(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 102934114 B
(45)授权公告日 2015.11.25

(21)申请号 201180029522.9
(22)申请日 2011.06.01
(30)优先权数据
12/815418 2010.06.15 US
(85)PCT国际申请进入国家阶段日
2012.12.14
(86)PCT国际申请的申请数据
PCT/US2011/038811 2011.06.01
(87)PCT国际申请的公布数据
WO2011/159476 EN 2011.12.22
(73)专利权人 微软技术许可有限责任公司
地址 美国华盛顿州
(72)发明人 J.M.卡吉尔 T.J.米勒
W.R.蒂普顿
(74)专利代理机构 中国专利代理(香港)有限公司 72001
代理人 董宁 汪扬
(51)Int.Cl.
G06F 17/30(2006.01)
(56)对比文件
US 2008040385 A1 ,2008.02.14, 参见全文.
US 2009094582 A1 ,2009.04.09, 参见全文.
US 7412460 B2 ,2008.08.12, 参见全文.
WO 2007127361 A2 ,2007.11.08, 参见全文.
审查员 于白

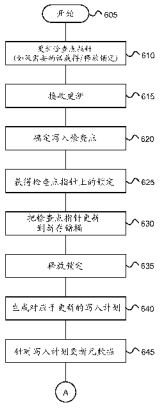
权利要求书2页 说明书14页 附图8页

(54)发明名称

用于文件系统的检查点

(57)摘要

这里所描述的主题的各个方面涉及用于文件系统的检查点。在一些方面中,针对文件系统的更新被组织成检查点存储桶。当希望有检查点时,后续的更新被导向另一个检查点存储桶。在针对当前检查点存储桶中的各项更新对各个全局表进行了更新之后,创建所述全局表的逻辑拷贝。该逻辑拷贝被存储为检查点数据的一部分。为了帮助恢复,检查点管理器可以等到当前检查点存储桶的所有更新都被写入到存储装置之后才把最终检查点数据写入到存储装置。该最终检查点数据可以涉及所述全局表的逻辑拷贝,并且包括用以验证检查点数据的正确性的验证代码。



CN 102934114 B

1. 一种至少部分地由计算机实施的方法，所述方法包括：

指示将把第一更新集合与第一检查点相关联；

确定把关于第一检查点的检查点数据写入到文件系统的存储装置，所述文件系统使用写时拷贝来更新文件系统的数据；

指示将把在第一更新集合之后发生的任何更新与后续检查点相关联；

生成针对第一更新集合的写入计划，每一个写入计划指示对应于表示第一更新集合当中的至少一项更新的数据的存储装置上的至少一个计划位置；

更新元数据以便指示文件系统的分配数据以及对应于由所述写入计划修改的文件系统对象的存储位置；

创建元数据的逻辑拷贝；

创建针对写入第一检查点数据的写入计划，同时允许与创建所述写入计划并行地生成对应于后续更新的写入计划；以及

把至少一项验证代码写入到存储装置，所述至少一项验证代码是检查点数据的一部分，所述至少一项验证代码可用来确定第一更新集合是否被正确地写入到存储装置。

2. 权利要求 1 的方法，其还包括：在把所述至少一项验证代码写入到存储装置之前，等待表示第一更新集合的数据被写入到存储装置。

3. 权利要求 1 的方法，其中，把至少一项验证代码写入到存储装置包括与涉及表示所述元数据的逻辑拷贝的至少一个树状数据结构的根节点的其他数据一起在一个块中把单项验证代码写入到存储装置，并且还包括计算所述单项验证代码以便验证所述块。

4. 权利要求 1 的方法，其还包括：读取所述至少一项验证代码，从存储装置上的数据计算至少一项其他验证代码，把所述至少一项验证代码与所述至少一项其他验证代码进行比较，以及由此确定表示第一更新集合的所有数据是否已被成功写入到存储装置。

5. 权利要求 1 的方法，其中，指示将把第一更新集合与第一检查点相关联包括：更新一个数据结构，其指示对于在所述数据结构被更新为指示另一个检查点之前所发生的任何更新将使用第一检查点。

6. 权利要求 1 的方法，其中，创建元数据的逻辑拷贝包括：与把表示第一更新集合当中的至少一项更新的数据写入到存储装置并行地创建所述逻辑拷贝。

7. 计算环境中的一种系统，其包括：

可操作来接收针对更新文件系统的文件系统对象的请求的接口；

可操作来确定将要发送到存储库以满足所述请求的一项或更多项 I/O 请求的 I/O 管理器；以及

检查点管理器，其可操作来：

确定将与针对更新文件系统对象的请求相关联的第一检查点，其中所述检查点管理器能够将各项请求指派给不同的检查点；

确定将把与所述检查点相关联的检查点数据写入到文件系统的存储装置；

确定对应于针对更新文件系统对象的后续请求的第二检查点；

等待文件系统的一致状态并且同时允许准备针对后续请求写入数据；

创建文件系统的元数据的逻辑拷贝；

把所述逻辑拷贝写入到存储装置；以及

把至少一项验证代码写入到存储装置，所述至少一项验证代码可用来确定所述检查点之前的更新是否已被写入到存储装置。

8. 权利要求 7 的系统，其中，所述检查点管理器可操作来确定将与针对更新文件系统对象的请求相关联的检查点包括：所述检查点管理器可操作来更新一个数据结构，其指示对于在确定把与第一检查点相关联的检查点数据写入到文件系统的存储装置之前所发生的更新将使用该检查点，并且对于此后发生的更新将使用第二检查点。

9. 权利要求 7 的系统，其中，所述检查点管理器可操作来确定把与检查点相关联的检查点数据写入到文件系统的存储装置包括：所述检查点管理器可操作来确定检查点定时器已到期，所述检查点定时器是基于恢复窗口。

10. 权利要求 7 的系统，其还包括可操作来生成写入计划的写入计划管理器，所述写入计划指示将与文件系统对象相结合地更新以便保持文件系统的一致状态的所有文件系统对象在所述存储装置上的位置。

11. 权利要求 7 的系统，其中，所述检查点管理器可操作来等待文件系统的一致状态包括：所述检查点管理器可操作来等到与第一检查点文件系统对象相关联的所有更新都被表示在文件系统的存储装置上为止。

12. 权利要求 7 的系统，其中，所述检查点管理器可操作来允许准备针对后续请求写入数据包括：所述检查点管理器可操作来允许针对后续请求生成写入计划并且将其写入到存储装置，但是直到创建了元数据的逻辑拷贝之后才允许更新所述元数据。

13. 一种至少部分地由计算机实施的方法，所述方法包括：

接收针对文件系统的恢复请求；

在文件系统的存储装置上定位检查点的检查点数据，所述检查点数据是先前通过包括以下动作而生成的：

指示将把在与所述检查点相关联的各项更新之后发生的任何更新指派给后续检查点；

针对与所述检查点相关联的各项更新生成写入计划，每一个写入计划指示用于表示至少其中一项所述更新的存储装置上的至少一个计划位置；

更新元数据以便指示由所述写入计划修改的对象的存储位置；

创建元数据的逻辑拷贝；以及

关于所述检查点把至少一项验证代码写入到存储装置；以及

利用所述验证代码验证所述检查点数据。

14. 权利要求 13 的方法，其中，利用所述验证代码验证所述检查点数据包括：计算所述检查点数据的校验和，并且把所述检查点数据的校验和与所述验证代码进行比较。

用于文件系统的检查点

背景技术

[0001] 在向存储器件写入数据的过程当中可能会发生停电或系统故障。当这种情况发生时，数据可能会丢失或者变得不一致。举例来说，如果在帐户持有者从 ATM 提取现金的过程当中发生系统故障，则所述交易可能会不公平地倾向于银行或帐户持有者。作为另一个例子，如果在涉及盘存取的长时间计算期间发生系统故障，则可能要花费很长时间来重做所述计算。

[0002] 这里所要求保护的主题不限于解决任何缺点或者只在例如前面所描述的环境中操作的实施例。相反，提供本背景描述只是为了说明可以在其中实践这里所描述的一些实施例的一个示例性技术领域。

发明内容

[0003] 简而言之，这里所描述的主题的各个方面涉及用于文件系统的检查点。在一些方面中，针对文件系统的更新被组织到检查点存储桶。当希望有检查点时，后续的更新被导向另一个检查点存储桶。在对于当前检查点存储桶中的各项更新对各个全局表进行了更新之后，创建所述全局表的逻辑拷贝。该逻辑拷贝被存储为检查点数据的一部分。为了帮助恢复，检查点管理器可以等到当前检查点存储桶的所有更新都被写入到存储装置之后才把最终检查点数据写入到存储装置。该最终检查点数据可以涉及所述全局表的逻辑拷贝，并且包括用以验证检查点数据的正确性的验证代码。

[0004] 提供本发明内容是为了简要地标识出将在下面的具体实施方式部分中进一步描述的主题的一些方面。本发明内容不意图标识出所要求保护的主题的关键或必要特征，也不意图被用来限制所要求保护的主题的范围。

[0005] 除非上下文中明确地另有所指，否则“这里所描述的主题”这一表达法指的是在具体实施方式部分中所描述的主题。术语“各个 / 一些方面”应被理解为“至少一个方面”。标识出在具体实施方式部分中所描述的主题的一些方面并不意图标识出所要求保护的主题的关键或必要特征。

[0006] 在附图中作为举例而非限制示出了这里所描述的主题的前述各个方面和其他方面，其中相同的附图标记指代类似的元件，并且在附图中：。

附图说明

[0007] 图 1 是表示可以把这里所描述的主题的各个方面合并到其中的示例性通用计算环境的方框图；

[0008] 图 2 是表示这里所描述的主题的各个方面可以在其中操作的系统的组件的示例性布置的方框图；

[0009] 图 3 是示出这里所描述的主题的各个方面的方框图；

[0010] 图 4 是总体上表示根据这里所描述的主题的各个方面的针对文件系统的更新的图示；

[0011] 图 5 是示出根据这里所描述的主题的各个方面的示例性检查点存储桶的方框图；以及

[0012] 图 6-8 是总体上表示根据这里所描述的主题的各个方面所可能发生的示例性动作的流程图。

具体实施方式

[0013] 定义

[0014] 这里所使用的术语“包括”及其变体应被理解为意味着“包括但不限于”的开放性术语。除非上下文中明确地另有所指，否则术语“或者”应被理解为“以及 / 或者”。术语“基于”应被理解为“至少部分地基于”。术语“一个实施例”和“一实施例”应被理解为“至少一个实施例”。术语“另一个实施例”应被理解为“至少一个其他实施例”。其他明确的和隐含的定义可以被包括在下面。

[0015] 示例性操作环境

[0016] 图 1 示出了可以在其上实施这里所描述的主题的各个方面的适当计算系统环境 100 的一个例子。计算系统环境 100 仅仅是适当计算环境的一个例子，而不意图暗示关于这里所描述的主题的各个方面的使用或功能范围的任何限制。计算环境 100 也不应当被解释为关于在示例性操作环境 100 中所示出的任何一个组件或组件组合有任何依赖性或要求。

[0017] 这里所描述的主题的各个方面适于与多种其他通用或专用计算系统环境或配置进行操作。适用于这里所描述的主题的各个方面的公知的计算系统、环境或配置的例子包括个人计算机、服务器计算机、手持式或膝上型器件、多处理器系统、基于微控制器的系统、机顶盒、可编程消费电子装置、网络 PC、小型计算机、大型计算机、个人数字助理(PDA)、游戏器件、打印机、各种电器(包括机顶盒、媒体中心或其他电器)、嵌入或附着到汽车上的计算器件、其他移动器件、包括任何前述系统或器件的分布式计算环境等等。

[0018] 可以在例如由计算机执行的程序模块之类的计算机可执行指令的一般情境中描述这里所描述的主题的各个方面。一般来说，程序模块包括施行特定任务或者实施特定抽象数据类型的例程、程序、对象、组件、数据结构等等。这里所描述的主题的各个方面还可以被实践在分布式计算环境中，其中各项任务由通过通信网络链接在一起的远程处理器件施行。在分布式计算环境中，各个程序模块可以同时处在包括存储器存储器件的本地和远程计算机存储介质中。

[0019] 参照图 1，用于实施这里所描述的主题的各个方面的示例性系统包括采取计算机 110 的形式的通用计算器件。计算机可以包括能够执行指令的任何电子器件。计算机 110 的组件可以包括处理单元 120、系统存储器 130 以及把包括系统存储器在内的各种系统组件耦合到处理单元 120 的系统总线 121。系统总线 121 可以是几种类型的总线结构当中的任一种，其中包括使用多种总线体系结构当中的任一种的存储器总线或存储器控制器、外围总线以及局部总线。作为举例而非限制，这样的体系结构包括工业标准体系结构(ISA)总线、微通道体系结构(MCA)总线、增强型 ISA (EISA)总线、视频电子标准协会(VESA)局部总线、外围组件互连(PCI)总线(其也被称作夹层总线)、外围组件互连扩展(PCI-E)总线、高级图形端口(AGP)以及快速 PCI (PCIe)。

[0020] 计算机 110 通常包括多种计算机可读介质。计算机可读介质可以是能够由计算机

110 访问的任何可用介质，并且包括易失性和非易失性介质以及可移除和不可移除介质。作为举例而非限制，计算机可读介质可以包括计算机存储介质和通信介质。

[0021] 计算机存储介质包括按照任何方法或技术实施的易失性和非易失性、可移除和不可移除介质，以用于存储诸如计算机可读指令、数据结构、程序模块或者其他数据之类的信息。计算机存储介质包括 RAM、ROM、EEPROM、闪存或其他存储器技术、CD-ROM、数字通用盘（DVD）或其他光盘存储装置、磁盒、磁带、磁盘存储装置或其他磁性存储器件或者可以被用来存储所期望的信息并且可由计算机 110 访问的任何其他媒体。

[0022] 通信介质通常把计算机可读指令、数据结构、程序模块或其他数据实现在调制的数据信号中，比如载波或其他传输机制，并且包括任何信息递送介质。术语“调制的数据信号”意味着信号的一项或更多项特性被以在所述信号中编码信息的方式设定或改变。作为举例而非限制，通信介质包括例如有线网络或直接连线连接之类的有线介质，以及例如声学、RF、红外和其他无线介质之类的无线介质。前述任何内容的各种组合也应当被包括在计算机可读介质的范围内。

[0023] 系统存储器 130 包括易失性和 / 或非易失性存储器的形式的计算机存储介质，比如只读存储器（ROM）131 和随机存取存储器（RAM）132。基本输入 / 输出系统 133（BIOS）通常被存储在 ROM 131 中，其包含例如在开机期间帮助在计算机 110 内的各个元件之间传输信息的基本例程。RAM 132 通常包含可由处理单元 120 立即访问和 / 或当前正由其操作的数据和 / 或程序模块。作为举例而非限制，图 1 示出了操作系统 134、应用程序 135、其他程序模块 136 以及程序数据 137。

[0024] 计算机 110 还可以包括其他可移除 / 不可移除、易失性 / 非易失性计算机存储介质。仅仅作为举例，图 1 示出了从不可移除的非易失性磁性介质进行读取或向其写入的硬盘驱动器 141，从不可移除的非易失性磁盘 152 进行读取或向其写入的磁盘驱动器 151，以及从可移除的非易失性光盘 156（比如 CD ROM 或其他光学介质）进行读取或向其写入的光盘驱动器 155。可以用在所述示例性操作环境中的其他可移除 / 不可移除、易失性 / 非易失性计算机存储介质包括磁带盒、闪存卡、数字通用盘、其他光盘、数字视频带、固态 RAM、固态 ROM 等等。硬盘驱动器 141 通常通过不可移除存储器接口（比如接口 140）连接到系统总线 121，并且磁盘驱动器 151 和光盘驱动器 155 通常通过可移除存储器接口（比如接口 150）连接到系统总线 121。

[0025] 前面所讨论并且在图 1 中示出的驱动器及其相关联的计算机存储介质对用于计算机 110 的计算机可读指令、数据结构、程序模块以及其他数据提供存储。在图 1 中，举例来说，硬盘驱动器 141 被图示为存储操作系统 144、应用程序 145、其他程序模块 146 以及程序数据 147。注意这些组件可以是与操作系统 134、应用程序 135、其他程序模块 136 以及程序数据 137 相同或不同的。操作系统 144、应用程序 145、其他程序模块 146 以及程序数据 147 在这里被给出不同的附图标记以便说明其至少是不同的拷贝。

[0026] 用户可以通过输入器件向计算机 110 输入命令和信息，所述输入器件例如是键盘 162 以及通常被称作鼠标、轨迹球或触摸板的指示器件 161。其他输入器件（未示出）可以包括麦克风、操纵杆、游戏手柄、碟形卫星信号收发天线、扫描仪、触敏屏、手写板等等。这些和其他输入器件常常通过耦合到系统总线的用户输入接口 160 连接到处理单元 120，但是也可以通过其他接口和总线结构连接，比如并行端口、游戏端口或通用串行总线（USB）。

[0027] 监视器 191 或其他类型的显示器件也通过诸如视频接口 190 之类的接口连接到系统总线 121。除了监视器之外，计算机还可以包括其他外围输出器件，比如扬声器 197 和打印机 196，其可以通过输出外围接口 195 连接。

[0028] 计算机 110 可以利用到一台或更多台远程计算机（比如远程计算机 180）的逻辑连接而操作在联网环境中。远程计算机 180 可以是个人计算机、服务器、路由器、网络 PC、对等器件或其他普通网络节点，并且通常包括前面关于计算机 110 所描述的许多或所有元件，但是在图 1 中仅仅示出了存储器存储器件 181。图 1 中所描绘的逻辑连接包括局域网（LAN）171 和广域网（WAN）173，但是也可以包括其他网络。这样的联网环境在办公室、企业范围计算机网络、内联网和因特网中是常见的。

[0029] 当被使用在 LAN 联网环境中时，计算机 110 通过网络接口或适配器 170 连接到 LAN 171。当被使用在 WAN 联网环境中时，计算机 110 可以包括调制解调器 172 或用于通过 WAN 173（比如因特网）建立通信的其他装置。调制解调器 172 可以是内部的或外部的，其可以通过用户输入接口 160 或其他适当机制连接到系统总线 121。在联网环境中，关于计算机 110 描绘的程序模块或其各个部分可以被存储在远程存储器存储器件中。作为举例而非限制，图 1 将远程应用程序 185 图示为驻留在存储器器件 181 上。应当认识到，所示出的网络连接是示例性的，并且可以使用在计算机之间建立通信链接的其他装置。

[0030] 检查点处理

[0031] 正如前面所提到的那样，在向存储器件写入数据的同时可能会发生停电和系统故障。这可能导致存储在存储器件上的数据处于不一致状态。为了解决这一问题和其他问题，可以向存储器件写入检查点。

[0032] 图 2 是表示这里所描述的主题的各个方面可以在其中操作的系统的组件的示例性布置的方框图。图 2 中示出的组件是示例性的，并且不意图全部包括可能需要或包括的所有组件。在其他实施例中，在不背离这里所描述的主题的各个方面的精神或范围的情况下，结合图 2 描述的组件和 / 或功能可以被包括在其他（所示出或未示出的）组件中或者被放置在子组件中。在一些实施例中，结合图 2 描述的组件和 / 或功能可以分布在多个器件上。

[0033] 翻到图 2，系统 205 可以包括一项或更多项应用 210、API 215、文件系统组件 220、存储库 250、通信机制 255 以及其他组件（未示出）。系统 205 可以包括一个或更多计算器件。这样的器件例如可以包括个人计算机、服务器计算机、手持式或膝上型器件、多处理器系统、基于微控制器的系统、机顶盒、可编程消费电子装置、网络 PC、小型计算机、大型计算机、蜂窝电话、个人数字助理（PDA）、游戏器件、打印机、各种电器（包括机顶盒、媒体中心或其他电器）、嵌入或附着到汽车上的计算器件、其他移动器件、包括任何前述系统或器件的分布式计算环境等等。

[0034] 在系统 205 包括单个器件的情况下，可以被配置成充当系统 205 的示例性器件包括图 1 的计算机 110。在系统 205 包括多个器件的情况下，所述多个器件当中的每一个可以包括类似地或不同地配置的图 1 的计算机 110。

[0035] 文件系统组件 220 可以包括恢复管理器 225、检查点管理器 230、I/O 管理器 235、写入计划管理器 237 以及其他组件（未示出）。这里所使用的术语“组件”应当被理解成包括一个器件的全部或一部分、一个或更多软件模块或其各个部分的总集、一个或更多软件

模块或其各个部分的某种组合以及一个或更多器件或其各个部分等等。

[0036] 通信机制 255 允许系统 205 与其他实体通信。举例来说，通信机制 255 可以允许系统 205 与远程主机上的应用进行通信。通信机制 255 可以是网络接口或适配器 170、调制解调器 172 或者用于建立结合图 1 所描述的通信的任何其他机制。

[0037] 存储库 250 是能够提供对于数据的访问的任何存储介质。所述存储库可以包括易失性存储器（例如高速缓存）和非易失性存储器（例如永久性存储装置）。“术语”数据应当被广泛地理解成包括可以由一个或更多计算机存储元件表示的任何事物。在逻辑上，数据可以在易失性或非易失性存储器中被表示为一系列 1 和 0。在具有非二进制存储介质的计算机中，可以根据存储介质的能力来表示数据。数据可以被组织成不同类型的数据结构，其中包括例如数字、字母等等的简单数据类型，分级、链接或其他有关数据类型，包括多个其他数据结构或简单数据类型的数据结构等等。数据的一些例子包括信息、程序代码、程序状态、程序数据、其他数据等等。

[0038] 存储库 250 可以包括硬盘存储装置、其他非易失性存储装置、例如 RAM 之类的易失性存储器、其他存储装置、前述各项的某种组合等等，并且可以跨多个器件分布。存储库 250 可以是外部的、内部的，或者同时包括处于系统 205 内部和外部的组件。

[0039] 可以通过存储控制器 240 来访问存储库 250。这里所使用的“访问”可以包括读取数据、写入数据、删除数据、更新数据、包括前述各项当中的两项或更多项的某种组合等等。存储控制器 240 可以接收针对访问存储库 250 的请求，并且可以适当地满足这样的请求。存储控制器 240 可以被布置成使其不保证将按照接收到数据的顺序将其写入到存储库 250。此外，在存储控制器 240 把所请求的数据实际写入到存储库 250 的非易失性存储器之前，存储控制器 240 就可以指示其已经写入了所述数据。

[0040] 所述一项或更多项应用 210 包括在创建、删除或更新数据的过程中所可能涉及的任何处理。这样的处理可以在用户模式或内核模式下执行。这里所使用的术语“处理”及其变型可以包括一个或更多传统进程、线程、组件、库、施行任务的对象等等。一个处理可以用硬件、软件或者硬件与软件的组合来实施。在一个实施例中，一个处理是能够施行一项动作或者被用于施行一项动作的任何机制（不管如何称呼）。一个处理可以分布在多个器件或单个器件上。所述一项或更多项应用 210 可以通过 API 215 向 I/O 管理器 235 做出文件系统请求（例如通过函数 / 方法调用）。

[0041] I/O 管理器 235 可以确定将向存储控制器 240（或者某一其他中间组件）发出哪一条或哪些 I/O 请求。I/O 管理器 235 还可以随着与文件系统请求相关联的操作的继续、完成或失败而向一项或更多项应用 210 返回数据。当文件系统请求涉及事务时，I/O 管理器 235 可以通知事务管理器（未示出），从而事务管理器可以适当地管理事务。在一些实施例中，事务管理器的功能可以被包括在 I/O 管理器 235 中。

[0042] 文件系统组件 220 可以使用写时拷贝、原位写入、前述各项的某种组合等来把文件系统对象或者关于文件系统对象的元数据写入到存储库 250。术语“文件”可以包括目录、不具有子代（其例如有时被视为文件）的文件系统对象、其他文件系统对象等等。

[0043] 在写时拷贝中，在修改文件的数据之前，把将被修改的数据的一份拷贝拷贝到另一个位置。在原位写入中，可以在原位修改文件的数据，而不把原始数据拷贝到另一个位置。写时拷贝和原位写入的混合可以包括针对关于文件的元数据施行写时拷贝，同时针对

包括在文件中的数据施行原位写入。

[0044] 可以在事务的情境中更新文件系统的对象。事务是可以通过各种属性描述的一组操作，所述属性例如包括原子性、一致性、隔离性和持久性。这里所使用的“事务”可以至少由一致性属性定义，并且还可以由前面的一项或更多项其他属性定义。

[0045] 一致性属性指的是关于一个或更多文件所允许的数据状态。在事务开始之前或者在事务完成之后，文件系统的各个文件应当处于所允许的状态(尽管其在事务期间可以经历不被允许的状态)。举例来说，银行事务可以被实施为由两项操作构成的集合：来自一个帐户的借记，以及针对另一个帐户的信贷。在该例中，一致性可以被定义为使得银行和帐户持有者的组合帐户结余是一个常数(例如 T=A+B，其中 T 是一个常数，A= 银行结余，B= 帐户持有者结余)。为了在该例中实施一致性，所述借记和信贷操作只需要是针对相同的金额，并且或者是二者全部完成，或者是在每一个帐户上都没有完成。

[0046] 可以写入检查点以指示文件系统的一致状态。检查点可以包括一个或更多验证代码(例如一个或更多校验和、散列或其他数据)，其可以被用来确定检查点和 / 或与检查点相关联的数据是否被正确地写入到盘上。在恢复时，可以找到最近一次写入的检查点。随后可以使用所述检查点的(多个)验证代码来确定该检查点和 / 或与该检查点相关联的数据是否被正确地写入到盘上。如果不是，则可以定位前一个检查点并且检查其有效性，直到找到有效的检查点为止。一旦找到最近的有效检查点，就知道文件系统的最近的一致状态。在该点之后发生的文件系统操作可以被丢弃，或者可以按照希望施行附加的恢复动作。

[0047] 在一个实施例中，文件系统上的对象可以用 D_n 表示，其中 n 向系统标识该对象。文件系统上的各个对象被串行化(即能够被表示为存储库 250 上的数据)及解串行化。一个对象表将每一个对象标识符与其在存储库 250 上的位置相关联。

[0048] 在修改事务中第一次更新 D_n 时，通过利用 n 查找其在对象表中的位置来找到 D_n。为了用在一个例子中，D_n 在存储库 250 上的存储位置被称作 L_1。

[0049] 随后从存储库 250 读取 L_1 的内容，可以将所述对象解串行化(例如从串行化格式转换成所述对象的结构)，并且把所述对象的将要修改的各个部分拷贝到主系统存储器中。在存储器中对所述各个部分(或其拷贝)施行更新。结合存储器中的各个部分被修改，在存储库 250 上为经过修改的部分指定一个或更多新位置(其被称作 L_2)。

[0050] 主系统存储器中的这些拷贝有时在这里被称作对象的“逻辑拷贝”。对象的逻辑拷贝包括可以被用来表示该对象的一个或更多数据结构。在逻辑上，逻辑拷贝是对象的复制品。在物理上，逻辑拷贝可以包括能够被用来创建对象的复制品的数据(其中包括指向其他数据的指针)。举例来说，在一种实现方式中，逻辑拷贝可以是对象的实际拷贝(例如逐比特拷贝)，或者是包括可以被用来创建所述对象的数据的数据结构。

[0051] 在另一种实现方式中，未经修改的逻辑拷贝可以包括指向原始对象的一个或更多指针。随着所述逻辑拷贝被修改，该逻辑拷贝中的一些指针可以指向新的存储器位置(例如对于所述逻辑拷贝的已改变部分)，而其他指针则可以指向原始对象的各个部分(例如对于所述逻辑拷贝的未改变部分)。利用所述指针，可以利用经过修改的数据连同原始对象的未经修改的数据一起构造经过修改的拷贝。例如可以施行创建逻辑拷贝以便减少创建对象的复制品所需的存储空间。

[0052] 此外，虽然在这里有时提到了串行化和解串行化，但是不意图把这里所描述的主

题的各个方面限制到习惯上所考虑的串行化和解串行化。在一个实施例中，串行化版本可以与解串行化版本逐比特完全相同。在另一个实施例中，串行化版本的比特可以按照不同于解串行化版本的格式和顺序被打包。实际上，在一个实施例中，串行化和解串行化应当被理解成意味着用于存储以及从存储库取回表示对象的数据的任何机制。其他机制例如可以包括以文字格式把对象的属性写入到存储库、用标记语言把对象的属性编码在存储库中、将对象的属性和其他特征存储在存储库上的其他方式等等。

[0053] 在系统的意愿下(例如在事务提交之后或者某一其他时间)，所述系统可以把经过修改的逻辑拷贝串行化回到稳定介质，但是这是在位置 L_2 处实施的。把经过修改的逻辑拷贝写回到新位置的所述意图被称作写入计划。写入计划可以标识针对一个或更多对象的任意数目的更新。写入计划可以涉及发生在多于一项事务中的改变。可以把多个写入计划组合成单个写入计划。

[0054] 写入计划管理器 237 可以参与创建针对各项更新的写入计划。当写入计划涉及多个文件系统对象时(例如在一项事务的情境中)，写入计划管理器 237 可以操作来生成指示在所述事务中所涉及的所有文件系统对象在存储装置上的位置的写入计划，以便对于文件系统保持一致状态。

[0055] 当修改紧接在检查点之后发生时，可以修改被称作恢复块的一个块(其可以被复制在多个位置处)，以便指向经过修改的逻辑拷贝的起始(即 L_2)。L_2 处的对象中的一个字段指向接下来将被写入的位置。该字段表示发生在检查点之间的写入计划链中的一环。

[0056] 结合发送针对写入逻辑拷贝的请求，可以对所述对象表做出修改。具体来说，由对象标识符索引的位置值可以被设定到将要在该处存储经过修改的逻辑拷贝的位置值(即 L_2)。通过这样做使得对于对象 D_n 的位置的后续查找将被引到位置 L_2，即对象的新版本。

[0057] 如果一项事务修改了多于一个对象，例如 D_i 和 D_j，则所述对象被视为彼此“原子结合”，并且在一个写入计划中被写入。写入计划可以指明这一关系(例如以针对所涉及对象的链接的形式)。

[0058] 按照这种方式，可以延续任意数目的对象。还可以按照与任何其他对象相同的方式将对象表周期性地写入到存储库 250。

[0059] 结合发送针对把对象表写入到存储库 250 的请求，还可以向存储控制器 240 发送冲刷命令。冲刷命令指示存储控制器 240 把来自其易失性存储器的尚未被写入的所有数据写入到存储库 250 的非易失性存储器。

[0060] 可以周期性地把检查点写入到存储装置，正如下面将更加详细地描述的那样。可以通过由存储库 250 存储的检查点记录来指示检查点。检查点可以在任何时间被写入，并且可以在冲刷之后变得稳定 / 持久。稳定 / 持久指的是检查点被存储在存储库的非易失性存储器上。

[0061] 在检查点稳定 / 持久之后，可以对被用于任何早前的和未被使用的对象拷贝(或其部分)的空间进行再利用。在冲刷完成之后，随后将恢复块指向接下来的写入计划链的起始。在一个实施例中，所述恢复块可以把所述写入计划链的起始指向对象表的新位置。

[0062] 结合图 3 描述一个更加具体的例子，该图是示出这里所描述的主题的各个方面的方框图。如图所示，图 3 示出了主存储器 305 和存储库 250。线 307 代表主存储器 305 与存储库 250 之间的划分。线 310 上方的对象处于主存储器中，而线 310 下方的对象则处于存

储库 250 的易失性或非易失性存储器中。

[0063] 在主存储器 305 中示出了对象 314-316。在实施过程中，对象 314-316 可以分别是对象 319-321 的解串行化逻辑拷贝。对象 319 处于存储库 250 上的位置 1550 处，对象 320 处于存储库 250 上的位置 200 处，以及对象 321 处于存储库 250 上的位置 800 处。

[0064] 对象表 310 包括指示对象 314-316 在存储库 250 上的位置的关键值对。利用对象 314-316 的标识符（n）对各个关键值对进行索引。

[0065] 当事务修改对象 316 时（例如通过将其名称改变为 foo.txt），一致性组件（例如图 2 的一致性组件 220）可以为更新后的对象确定新的存储位置（例如位置 801）。如果所述对象是一个文件，则在事务的情境中更新其名称还可能导致包括该文件的目录也被涉及在所述事务中。举例来说，当文件名被改变时，表示该文件的对象和表示包括该文件的目录的对象可能需要被涉及在所述事务中。在这种情况下，包括所述对象的目录被表示为对象 314，并且更新后的目录（例如对象 318）的逻辑拷贝被表示为存储库 250 中的对象 323。此外，表 310 已被逻辑地更新到表 311 以指示经过修改的对象（即对象 317 和 318）的新的存储位置（即 801 和 1000）。

[0066] 例如可以由 I/O 管理器 235 或图 2 的某一其他组件明确地指示或确定在事务的情境内修改某一对象还会影响另一个对象。

[0067] 当在一项事务的更新中涉及两个或更多对象时，所述对象被视为“原子结合”，正如前面所提到的那样。在恢复操作中，除非在存储库 250 中找到对应于在所述事务的情境中改变的所有对象的改变，否则所找到的所有改变都被丢弃。换句话说，如果找到了对应于其中一个对象的改变但是没有找到对应于另一个对象的改变，则丢弃对应于所述其中一个对象的改变。

[0068] 为了原子结合两个或更多对象，在一个实施例中，可以在存储库 250 中存储一个指针或者通过其他方式将其与每一个对象相关联。一个指针可以指示在事务中涉及到的另一个对象（或其部分）的存储位置。如果在事务中没有涉及附加的对象，则所述指针可以指向一个“死块”，或者指示另一个写入计划的“头”对象的存储位置。该头对象可以包括一个写入计划、所述写入计划的经过修改的对象（或其部分）等等。

[0069] 除了指向下一个存储位置的指针之外，还可以在存储库 250 中存储用以指示所“指向”的对象的正确内容的数据。举例来说，可以存储指示所指向的对象的正确内容的散列。

[0070] 在图 3 给出的例子中，与对象 322 相关联的指针可以指向与对象 323 相关联的存储位置。所述指针把这两个对象结合在一起。如果在恢复期间没有找到其中的一个对象或者其不具有正确的内容，则可以丢弃由所找到的对象表示的改变。

[0071] 由于存储库 250 的性质，可能无法保证哪一个对象将首先被写入到存储库 250 的非易失性存储器。如果对象 322 首先被写入并且对象 323 没有被写入，则来自对象 322 的指针将指向可能具有虚假数据的存储位置。但是通过计算所述存储位置处的数据的散列并且将该散列与对于对象 322 所存储的散列进行比较，可以把位置 1000 处的数据检测为具有无效数据。在这种情况下，在恢复期间，恢复管理器（例如图 2 的恢复管理器 225）可以丢弃由对象 322 和 323 所表示的改变。

[0072] 恢复块 330 指向在检查点之后本应在该处存储数据的第一存储位置（在本例中是

位置 801)。恢复块 330 还可以包括利用存储在所述第一存储位置处的对象的正确内容计算的散列或者与之相关联。

[0073] 图 4 是总体上表示根据这里所描述的主题的各个方面的发生在文件系统上的更新的图示。全局表 405 包括标识出各个对象在存储库上的位置的对象表以及关于存储库 250 上的已被分配的空间的分配数据。此外还示出了进行中的更新 410。当更新接触到时间轴 415 时,所述更新完成,并且不再需要修改全局表 405 的任何内容。更新 410 的每一条更新线可以表示多项更新。如果需要一起做出多项更新以便保持一致性,则可以在事务的情境中做出所述更新。

[0074] 为了使得检查点有效,需要在一致状态下写入所述检查点。对于写时拷贝文件系统,当更新某一对象时,所修改的对象的逻辑拷贝被存储在文件系统的一个新位置处。该新位置在对象表中由对于对象表的更新反映出来。为了一致性,对象表反映出尚未被写入到盘上的更新将是错误的,这是因为在系统故障之前所述更新可能没有完全被写入到盘上。类似地,如果更新完成并被写入到盘上并且其他事务相关的更新也被完成但是对象表没有表现出所述更新,则这也将是错误的。

[0075] 为了确保一致性,需要在全局表中反映出对应于更新的元数据时选择检查点。如果表示更新 410 的每一条线指示可以针对所述更新对全局表 405 做出更新的一个周期,则在时间 520 施行检查点可能产生不一致状态,而在时间 525 施行检查点则将产生一致状态。

[0076] 图 5 是示出根据这里所描述的主题的各个方面的示例性检查点存储桶的方框图。为了解决前面提到的问题和其他问题,每一项更新可以与一个检查点存储桶(例如其中一个存储桶 515)相关联。检查点存储桶是指示在把检查点的检查点数据写入到盘上之前需要更新全局表以便至少考虑到与所述检查点存储桶相关联的更新的写入计划的逻辑概念。换句话说,需要更新全局表以便考虑到存储桶的各项更新的位置和分配信息,尽管所述更新当前可能或者可能没有被写入到这些位置。

[0077] 可以周期性地(例如在基于恢复窗口的检查点定时器到期时、在发生了一定次数的写入之后、在超出了某一其他阈值之后等等)做出生成检查点的决定。当这种情况发生时,检查点管理器可以更新指示检查点存储桶与后续更新相关联的数据(例如数据结构 510)。举例来说,检查点管理器可以获得指示当前检查点存储桶的数据(例如数据结构 510)上的互斥锁定(例如锁定 505)。在检查点管理器获得了所述数据上的互斥锁定之后,检查点管理器可以更新所述数据以指示对应于后续更新的新的检查点存储桶。所有后续更新都与所述新的检查点存储桶相关联,直到所述数据被改变以指示对应于后续更新的另一个检查点存储桶为止。

[0078] 检查点存储桶可以被视为一种逻辑概念,并且可以通过多种方式来实施。举例来说,在一种实现方式中,检查点存储桶可以被实施为诸如列表之类的数据结构,所述列表具有指向与检查点存储桶相关联的每一项更新的指针。作为另一个例子,检查点存储桶可以被实施为针对每一项更新保持的数据,其中所述数据指示与所述更新相关联的检查点。作为另一个例子,检查点存储桶可以被实施为计数信号量。在该例中,可能不知道哪些更新仍需要被写入到盘上,但是知道仍需要被写入到盘上的更新的计数。在该例中可以使用读取 / 写入锁定。

[0079] 前面的例子不意图包含或穷举实施检查点存储桶的所有方式。实际上,基于这里

的教导，本领域技术人员可以认识到用于实施检查点存储桶的许多其他机制。

[0080] 在指示对应于后续更新的检查点存储桶（例如通过改变数据结构 510）之后，检查点管理器可以等待生成对应于当前检查点存储桶中的所有更新的写入计划。在生成了对应于当前检查点存储桶中的所有更新的写入计划之后（但是可能未将其写入到存储装置），检查点管理器可以取得图 4 的全局表 405 的快照，并且创建针对把全局表 405 的快照写入到存储库的写入计划。可以通过写时拷贝或其他机制将快照创建为全局表 405 的一份逻辑拷贝。

[0081] 回到图 4，可以生成对应于检查点之后的更新的写入计划并且将其写入到盘上，同时检查点管理器等待生成当前检查点存储桶中的所有更新，并且同时检查点管理器还生成针对写入检查点的写入计划。但是当检查点管理器尝试获得全局表的快照时，检查点管理器可以在创建所述快照之前获得全局表 405 上的互斥锁定。在检查点管理器具有互斥锁定的同时，仍然可以对于其他更新生成写入计划并且甚至可以将这些写入计划存储在存储库上，但是直到检查点管理器释放其互斥锁定之后，全局表（例如对象表）才可以被更新以便指向这些写入计划。结合释放所述锁定，检查点管理器可以发送一个信号（例如引发某一事件），其指示后续检查点已被启用并且后续的更新可以对全局表进行更新。

[0082] 为了帮助恢复，可以将检查点与验证代码一起写入到盘上以便根据下面的规则验证所述检查点：

[0083] 1、等待由写入计划指示的数据被写入到盘上（例如等待与检查点相关联的所有更新被写入到盘上）；

[0084] 2、请求把与检查点相关联的所有数据写入到盘上（例如请求把元数据的逻辑拷贝写入到盘上）；

[0085] 3、发起或等待冲刷并且等待表明冲刷已成功完成的确认。

[0086] 4、生成对应于已被写入到盘上的检查点的验证代码。在一个实施例中，所述验证代码可以对应于已被写入到盘上的数据的一个子集。举例来说，如果对应于文件的数据被存储在树中，其中所述树的每一个节点包括对应于其子代的验证代码，则所述验证代码可以是对应于树的根节点。在该实施例中，验证代码可以与根节点一起写入，并且还可以被用来验证所述验证代码是正确的。

[0087] 5、请求把验证代码（以及比如根节点之类的任何相关联的数据）写入到盘上。注意在系统发生故障之前，验证代码可能没有实际上到达盘上。如果没有的话，则所述检查点不是有效检查点。

[0088] 有了这些规则，在恢复期间，如果在存储装置上找到检查点并且检查点的内部验证代码是有效的，则与检查点相关联的其他数据也预期会被存储在所述存储装置上并且是有效的。如果验证代码被包括在根节点中，则根节点中的其他数据（例如指向树中的其他节点的指针）可以被用来找到对应于检查点的其余数据。

[0089] 作为一种替换方案，对应于与检查点相关联的每一项更新的验证代码可以被写入到存储装置。举例来说，检查点可以指示本应在该检查点之前并且在前一个检查点之后发生的所有更新的各个块。对于所指示的每一个块，检查点可以存储指示该块的正确内容的验证代码。在这种替换方案中的恢复期间，为了验证检查点，可以将每一个块关于其检查点的相关联的验证代码进行验证。

[0090] 回到图 2，在一个实施例中，检查点管理器 230 可以操作来施行包括以下各项动作：

[0091] 1、确定将与针对更新文件系统对象的请求相关联的第一检查点。如前所述，检查点管理器 230 可以通过更新某一数据结构（例如图 5 的数据结构 510）以便指向新的检查点存储桶来实现这一点。随后随着接收到针对更新的每一则后续请求，可以把该请求指派到该新的检查点存储桶。

[0092] 注意这里所使用的术语“第一”并不意味着实际的第一个检查点；相反其被用来与“第二”检查点进行区分。换句话说，如果存在 N 个检查点，则第一检查点可以是其中 1<=X<=N 的任意 X，并且第二检查点可以是其中 1<=Y<=N 并且 X<>Y 的任意 Y。

[0093] 2、确定何时把与检查点相关联的检查点数据写入到文件系统的存储装置。举例来说，可以是检查点定时器到期、可以是更新数据被超过，或者某一其他阈值可以用来确定到了写入检查点数据的时间。

[0094] 3、对于针对更新文件系统对象的后续请求确定第二检查点。如前所述，检查点管理器 230 可以通过在数据结构（例如图 5 的数据结构 510）上获得互斥锁定（例如锁定 505）之后更新所述数据结构来实现这一点。

[0095] 4、等待文件系统的一致状态并且同时允许准备对于后续请求写入数据。当与当前检查点存储桶相关联的所有更新被表示在（例如已被成功地写入到）存储装置上时，一致状态发生。允许准备对于后续请求写入数据包括允许对于后续请求生成写入计划并且写入到存储装置，但是直到创建了元数据（例如全局表）的逻辑拷贝之后才允许更新所述元数据。

[0096] 5、创建文件系统的元数据的逻辑拷贝。这一点可以通过如前所述地取得全局表的快照来实现。

[0097] 6、把元数据的逻辑拷贝写入到存储装置。在一个实施例中，这方面可以包括请求把逻辑拷贝写入到存储装置并且等待关于所述逻辑拷贝已被写入到存储装置的证实。在另一个实施例中，这方面可以包括在允许对于元数据的后续更新之前将存储装置上的所述拷贝标记为清洁，从而使得所述更新导致写时拷贝。

[0098] 7、将至少一项验证代码写入到存储装置。如前所述，可以使用验证代码来确定检查点之前的更新是否被写入到存储装置以及检查点记录本身是否有效。

[0099] API 215 可以接收针对修改在一项事务中涉及的对象的请求。作为响应，I/O 管理器 235 可以在存储库的某一存储位置（例如 L_1）处定位所述对象，创建所述对象的逻辑拷贝，在事务的情境中对所述对象做出改变，确定用于存储经过改变的逻辑拷贝的第二存储位置（例如 L_2），向存储控制器 240 发送针对写入经过改变的逻辑拷贝的请求，以及更新易失性数据结构（例如对象表 310）以指示逻辑拷贝被存储在第二存储位置处。

[0100] 如果 API 215 接收到针对在所述事务中涉及的另一个对象的请求，则 I/O 管理器 235 可以施行附加的动作，其中包括创建将所述其他对象与第一对象结合在一起的关联（例如写入计划）。随后，结合发送针对把各个对象的修改写入到存储装置的请求，I/O 管理器 235 还可以向存储控制器 240 发送针对把所述关联写入到存储装置的请求。

[0101] 图 6-8 是总体上表示根据这里所描述的主题的各个方面所可能发生的示例性动作的流程图。为了解释起来简单，结合图 6-8 描述的方法被描绘并且描述为一系列步骤。应当理解并且认识到的是，这里所描述的主题的各个方面不限于所示出的步骤和 / 或各个步

骤的顺序。在一个实施例中，各个步骤按照下面所描述的顺序发生。但是在其他实施例中，各个步骤可以并行地发生、按照另一顺序发生以及 / 或者与未在这里给出并描述的其他步骤一起发生。此外，可能并不需要所示出的所有步骤来实施根据这里所描述的主题的各个方面的方法。此外本领域技术人员还将理解并且认识到，所述方法可以替换地通过状态图被表示为一系列互相关的状态或者被表示为事件。

[0102] 翻到图 6，在方框 605 中，所述动作开始。在方框 610 中，做出指示将把第一更新集合与第一检查点相关联的指示。这方面可以通过修改某一数据结构以指示将把后续更新与第一检查点相关联来实现。这方面例如可以涉及获得并且释放锁定以及更新指针或其他数据结构以便指向某一检查点存储桶，正如前面所提到的那样。再次注意，“第一”可以意味着文件系统的任意检查点，并且被用来把该检查点与后续检查点进行区分。举例来说，参照图 2 和 5，检查点管理器 230 可以获得数据结构 510 上的锁定 505，并且更新所述指针以便指向其中一个检查点存储桶 515。

[0103] 在方框 615 中，接收更新并且将其与第一检查点相关联。举例来说，参照图 2，I/O 管理器 235 可以通过 API 215 从（多个）应用 210 接收更新请求。在接收到更新时，可以将其与检查点相关联。

[0104] 在方框 620 中，确定将第一检查点的检查点数据写入到文件系统的存储装置。举例来说，参照图 2，检查点管理器 230 可以确定检查点定时器已到期，并且可以由此确定将把检查点写入到存储库 250。

[0105] 在方框 625 中，在用于指示对应于后续更新的检查点的数据结构上获得锁定。举例来说，参照图 2 和 5，检查点管理器 230 可以获得数据结构 510 上的锁定 505。

[0106] 在方框 630 中，更新所述数据结构以便指向另一个检查点。修改该数据结构指示将把在第一更新集合之后发生的任何更新与后续检查点相关联。举例来说，参照图 2 和 5，检查点管理器 230 可以更新数据结构 510 以便指向另一个检查点存储桶 515。

[0107] 在方框 635 中，释放所述锁定。举例来说，参照图 2 和 5，检查点管理器 230 可以释放锁定 505。

[0108] 在方框 640 中，生成对应于所述更新的写入计划。每一个写入计划对于表示第一更新集合当中的至少一项更新的数据指示存储装置上的至少一个计划位置。举例来说，参照图 2，写入计划管理器 237 可以参与创建针对与某一检查点相关联的更新的写入计划。

[0109] 在方框 645 中，对于所述写入计划更新元数据。该元数据指示对应于各个写入计划的存储位置（尽管所述写入计划可能已被或者可能尚未被写入到存储装置）。举例来说，参照图 2，写入计划管理器 237 可以更新全局表以便指示由所述写入计划修改的存储位置对象。

[0110] 在方框 645 之后，所述动作在图 7 的方框 705 中继续。翻到图 7，在方框 705 中，对于所述元数据获得锁定。举例来说，参照图 2 和 4，检查点管理器 230 可以在全局表 405 上获得锁定。检查点管理器 230 可以等到元数据反映出对应于第一更新集合当中的所有更新的存储位置为止（尽管所有这些更新可能已被或者可能尚未被写入到这些存储位置）。

[0111] 在方框 710 中，创建元数据的逻辑拷贝。如前所述，这方面可以涉及创建元数据的新拷贝，将元数据标记为清洁从而使得对于所述元数据的后续更新导致写时拷贝，或者某种其他逻辑拷贝机制。举例来说，参照图 2 和 4，检查点管理器 230 可以制作全局表 405 的

逻辑拷贝。

[0112] 在方框 715 中，释放所述锁定。举例来说，参照图 2 和 4，检查点管理器 230 可以释放全局表 405 上的锁定。

[0113] 在方框 720 中，创建针对写入第一检查点数据的写入计划。创建该写入计划可以与对于所述检查点之后的更新生成写入计划（并且将其写入到盘上）以及把对应于当前写入计划的数据写入到盘上并行地发生。举例来说，参照图 2，检查点管理器 230 可以使用写入计划管理器 237 来创建针对第一检查点的检查点数据的写入计划。该数据可以包括全局表的逻辑拷贝，正如前面所提到的那样。

[0114] 在方框 725 中，在一个实施例中，检查点管理器可以等到第一更新集合的所有更新都被成功地写入到存储装置为止。在所有更新都被成功地写入到存储装置之后，更新管理器随后可以写入包括验证代码的最终检查点记录。正如前面所提到的那样，这样允许恢复处理简单地检查所述验证代码以便确定是否对应于所述检查点的所有更新都预期已被写入到存储装置。

[0115] 在另一个实施例中，检查点管理器可以在一条检查点记录中写入几项验证代码。这些验证代码可以与第一更新集合当中的各项更新的存储位置相关联。在该实施例中，检查点管理器可以等到这些更新被写入到存储装置，或者可以在不做等待的情况下写入检查点记录。如果选择了后一选项，则与验证有效检查点记录处于盘上相比，在恢复期间可能更多地涉及找到适当的检查点。

[0116] 在方框 730 中，可以把检查点数据写入到存储装置。这方面例如可以涉及把与检查点数据相关联的写入计划写入到存储装置。作为另一个例子，这方面可以涉及把涉及各个全局表的逻辑拷贝的检查点记录写入到存储装置。举例来说，参照图 2，检查点管理器 230 可以请求把对应于检查点数据的写入计划写入到存储装置。

[0117] 在方框 735 中，将至少一项验证代码写入到存储装置。可以将把至少一项验证代码写入到存储装置的措施与把涉及各个全局表的逻辑拷贝的检查点记录写入到存储装置的措施相组合。举例来说，参照图 2，检查点管理器 230 可以把检查点记录写入到存储装置，所述检查点记录涉及各个全局表的逻辑拷贝并且包括用于验证所述检查点记录的内容的验证代码。

[0118] 在方框 740 中，可以施行其他动作（如果有的话）。

[0119] 翻到图 8，在方框 805 中，所述动作开始。在方框 810 中，接收恢复请求。举例来说，参照图 2，恢复管理器 225 可以接收针对对于存储在存储库 250 上的数据施行恢复的恢复请求。

[0120] 在方框 815 中，定位检查点数据。举例来说，参照图 2，恢复管理器 225 可以定位存储在存储库 250（或某一其他存储库）上的最近的检查点数据。

[0121] 在方框 820 中，利用验证代码对所述检查点数据进行验证。举例来说，参照图 2，恢复管理器 225 可以计算检查点数据的校验和，并且把该校验和与和检查点数据一起存储的校验和进行比较。如果两个校验和匹配，则可以认为检查点是有效的。如果希望进行额外的验证，则恢复管理器可以尝试验证由检查点数据所涉及的各个全局表所指示的一个或更多对象。

[0122] 在方框 825 中，可以施行其他动作（如果有的话）。

[0123] 从前面的具体实施方式可以看到，前面描述了涉及用于文件系统的检查点的各个方面。虽然这里所描述的主题的各个方面可以有各种修改和替换构造，但是在附图中示出并且在前面具体实施方式了其特定所示实施例。但是应当理解的是，所要求保护的主题的各个方面不意图被限制到所公开的具体形式，相反，这里的意图是涵盖落在这里所描述的主题的各个方面的精神和范围内的所有修改、替换构造和等同方案。

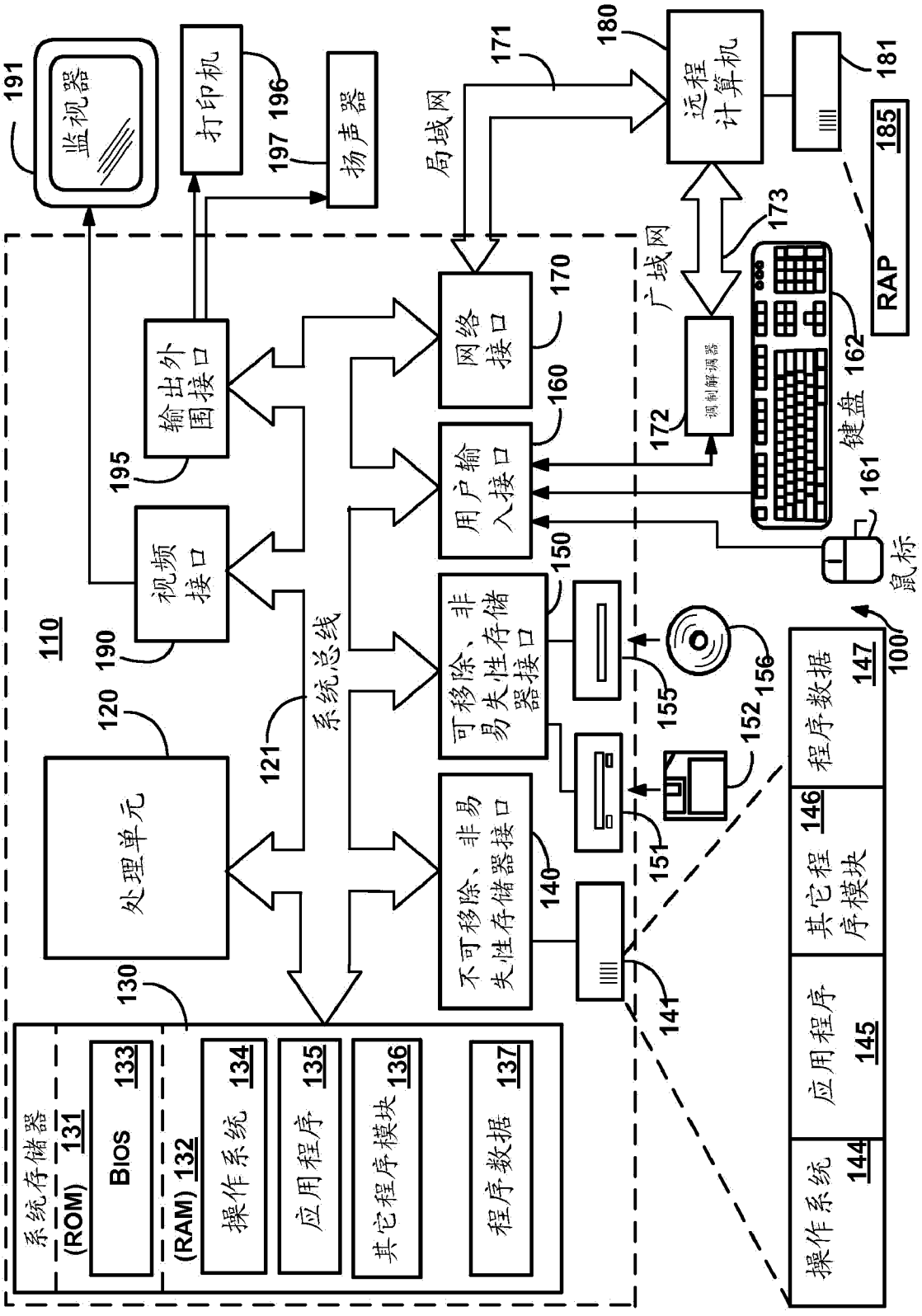

191
监视器
打印机
196
197
扬声器
局域网
171
180
远程计算机
181
185
RAP
173
广域网
调制解调器
172
键盘
162
161
鼠标
170
网络接口
160
输出外围接口
195
用户输入接口
150
视频接口
190
110
可移除、非易失性存储器接口
155
156
152
120
处理单元
121
系统总线
不可移除、非易失性存储器接口
140
151
141
130
系统存储器
(ROM) 131
BIOS 133
(RAM) 132
操作系统 134
应用程序 135
其它程序模块 136
程序数据 137
操作系统 144
应用程序 145
其它程序模块 146
程序数据 147
100

图 1

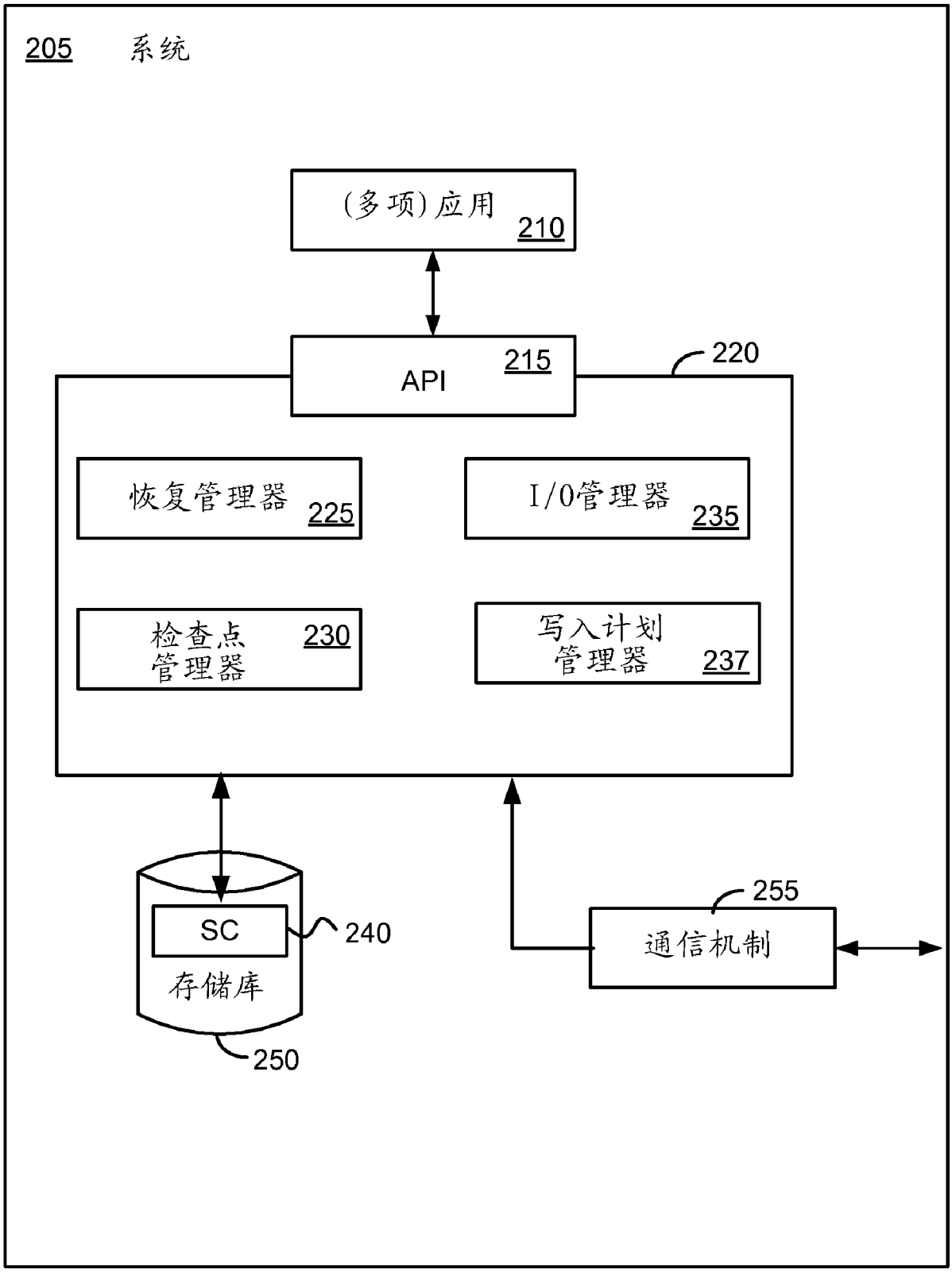

205 系统
(多项)应用 210
API 215
220
恢复管理器 225
I/O管理器 235
检查点管理器 230
写入计划管理器 237
SC 240
存储库
250
255
通信机制

图 2

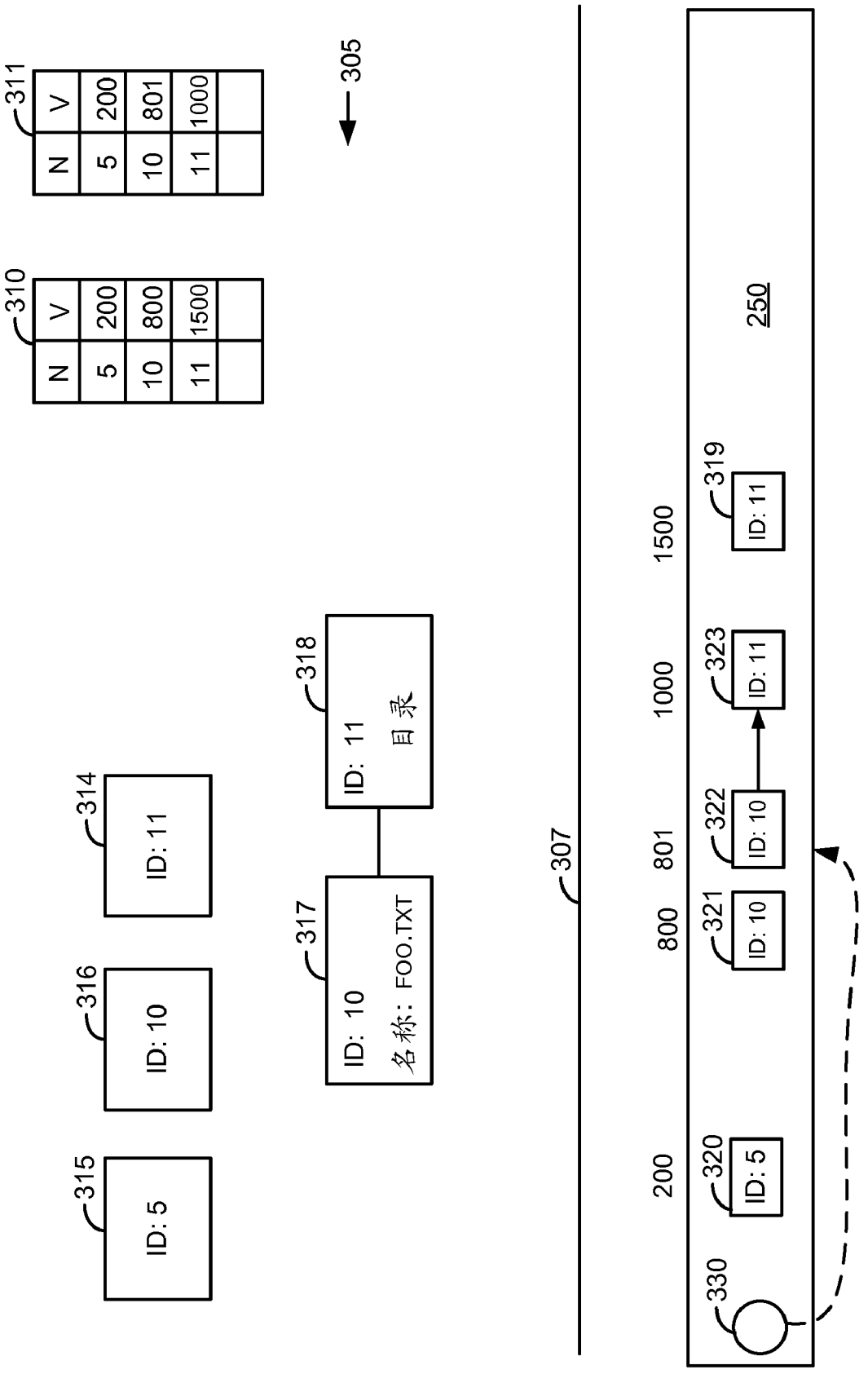

311
N
V
5
200
10
801
11
1000
305
310
N
V
5
200
10
800
11
1500
250
1500
319
ID: 11
1000
323
ID: 11
322
ID: 10
801
321
ID: 10
800
320
ID: 5
200
330
307
318
ID: 11
目录
317
ID: 10
名称：FOO.TXT
314
ID: 11
316
ID: 10
315
ID: 5

图 3

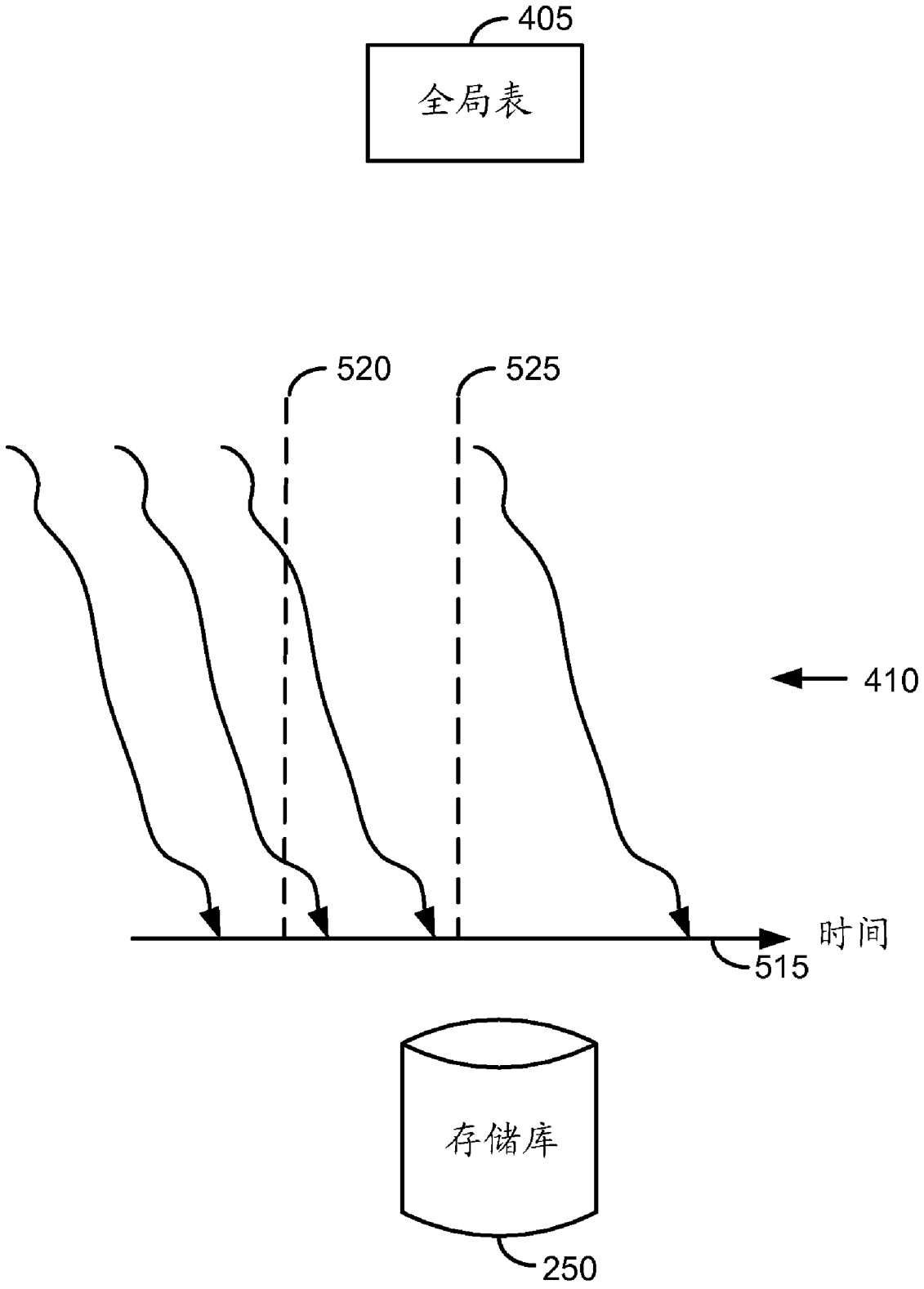


图 4

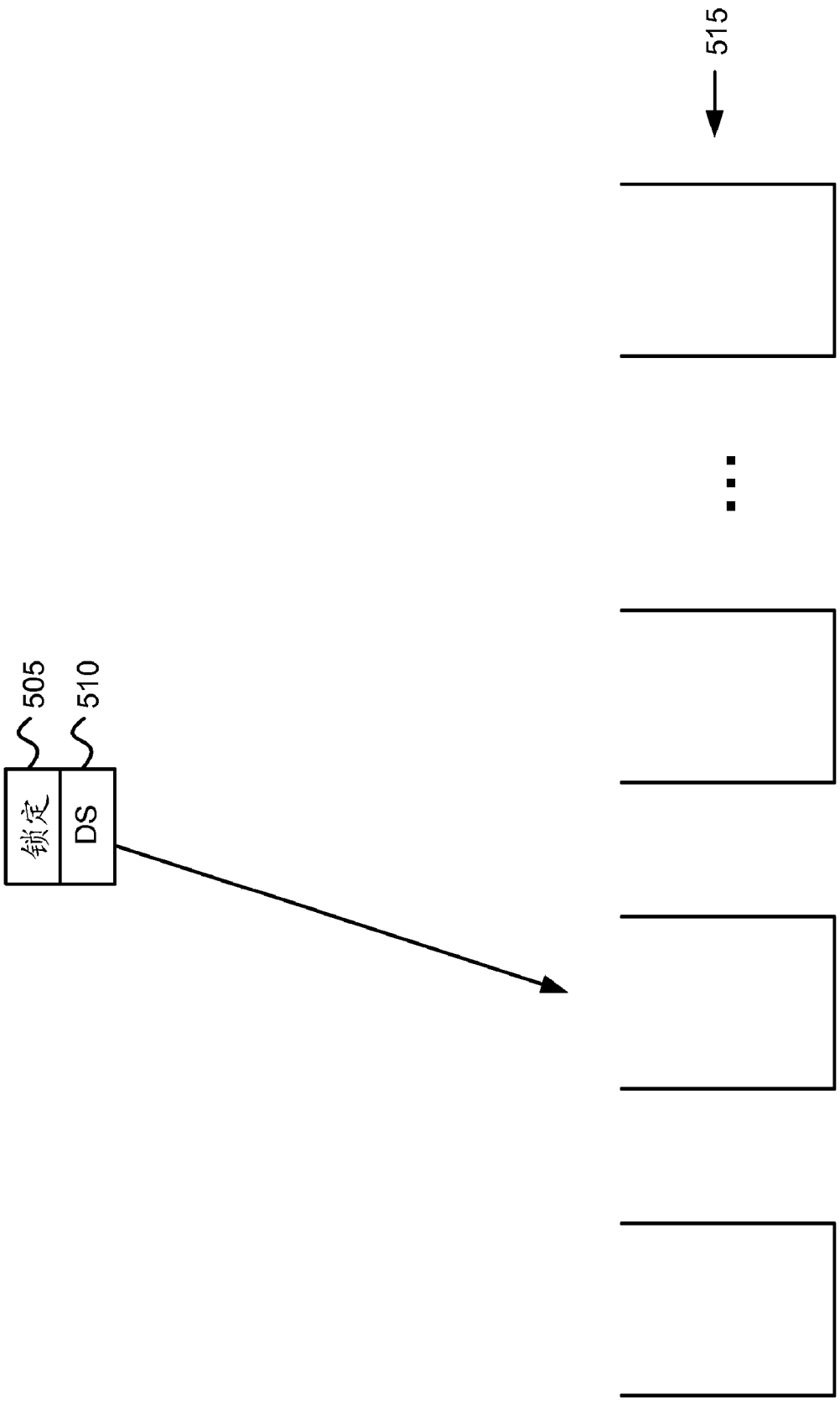


图 5

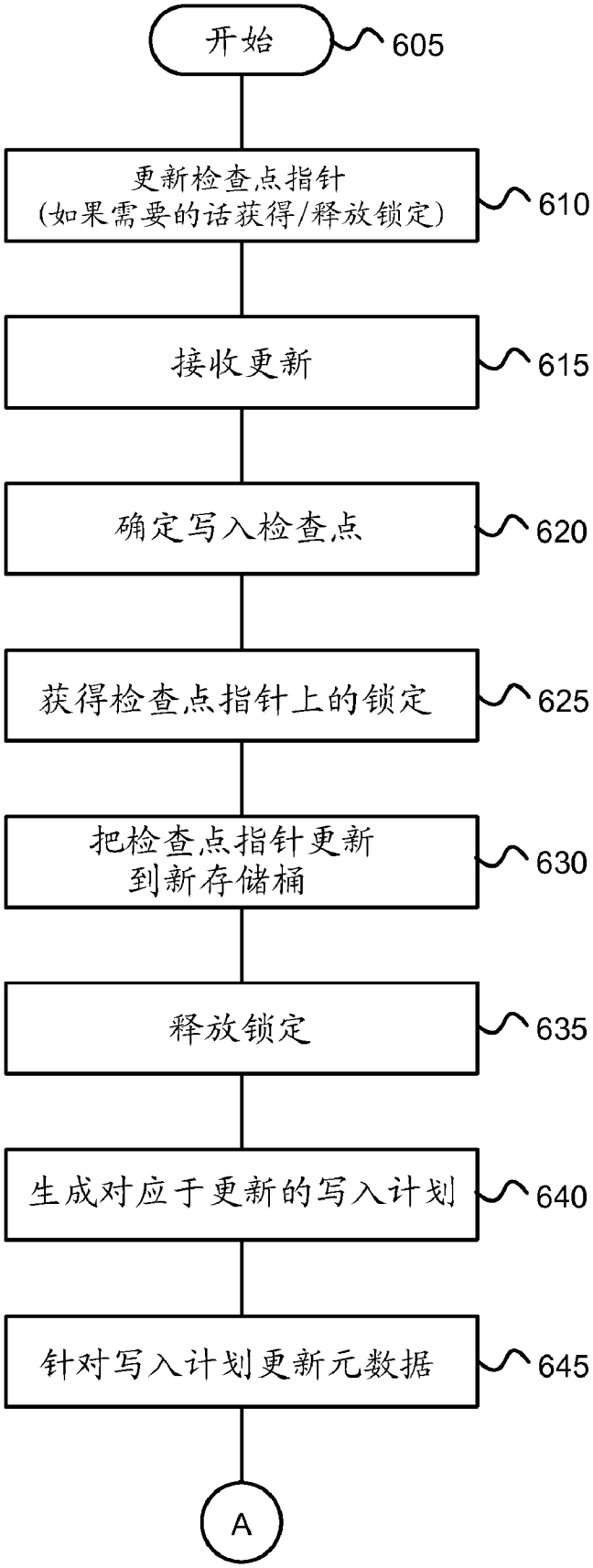

开始
605
更新检查点指针
(如果需要的话获得/释放锁定)
610
接收更新
615
确定写入检查点
620
获得检查点指针上的锁定
625
把检查点指针更新
到新存储桶
630
释放锁定
635
生成对应于更新的写入计划
640
针对写入计划更新元数据
645
A

图 6

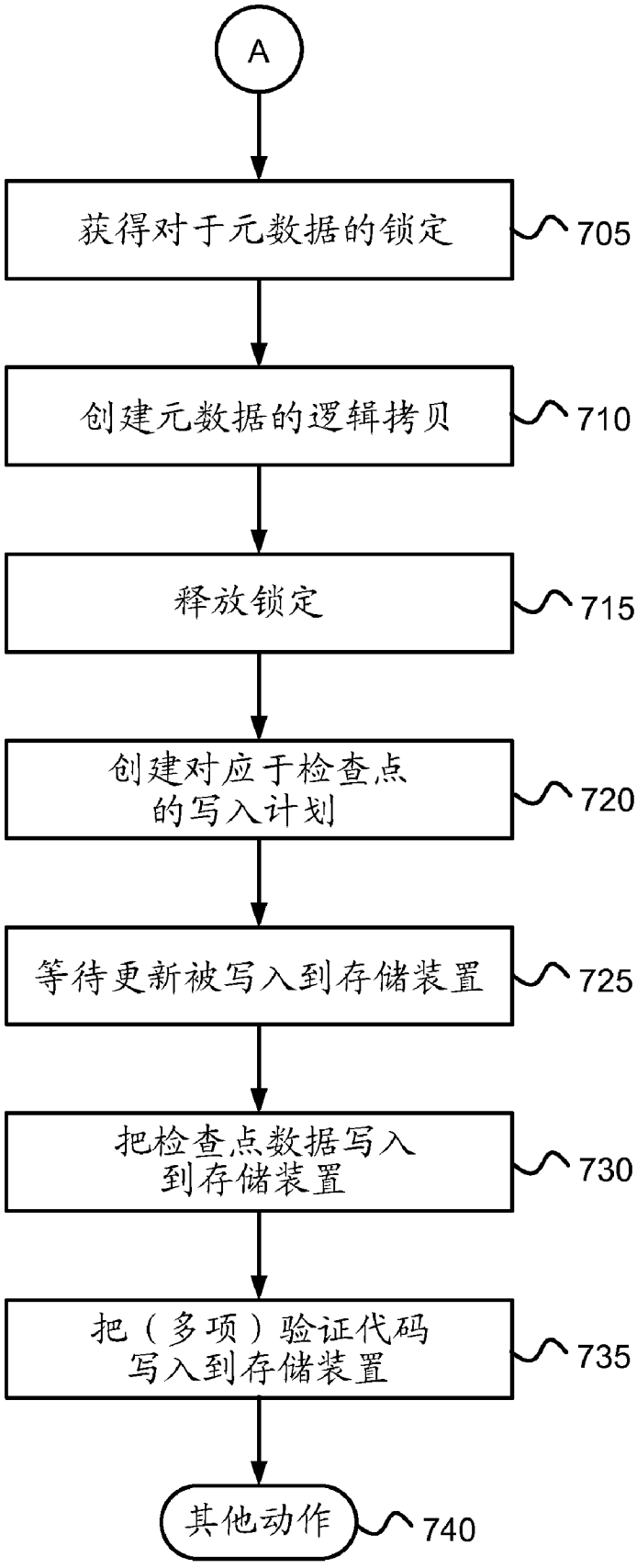

A
获得对于元数据的锁定
705
创建元数据的逻辑拷贝
710
释放锁定
715
创建对应于检查点的写入计划
720
等待更新被写入到存储装置
725
把检查点数据写入到存储装置
730
把（多项）验证代码写入到存储装置
735
其他动作
740

图 7

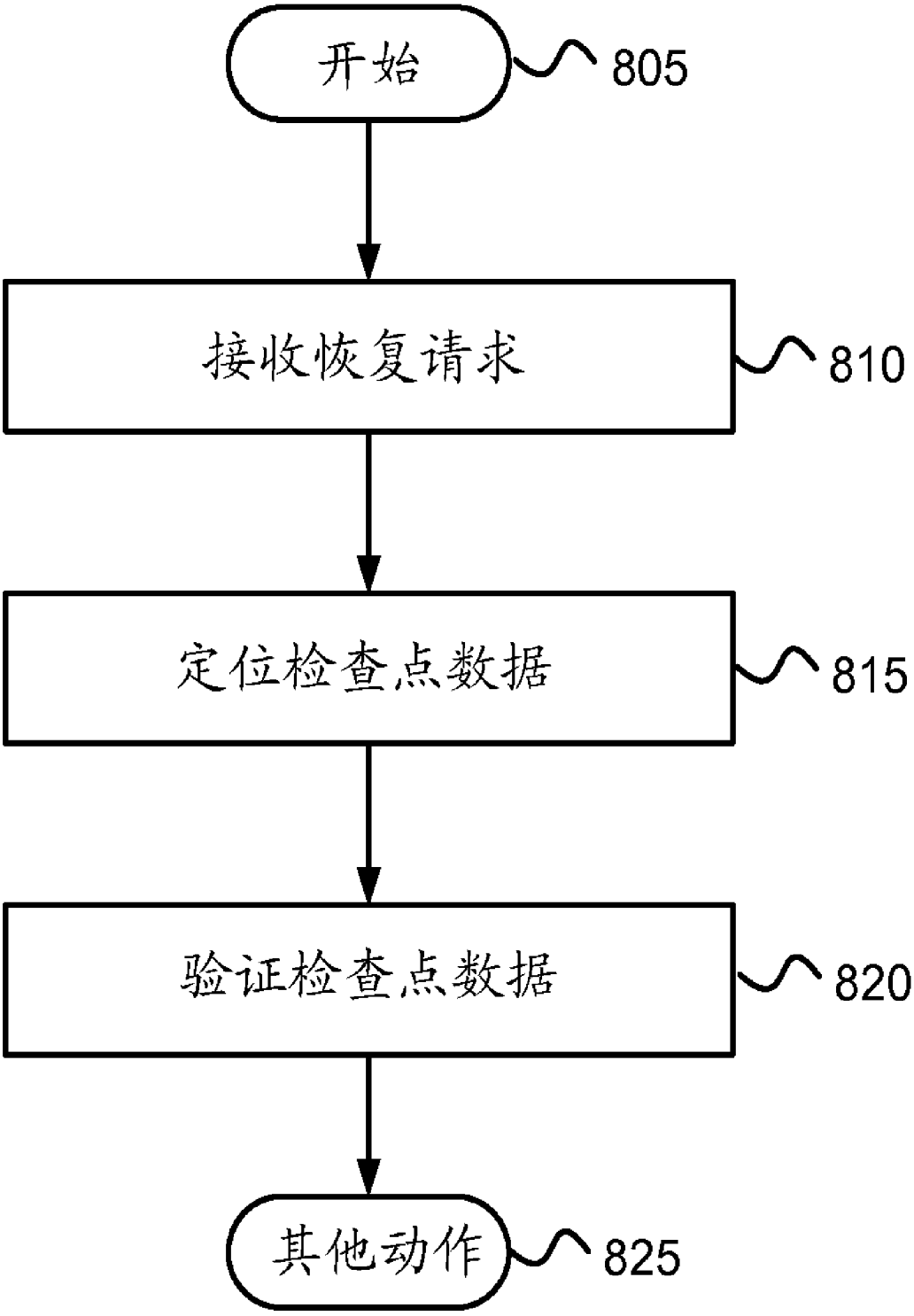


图 8