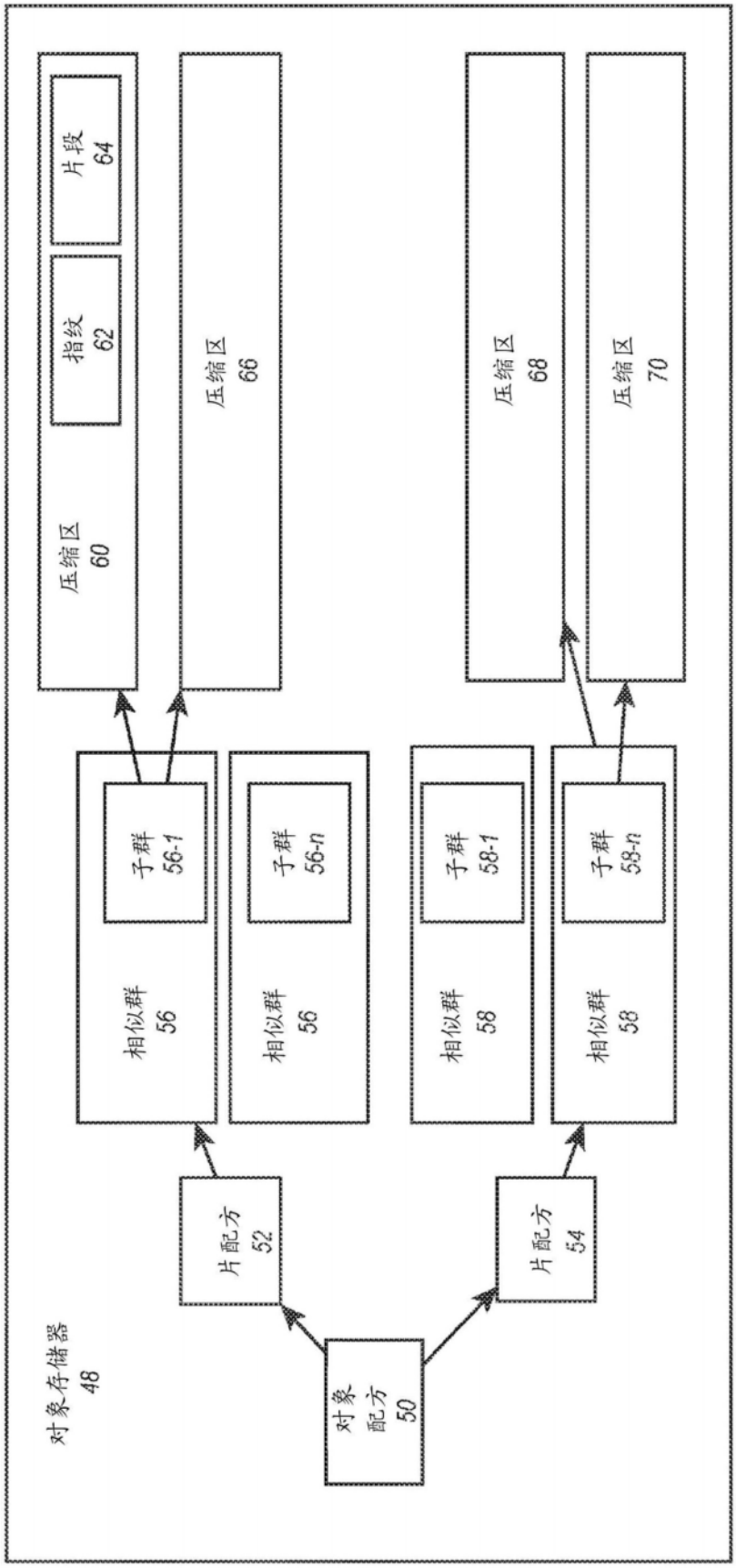

对象存储器
48
对象
配方
50
片配方
52
片配方
54
相似群
56
子群
56-1
子群
56-n
相似群
58
子群
58-1
子群
58-n
压缩区
60
指纹
62
片段
64
压缩区
66
压缩区
68
压缩区
70

图1A

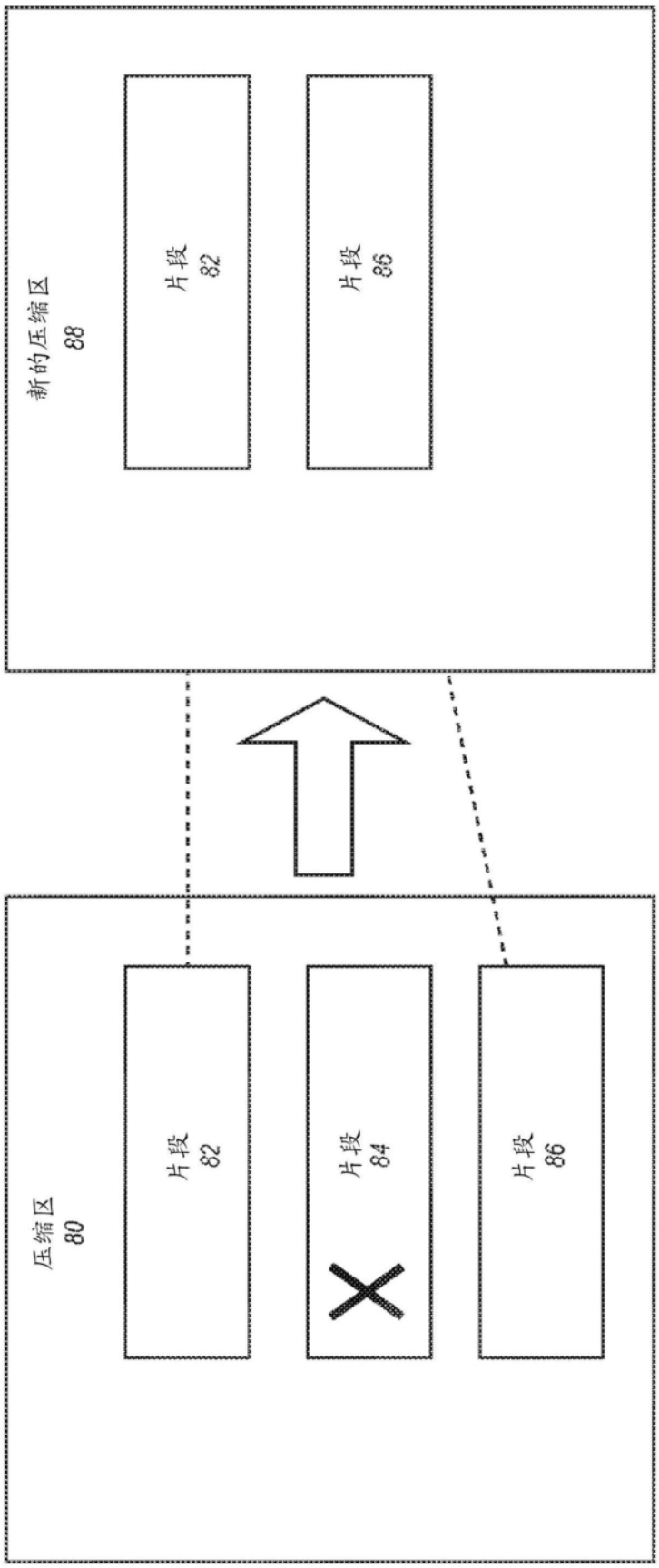


图1B

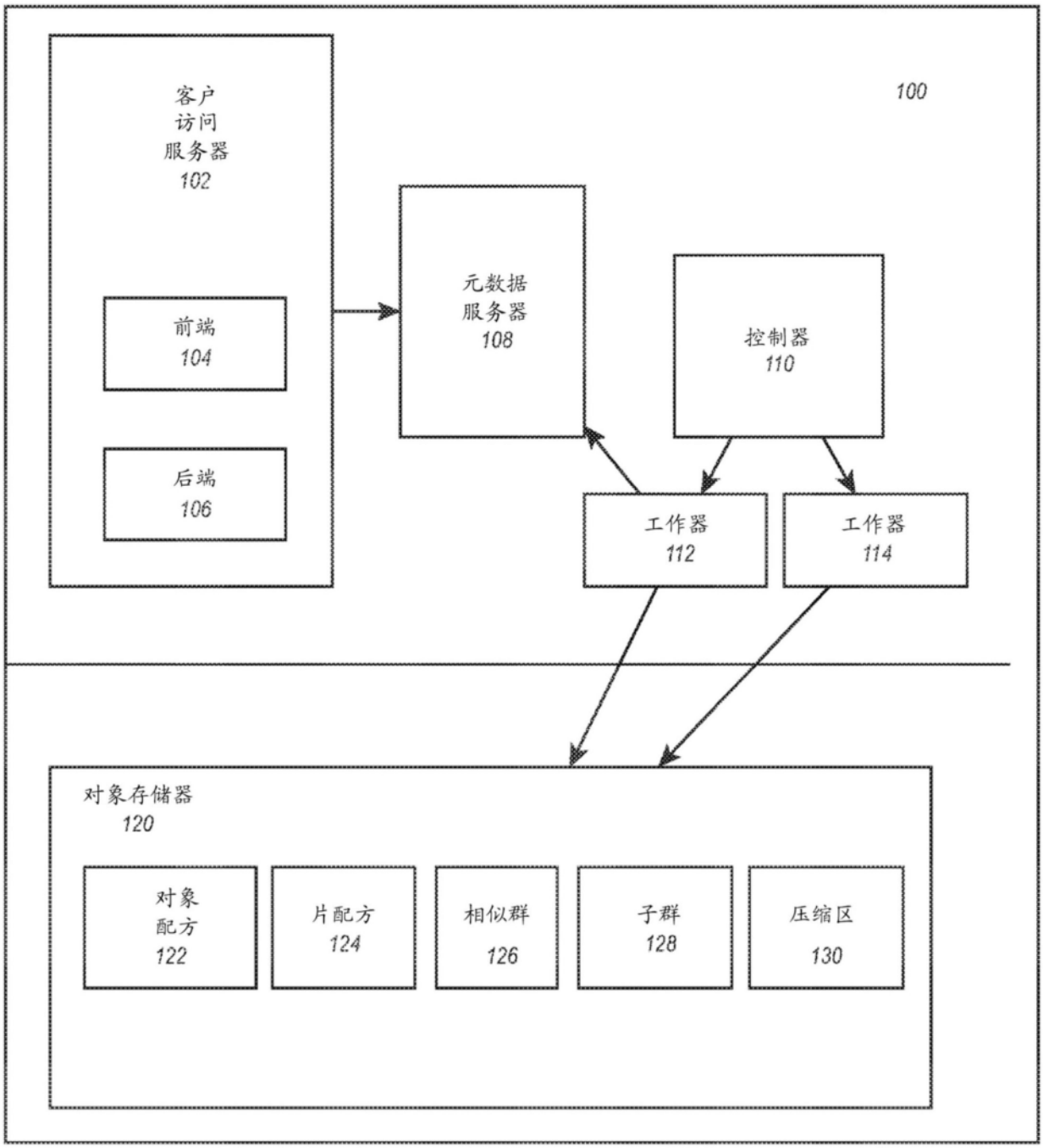

客户
访问
服务器
102
100
前端
104
后端
106
元数据
服务器
108
控制器
110
工作器
112
工作器
114
对象存储器
120
对象
配方
122
片配方
124
相似群
126
子群
128
压缩区
130

图1C

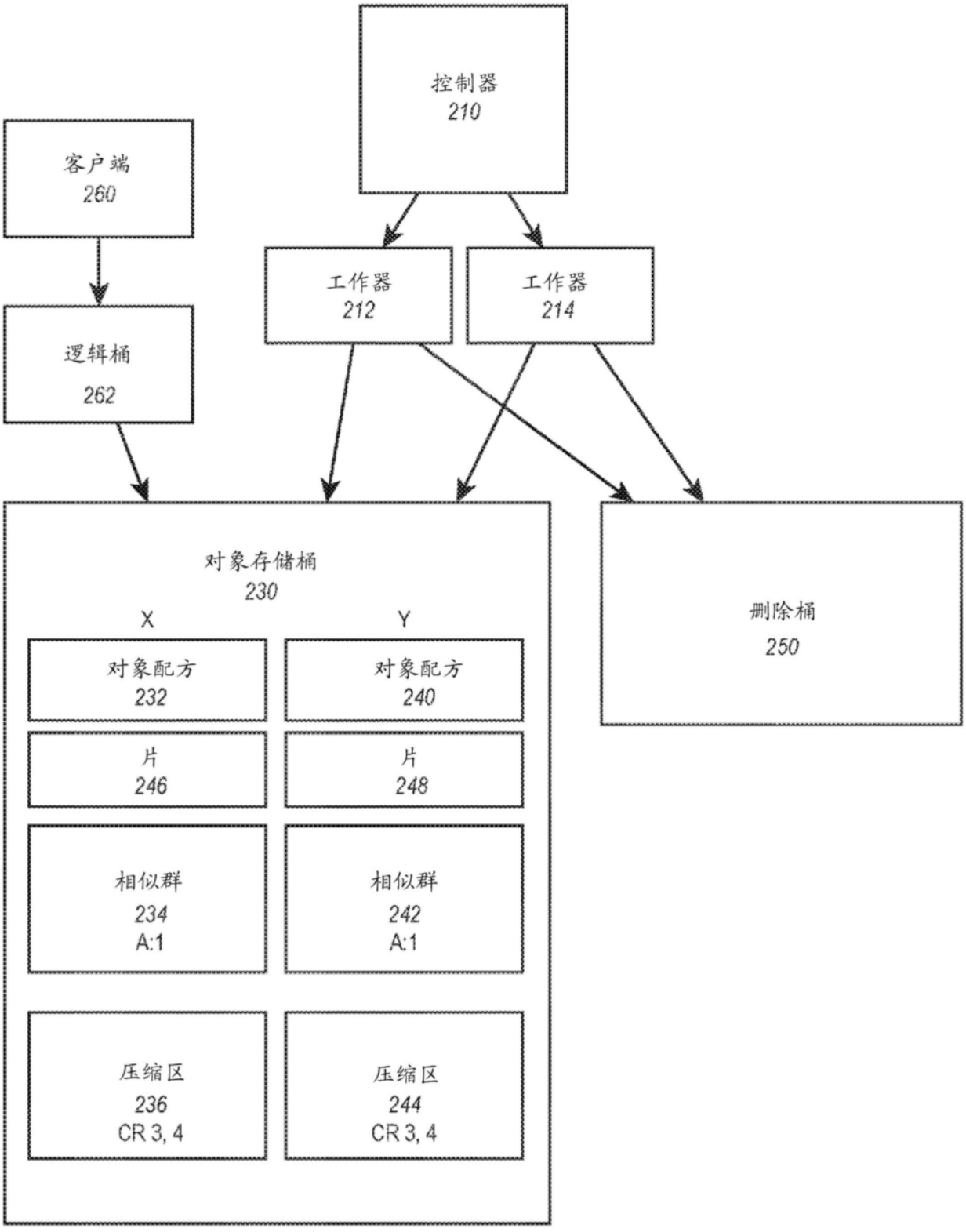

控制器
210
客户端
260
工作器
212
工作器
214
逻辑桶
262
对象存储桶
230
X
Y
对象配方
232
对象配方
240
片
246
片
248
相似群
234
A:1
相似群
242
A:1
压缩区
236
CR 3, 4
压缩区
244
CR 3, 4
删除桶
250

图2

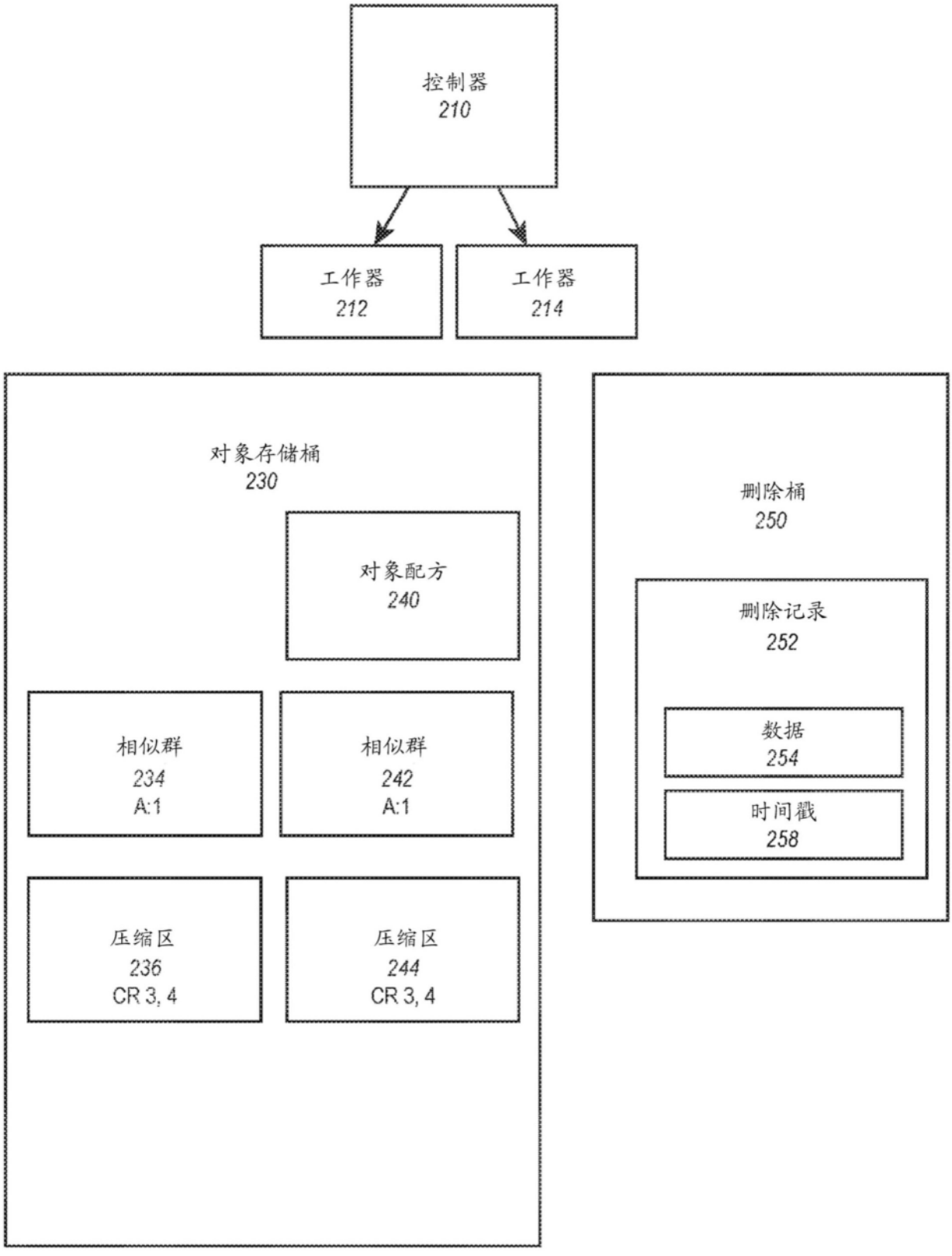

控制器
210
工作器
212
工作器
214
对象存储桶
230
对象配方
240
相似群
234
A:1
相似群
242
A:1
压缩区
236
CR 3, 4
压缩区
244
CR 3, 4
删除桶
250
删除记录
252
数据
254
时间戳
258

图3

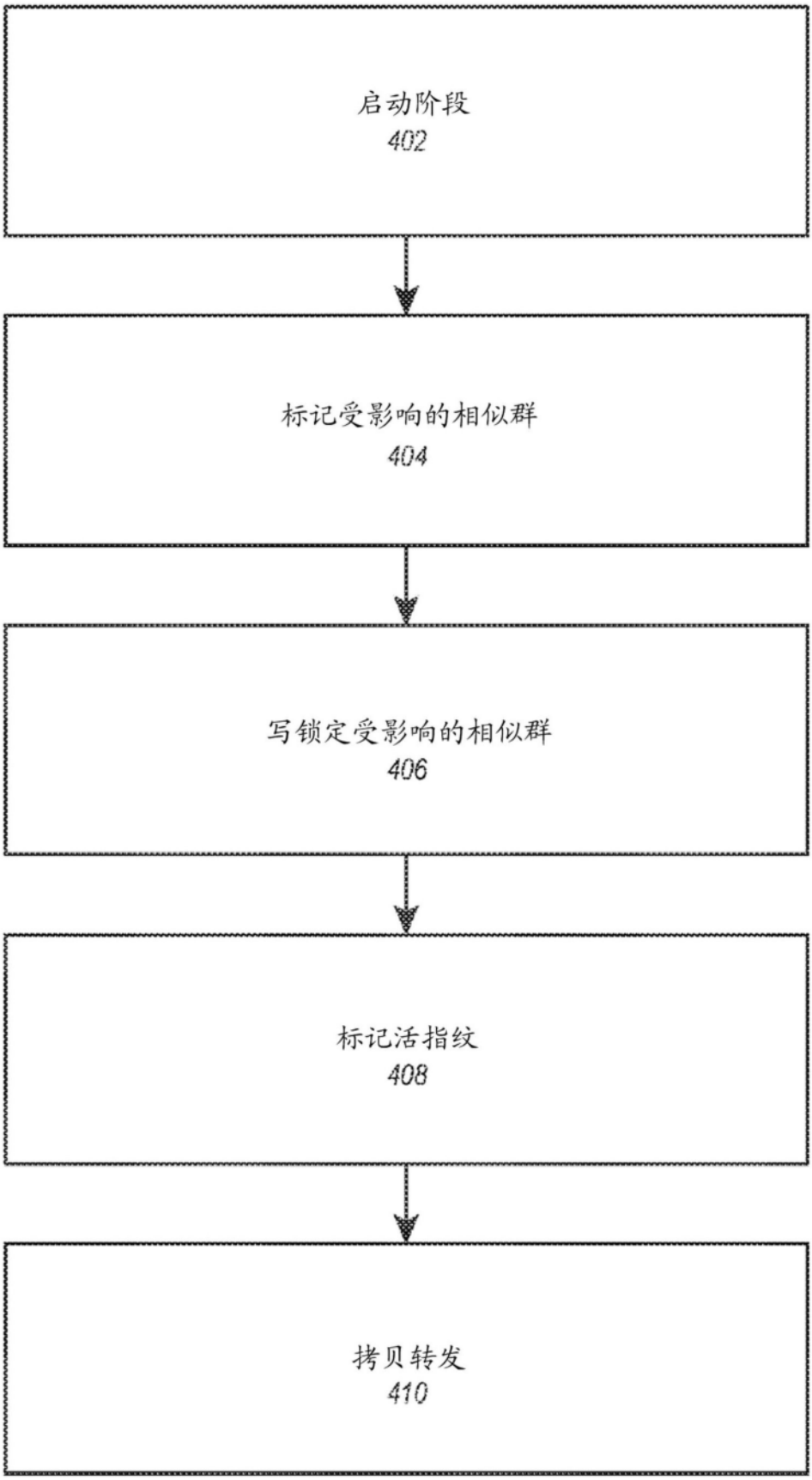


图4

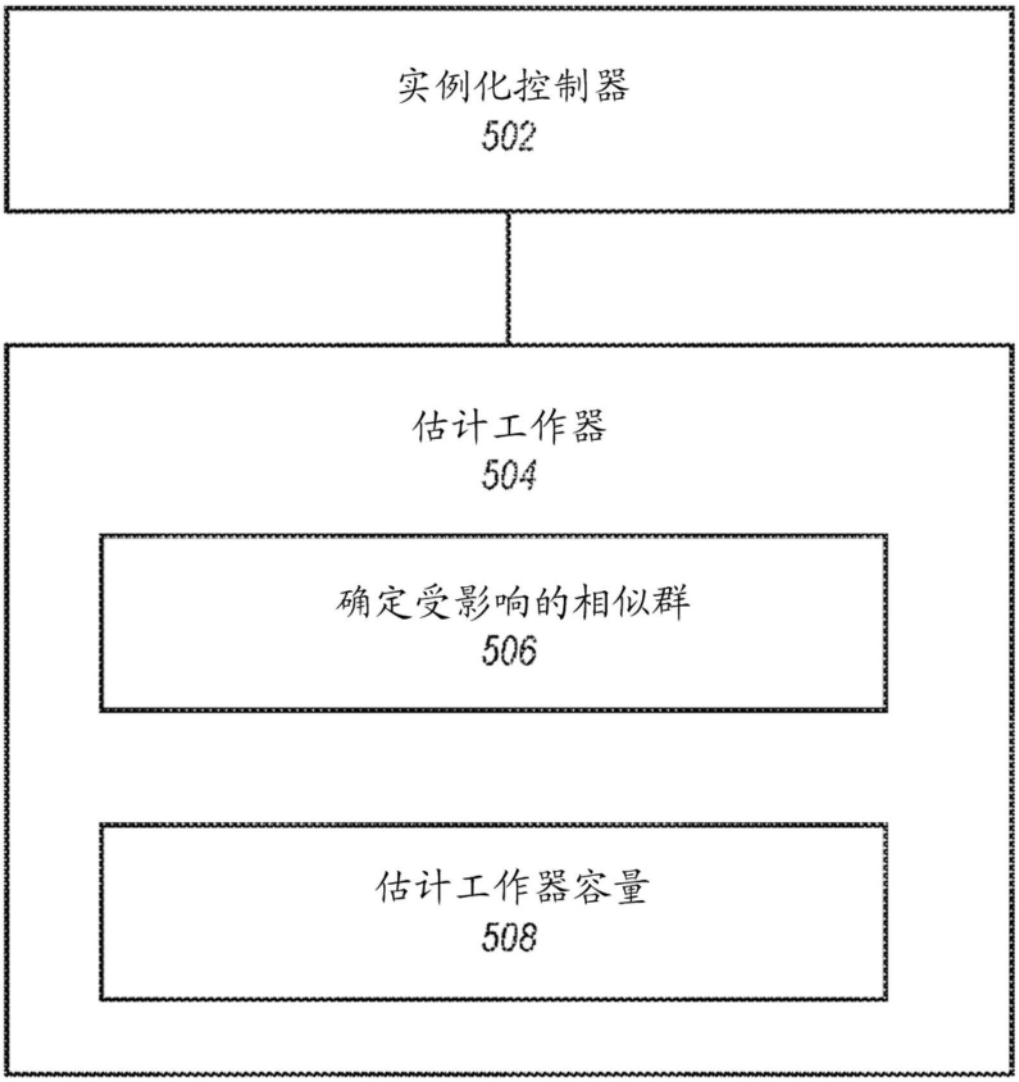


图5

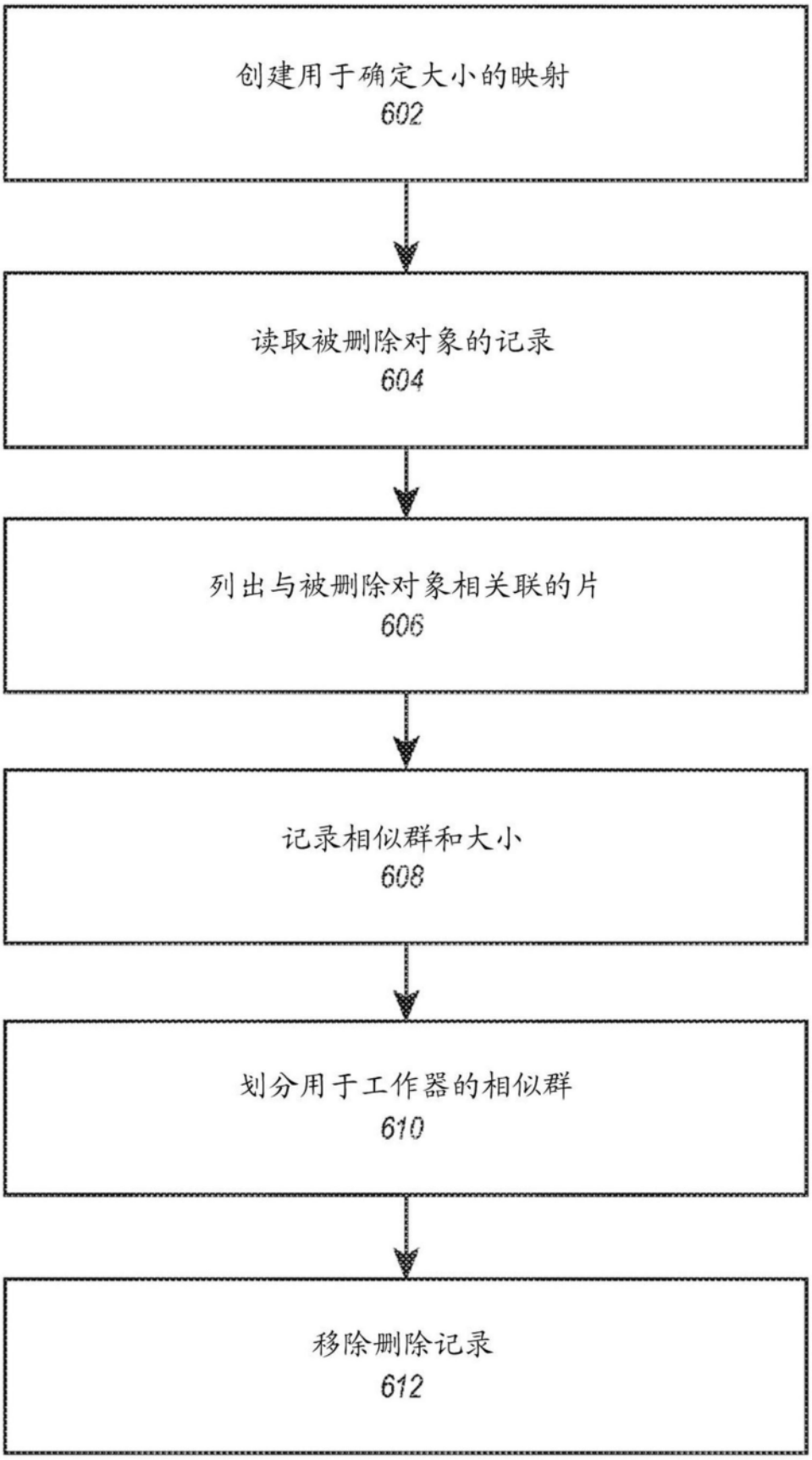

创建用于确定大小的映射
602
读取被删除对象的记录
604
列出与被删除对象相关联的片
606
记录相似群和大小
608
划分用于工作器的相似群
610
移除删除记录
612

图6

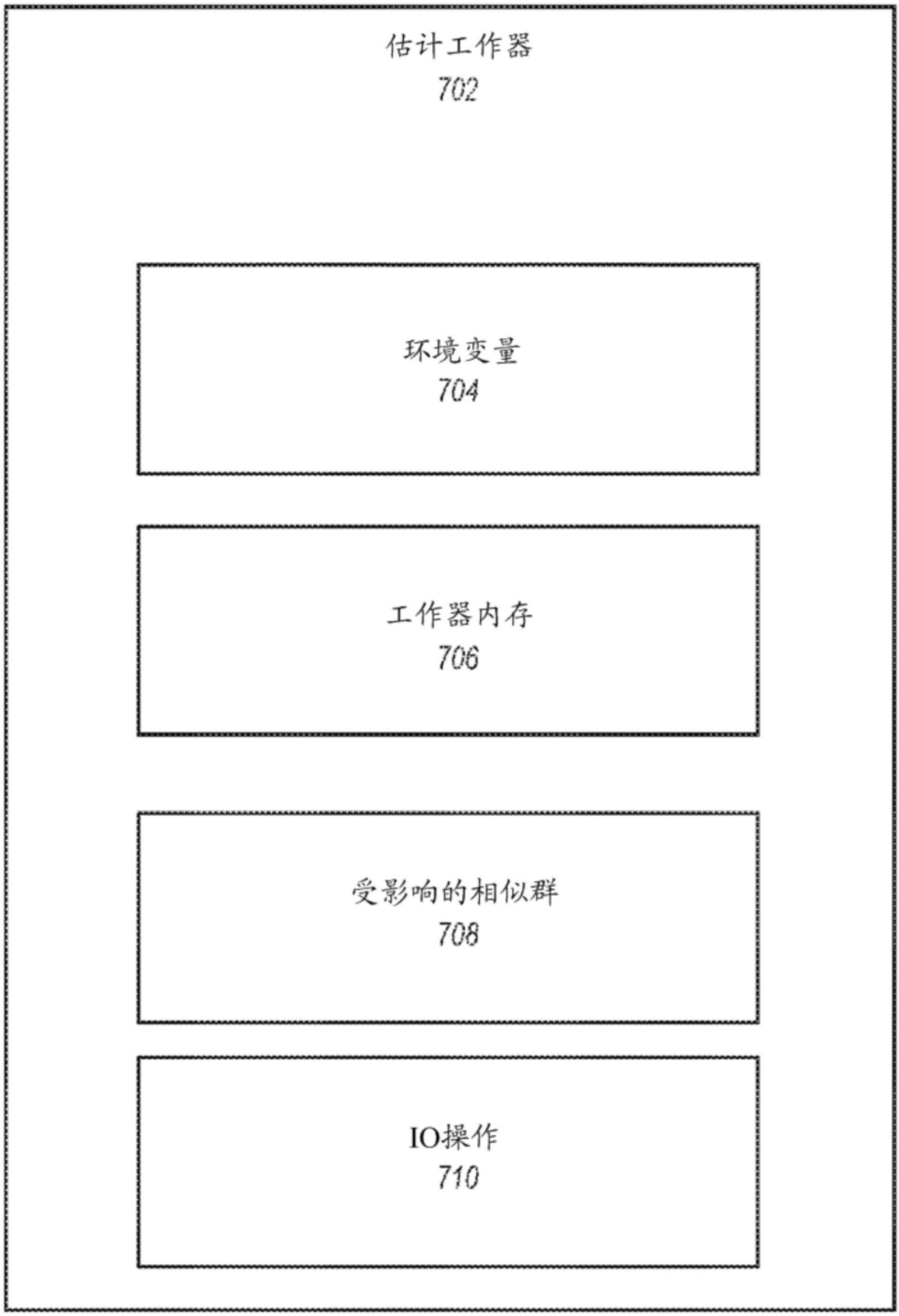


图7

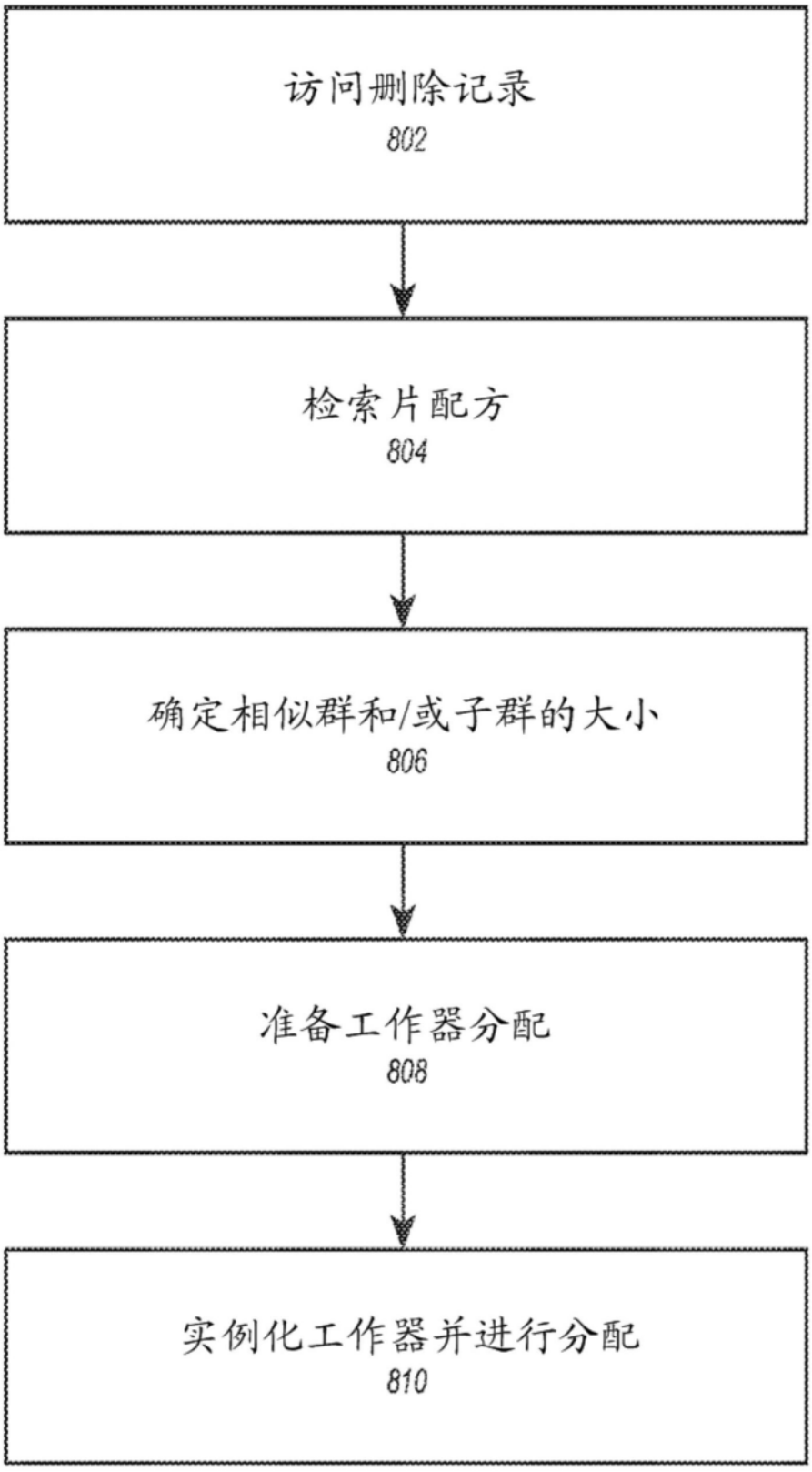

访问删除记录
802
检索片配方
804
确定相似群和/或子群的大小
806
准备工作器分配
808
实例化工作器并进行分配
810

图8A

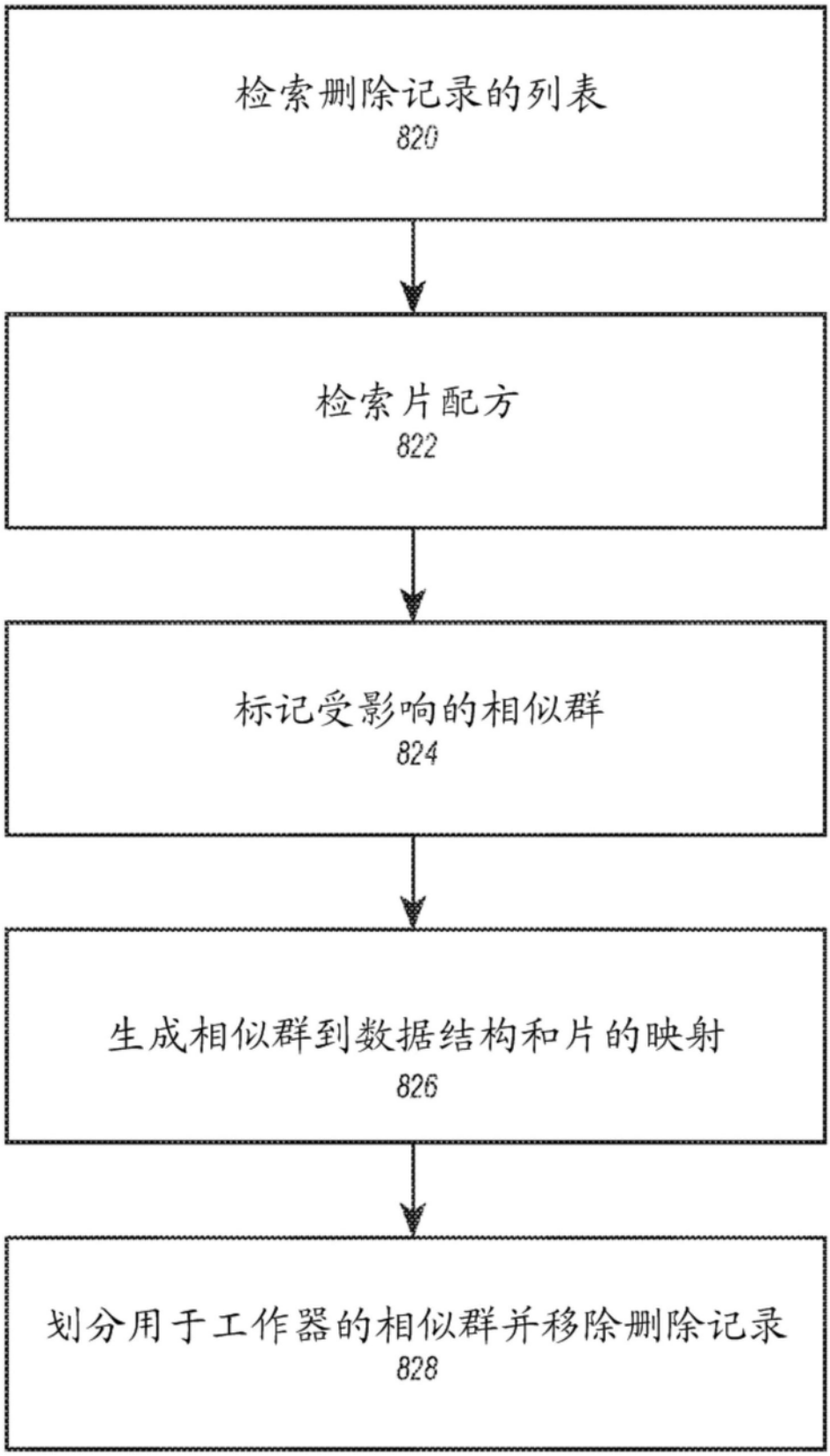

检索删除记录的列表
820
检索片配方
822
标记受影响的相似群
824
生成相似群到数据结构和片的映射
826
划分用于工作器的相似群并移除删除记录
828

图8B

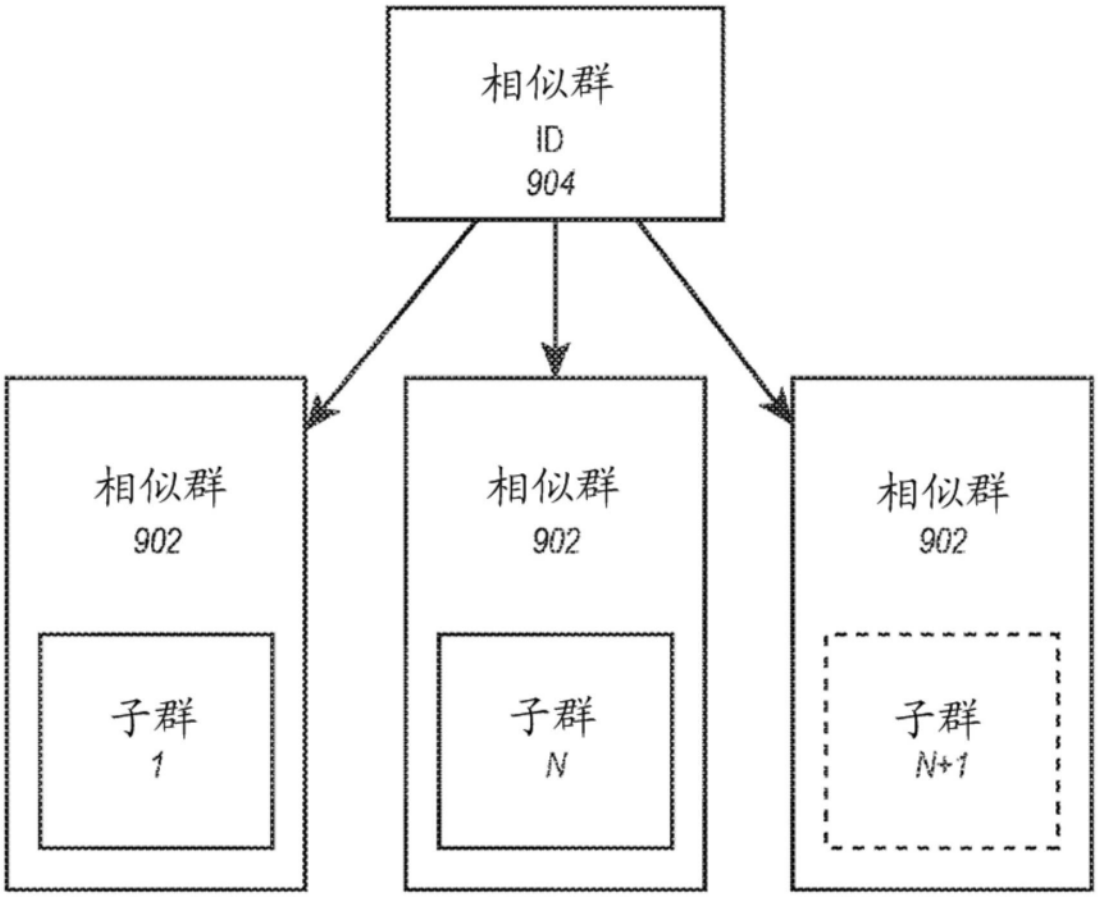


图9

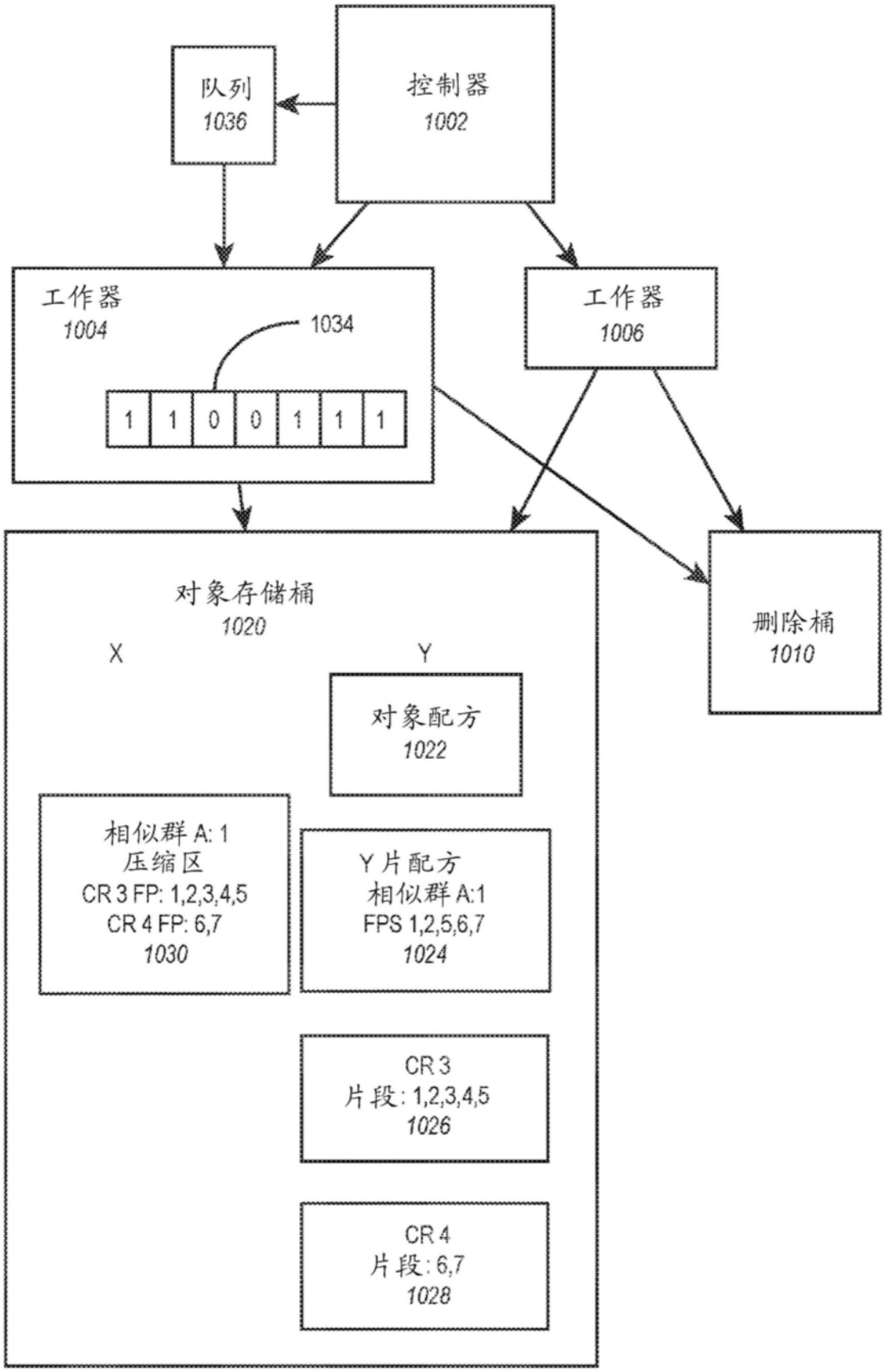

队列
1036
控制器
1002
工作器
1004
1034
1 1 0 0 1 1 1
工作器
1006
对象存储桶
1020
X
Y
删除桶
1010
对象配方
1022
相似群 A: 1
压缩区
CR 3 FP: 1,2,3,4,5
CR 4 FP: 6,7
1030
Y 片配方
相似群 A:1
FPS 1,2,5,6,7
1024
CR 3
片段: 1,2,3,4,5
1026
CR 4
片段: 6,7
1028

图10