(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 109697016 B
(45)授权公告日 2022.02.15

(21)申请号 201710981975.0
(22)申请日 2017.10.20
(65)同一申请的已公布的文献号
申请公布号 CN 109697016 A
(43)申请公布日 2019.04.30
(73)专利权人 伊姆西IP控股有限责任公司
地址 美国马萨诸塞州
(72)发明人 赵军平 郭帆 王鲲
(74)专利代理机构 北京市金杜律师事务所 11256
代理人 王茂华 张曦
(51)Int.Cl.
G06F 3/06(2006.01)

(56)对比文件
US 2013132967 A1,2013.05.23
US 2013132967 A1,2013.05.23
US 9588976 B1,2017.03.07
US 2005165975 A1,2005.07.28
EP 0766177 A1,1997.04.02
US 9256542 B1,2016.02.09
CN 101213510 A,2008.07.02
CN 102713864 A,2012.10.03
CN 101986285 A,2011.03.16
US 2013326181 A1,2013.12.05

审查员 王巧玲

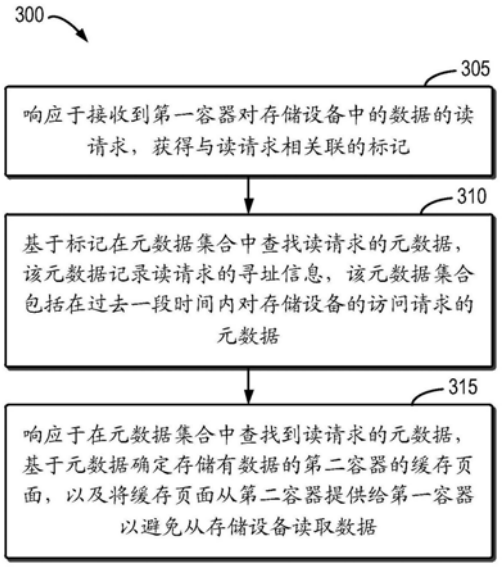
权利要求书3页 说明书12页 附图7页

(54)发明名称

用于改进容器的存储性能的方法和装置

(57)摘要

本公开的实施例提供了一种计算机实现的方法和用于存储系统的装置。该方法包括:响应于接收到第一容器对存储设备中的数据的读请求,获得与读请求相关联的标记;基于标记在元数据集合中查找读请求的元数据,元数据记录读请求的寻址信息,元数据集合包括在过去一段时间内对存储设备的访问请求的元数据;以及响应于在元数据集合中查找到读请求的元数据,基于元数据确定存储有数据的第二容器的缓存页面;以及将缓存页面从第二容器提供给第一容器以避免从存储设备读取数据。



CN 109697016 B

1.一种计算机实现的方法,包括:

响应于接收到第一容器对存储设备中的数据的读请求,获得与所述读请求相关联的标记;

基于所述标记在元数据集合中查找所述读请求的元数据,所述元数据记录所述读请求的寻址信息,所述元数据集合包括在过去一段时间内对所述存储设备的访问请求的元数据;以及

响应于在所述元数据集合中查找到所述读请求的所述元数据,

基于所述元数据确定存储有所述数据的第二容器的缓存页面;以及

将所述缓存页面从所述第二容器提供给所述第一容器以避免从所述存储设备读取所述数据;

其中获得与所述读请求相关联的所述标记包括:

将与所述读请求相关联的虚拟设备标识符和虚拟逻辑块编号转换为与所述读请求相关联的真实设备标识符和真实逻辑块编号。

2.根据权利要求1所述的方法,进一步包括:

响应于未在所述元数据集合中查找到所述元数据,从所述存储设备读取所述数据;以及

将所述元数据添加到所述元数据集合。

3.根据权利要求1所述的方法,其中基于所述标记在元数据集合中查找所述读请求的元数据包括:

选择与所述真实设备标识符相对应的索引表;以及

使用所述真实逻辑块编号作为关键值在所述索引表中查找所述元数据。

4.根据权利要求1所述的方法,其中所述元数据包括位于块层的元数据和位于文件层的元数据。

5.根据权利要求1所述的方法,其中将所述缓存页面从所述第二容器提供给所述第一容器包括:

将所述缓存页面从所述第二容器复制或迁移到所述第一容器。

6.根据权利要求5所述的方法,进一步包括:

基于以下至少一项来确定是将所述缓存页面复制到还是迁移到所述第一容器:所述缓存页面的访问频率、所述缓存页面的访问特性、以及所述缓存页面所在的存储器的剩余空间。

7.根据权利要求5所述的方法,进一步包括:

如果所述缓存页面从所述第二容器被复制到所述第一容器,将所述元数据添加到所述元数据集合;以及

如果所述缓存页面从所述第二容器被迁移到所述第一容器,利用所述元数据来修改所述元数据集合。

8.根据权利要求5所述的方法,进一步包括:

响应于所述缓存页面在被迁移到所述第一容器中之后在所述第一容器中被修改或删除,将所述缓存页面返回给所述第二容器。

9.根据权利要求1所述的方法,进一步包括:

响应于所述第二容器的缓存页面中存储有所述数据并且所述缓存页面的状态为“干净”,将所述缓存页面从所述第二容器提供到所述第一容器。

10.根据权利要求1所述的方法,进一步包括:

响应于所述第二容器的缓存页面中存储有所述数据并且所述缓存页面的状态为“脏”,

删除所述元数据集合中与所述缓存页面相关联的元数据;

从所述存储设备读取所述数据;以及

将所述读请求的所述元数据添加到所述元数据集合。

11.一种用于存储系统的装置,包括:

至少一个处理器;以及

包括计算机程序指令的至少一个存储器,所述至少一个存储器和所述计算机程序指令被配置为,与所述至少一个处理器一起,使得所述装置:

响应于接收到第一容器对存储设备中的数据的读请求,获得与所述读请求相关联的标记;

基于所述标记在元数据集合中查找所述读请求的元数据,所述元数据记录所述读请求的寻址信息,所述元数据集合包括在过去一段时间内对所述存储设备的访问请求的元数据;以及

响应于在所述元数据集合中查找到所述读请求的所述元数据,

基于所述元数据确定存储有所述数据的第二容器的缓存页面;以及

将所述缓存页面从所述第二容器提供给所述第一容器以避免从所述存储设备读取所述数据;

其中获得与所述读请求相关联的所述标记包括:

将与所述读请求相关联的虚拟设备标识符和虚拟逻辑块编号转换为与所述读请求相关联的真实设备标识符和真实逻辑块编号。

12.根据权利要求11所述的装置,其中所述至少一个存储器和所述计算机程序指令进一步被配置为,与所述至少一个处理器一起,使得所述装置:

响应于未在所述元数据集合中查找到所述元数据,从所述存储设备读取所述数据;以及

将所述元数据添加到所述元数据集合。

13.根据权利要求11所述的装置,其中所述至少一个存储器和所述计算机程序指令进一步被配置为,与所述至少一个处理器一起,使得所述装置:

选择与所述真实设备标识符相对应的索引表;以及

使用所述真实逻辑块编号作为关键值在所述索引表中查找所述元数据。

14.根据权利要求11所述的装置,其中所述元数据包括位于块层的元数据和位于文件层的元数据。

15.根据权利要求11所述的装置,其中所述至少一个存储器和所述计算机程序指令进一步被配置为,与所述至少一个处理器一起,使得所述装置:

将所述缓存页面从所述第二容器复制或迁移到所述第一容器。

16.根据权利要求15所述的装置,其中所述至少一个存储器和所述计算机程序指令进一步被配置为,与所述至少一个处理器一起,使得所述装置:

基于以下至少一项来确定是将所述缓存页面复制到还是迁移到所述第一容器:所述缓存页面的访问频率、所述缓存页面的访问特性、以及所述缓存页面所在的存储器的剩余空间。

17.根据权利要求15所述的装置,其中所述至少一个存储器和所述计算机程序指令进一步被配置为,与所述至少一个处理器一起,使得所述装置:

如果所述缓存页面从所述第二容器被复制到所述第一容器,将所述元数据添加到所述元数据集合;以及

如果所述缓存页面从所述第二容器被迁移到所述第一容器,利用所述元数据来修改所述元数据集合。

18.根据权利要求15所述的装置,其中所述至少一个存储器和所述计算机程序指令进一步被配置为,与所述至少一个处理器一起,使得所述装置:

响应于所述缓存页面在被迁移到所述第一容器中之后在所述第一容器中被修改或删除,将所述缓存页面返回给所述第二容器。

19.根据权利要求11所述的装置,其中所述至少一个存储器和所述计算机程序指令进一步被配置为,与所述至少一个处理器一起,使得所述装置:

响应于所述第二容器的缓存页面中存储有所述数据并且所述缓存页面的状态为“干净”,将所述缓存页面从所述第二容器提供到所述第一容器。

20.根据权利要求11所述的装置,其中所述至少一个存储器和所述计算机程序指令进一步被配置为,与所述至少一个处理器一起,使得所述装置:

响应于所述第二容器的缓存页面中存储有所述数据并且所述缓存页面的状态为“脏”,

删除所述元数据集合中与所述缓存页面相关联的元数据;

从所述存储设备读取所述数据;以及

将所述读请求的所述元数据添加到所述元数据集合。

21.一种计算机可读介质,存储机器可执行指令,所述机器可执行指令在被执行时使机器执行根据权利要求1-10中任一项所述的方法的步骤。

用于改进容器的存储性能的方法和装置

技术领域

[0001] 本公开一般地涉及计算机系统或存储系统，并且更特别地，涉及一种用于改进容器的存储性能的方法和装置。

背景技术

[0002] 近来，继虚拟化技术出现后，容器(container)技术逐渐成为对计算领域(特别是云计算领域)具有深远影响的变革技术。容器技术的发展和应用，为应用云计算提供了新思路，同时容器技术也将对云计算的交付方式、效率、PaaS(平台即服务)的构建等方面产生深远的影响。容器(例如，Docker等)使得更轻便、更快捷和更简便的服务开发与运营维护成为可能。尽管容器的性能通常是较好的，但是仍然严重地依赖于工作负载和底层存储驱动器。

[0003] 一般而言，取决于写时复制(Copy-On-Write)在何处实施，容器的存储驱动器可以具有两种类型，即基于文件的类型或基于块的类型。基于文件的类型的存储驱动器例如包括AUFS和Overlay/2等，而基于块的类型的存储驱动器(下文简称块存储驱动器)例如包括DevMapper和ZFS等。然而，已有的用于容器的块存储驱动器在存储性能上仍然存在各种缺陷和不足，在许多场合下无法满足计算系统或存储系统的性能要求。

发明内容

[0004] 本公开提供了一种计算机实现的方法和用于存储系统的装置。

[0005] 在本公开的第一方面，提供了一种计算机实现的方法。该方法包括：响应于接收到第一容器对存储设备中的数据的读请求，获得与读请求相关联的标记；基于标记在元数据集合中查找读请求的元数据，元数据记录读请求的寻址信息，元数据集合包括在过去一段时间内对存储设备的访问请求的元数据；以及响应于在元数据集合中查找到读请求的元数据，基于元数据确定存储有数据的第二容器的缓存页面；以及将缓存页面从第二容器提供给第一容器以避免从存储设备读取数据。

[0006] 在一些实施例中，该方法可以进一步包括：响应于未在元数据集合中查找到元数据，从存储设备读取数据；以及将元数据添加到元数据集合。

[0007] 在一些实施例中，获得与读请求相关联的标记可以包括：将与读请求相关联的虚拟设备标识符和虚拟逻辑块编号转换为与读请求相关联的真实设备标识符和真实逻辑块编号。

[0008] 在一些实施例中，基于标记在元数据集合中查找读请求的元数据可以包括：选择与真实设备标识符相对应的索引表；以及使用真实逻辑块编号作为关键值在索引表中查找元数据。

[0009] 在一些实施例中，元数据可以包括位于块层的元数据和位于文件层的元数据。

[0010] 在一些实施例中，将缓存页面从第二容器提供给第一容器可以包括：将缓存页面从第二容器复制或迁移到第一容器。

[0011] 在一些实施例中，该方法可以进一步包括：基于缓存页面的访问频率、缓存页面的

访问特性、以及缓存页面所在的存储器的剩余空间中的至少一项来确定是将缓存页面复制到还是迁移到第一容器。

[0012] 在一些实施例中，该方法可以进一步包括：如果缓存页面从第二容器被复制到第一容器，将元数据添加到元数据集合；以及如果缓存页面从第二容器被迁移到第一容器，利用元数据来修改元数据集合。

[0013] 在一些实施例中，该方法可以进一步包括：响应于缓存页面在被迁移到第一容器中之后在第一容器中被修改或删除，将缓存页面返回给第二容器。

[0014] 在一些实施例中，该方法可以进一步包括：响应于第二容器的缓存页面中存储有数据并且缓存页面的状态为“干净”，将缓存页面从第二容器提供到第一容器。

[0015] 在一些实施例中，该方法可以进一步包括：响应于第二容器的缓存页面中存储有数据并且缓存页面的状态为“脏”，删除元数据集合中与缓存页面相关联的元数据；从存储设备读取数据；以及将读请求的元数据添加到元数据集合。

[0016] 在本公开的第二方面，提供了一种用于存储系统的装置。该装置包括至少一个处理器、以及包括计算机程序指令的至少一个存储器。至少一个存储器和计算机程序指令被配置为，与至少一个处理器一起，使得装置：响应于接收到第一容器对存储设备中的数据的读请求，获得与读请求相关联的标记；基于标记在元数据集合中查找读请求的元数据，元数据记录读请求的寻址信息，元数据集合包括在过去一段时间内对存储设备的访问请求的元数据；以及响应于在元数据集合中查找到读请求的元数据，基于元数据确定存储有数据的第二容器的缓存页面；以及将缓存页面从第二容器提供给第一容器以避免从存储设备读取数据。

[0017] 在一些实施例中，至少一个存储器和计算机程序指令可以进一步被配置为，与至少一个处理器一起，使得装置：响应于未在元数据集合中查找到元数据，从存储设备读取数据；以及将元数据添加到元数据集合。

[0018] 在一些实施例中，至少一个存储器和计算机程序指令可以进一步被配置为，与至少一个处理器一起，使得装置：将与读请求相关联的虚拟设备标识符和虚拟逻辑块编号转换为与读请求相关联的真实设备标识符和真实逻辑块编号。

[0019] 在一些实施例中，至少一个存储器和计算机程序指令可以进一步被配置为，与至少一个处理器一起，使得装置：选择与真实设备标识符相对应的索引表；以及使用真实逻辑块编号作为关键值在索引表中查找元数据。

[0020] 在一些实施例中，元数据可以包括位于块层的元数据和位于文件层的元数据。

[0021] 在一些实施例中，至少一个存储器和计算机程序指令可以进一步被配置为，与至少一个处理器一起，使得装置：将缓存页面从第二容器复制或迁移到第一容器。

[0022] 在一些实施例中，至少一个存储器和计算机程序指令可以进一步被配置为，与至少一个处理器一起，使得装置基于缓存页面的访问频率、缓存页面的访问特性、以及缓存页面所在的存储器的剩余空间中的至少一项来确定是将缓存页面复制到还是迁移到第一容器。

[0023] 在一些实施例中，至少一个存储器和计算机程序指令可以进一步被配置为，与至少一个处理器一起，使得装置：如果缓存页面从第二容器被复制到第一容器，将元数据添加到元数据集合；以及如果缓存页面从第二容器被迁移到第一容器，利用元数据来修改元数

据集合。

[0024] 在一些实施例中，至少一个存储器和计算机程序指令可以进一步被配置为，与至少一个处理器一起，使得装置：响应于缓存页面在被迁移到第一容器中之后在第一容器中被修改或删除，将缓存页面返回给第二容器。

[0025] 在一些实施例中，至少一个存储器和计算机程序指令可以进一步被配置为，与至少一个处理器一起，使得装置：响应于第二容器的缓存页面中存储有数据并且缓存页面的状态为“干净”，将缓存页面从第二容器提供到第一容器。

[0026] 在一些实施例中，至少一个存储器和计算机程序指令可以进一步被配置为，与至少一个处理器一起，使得装置：响应于第二容器的缓存页面中存储有数据并且缓存页面的状态为“脏”，删除元数据集合中与缓存页面相关联的元数据；从存储设备读取数据；以及将读请求的元数据添加到元数据集合。

[0027] 在本公开的第三方面，提供了一种计算机程序产品。该计算机程序产品被有形地存储在非易失性计算机可读介质上并且包括机器可执行指令，机器可执行指令在被执行时使机器执行根据第一方面的方法的步骤。

附图说明

[0028] 通过参考附图阅读下文的详细描述，本公开的实施例的上述以及其他目的、特征和优点将变得容易理解。在附图中，以示例性而非限制性的方式示出了本公开的若干实施例，其中：

[0029] 图1示出了在已有的存储系统中容器通过块存储驱动器进行读操作的示意图。

[0030] 图2示出了在根据本公开的实施例的存储系统中容器通过块存储驱动器进行读操作的示意图。

[0031] 图3示出了根据本公开的实施例的存储方法的示意性流程图。

[0032] 图4示出了根据本公开的实施例的位于块层的元数据集合的示意图。

[0033] 图5示出了根据本公开的实施例的位于文件层的元数据集合的示意图。

[0034] 图6示出了根据本公开的实施例的将缓存页面从一个容器复制到另一容器的示意图。

[0035] 图7示出了根据本公开的实施例的将缓存页面从一个容器迁移到另一容器的示意图。

[0036] 图8示出了在根据本公开的实施例中容器读取数据的方法的示意性流程图。

[0037] 图9示出了根据本公开的实施例的用于存储系统的装置的示意性框图。

[0038] 图10示出了一种可以被用来实施本公开的实施例的设备的示意性框图。

[0039] 贯穿所有附图，相同或者相似的参考标号被用来表示相同或者相似的元件。

具体实施方式

[0040] 下面将参考附图中所示出的若干示例性实施例来描述本公开的原理和精神。应当理解，描述这些具体的实施例仅是为了使本领域的技术人员能够更好地理解并实现本公开，而并非以任何方式限制本公开的范围。

[0041] 如上文提到的，容器(诸如Docker)可以认为是一种轻便的OS虚拟化，用于更轻便、

更快速且更容易的服务构建和操作。通常，容器构建在许多镜像(image)层上。为了有效率地管理这些只读的镜像并且向(多个)运行中容器提供可写入的层，容器利用了一些已有的存储技术(例如，存储驱动器)来提供写时复制(CoW)快照机制。

[0042] 一般而言，取决于写时复制在何处实施，存在两种类型的存储驱动器。一种类型是基于文件的存储驱动器，例如AUFS、OverlayFS/Overlay2和VFS。另一种类型是基于块的存储驱动器，例如DevMapper、ZFS和Btrfs，注意，基于块的存储驱动器是块层级的快照。不同类型的驱动器具有不同的特性，因此具有不同的IO性能、存储器效率和存储效率。

[0043] 块存储驱动器相比文件存储驱动器通常具有更好的存储效率和写入性能，但是它们的存储器占用和读取性能不是最佳的，尤其是在许多容器被启动的情况下。例如，DevMapper是一种流行的块存储驱动器并且在CentOS、RHEL、Fedora和Oracle Linux等系统中是默认的选择。然而，如果DevMapper上存在多个容器，那么在读取相同的数据时，这些容器可能产生许多对存储盘的冗余读请求，并且在每个容器中生成冗余的缓存页面。

[0044] 下面的表1示出了在块存储驱动器与文件存储驱动器之间的简单比较。

[0045] 表1

[0046]

	块存储驱动器	文件存储驱动器
I/O	写性能较好：块级别的CoW 读性能较差：不能共享页面缓存器	读性能较好：共享页面缓存器 写性能较差：文件级别的CoW
存储器效率	较差：页面缓存器中存在数据的多个副本	较好：在页面缓存器中共享数据的副本
存储效率	较好：以块级别进行CoW	较差：对整个文件进行CoW

[0047]

[0048] 由表1可以看出，块存储驱动器与文件存储驱动器相比，主要是读性能和存储器效率较差。其根本原因在于多个容器之间不能对缓存页面进行共享，从而造成了块存储驱动器需要更多地从存储设备中读取数据，即使是需要读取的数据已经存储在其他容器的缓存页面中。此外，多个容器之间不能共享缓存页面还造成了需要读取相同数据的多个容器在存储器中创建多个相同的缓存页面，这导致了存储器占用较大。下面结合图1来详细地解释多个容器从已有的块存储驱动器读取数据的具体过程以及已有的块存储驱动器所存在的问题。

[0049] 图1示出了在已有的存储系统100中容器111-113通过块存储驱动器120进行读操作的示意图。如图1所示，存储系统100包括容器111-113、块存储驱动器120和存储设备130。容器111-113可以通过块存储驱动器120从存储设备130读取数据140。应当理解，尽管图1中示例性地示出了存储系统100包括三个容器111-113，但是在其他实施例中，存储系统100可以具有更多或更少的容器。

[0050] 以容器111为例，在容器111需要通过块存储驱动器120从存储设备130读取数据

140时，容器111向块存储驱动器120发送读请求131。在接收到读请求131以后，块存储驱动器120根据读请求131中包含的寻址信息通过与存储设备130的交互150而获得读请求131所针对的数据140，然后将数据140返回132给容器111。

[0051] 容器111将从块存储驱动器120收到的数据140缓存在缓存页面241中，缓存页面241具有特定的大小，其可以对应于块层中的特定数据块。如果容器111短时间内需要再次读取数据140，则容器111可以直接从缓存页面241获得数据140，而不必再通过块存储驱动器120来访问存储设备130。

[0052] 类似地，假设容器112、113也需要读取数据140，则它们可以向块存储驱动器120发送读请求133、135，可以从块存储驱动器120接收134、136到数据140，还可以将数据140存放在各自的缓存页面142、143中。

[0053] 如上文提到的，块存储驱动器120(例如，DevMapper)工作在块层级，因此在块存储驱动器120之上，容器111-113各自创建有它们对应的文件系统(未示出)以及它们各自的缓存页面241-143。然而，容器111-113之间不能共享这些缓存页面241-143，即使这些缓存页面241-143可能是从存储设备140的相同块读取的。在这样的情况下，块存储驱动器120引入了冗余的对存储设备130的读操作而降低了读性能。

[0054] 此外，当容器111-113使用块存储驱动器120时，通常将会共享镜像以节省存储设备130的存储空间。因此，存储设备130的具有数据140的相同数据块可能反复地被多个容器111-113缓存，且进一步在缓存页面241-143所在的存储器(未示出)中生成许多相同的缓存页面。这潜在地浪费了实现缓存页面241-143存储器的存储空间。例如，在块存储驱动器120具有数百个实例的情况下，一个块(4KB-8KB)的占用可能被放大数百倍而造成显著的存储器占用。

[0055] 本公开的实施例将解决上述技术问题。为此，本公开的实施例至少克服了如下技术挑战。首先，本公开的实施例能够准确地且有效率地检测到来自不同容器(例如，容器111-113)的对存储设备130的冗余读请求。而且，在确定冗余的读请求之后，能够确定哪个容器的缓存页面中存储有读请求131所针对的数据140，并且能够在不同的容器之间共享具有该数据的缓存页面。再者，本公开的实施例能够在存储器占用与读操作性能之间取得合理的平衡。如下文将描述的，上述诸多技术问题和挑战将通过本公开的实施例得到解决。下面结合图2和图3来描述根据本公开的实施例的改进的存储系统和方法。

[0056] 图2示出了在根据本公开的实施例的存储系统200中容器211-213通过块存储驱动器221进行读操作的示意图。如图2所示，与已有的存储系统100不同，存储系统200另外地包括快速旁路缓冲器(Fast Look-aside Buffer，FLB)222来解决上述的技术问题和挑战而无需对块存储驱动器221进行高成本的重新设计。在一些实施例中，快速旁路缓冲器222可以在内核空间与块存储驱动器221和系统缓存(或存储器、页面缓存器)一起工作。

[0057] 在图2中，使用附图标记230、250、270以及220描绘了存储系统200可以通过快速旁路缓冲器222确定来自容器211的冗余读请求230，从而避免270了对存储设备223的冗余访问。此外，图2使用标记240、260、280描绘了来自容器212的读请求240由快速旁路缓冲器222确认为非冗余的情况下，经由块存储驱动器221发送260给快速旁路缓冲器222，再经由快速旁路缓冲器222发送280给存储设备223。下文结合图3来具体地描述存储系统200及其各个组件在这两种情形中进行的操作。

[0058] 图3示出了根据本公开的实施例的存储方法300的示意性流程图。在一些实施例中，方法300可以由图2中所描绘的存储系统200来执行，例如可以由存储系统200中的各种单元或组件来执行。

[0059] 在305处，响应于接收到容器(例如，容器211)对存储设备223中的数据140的读请求230，块存储驱动器221获得与读请求230相关联的标记250，并将其发送给快速旁路缓冲器222。在一些实施例中，标记250也可以由快速旁路缓冲器222从读请求230中获得。在这种情况下，块存储驱动器221将读请求230转发给快速旁路缓冲器222。

[0060] 如下文详细描述的，快速旁路缓冲器222可以基于标记250来确定读请求230是否为冗余的读请求。在确定读请求230为冗余的情况下，快速旁路缓冲器222可以利用标记250来确定读请求230的元数据，该元数据记录读请求230的寻址信息，从而进一步确定读请求230针对的数据存储在哪个其他容器(例如，容器212-213)的缓存页面中。

[0061] 通常，块存储驱动器221将存储设备223映射为多个虚拟设备，并且向不同的容器(例如，容器211-213)提供不同的虚拟设备标识符和虚拟逻辑块编号。因此，来自容器211的读请求230中将带有其访问目标的虚拟设备标识符和虚拟逻辑块编号。如果可以将虚拟设备标识符(ID)和虚拟逻辑块编号(vLBN)转换为块存储驱动器221中的真实设备标识符(ID)和真实逻辑块编号(也称为物理块编号PBN)，就可以通过将读请求230的真实设备标识符和真实逻辑块编号与之前的访问请求的真实设备标识符和真实逻辑块编号进行比较来判断读请求230是否为冗余的。

[0062] 因此，在一些实施例中，快速旁路缓冲器222可以采用读请求230的访问目标的真实设备标识符和真实逻辑块编号作为与读请求230相关联的标记250。为了获得标记250，块存储驱动器221可以将与读请求230相关联的虚拟设备标识符和虚拟逻辑块编号转换为与读请求230相关联的真实设备标识符和真实逻辑块编号。在一些实施例中，该转换操作也可以由快速旁路缓冲器222来执行。

[0063] 在310处，快速旁路缓冲器222基于标记250在元数据集合中查找读请求230的元数据。如上文所述，该元数据记录有读请求230的寻址信息，因此可以通过该元数据来确定读请求230是否为冗余的读请求。此外，这里的元数据集合包括在过去一段时间内对存储设备223的访问请求的元数据。

[0064] 例如，容器212在一段时间(例如，五分钟)之前对存储设备223进行了访问，那么容器212的该访问请求(可能是读请求或写请求)可以被添加到该元数据集合中。应当理解，此处的5分钟时间段和特定容器仅为示例，无意以任何方式限定本公开的实施例的范围。在其他实施例中，本领域的技术人员可以根据具体情况来设置其他长度的时间段，并且其他容器的近期访问请求也包括该元数据集合中。

[0065] 在一些实施例中，元数据可以包括位于块层的元数据和位于文件层的元数据。相应地，元数据集合也可以包括位于块层的元数据集合和位于文件层的元数据集合。下文结合图4和图5来具体地描述这两种元数据和元数据集合的示例性组织形式。

[0066] 图4示出了根据本公开的实施例的位于块层的元数据集合400的示意图。如图4所示，在一些实施例中，位于块层的元数据420、430等可以包括设备标识符、逻辑块编号、inode和偏移等寻址信息。采用这些寻址信息的原因如下。

[0067] 在存储系统200中，文件的缓存页面可以在存储器中被组织为基数树(radix

tree)，而基数树的根存储在文件的inode中。此外，缓存页面在文件中的偏移被用作基数树的索引。因此，通过设备标识符和逻辑块编号可以查找到特定的元数据，而通过特定元数据中包括的inode和偏移可以进一步查找到特定的缓存页面。此外，如图4所示，元数据420、430等可以按照索引表(例如，哈希表)410的形式进行组织。在一些实施例中，存储系统200中的每个存储设备可以具有各自的索引表(例如，哈希表)。

[0068] 应当注意，图4中描绘的元数据集合400的组织形式和元数据420、430等的内容仅为示例，无意以任何方式限制本公开的实施例。在其他实施例中，元数据集合400也可以采用其他适合的形式来组织，而元数据420、430等可以包括其他适合的寻址信息。

[0069] 在图4中的索引表410的组织下，快速旁路缓冲器222可以选择与真实设备标识符相对应的索引表410，然后使用真实逻辑块编号作为关键值在索引表410中查找读请求230的元数据420。

[0070] 在进一步的实施例中，为了减少从文件层到块层的长调用堆栈路径以进一步改进性能，本公开的实施例还可以在文件层或页面缓存器层中维持元数据的卷影副本(shadow copy)，从而查找可以快速被执行而无需总是访问块层，因此也跳过了任何IO调度器。下面结合图5来详细地描述这样的实施例。

[0071] 图5示出了根据本公开的实施例的位于文件层的元数据集合500的示意图。如图5所示，在一些实施例中，位于文件层的元数据520、530等可以包括文件系统标识符、inode、偏移、“干净/脏”状态等。此外，如图5所示，元数据520、530等可以按照索引表(例如，哈希表)510的形式进行组织。

[0072] 由于这种文件层的元数据520、530等可以在文件系统之间被共享，因此是全局性的。元数据520、530等中的条目仅在块层确认了冗余的访问请求时才被添加，并且这些条目可以在文件层快速地被查询和移除(例如，在页面为“脏”时)。文件层的元数据520、530等相比于块层的元数据420、430等以稍多一些的存储器占用为代价实现了更快速的读性能。

[0073] 应当注意，图5中描绘的元数据集合500的组织形式和元数据520、530等的内容仅为示例，无意以任何方式限制本公开的实施例。在其他实施例中，元数据集合500也可以采用其他适合的形式来组织，而元数据520、530等可以包括其他适合的寻址信息。

[0074] 返回参考图3，在315处，响应于在元数据集合400(或500)中查找到读请求230的元数据420(或520)，快速旁路缓冲器222基于元数据420(或520)确定存储有数据140的容器(例如，容器212)的缓存页面242。

[0075] 在确定容器212的缓存页面242之后，容器212向容器211提供缓存页面242。由于缓存页面242中存储有读请求230针对的数据140，所以在缓存页面242被提供220给容器211之后，容器211可以从自己的缓存页面241(与缓存页面242的内容相同)中读取数据140，从而避免270了从存储设备223读取数据140。

[0076] 此外，在一些实施例中，在将缓存页面242提供220给容器211之前，可以先检查缓存页面242的“干净/脏”状态。例如，响应于容器212的缓存页面242中存储有数据140并且缓存页面242的状态为“干净”，可以将缓存页面242从容器212提供220到容器211。另一方面，响应于容器212的缓存页面242中存储有数据140并且缓存页面242的状态为“脏”，可以删除元数据集合400(或500)中与缓存页面242相关联的元数据420(或520)，然后可以从存储设备223读取数据140，以及将读请求230的元数据添加到元数据集合400(或500)中，以便由其

他容器对数据140的后续访问加以利用。

[0077] 在一些情况下，如果未在元数据集合400(或500)中查找到元数据420(或520)，则快速旁路缓冲器222可以从存储设备223读取数据140。在这种情况下，也可以将读请求230的元数据添加到元数据集合400(或500)中，以便由其他容器对数据140的后续访问加以利用。

[0078] 再者，在本公开的一些实施例中，在需要向容器211提供缓存页面242的情况下，容器212可以将缓存页面242复制到或迁移到容器211。换句话说，本公开的实施例提供了复制和迁移这两种方式从源缓存页面242向目标容器211提供数据140。下面结合图6和图7分别描述这两种方式。

[0079] 图6示出了根据本公开的实施例的将缓存页面242从容器212复制650到容器211的示意图。如图6所示，容器211通过与块存储驱动器221的交互630发送对数据140的读请求230，而容器212通过与块存储驱动器221的交互640知晓需要将包括数据140的缓存页面242复制650到容器211而形成缓存页面241。此后，容器211可以从缓存页面241读取数据140。此外，在一些实施例中，如果缓存页面242从容器212被复制到容器211，则可以将读请求230的元数据添加到元数据集合400(或500)。

[0080] 图7示出了根据本公开的实施例的将缓存页面242从容器212迁移750到容器211的示意图。如图7所示，容器211通过与块存储驱动器221的交互630发送对数据140的读请求230，而容器212通过与块存储驱动器221的交互640知晓需要将包括数据140的缓存页面242迁移710到容器211而形成缓存页面241。注意，与图6中描绘的“复制650”不同，此处的“迁移710”意味着缓存页面242从容器212中被删除，因此释放了缓存页面242的存储空间。此后，容器211可以从缓存页面241读取数据140。

[0081] 此外，如图7所示，在一些实施例中，响应于缓存页面242在被迁移到容器211中之后在容器11中被修改或删除(例如，被调换)，缓存页面242可以被返回720给容器212。这具体可以通过维持(多个)近期历史拥有缓存页面242的容器的列表来实现。此外，如果缓存页面242从容器212被迁移到容器211，则可以利用读请求230的元数据来修改元数据集合400(或500)。例如，将原来指向容器212的元数据420(或520)修改为指向容器211。

[0082] 从上面的描述可知，图6中的复制方式在存储器中生成了重复的副本，即缓存页面241和242。如果缓存页面241和242都具有较高访问频率，则在存储器中维持重复副本可能有利于IO性能。但是，这种方式可能会浪费存储器的存储空间。相反地，图7中的迁移方式在存储器中针对每个数据块仅维持单个副本。这节省了存储器的存储空间。但是，在缓存页面241(或242)持续地被容器211和212两者访问的情况下，这种迁移方式可能会导致缓存页面241(或242)在容器211与212之间过于频繁的迁移，这将会降低性能。

[0083] 为了在存储器占用与IO性能之间实现合理的平衡，本公开的实施例提供了一种存储器中管理缓存页面的重复副本的方案。具体地，在处理冗余的读请求230时，快速旁路缓冲器222可以首先检查作为源的缓存页面242的访问频率。如果缓存页面242的访问频率较低，则快速旁路缓冲器222可以将它迁移到容器211的缓存页面241。否则，快速旁路缓冲器222可以将缓存页面242复制到容器211的缓存页面241。

[0084] 另一方面，当存储器的存储空间存在压力时，在存储器中保持缓存页面的较多重复副本可能会驱逐掉其他的缓存页面，而减小在存储器中查找到缓存页面的命中率并降低

IO性能。因此，快速旁路缓冲器222可以扫描所有的缓存页面，并且在存储器的剩余空间达到特定阈值(例如，10％)时优先调换出重复的缓存页面和访问频率较低的缓存页面(冷页面)。

[0085] 此外，如果某个数据块时常被多个容器(例如，容器211-213)访问，但是它对应的缓存页面经常处于“脏”状态，则快速旁路缓冲器222可以在属于它自己的存储空间中保存该数据块的“干净”副本而无需将它添加到容器211-213的缓存页面241-243。

[0086] 也就是说，快速旁路缓冲器222可以基于缓存页面242的访问频率、缓存页面242的访问特性、以及缓存页面242所在的存储器的剩余空间中的至少一项来确定是将缓存页面242复制650到还是迁移710到容器211。通过这样的方式，快速旁路缓冲器222可以取决于各种因素而混合地使用图6和图7中描绘的复制方式和迁移方式。换言之，假定N个容器共享特定的缓存页面，则在特定时刻，存储器中可能存在M个缓存页面的副本，其中M的取值在1与N之间。

[0087] 从上文的描述可知，由于存储系统200在已有的存储系统100基础上添加了快速旁路缓冲器222以及对应的操作，所以在根据本公开的实施例的存储系统200中，当容器(例如，容器211)需要读取数据140时，将会具有与存储系统100中不同的流程。下文结合图8来描述在存储系统200包括快速旁路缓冲器222之后的读取数据140的具体流程。

[0088] 图8示出了在根据本公开的实施例中容器211读取数据140的方法800的示意性流程图。在图8的流程图中，以图4中描绘的位于块层的元数据集合400和元数据420、430等作为示例进行描述。

[0089] 如图8所示，在框802处，容器211发起对数据140的读请求230。在框804处，块存储驱动器221获得读请求230的标记250。在一些实施例中，标记250可以是与读请求230相关联的真实设备标识符和真实逻辑块编号。例如，块存储驱动器221或快速旁路缓冲器222可以将读请求230的虚拟设备标识符和虚拟逻辑块编号转换为读请求230的真实设备标识符和真实逻辑块编号从而获得标记250。

[0090] 在框806处，快速旁路缓冲器222可以基于请求230的标记250来确定读请求230是否为冗余的。具体地，快速旁路缓冲器222基于标记250在元数据集合400中查找读请求230的元数据420。如果在元数据集合400中查找到元数据420，则可以确定读请求230是冗余的并且进一步基于元数据420确定数据140被存储在容器212的缓存页面242中。然后，方法800前进到框808。

[0091] 在框808处，快速旁路缓冲器222确定缓存页面242的“干净/脏”状态。如果确定缓存页面242是“干净”的，则方法800前进到框810。在框810处，快速旁路缓冲器222基于缓存页面242的访问频率、缓存页面242的访问特性、以及缓存页面242所在的存储器的剩余空间等各种因素之一或组合来确定是采用复制方式还是迁移方式将缓存页面242从容器212提供220到容器211。

[0092] 如果选择采用复制方式，则方法800前进到框812。在框812处，在完成复制后，将读请求230的元数据添加到元数据集合400中。如果选择采用迁移方式，则方法800前进到框814。在框814处，在完成迁移后，利用读请求230的元数据来修改元数据集合400。例如，将指向容器212的元数据420修改为指向容器211。

[0093] 此外，如果在框806处确定读请求230不是冗余的，则方法800前进到框816。在框

816处，从存储设备223读取数据140。接着，方法800前进到框812。在框812处，将读请求230的元数据添加到元数据集合400中。

[0094] 此外，如果在框808处确定缓存页面242的状态为“脏”，则方法800前进到框813。在框813处，删除元数据集合400中与数据140有关的元数据420。接着，方法800前进到框816。在框816处，从存储设备223读取数据140。然后，方法800前进到框812。在框812处，将读请求230的元数据添加到元数据集合400中。

[0095] 图9示出了根据本公开的实施例的用于存储系统的装置900的示意性框图。本领域的技术人员可以理解，图9仅示出了装置900中与本公开紧密相关的单元或组件。在一些实施例中，装置900还可以包括使其能够正常运转的其他功能单元或组件。此外，图9中所示出的各个单元或组件之间可以存在必要的连接关系，但是出于简洁的考虑，图9中并没有描绘出这些连接关系。在一些实施例中，装置900可以被配置为实施上文结合图3所描述的方法300。

[0096] 如图9中所示出的，装置900包括获得单元910、查找单元920、确定单元930和提供单元940。获得单元910被配置为响应于接收到第一容器对存储设备中的数据的读请求，获得与读请求相关联的标记250。查找单元920被配置为基于标记250在元数据集合中查找读请求的元数据，该元数据记录读请求的寻址信息，该元数据集合包括在过去一段时间内对存储设备的访问请求的元数据。控制单元930被配置为：响应于查找单元920在元数据集合中查找到读请求的元数据，基于元数据确定存储有数据的第二容器的缓存页面。提供单元940被配置为：响应于控制单元930确定存储有数据的第二容器的缓存页面，将缓存页面从第二容器提供220给第一容器以避免270从存储设备读取数据。

[0097] 在一些实施例中，控制单元930可以进一步被配置为：响应于查找单元920未在元数据集合中查找到元数据，从存储设备读取数据，以及将元数据添加到元数据集合。

[0098] 在一些实施例中，获得单元910可以进一步被配置为：将与读请求相关联的虚拟设备标识符和虚拟逻辑块编号转换为与读请求相关联的真实设备标识符和真实逻辑块编号。

[0099] 在一些实施例中，查找单元920可以进一步被配置为：选择与真实设备标识符相对应的索引表；以及使用真实逻辑块编号作为关键值在索引表中查找元数据。

[0100] 在一些实施例中，元数据可以包括位于块层的元数据和位于文件层的元数据。

[0101] 在一些实施例中，提供单元940可以进一步被配置为：将缓存页面从第二容器复制或迁移到第一容器。

[0102] 在一些实施例中，提供单元940可以进一步被配置为：基于以下至少一项来确定是将缓存页面复制到还是迁移到第一容器：缓存页面的访问频率、缓存页面的访问特性、以及缓存页面所在的存储器的剩余空间。

[0103] 在一些实施例中，控制单元930可以进一步被配置为：如果缓存页面从第二容器被复制到第一容器，将元数据添加到元数据集合；以及如果缓存页面从第二容器被迁移到第一容器，利用元数据来修改元数据集合。

[0104] 在一些实施例中，提供单元940可以进一步被配置为：响应于缓存页面在被迁移到第一容器中之后在第一容器中被修改或删除，将缓存页面返回给第二容器。

[0105] 在一些实施例中，提供单元940可以进一步被配置为：响应于第二容器的缓存页面中存储有数据并且缓存页面的状态为“干净”，将缓存页面从第二容器提供220到第一容器。

[0106] 在一些实施例中，控制单元930可以进一步被配置为：响应于第二容器的缓存页面中存储有数据并且缓存页面的状态为“脏”，删除元数据集合中与缓存页面相关联的元数据；从存储设备读取数据；以及将读请求的元数据添加到元数据集合。

[0107] 图10示意性地示出了一种可以被用来实施本公开的实施例的设备1000的框图。如图10中所示出的，设备1000包括中央处理单元(CPU)1001，其可以根据存储在只读存储设备(ROM)1002中的计算机程序指令或者从存储单元1008加载到随机访问存储设备(RAM)1003中的计算机程序指令，来执行各种适当的动作和处理。在RAM 1003中，还可存储设备1000操作所需的各种程序和数据。CPU 1001、ROM 1002以及RAM 1003通过总线1004彼此相连。输入/输出(I/O)接口1005也连接至总线1004。

[0108] 设备1000中的多个部件连接至I/O接口1005，包括：输入单元1006，例如键盘、鼠标等；输出单元1007，例如各种类型的显示器、扬声器等；存储单元1008，例如磁盘、光盘等；以及通信单元1009，例如网卡、调制解调器、无线通信收发机等。通信单元1009允许设备1000通过诸如因特网的计算机网络和/或各种电信网络与其他设备交换信息/数据。

[0109] 上文所描述的各个过程和处理，例如方法300，可由处理单元1001来执行。例如，在一些实施例中，方法300可被实现为计算机软件程序，其被有形地包含于机器可读介质，例如存储单元1008。在一些实施例中，计算机程序的部分或者全部可以经由ROM 1002和/或通信单元1009而被载入和/或安装到设备1000上。当计算机程序被加载到RAM 1003并由CPU 1001执行时，可以执行上文描述的方法300的一个或多个步骤。

[0110] 与已有的存储系统中的容器读取数据的方法相比，根据本公开的实施例的存储系统和存储方法通过充分利用块层级的数据共享而实现了高性能和高效率的存储系统和方法，其至少提供了以下的技术优势。首先，本公开的实施例提供了对存储设备的冗余读请求的检测和减少。通过使用少量的元数据，实施例实现在从存储设备取回数据之前的冗余读请求的检测。此外，实施例通过复制或迁移的方式，利用已经存在于其他容器的缓存页面的内容来服务于检测到的冗余读请求。

[0111] 另外，本公开的实施例可以处理存储器中的缓存页面的重复副本。在发现冗余的读请求时，实施例可以在容器之间迁移缓存页面以避免在存储器中生成重复的缓存页面。特别是在存储器空间不足时，实施例可以调换出重复的缓存页面以得到更多的存储器空间。

[0112] 另外，本公开的实施例可以自适应地在IO性能与存储器效率之间进行折中。实施例可以通过合理地管理重复缓存页面的数量而在在IO性能与存储器占用之间进行权衡。具体地，实施例可以基于缓存页面的访问频率、缓存页面的访问特性、以及缓存页面所在的存储器的剩余空间等因素之一或组合来确定重复缓存页面的数量。

[0113] 再者，本公开的实施例可以通过灵活的添加(add-on)模块来实现。实施例提供的快速旁路缓冲器可以是与块存储驱动器协作的内核模块，并且可以灵活地被加载/卸载。实施例提供的快速旁路缓冲器几乎重用了已有块存储驱动器的所有的基础结构并且减少了重新设计的风险或保护了已有的投入。作为一种增强的基础结构，本公开的实施例还可以应用到融合系统或超融合系统、云系统(例如，VirtuStream)、或任何其他块共享的系统中，甚至应用到超过容器技术环境以外的环境中。

[0114] 如本文所使用的，术语“包括”及其类似用语应当理解为开放性包含，即“包括但不

限于”。术语“基于”应当理解为“至少部分地基于”。术语“一个实施例”或“该实施例”应当理解为“至少一个实施例”。术语“第一”、“第二”等等可以指代不同的或相同的对象。本文还可能包括其他明确的和隐含的定义。

[0115] 如本文所使用的，术语“确定”涵盖各种各样的动作。例如，“确定”可以包括运算、计算、处理、导出、调查、查找(例如，在表格、数据库或另一数据结构中查找)、查明等。此外，“确定”可以包括接收(例如，接收信息)、访问(例如，访问存储器中的数据)等。此外，“确定”可以包括解析、选择、选取、建立等。

[0116] 应当注意，本公开的实施例可以通过硬件、软件或者软件和硬件的结合来实现。硬件部分可以利用专用逻辑来实现；软件部分可以存储在存储器中，由适当的指令执行系统，例如微处理器或者专用设计硬件来执行。本领域的技术人员可以理解上述的设备和方法可以使用计算机可执行指令和/或包含在处理器控制代码中来实现，例如在可编程的存储器或者诸如光学或电子信号载体的数据载体上提供了这样的代码。

[0117] 此外，尽管在附图中以特定顺序描述了本公开的方法的操作，但是这并非要求或者暗示必须按照该特定顺序来执行这些操作，或是必须执行全部所示的操作才能实现期望的结果。相反，流程图中描绘的步骤可以改变执行顺序。附加地或备选地，可以省略某些步骤，将多个步骤组合为一个步骤执行，和/或将一个步骤分解为多个步骤执行。还应当注意，根据本公开的两个或更多装置的特征和功能可以在一个装置中具体化。反之，上文描述的一个装置的特征和功能可以进一步划分为由多个装置来具体化。

[0118] 虽然已经参考若干具体实施例描述了本公开，但是应当理解，本公开不限于所公开的具体实施例。本公开旨在涵盖所附权利要求的精神和范围内所包括的各种修改和等效布置。

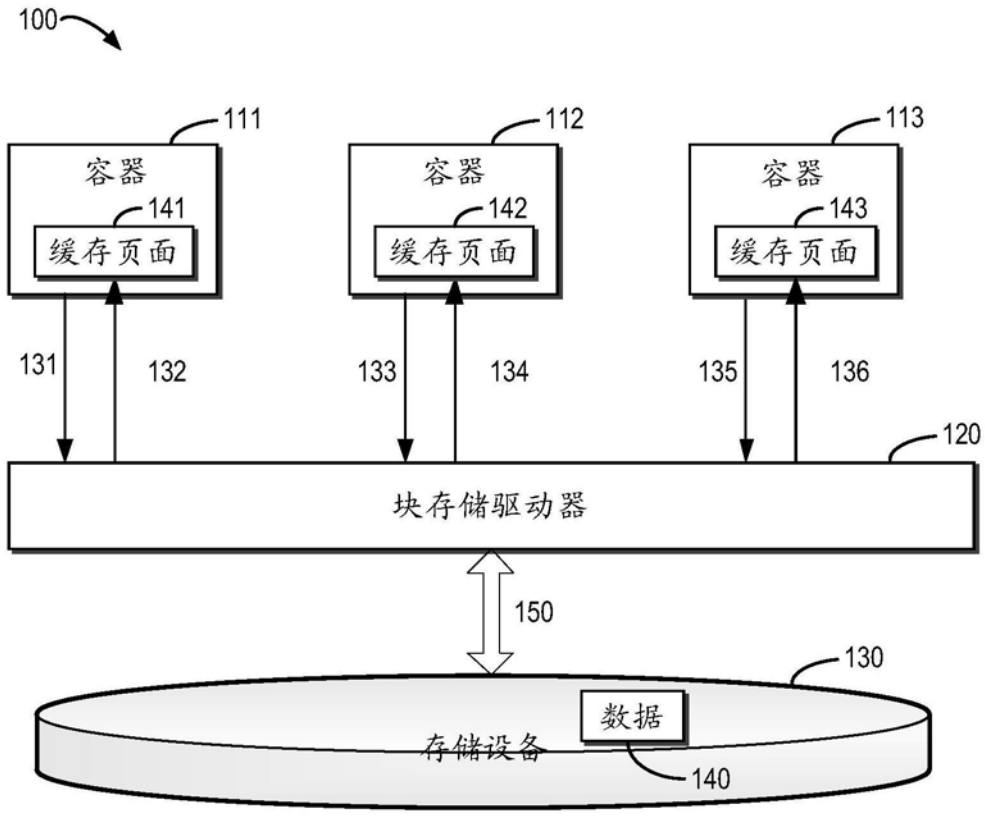

100
111
容器
141
缓存页面
112
容器
142
缓存页面
113
容器
143
缓存页面
131
132
133
134
135
136
120
块存储驱动器
150
130
数据
存储设备
140

图1

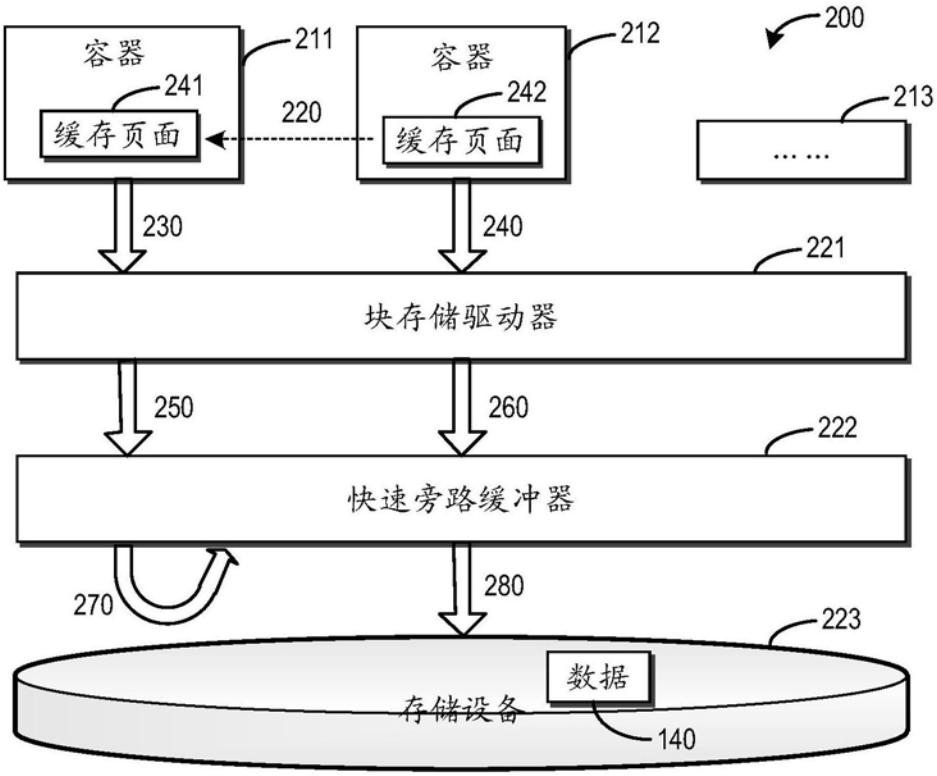

容器
211
241
缓存页面
220
容器
212
242
缓存页面
200
213
……
230
240
221
块存储驱动器
250
260
222
快速旁路缓冲器
270
280
223
数据
存储设备
140

图2

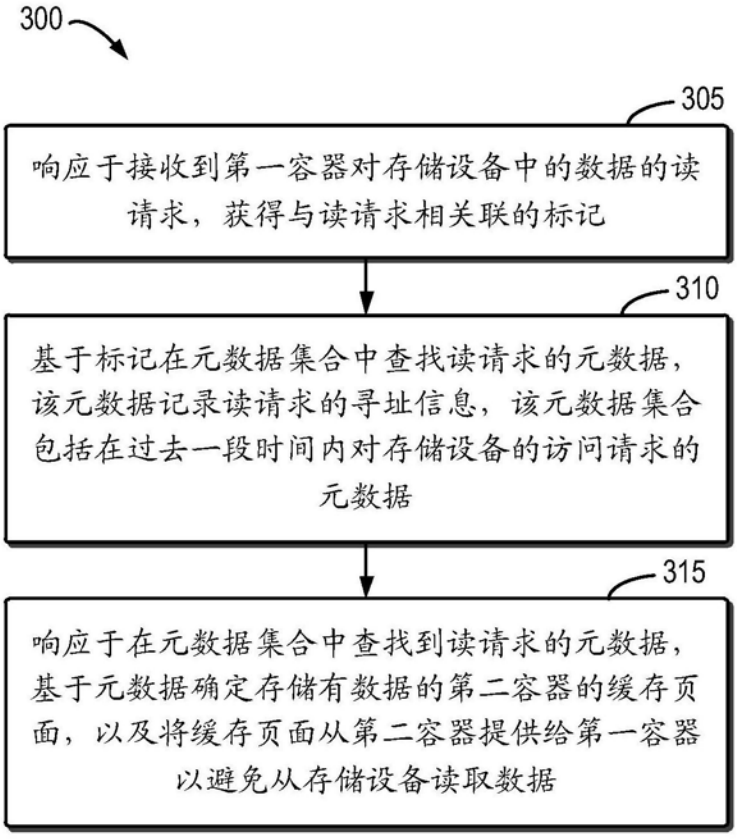

300
305
响应于接收到第一容器对存储设备中的数据的读请求，获得与读请求相关联的标记
310
基于标记在元数据集合中查找读请求的元数据，该元数据记录读请求的寻址信息，该元数据集合包括在过去一段时间内对存储设备的访问请求的元数据
315
响应于在元数据集合中查找到读请求的元数据，基于元数据确定存储有数据的第二容器的缓存页面，以及将缓存页面从第二容器提供给第一容器以避免从存储设备读取数据

图3

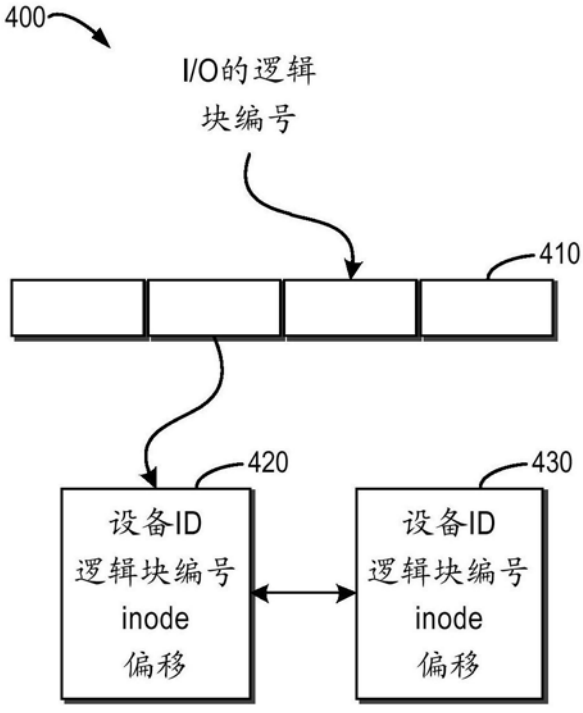

400
I/O的逻辑块编号
410
420
设备ID
逻辑块编号
inode
偏移
430
设备ID
逻辑块编号
inode
偏移

图4

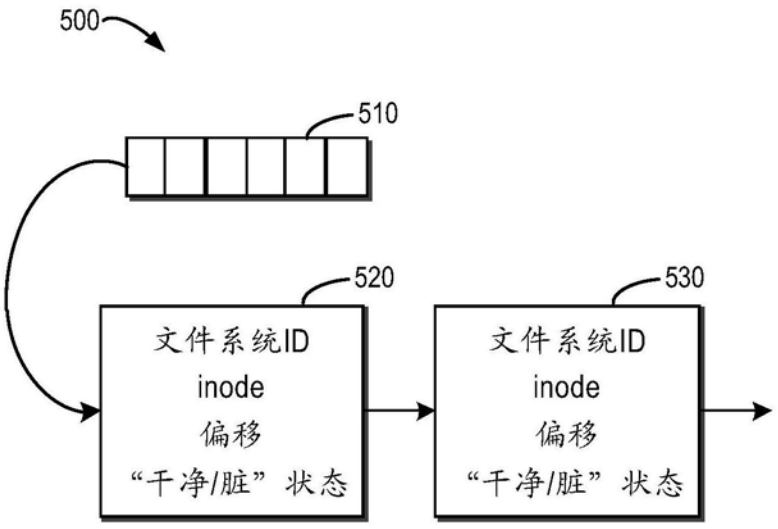

500
510
520
文件系统ID
inode
偏移
“干净/脏”状态
530
文件系统ID
inode
偏移
“干净/脏”状态

图5

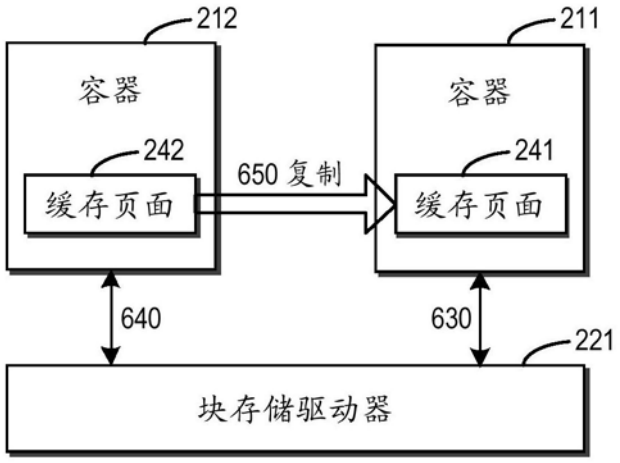

212
容器
242
缓存页面
650 复制
211
容器
241
缓存页面
640
630
221
块存储驱动器

图6

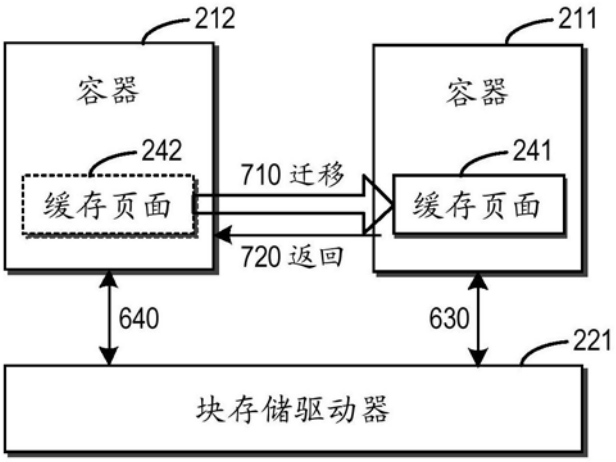


图7

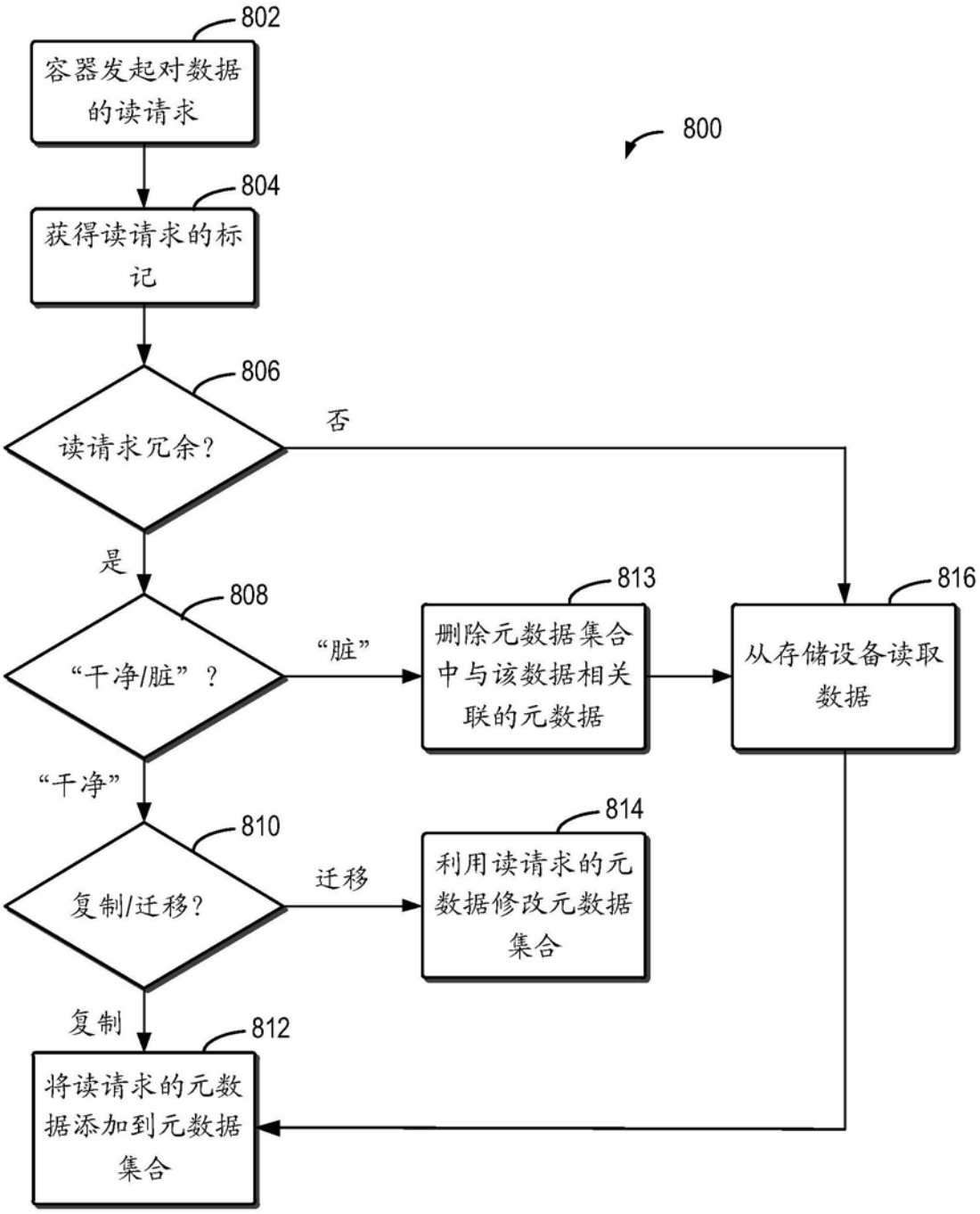

802
容器发起对数据的读请求
800
804
获得读请求的标记
806
读请求冗余?
否
是
808
“干净/脏”?
“脏”
813
删除元数据集合中与该数据相关联的元数据
816
从存储设备读取数据
“干净”
810
复制/迁移?
迁移
814
利用读请求的元数据修改元数据集合
复制
812
将读请求的元数据添加到元数据集合

图8

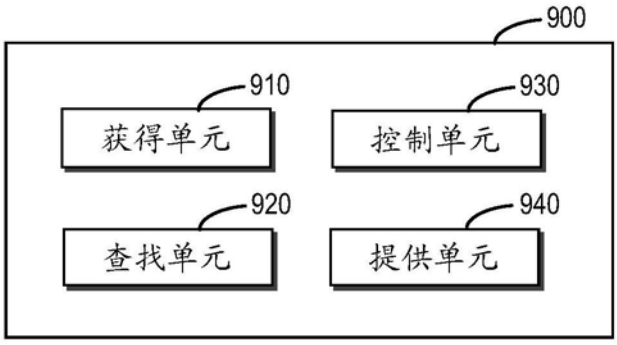

900
910
获得单元
930
控制单元
920
查找单元
940
提供单元

图9

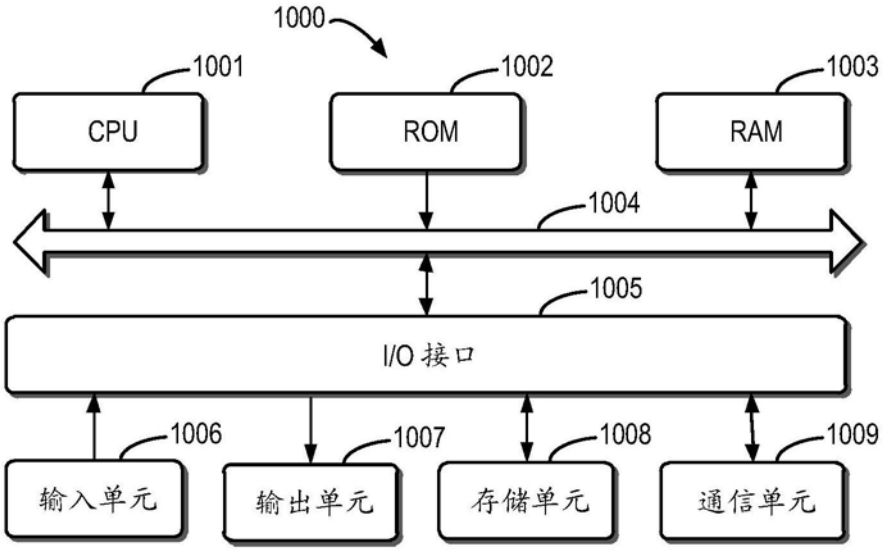

1000
1001
CPU
1002
ROM
1003
RAM
1004
1005
I/O 接口
1006
输入单元
1007
输出单元
1008
存储单元
1009
通信单元

图10