



(12) 发明专利

(10) 授权公告号 CN 114641977 B

(45) 授权公告日 2024. 04. 09

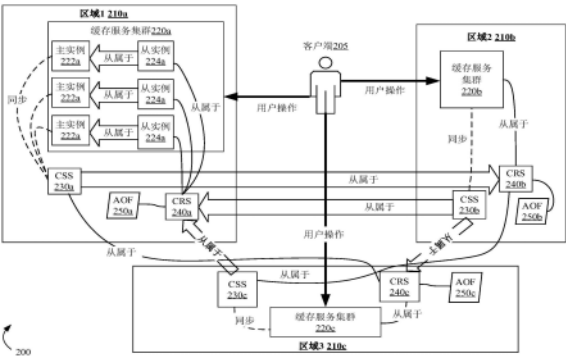
(21) 申请号 202080077210.4	(72) 发明人 邝彬 陈明 卡加鲁班·苏伦德兰 胡振华
(22) 申请日 2020.08.06	
(65) 同一申请的已公布的文献号 申请公布号 CN 114641977 A	(74) 专利代理机构 北京同立钧成知识产权代理有限公司 11205 专利代理师 杨贝贝 臧建明
(43) 申请公布日 2022.06.17	
(30) 优先权数据 16/731,771 2019.12.31 US	(51) Int.Cl. H04L 67/568 (2022.01) H04L 67/1095 (2022.01)
(85) PCT国际申请进入国家阶段日 2022.05.06	(56) 对比文件 CN 109582734 A, 2019.04.05 CN 110247944 A, 2019.09.17 US 2016321294 A1, 2016.11.03 US 2017024451 A1, 2017.01.26 US 9633051 B1, 2017.04.25
(86) PCT国际申请的申请数据 PCT/CN2020/107506 2020.08.06	
(87) PCT国际申请的公布数据 W02021/135227 EN 2021.07.08	
(73) 专利权人 华为云计算技术有限公司 地址 550025 贵州省贵阳市贵安新区黔中大道交兴功路华为云数据中心	审查员 黄睿
	权利要求书3页 说明书18页 附图10页

(54) 发明名称

用于在双活架构中进行跨区域数据管理的系统和方法

(57) 摘要

描述了用于跨区域双活数据复制的系统和方法。提供了一种系统,所述系统包括跨区域复制服务(cross-region replication service, CRS)、跨区域同步服务(cross-region synchronizationservice, CSS)和由缓存服务实例执行的无冲突复制数据类型(conflict-free replication data type, CRDT)模块。



1. 一种用于在双活架构中进行跨区域数据管理的系统,其特征在于,所述系统包括:
 - 属于第一区域的至少一个缓存服务实例;
 - 属于所述第一区域的第一跨区域复制服务CRS;
 - 属于所述第一区域的第一跨区域同步服务CSS;
 - 所述第一CRS用于:
 - 从所述至少一个缓存服务实例接收同区域键值缓存操作;
 - 将所述同区域键值缓存操作复制到属于第二区域的第二CSS;
 - 所述第一CSS用于:
 - 从属于所述第二区域的第二CRS接收跨区域键值缓存操作;
 - 对所述至少一个缓存服务实例重放所述跨区域键值缓存操作;
 - 所述至少一个缓存服务实例用于执行无冲突复制数据类型CRDT模块以:
 - 从所述第一区域的客户端接收包含在所述同区域键值缓存操作中的缓存数据,或者从所述第一CSS接收包含在所述跨区域键值缓存操作中的缓存数据;
 - 通过向非幂等键值缓存操作附加状态元数据,将所述非幂等键值缓存操作转换为幂等键值缓存操作,其中所述状态元数据包括区域标识符ID和/或集群ID来源信息,其用于标识所述缓存数据的来源区域和/或集群;
 - 解决所述缓存数据与属于所述第一区域的缓存数据库中现有条目之间的任何冲突;
 - 将所述缓存数据写入所述缓存数据库;
 - 将所述写入的缓存数据复制到所述第一CRS。
2. 根据权利要求1所述的系统,其特征在于,所述至少一个缓存服务实例还用于执行所述CRDT模块以:
 - 将所述状态元数据与相关缓存数据一起存储在单个数据结构中。
3. 根据权利要求1所述的系统,其特征在于,所述缓存数据与相关冲突解决元数据相关联,并且所述冲突解决元数据包括用于解决与所述缓存数据库中现有条目的任何冲突的信息。
4. 根据权利要求3所述的系统,其特征在于,所述至少一个缓存服务实例还用于执行所述CRDT模块以:
 - 确定所述缓存数据与所述缓存数据库中现有条目之间的冲突;
 - 根据与所述缓存数据相关联的第一冲突解决元数据和与所述现有条目相关联的第二冲突解决元数据之间的比较来解决所述冲突。
5. 根据权利要求3或4所述的系统,其特征在于,所述缓存数据与冲突解决元数据一起存储在单个数据结构中。
6. 根据权利要求1至4中任一项所述的系统,其特征在于,所述第一CRS还用于:
 - 压缩所述同区域键值缓存操作,所述同区域键值缓存操作作为压缩数据复制到所述第二CSS。
7. 根据权利要求1至4中任一项所述的系统,其特征在于,所述同区域键值缓存操作存储在所述第一CRS的队列中,并且所述第一CRS还用于:
 - 当所述队列中的数据至少为阈值大小时,压缩所述队列中的所述数据;
 - 所述同区域键值缓存操作作为压缩数据复制到所述第二CSS。

8. 根据权利要求1至4中任一项所述的系统,其特征在于,所述同区域键值缓存操作存储在所述第一CRS的队列中,并且所述第一CRS还用于:

使用选自多个不同压缩算法的选定压缩算法压缩所述队列中的数据,所述选定压缩算法是根据存储在所述队列中的数据类型选择的;

所述同区域键值缓存操作作为压缩数据复制到所述第二CSS。

9. 根据权利要求8所述的系统,其特征在于,所述第一CRS还用于:

生成提供关于所述选定压缩算法的信息的压缩元数据;

将所述压缩元数据附加到所述压缩数据。

10. 根据权利要求9所述的系统,其特征在于,所述第一CRS还用于:

将所述压缩数据与所述压缩元数据一起写入所述第一区域中维护的缓存操作日志。

11. 一种用于在双活架构中进行跨区域数据管理的方法,其特征在于,所述方法包括:

从第一区域的客户端接收包含在同区域键值缓存操作中的缓存数据,或者从属于第一区域的跨区域同步服务CSS接收包含在跨区域键值缓存操作中的缓存数据;

通过向非幂等键值缓存操作附加状态元数据,将所述非幂等键值缓存操作转换为幂等键值缓存操作,其中所述状态元数据包括区域标识符ID和/或集群ID来源信息,其用于标识所述缓存数据的来源区域和/或集群;

解决所述缓存数据与属于所述第一区域的缓存数据库中现有条目之间的任何冲突;

将所述缓存数据写入所述缓存数据库;

将所述写入的缓存数据复制到属于所述第一区域的跨区域复制服务CRS。

12. 根据权利要求11所述的方法,其特征在于,还包括:

将所述状态元数据与相关缓存数据一起存储在单个数据结构中。

13. 根据权利要求11所述的方法,其特征在于,所述缓存数据与相关冲突解决元数据相关联,并且所述冲突解决元数据包括用于解决与所述缓存数据库中现有条目的任何冲突的信息。

14. 根据权利要求13所述的方法,其特征在于,还包括:

确定所述缓存数据与所述缓存数据库中现有条目之间的冲突;

根据与所述缓存数据相关联的第一冲突解决元数据和与所述现有条目相关联的第二冲突解决元数据之间的比较来解决所述冲突。

15. 根据权利要求13或14所述的方法,其特征在于,所述缓存数据与冲突解决元数据一起存储在单个数据结构中。

16. 一种用于在双活架构中进行跨区域数据管理的方法,其特征在于,所述方法包括:

从属于第一区域的至少一个缓存服务实例接收同区域键值缓存操作;

压缩所述同区域键值缓存操作;

将所述同区域键值缓存操作作为压缩数据复制到属于第二区域的跨区域同步服务CSS。

17. 根据权利要求16所述的方法,其特征在于,所述同区域键值缓存操作存储在队列中,并且所述方法还包括:

当所述队列中的数据至少为阈值大小时,压缩所述队列中的所述数据。

18. 根据权利要求16或17所述的方法,其特征在于,所述同区域键值缓存操作存储在队

列中,并且所述方法还包括:

使用选自多个不同压缩算法的选定压缩算法压缩所述队列中的数据,所述选定压缩算法是根据存储在所述队列中的数据类型选择的。

19. 根据权利要求18所述的方法,其特征在于,还包括:

生成提供关于所述选定压缩算法的信息的压缩元数据;

将所述压缩元数据附加到所述压缩数据;

将所述压缩数据与所述压缩元数据一起写入所述第一区域中维护的缓存操作日志。

用于在双活架构中进行跨区域数据管理的系统和方法

[0001] 相关申请案的交叉申请

[0002] 本申请要求于2019年12月31日递交的申请号为16/731,771、发明名称为“用于在双活架构中进行跨区域数据管理的系统和方法 (SYSTEMS AND METHODS FOR CROSS-REGION DATA MANAGEMENT IN AN ACTIVE-ACTIVE ARCHITECTURE)”的美国专利申请案的优先权,其全部内容通过引用结合在本申请中。

技术领域

[0003] 本发明涉及用于在双活架构中进行数据管理的系统和方法,包括用于在云计算中的键值缓存服务之间进行跨区域双活数据复制的系统和方法。

背景技术

[0004] 云计算(也称为“云”)是一种基于网络的计算(例如,基于互联网的计算)形式,支持访问共享的可配置计算资源池和更高层服务,而这些计算资源和服务通常可以通过互联网使用极少的管理工作量快速调配。云计算是继从基于大型机的计算向基于客户端-服务器的计算转变之后的另一行为方式的转变,并作为服务实现。云计算服务商通常提供三种主要类型的服务(以下简称云计算服务):基础设施即服务(infrastructure as a service,IaaS)、平台即服务(platform as a service,PaaS)和软件即服务(software as a service,SaaS),而这些服务是通过根据客户使用需求创建虚拟机提供的。IaaS提供可供客户租用和使用的计算基础设施。计算基础设施包括虚拟化和在客户之间共享的物理计算资源(例如处理器、内存、存储、服务器、网络组件等)。PaaS提供一种客户在开发、运行和管理软件应用时可以无需构建和维护计算基础设施的平台。SaaS在订阅的基础上通过互联网按需提供在计算基础设施上运行的软件应用。一种类型的PaaS是缓存服务。缓存服务是一种为存储在云计算基础设施上运行的数据应用提供缓存的服务。一种特定类型的缓存服务是键值缓存服务。

[0005] 在云计算中,通常使用键值缓存服务提高缓存服务的性能,这适用于即使在峰值负载期间也要求最小响应时间的各种基于云的应用(即,在“云”的计算基础设施上运行的应用)。用于缓存服务的常见架构是双活架构。在双活架构(也称为双活系统或双活服务)中,存在多个独立区域(每个区域可以包括一个处理节点集群),每个区域提供缓存服务实例,每个缓存服务实例都有一个本地缓存(即本地数据存储)用于存储相同数据。每个区域提供的缓存服务的实例都有读写权限,可以独立执行写命令以修改本地缓存。当在两个不同区域中提供缓存服务的实例并且缓存服务的每个实例同时尝试修改存储在其本地缓存中的相同数据时,可能会发生跨区域冲突。

发明内容

[0006] 在各示例中,本发明描述了用于在双活架构进行跨区域数据管理的系统和方法。

[0007] 在一些示例中,可以使用选自不同可用压缩算法的合适压缩算法动态压缩键值缓

存操作。根据所述键值缓存操作的有效负载类型和/或大小,可以动态选择合适压缩算法,也可以完全跳过压缩。压缩元数据可以记录在操作日志中,使得所述压缩键值缓存操作能够随后读取。

[0008] 在一些示例中,本发明描述了用于将缓存服务的非幂等键值缓存操作转换为幂等操作的技术。例如,可以添加基于状态的冲突解决元数据。然后,可以使用基于状态的无冲突复制数据类型(conflict-free replicated data type,CRDT)来解决任何冲突操作。冲突解决元数据可以与相关缓存数据一起存储,以使数据能够与相关元数据一起迁移。

[0009] 本文公开的各示例可以实现不会显著影响缓存服务的现有实现的冲突解决机制。

[0010] 在一些方面中,本发明描述了一种用于在双活架构中进行跨区域数据管理的系统。所述系统包括:属于第一区域的至少一个缓存服务实例;属于所述第一区域的第一跨区域复制服务(cross-region replication service,CRS);属于所述第一区域的第一跨区域同步服务(cross-region synchronization service,CSS)。所述CRS用于:从所述至少一个缓存服务实例接收同区域键值缓存操作;将所述键值同区域缓存操作复制到属于第二区域的第二CSS。所述CSS用于:从属于所述第二区域的第二CRS接收跨区域键值缓存操作;对所述至少一个缓存服务实例重放所述跨区域键值缓存操作。所述至少一个缓存服务实例用于执行无冲突复制数据类型(conflict-free replicated data type,CRDT)模块以:从所述第一区域的客户端接收包含在所述同区域键值缓存操作中的缓存数据,或者从所述CSS接收包含在所述跨区域键值缓存操作中的缓存数据;解决所述缓存数据与属于所述第一区域的缓存数据库中现有条目之间的任何冲突;将所述缓存数据写入所述缓存数据库;将所述写入的缓存数据复制到所述CRS。

[0011] 在上述任一方面中,所述至少一个缓存服务实例还可以用于执行所述CRDT模块以:通过向非幂等缓存数据附加状态元数据,将所述非幂等缓存数据转换为幂等缓存数据。

[0012] 在上述任一方面中,所述至少一个缓存服务实例还可以用于执行所述CRDT模块以:将状态元数据与相关缓存数据一起存储在单个数据结构中。

[0013] 在上述任一方面中,所述缓存数据可以与相关冲突解决元数据相关联,并且所述冲突解决元数据可以包括用于解决与所述缓存数据库中现有条目的任何冲突的信息。

[0014] 在上述任一方面中,所述至少一个缓存服务实例还可以用于执行所述CRDT模块以:确定所述缓存数据与所述缓存数据库中现有条目之间的冲突;根据与所述缓存数据相关联的第一冲突解决元数据和与所述现有条目相关联的第二冲突解决元数据之间的比较来解决所述冲突。

[0015] 在上述任一方面中,所述缓存数据可以与所述冲突解决数据一起存储在单个数据结构中。

[0016] 在上述任一方面中,所述CRS模块还可以用于:压缩所述同区域键值缓存操作,其中所述同区域键值缓存操作作为压缩数据复制到所述第二CSS。

[0017] 在上述任一方面中,所述同区域键值缓存操作可以存储在所述CRS模块的队列中,并且所述CRS模块还可以用于:当所述队列中的数据至少为阈值大小时,压缩所述队列中的所述数据。所述同区域键值缓存操作可以作为压缩数据复制到所述第二CSS。

[0018] 在上述任一方面中,所述同区域键值缓存操作可以存储在所述CRS模块的队列中,并且所述CRS模块还可以用于:使用选自多个不同压缩算法的选定压缩算法压缩所述队列

中的数据,所述选定压缩算法是根据存储在所述队列中的数据类型选择的。所述同区域键值缓存操作可以作为压缩数据复制到所述第二CSS。

[0019] 在上述任一方面中,所述CRS模块还可以用于:生成提供关于所述选定压缩算法的信息的压缩元数据;将所述压缩元数据附加到所述压缩数据。

[0020] 在上述任一方面中,所述CRS模块还可以用于:将所述压缩数据与所述压缩元数据一起写入所述第一区域中维护的缓存操作日志。

[0021] 在一些方面中,本发明描述了一种用于在双活架构中进行跨区域数据管理的方法。所述方法包括:从所述第一区域的客户端接收包含在同区域键值缓存操作中的缓存数据,或者从属于所述第一区域的跨区域同步服务(cross-region synchronization service,CSS)接收包含在跨区域键值缓存操作中的缓存数据;解决所述缓存数据与属于所述第一区域的缓存数据库中现有条目之间的任何冲突;将所述缓存数据写入所述缓存数据库;将所述写入的缓存数据复制到属于所述第一区域的跨区域复制服务(cross-region replication service,CRS)。

[0022] 在上述任一方面中,所述方法还包括:通过向非幂等键值缓存操作附加状态元数据,将所述非幂等键值缓存操作转换为幂等键值缓存操作。

[0023] 在上述任一方面中,指令还可以使装置执行所述CRDT模块以:将状态元数据与相关缓存数据一起存储在单个数据结构中。

[0024] 在上述任一方面中,所述缓存数据可以与相关冲突解决元数据相关联。所述冲突解决元数据可以包括用于解决与所述缓存数据库中现有条目的任何冲突的信息。

[0025] 在上述任一方面中,所述方法包括:确定所述缓存数据与所述缓存数据库中现有条目之间的冲突;根据与所述缓存数据相关联的第一冲突解决元数据和与所述现有条目相关联的第二冲突解决元数据之间的比较来解决所述冲突。

[0026] 在上述任一方面中,所述缓存数据可以与所述冲突解决数据一起存储在单个数据结构中。

[0027] 在一些方面中,本发明描述了一种用于在双活架构中进行跨区域数据管理的方法。所述方法包括:从属于第一区域的至少一个缓存服务实例接收同区域键值缓存操作;压缩所述同区域键值缓存操作;将所述同区域键值缓存操作作为压缩数据复制到属于第二区域的跨区域同步服务(cross-region synchronization service,CSS)。

[0028] 在上述任一方面中,所述同区域键值缓存操作可以存储在队列中,并且所述方法还包括:当所述队列中的数据至少为阈值大小时,压缩所述队列中的所述数据。

[0029] 在上述任一方面中,所述同区域键值缓存操作可以存储在队列中,并且所述方法还包括:使用选自多个不同压缩算法的选定压缩算法压缩所述队列中的数据,所述选定压缩算法是根据存储在所述队列中的数据类型选择的。

[0030] 在上述任一方面中,所述方法还包括:生成提供关于所述选定压缩算法的信息的压缩元数据;将所述压缩元数据附加到所述压缩数据;将所述压缩数据与所述压缩元数据一起写入所述第一区域中维护的缓存操作日志。

附图说明

[0031] 现在通过示例参考示出本申请的示例性实施例的附图,其中:

[0032] 图1是示出了本文描述的各示例性实施例提供的用于提供云计算服务的云计算架构的框图;

[0033] 图2是示出了本文描述的各示例性实施例提供的实现双活架构的示例性系统的框图;

[0034] 图3是示出了处理单元的简化示例的框图,该处理单元可以用于实现图2的架构中所示的一个或多个实体;

[0035] 图4是示出了可以由图2的架构中的CRS执行的示例性方法的流程图;

[0036] 图5示出了可以如何压缩操作记录并将其写入操作日志的示例;

[0037] 图6是示出了可以由图2的架构中的CSS执行的示例性方法的流程图;

[0038] 图7是示出了可以由图2的架构中的缓存服务服务器执行的示例性方法的流程图;

[0039] 图8是示出了可以由图2的架构中的缓存服务服务器执行的另一个示例性方法的流程图;

[0040] 图9是示出了由图2的架构中的缓存服务服务器执行的示例性进程的简化框图;

[0041] 图10示出了本文描述的各示例性实施例提供的数据结构的数据结构定义,该数据结构将缓存数据与其相关元数据一起放置。

[0042] 不同附图中可以使用相同的附图标记来表示相同的组件。

具体实施方式

[0043] 本发明描述了使用双活架构的云计算环境中的示例。尽管示出并讨论了某些系统图和流程图,但这些仅作为示例提供,并非为了进行限制。

[0044] 图1是示出了可以提供云计算服务的云计算架构的示意性逻辑框图。所示的云计算架构100 (以下简称云100) 的逻辑图通常包括基础设施平台102 (例如,基础设施即服务 (infrastructure as a service, IaaS) 层)、应用平台104 (例如,平台即服务 (platform as a service, PaaS) 层) 和应用106 (例如,软件即服务 (software as a service, SaaS) 层)。基础设施平台102包括物理硬件资源108和虚拟化层110,虚拟化层110向应用平台104提供物理硬件资源108的抽象。虚拟化层110提供的抽象取决于托管在应用平台104上的应用112的要求。物理硬件资源108包括:物理机114,物理机114包括处理资源 (例如,中央处理器 (central processing units, CPU)、图形处理器 (graphic processing unit, GPU)、加速器、张量处理器 (tensor processing unit, TPU));物理存储116,物理存储116包括存储资源,例如存储器 (例如,静态随机存取存储器 (static random access memory, SRAM)、动态随机存取存储器 (dynamic random access memory, DRAM)、同步DRAM (synchronous DRAM, SDRAM)、只读存储器 (read-only memory, ROM))、持久性存储设备 (例如,硬盘驱动器、光盘驱动器或它们的组合) 和网络资源 (未示出),所述网络资源通常位于数据中心内。本领域的人员应当理解,数据中心包括物理硬件资源108的集合 (通常是服务器的形式),其可以用作包括处理、存储和网络资源的集体计算资源。在数据中心内,可以将多个服务器连接在一起以提供计算资源池,在该计算资源池上可以实例化虚拟化实体。数据中心可以相互连接,形成计算资源池,这些计算资源池通过连接资源相互连接。所述连接资源可以采取物理连接的形式,例如以太网或光通信链路。

[0045] 虚拟化层110支持通过提供IaaS设施为应用112提供灵活且高效的多租户运行时

和托管环境。虚拟化层110包括可以为应用平台104托管的每个应用112提供安全和资源“沙盒”的虚拟化管理器或虚拟机监视器(未示出)。每个“沙盒”可以实现为虚拟机(Virtual Machine, VM) 118, VM 118可以包括适当的操作系统和对虚拟化存储资源120的受控访问。

[0046] 虚拟化层110对物理硬件资源108的虚拟化被认为是云100的基础技术。虚拟化是一种可以创建计算资源(例如,处理、存储和网络资源)的虚拟计算资源池的技术,其中所述计算资源通过连接资源相互连接。虚拟化可以采取实例化VM 118的形式,VM 118对于网络上的其它实体以及在VM 118上执行的软件而言与物理计算设备并无不同。VM 118具有其自己的一组计算资源(例如,处理、存储和连接资源),在该组计算资源上可以执行操作系统。VM 118可以具有可被分配网络地址的虚拟网络接口。在底层资源和VM 118之间,通常存在管理资源隔离和网络交互的虚拟机监视器(未示出)。VM 118的一个目的是提供与在云100上运行的其它进程的隔离。最初开发时,VM 118是一种实现在不同进程运行时可以不必担心单个错误进程会导致系统彻底崩溃的机制。相反,错误进程将包含在其自己的VM 118中。这种隔离可以实现每个VM 118都具有其自己的一组网络接口。通常,单个底层计算资源可以支持多个虚拟化实体。

[0047] 本领域的技术人员应当理解,最新的发展是使用容器代替VM 118。如上所述,每个VM 118通常包括其自己的操作系统,这通常会增加冗余的计算、存储和连接资源使用量。容器可以实现单个操作系统(operating system, OS)内核支持多个隔离的应用。代替可以实现每个VM 118都运行其自己OS的虚拟机监视器的是单个OS,所述单个OS托管负责执行本应由VM 118提供的资源隔离的容器。

[0048] 应用平台104提供用于托管应用112的功能并且包括应用平台服务122。应用平台服务122向托管在应用平台104上的应用112提供一组中间件应用服务和基础设施服务。托管在应用平台104上的应用112可以在所述VM或所述物理机上运行。在图1所示的示例中,应用平台服务122包括缓存服务124、数据库服务126、消息服务128和应用程序接口(application program interface, API)网关服务130,其中,缓存服务124用于内存数据存储;数据库服务126用于应用;消息服务128用于向用户客户发布消息;应用程序接口网关服务130使客户能够创建、发布和维护API以访问其它云服务。本领域的技术人员应当理解,应用平台服务112可以向客户提供其它中间件应用服务,例如通知服务、运行时服务等。来自客户的应用112可以在相应VM 118或物理机114中部署和执行。

[0049] 在一些示例中,缓存服务124可以使用跨区域双活键值缓存服务来实现。设计良好的跨区域缓存服务124应在区域之间提供快速可靠的双向数据复制。缓存服务124应满足(或接近满足)多个性能方面,以便具有实际的商业用途。应避免区域之间的循环数据复制。当源自一个区域的缓存数据在另一个区域中复制并执行后在原始区域中被复制回来并再次执行(导致操作重复)时,就会发生循环数据复制。数据复制(即,缓存数据的复制)应足够快,以使缓存服务能够跟上所有区域的并发(或近并发)数据更改,即使跨区域网络可能出现缓慢和/或不稳定的情况。缓存服务124应支持在具有多个缓存服务实例的一个缓存服务集群与具有多个缓存服务实例的另一个缓存服务集群之间进行数据复制。所有区域的本地缓存中存储的数据(以下简称缓存数据)在数据复制后最终应是一致的。缓存服务124应为数据更改提供冲突识别和冲突解决,并且该冲突识别和解决应是一致的并且相对较快(例如,在1ms内或更快时间内完成)。用于解决数据更改冲突的任何元数据,即提供有关数据的

信息(例如数据的创建日期、数据类型和数据来源)的任何数据,不应影响在缓存服务实例之间移动数据的数据移动性。从缓存服务用户和云计算服务提供商的角度来看,用于解决上述任何问题的任何方案都不应明显影响缓存服务124的性能、应用和实现。

[0050] 为了解决缓存服务(包括键值缓存服务)的一个或多个性能方面问题,已经进行了一些尝试。例如,当某个区域中的键值缓存服务实例执行键值缓存操作以插入、删除或更新存储在其本地缓存中的数据时,所述键值缓存操作可以在被广播到其它区域中的缓存服务实例之前被压缩,这有助于减少跨区域传输的数据量,并缩短复制时间(即,将数据复制到其它区域中的缓存服务实例的本地缓存的时间)。然而,目前的方法不能动态压缩键值缓存操作,因为压缩键值缓存操作和相关压缩属性(例如,选定压缩算法、原始未压缩大小、压缩大小等)必须写入操作日志(例如,Redis环境下的只追加文件(append only file,AOF))和/或从操作日志中读取。此外,目前的方法不能根据键值缓存操作的不同有效负载类型和大小,从不同的可能压缩算法中选择合适的压缩算法来压缩键值缓存操作,并且不允许完全跳过键值缓存操作的压缩。这可能会导致低效和/或不必要的压缩,因而可能会引入额外的操作延迟和复制时间,从而对键值缓存服务的性能产生不利影响。

[0051] 目前用于解决跨区域键值缓存操作冲突的方案也存在影响键值缓存服务性能的缺点。现有冲突解决方法的一个示例是基于区域之间的一致/仲裁(例如,多数获胜),其吞吐量相对较低,从而影响缓存服务的速度。另一种现有方案是“最后写入者获胜”方法,但这种方法可能会导致应用不准确/不一致。其它现有方案对于解决非幂等操作(例如,基于无冲突复制数据类型的操作)有局限性。

[0052] 目前的方法不支持从多个主实例复制键值缓存操作的从实例,因为从实例中没有用于解决冲突的内置功能。目前的一些方法直接将冲突解决功能添加到从实例,但这需要对缓存服务代码进行重大更改,这使得代码与其开源社区分离和/或导致难以支持各种用户场景。

[0053] 用于解决键值缓存操作之间的冲突(也称为键值缓存操作冲突)的一些现有方法依赖于使用元数据来解决冲突(以下称为冲突解决元数据)。然而,这类方法中的冲突解决元数据会影响在缓存服务实例之间移动缓存数据的数据移动性。例如,目前的一些方法将冲突解决元数据与缓存数据分开存储在本地缓存(例如缓存数据库)中,因此不能保证在重新分片缓存集群时元数据和缓存数据会一起移动。通过重新分片集群,云服务提供商能够增加或减少集群中的分片或节点的数量以适应变化。目前的一些方法通过改变缓存数据的数据结构来将冲突解决元数据与缓存数据一起存储,但结果导致缓存数据不能移动到不支持新数据结构的其它缓存服务实例中。本发明描述了解决上述至少一些缺点的示例。

[0054] 下文提供了可用于以下讨论的一些术语的定义。在适当的情况下,这些术语可以在详细讨论中进一步定义和/或扩展。

[0055] **Redis**:一种内存数据存储,可用于实现缓存或数据库。Redis是一种开源软件,一直用于开发缓存服务。在Redis环境中,Redis核心是指处理缓存服务请求(例如,来自客户端)的核心代码的实现,而Redis模块是一个插件,它扩展Redis功能并实现新的Redis命令,且不影响Redis核心。尽管在本发明中将Redis称为内存数据存储的示例形式,但应当理解这仅是示例性的并且可以使用其它类型的内存存储。

[0056] **区域**:具有数据中心的逻辑和/或地球物理区域。每个数据中心提供一个缓存服务

的实例(在本文中也称为缓存服务实例)。

[0057] 缓存服务集群:缓存服务实例的集群。

[0058] 主实例:在主从对中充当主角色的缓存服务实例。主实例处理客户端请求。

[0059] 缓存数据库:缓存服务的数据库实现。缓存数据库可以是存储缓存数据的数据存储(即,本地缓存)。缓存数据库可以是将缓存数据存储为键值对的Redis数据库。

[0060] 从实例:在主从对中充当从角色的缓存服务实例。从实例从其配对的主实例复制数据。如果成对的主实例离线、崩溃或无法执行其功能,则从实例会被提升为主实例的角色。

[0061] 跨区域复制服务(cross-region replication service, CRS):CRS从同区域的从实例复制缓存数据。CRS可以被认为是从实例中的从实例。

[0062] 跨区域同步服务(cross-region synchronization service, CSS):CSS从所有其它区域的CRS复制缓存数据,并对自己区域的主实例重放缓存数据。在本发明中,“重放”缓存数据可以涉及将缓存数据从一个实体(例如,缓存客户端角色的CSS)应用和/或写入另一实体的数据库(例如,缓存服务角色的主实例)。

[0063] 通常,主实例、从实例、CRS和CSS的实例可以是在一个或多个物理机、虚拟机118或由IaaS提供的容器上运行的进程。通常,当任何进程可以在物理机(例如,物理服务器)上运行时,可以认为主实例、从实例、CRS或CSS的实例具有物理性。为简单起见,本发明将提及主实例、从实例、CRS和CSS,而不是提及主实例、从实例、CRS和CSS的实例。

[0064] 只追加文件(append only file, AOF):AOF是一个日志文件,用于记录改变缓存服务的本地缓存数据库状态的键值缓存操作。读取命令不会记录在日志文件中,因为它不会改变数据库的状态。日志文件的内容可以用于恢复缓存数据(例如,可以加载到重新启动的缓存服务以在缓存服务崩溃的情况下恢复旧的缓存数据)。

[0065] 客户端:客户端向缓存服务发送缓存请求。

[0066] 无冲突复制数据类型(conflict-free replication data type, CRDT):一种数据结构类型,当跨多个缓存服务实例复制缓存数据和元数据时,它支持解决键值缓存操作之间的冲突,包括写入命令、删除命令、更新命令。可以使用添加到现有数据存储(例如,Redis核心)的功能的CRDT模块(也可以称为CRDT代理)来实现CRDT数据结构。可以在缓存服务实例启动时加载和初始化CRDT模块。可以使用(例如)新实现的Redis命令CRDT.SET来实现CRDT相关功能;可以使用(例如)新实现的Redis命令NATIVE.SET来实现Redis核心的原始SET命令。

[0067] 下文结合附图进行的说明将有助于理解进一步的细节。

[0068] 图2是示出了本文公开的各示例提供的实现双活架构的示例性系统200的框图。系统200可以用于实现图1所示的缓存服务124。系统200可以实现多个区域中的键值缓存服务实例之间的跨区域双活缓存数据复制。

[0069] 系统200包括跨不同区域分布的缓存服务集群。缓存服务集群,连同缓存服务集群本地的任何支持服务器或模块,可以被称为区域,每个区域实例化缓存服务124的相应实例(例如,键值缓存服务的相应实例)。在所示的示例中,系统200包括三个区域,即区域1 210a、区域2 210b和区域3 210c(通常称为区域210)。尽管示出了三个区域210,但是系统200可以具有更多或更少数量的区域210。例如,区域210可以位于不同的地理位置,并且可

以是全球分布的。尽管本发明描述了在每个区域210中使用集群的示例,但是在一些示例中,可以使用缓存服务的单个实例(例如,键值缓存服务的单个实例)来代替缓存服务集群。

[0070] 每个区域210包括相应缓存服务集群220a、220b、220c(通常称为缓存服务集群220)、相应跨区域同步服务(cross-region synchronization service,CSS) 230a、230b、230c(通常称为CSS 230)以及相应跨区域复制服务(cross-region replication service,CRS) 240a、240b、240c(通常称为CRS 240)。每个CRS 240创建并维护各自的仅追加文件(append only file,AOF) 250a、250b、250c(通常称为AOF 250,统称为AOF),其记录键值缓存操作。客户端205(或应用用户)可以向任何区域210发送用户操作,例如用于插入、删除和更新缓存数据的写操作,以及用于读取缓存数据的读操作。发送到给定区域210的用户操作由相应区域210中的缓存服务集群220接收。

[0071] 为了便于理解,仅详细示出了区域1 210a,简化了区域2 210b和3 210c。现在描述区域1 210a的缓存服务集群220a。应当理解,系统200中的每个区域210都可以类似于如下所述的区域1 210a来运行。缓存服务集群220a包括多个主从对(三个这样的主从对仅作为示例示出),每个主从对具有主实例222a和从实例224a(通常称为主实例222和从实例224)。

[0072] 图3是示出了处理单元300的简化示例的框图,该处理单元可以用于运行图2所示的一个或多个进程,例如实例化主实例222、从实例224、CSS 230和/或CRS 240的进程。可以使用适用于实现本发明中描述的实施例的其它示例,其可以包括与下文所述不同的组件。虽然图3示出了每个组件的单个实例,但是处理单元300中可以具有每个组件的多个实例。

[0073] 处理单元300可以包括一个或多个处理设备302,例如处理器、微处理器、数字信号处理器、专用集成电路(application-specific integrated circuit,ASIC)、现场可编程门阵列(programmable gate array,FPGA)、专用逻辑电路、专用人工智能处理器单元或其组合。处理单元300还可以包括一个或多个可选输入/输出(I/O)接口304,其可以实现与一个或多个可选输入设备306和/或可选输出设备308的连接。

[0074] 在所示的示例中,输入设备306(例如,键盘、鼠标、麦克风、触摸屏和/或小键盘)和输出设备308(例如,显示器、扬声器和/或打印机)被示为可选的并且在处理单元300外部。在其它示例中,可以没有任何输入设备306和输出设备308,在这种情况下可以不需要I/O接口304。

[0075] 处理单元300可以包括一个或多个网络接口310,用于与区域210中的实体进行有线或无线通信(也称为同区域通信)或与系统200中的另一个区域210中的实体进行有线或无线通信(也称为跨区域通信)。网络接口310可以包括用于同区域和/或跨区域通信的有线链路(例如,以太网线)和/或无线链路(例如,一根或多根天线)。

[0076] 处理单元300还可以包括一个或多个存储单元312,一个或多个存储单元312可以包括大容量存储单元,例如固态驱动器、硬盘驱动器、磁盘驱动器和/或光盘驱动器。

[0077] 处理单元300可以包括一个或多个存储器314,一个或多个存储器314可以包括易失性或非易失性存储器(例如,闪存、随机存取存储器(random access memory,RAM)和/或只读存储器(read-only memory,ROM))。非瞬时性存储器314可以存储由处理设备302执行的指令,例如,以执行本发明中所描述的示例。例如,在处理单元300是缓存服务服务器(例如,主实例222或从实例224)的情况下,存储器314可以存储用于执行无冲突复制数据类型(conflict-free replicated data type,CRDT)模块的功能的指令,如下文进一步讨论的。

[0078] 存储器314可以包括其它软件指令,例如用于实现操作系统和其它应用/功能。在一些示例中,处理单元300可以附加地或替代地执行来自外部存储器(例如,与处理单元300进行有线或无线通信的外部驱动器)的指令,或者可以接受由瞬时性或非瞬时性计算机可读介质提供的可执行指令。非瞬时性计算机可读介质的示例包括RAM、ROM、可擦除可编程ROM(erasable programmable ROM,EPR0M)、电可擦除可编程ROM(electrically erasable programmable ROM,EEPROM)、闪存、CD-ROM或其它便携式存储器。

[0079] 处理单元300还可以包括总线316,总线316提供处理单元300的组件(包括上述组件)之间的通信。总线316可以是任何合适的总线架构,例如,包括存储器总线、外设总线或视频总线。

[0080] 再次参考图2。缓存服务集群220是提供缓存服务124(见图1)的缓存服务实例(例如,主实例222和从实例224)的集群,缓存服务124是应用平台服务122之一。还提供缓存数据库的(例如,分布式)缓存服务实例(例如,主实例222和从实例224)。CSS 230和CRS 240还用于提供缓存服务124,如下文进一步讨论的。无冲突复制数据类型(conflict-free replicated data type,CRDT)模块由缓存服务集群220的每个缓存服务实例实现。主实例222、从实例224、CRS 240和CSS 230的实例可以是在一个或多个物理机或虚拟机118上或在容器中运行的进程。为简单起见,本发明将提及主实例222、从实例224、CRS 240和CSS 230,而不是提及主实例222、从实例224、CRS 240和CSS 230的实例。

[0081] Redis是一种开源软件,可以实现键值缓存。本文公开的CRDT模块也可以实现为Redis模块。

[0082] 图2示出了系统200中的实体之间的一些示例性关系,现在将对其进行讨论。在本发明中,同区域210中的实体之间的关系可以称为同区域键值操作,不同区域210中的实体之间的关系可以称为跨区域键值操作。

[0083] 每个区域210中的CRS 240用于复制相应区域210中的主从对的键值缓存操作。因此,CRS 240充当相应区域210的缓存服务集群220中的从实例224的同区域从实例(例如,如图2中的箭头所示)。这种配置利用了键值缓存服务本地的现有主从复制机制,这可以更容易地实现,而对其它缓存服务代码的影响很小或没有影响。此外,通过使用本地缓存服务代码提供的现有机制,本发明的各示例可以实现为现有核心代码之上的补丁或插件模块,并且可以通过对缓存服务的客户端透明的方式实现(即,客户端的过程保持不变,并且客户端不知道缓存服务的任何变化)。应当理解,缓存服务或应用的用户通过客户端访问本地缓存(例如,缓存数据库)。在将键值缓存操作跨区域复制到其它缓存服务实例之前,CRS 240还可以对复制的键值缓存操作执行压缩。所述压缩可以动态执行,可以使用选自多种不同可能算法的压缩算法,并且可以完全省略,如下文进一步讨论的。

[0084] 一个区域210的CRS 240将键值缓存操作复制到每个其它区域210的相应CSS 230。因此,每个区域210中的CSS 230充当每个其它CRS区域210的CRS 240的跨区域从实例。在图2的示例中,箭头指示每个CSS 230与每个跨区域CRS 240的跨区域从属关系。例如,区域1 210a的CSS 230a充当区域2 210b和3 210c的CRS 240b、240c的从实例。

[0085] 给定区域210的CSS 230用于将收到的键值缓存操作同步到相应区域210的缓存服务集群220中的同区域主实例222。主实例222将缓存键值操作复制到相应从实例224。

[0086] 缓存服务集群220的每个缓存服务实例(例如,主实例222和从实例224)实现CRDT

模块。在一些示例中,CRDT模块提供用于将非幂等键值缓存操作转换为幂等键值操作的功能。在本发明中,幂等键值操作是指无论执行一次还是多次,都实现相同结果状态的键值操作。换言之,幂等键值操作是在不小心重复键值缓存操作时不会导致本地缓存(例如,缓存数据库)状态错误的键值操作。例如,键值缓存操作“set count=2”是幂等操作,因为无论该操作被发送和执行多少次,结果都是将计数设置为值2。但是,键值缓存操作“increment count by 1”是非幂等操作,因为键值操作的每次重复都会改变计数值。在一些示例中,幂等键值操作也可以称为基于状态的键值操作或状态定义的键值操作。对于双活数据复制系统,幂等键值操作可能优于非幂等键值操作,可避免由于意外重复的复制键值操作而导致错误数据。

[0087] 在一些示例中,CRDT模块还提供使用基于状态的CRDT来解决键值缓存操作冲突(即,键值缓存操作之间的冲突)的功能。在一些示例中,CRDT模块还提供用于将冲突解决元数据与相关缓存数据一同放置的功能。如本文所讨论的,冲突解决可以避免侵入现有的数据结构和代码。

[0088] CRS 240、CSS 230和CRDT模块(在缓存服务中实现)一起运行,以在键值缓存服务中提供跨区域双活数据复制。本发明的一些方面还单独描述了这些实体中的每一个的操作。因此,本发明包括系统200作为整体的操作,以及系统200内的各个实体的操作。

[0089] 图4是示出了可以由CRS 240作为其功能的一部分来执行的示例性方法400的流程图。CRS 240可以执行方法400以实现缓存数据的跨区域复制。

[0090] 在402处,给定区域210中的CRS 240从同区域缓存服务集群230接收键值缓存操作。例如,可以从缓存服务集群230中的从实例224的CRDT模块转发键值缓存操作。

[0091] 在404处,将收到的键值缓存操作放入CRS 240的队列。可以在将键值缓存操作放入队列之前或之时生成序列号并将其分配给该键值缓存操作。CRS 240可以继续将收到的键值缓存操作推送到队列,直到存在将排队的键值缓存操作发送到跨区域CSS 230的触发。例如,触发可以是队列的阈值大小(例如,只要有至少10个排队的键值缓存操作,或者当排队的键值缓存操作总计至少1Mb时,就会发送队列键值缓存操作)、计时器到期(例如,排队的键值缓存操作每1秒发送一次)或任何其它合适的触发。通过触发可以在比现有方法更小的块中传输(具有可选压缩)排队的键值缓存操作。目前的方法通常压缩和传输整个队列,这可以导致操作变慢,并且单个错误将导致重新发送整个队列的可能性更高。

[0092] 可选地,在405处,压缩排队的键值缓存操作,例如使用静态压缩算法。

[0093] 替代地和可选地,代替步骤405,在406处,可以为排队的键值缓存操作执行确定的压缩操作。步骤406可以包括:(可选地)确定是否应执行压缩(在步骤408中);(可选地)从不同的可用压缩算法中选择压缩算法并使用选定的压缩算法执行压缩(在步骤410中);和/或(可选地)确定在执行压缩之后压缩是否满足预定义阈值(在412中)。

[0094] 在可选的步骤408处,可以评估队列的大小(例如,排队的键值缓存操作的总字节大小)并将其与预定义阈值大小(例如,1kB、5kB或10kB等)进行比较。如果队列大小低于预定义阈值大小,则可以完全跳过压缩,所述方法可以直接转到步骤414。当队列的大小较小时可能适合跳过压缩,以减少处理器时间(在压缩和随后的解压缩过程中),因为队列的小尺寸意味着压缩对于改进跨区域传输可能具有很小的益处或没有益处。

[0095] 如果在408中确定应执行压缩,或者如果不执行步骤408,则可以执行可选步骤

410。在可选的步骤408处,使用选定的压缩算法来执行队列的压缩。所述选定的压缩算法可以从多个不同的可用压缩算法中选择。所述选定的压缩算法可以通过在多个不同的可用压缩算法中识别出预期会实现良好或最佳压缩比/压缩效率的压缩算法来选择。例如,可以识别队列中出现最多的键值操作类型,并可以选择最适合该类型(例如,映射到该类型,比如基于查找表、预定义规则或其它参考)的压缩算法。然后可以使用选定的压缩算法来压缩队列中的键值缓存操作。

[0096] 可选地,在412处,在队列已被压缩之后,可以确定压缩是否满足预定义性能阈值。例如,可以评估压缩之前和之后排队的键值缓存操作的大小(例如,通过将压缩数据计算为未压缩数据的百分比),并与预定义性能阈值进行比较(例如,压缩数据不应超过未压缩数据的50%)。如果不满足性能阈值,则可以确定不应执行压缩。这可能是合适的,因为压缩的好处不会超过后续解压缩的开销。如果确定不应执行压缩,则在后续步骤中使用未压缩操作。

[0097] 如果确定应执行压缩,则在后续步骤中使用压缩键值缓存操作。还可以生成压缩元数据以提供关于压缩属性的信息(例如,选定的压缩算法、原始未压缩大小、压缩大小等),这些信息可以用于以后的解压缩。通过将压缩元数据附加到压缩缓存数据,可以将压缩元数据与压缩缓存数据一起打包。打包后的压缩元数据和压缩后的缓存数据具有与键值缓存服务的AOF中的记录(例如,用于Redis的RESP)相同的格式。这种压缩元数据可以是唯一的,因为它提供了对本文公开的动态压缩唯一的信息。依赖于静态压缩的当前技术可能不使用这种压缩元数据,因为压缩操作是固定的并且是已知的。

[0098] 图5示出了可以如何压缩AOF记录并将其写入日志文件(即AOF)的示例,如本文所公开的。通常,当前技术必须压缩/解压缩整个AOF日志,但无法在运行时动态压缩/解压缩AOF日志中的单个AOF记录。相比之下,使用本发明的各示例,可以对单个AOF记录进行这种动态压缩/解压缩。

[0099] 图5示出了可以如何在AOF文件中压缩记录的示例性高级视图。例如,常规未压缩AOF记录505的大小可以超过1kB。这些记录505可以被压缩(例如,使用compress()函数)成单个压缩记录510。在该示例中,压缩记录510具有COMPRESZ格式。然后将压缩记录510写入AOF日志515(例如,使用write()函数),其现在包含多个压缩记录(在该示例中,全部为COMPRESZ格式)。将一串原始/常规记录(左框)压缩为COMPRESZ记录(中间框);一串COMPRESZ记录生成新的/压缩的AOF文件(右框)。

[0100] 图5还示出了可以如何通过压缩来改变记录的示例性详细视图。示出了依照Redis定义的协议的记录的示例性格式520。还示出了遵循Redis协议定义的格式520的示例性记录525。还示出了遵循COMPRESZ格式的示例性压缩记录530。应当注意,压缩记录530与压缩相关元数据(例如描述原始长度和压缩长度的数据)相关联。

[0101] 应当注意,可以将关于压缩属性的信息(例如,压缩元数据)添加到AOF记录中。这可以使数据能够从AOF记录中正确恢复。应当注意,如果需要(例如,如果AOF记录将由不识别压缩元数据的进程使用),也可以从AOF记录中删除压缩元数据。

[0102] 再次参考图4。在414处,队列中的键值缓存操作(可能已被压缩)由CRS 240发送到一个或多个跨区域CSS 230。通常,一个给定区域210中的CRS 240向系统200的所有其它区域210中的所有CSS 230发送排队的键值缓存操作,以确保系统200上的缓存数据复制。

[0103] 在416处,CRS 240生成要添加到其日志文件(例如,AOF 250)的日志记录以及关于发送的操作的信息。记录可以包括在414中发送的键值缓存操作的列表,每个键值缓存操作是一条单独的记录。在一些示例中,可以在将记录添加到(例如,写入)缓存操作日志之前,将记录压缩成单个压缩记录。

[0104] 在一些示例中,可以并行执行步骤416和步骤414(例如,当将记录的键值缓存操作发送到跨区域CSS 230时,生成日志记录并将其添加到(例如,写入)缓存操作日志中)。在一些示例中,可以在步骤414之前执行步骤416(例如,在将记录的键值缓存操作发送到跨区域CSS 230之前,生成日志记录并将其添加到(例如写入)缓存操作日志中)。

[0105] 图6是示出了可以由CSS 230作为其功能的一部分来执行的示例性方法600的流程图。CSS 230可以执行方法600以实现缓存数据的跨区域复制。

[0106] 在602处,给定区域210中的CSS 230从至少一个跨区域CRS 240接收一个或多个键值缓存操作。通常,CSS 230可以从系统200的所有其它区域210中的CRS 240接收键值缓存操作。跨区域CRS 240可以通过异步方式向CSS 230发送键值缓存操作,并且CSS 230可以在每一组操作到达时按照到达的顺序处理每一组操作。

[0107] 可选地,在604处,如果已在CRS 240处压缩收到的键值缓存操作(例如,如方法400中所述),则CSS 230可以解压缩键值缓存操作。在一些示例中,CSS 230可以通过检查收到的键值缓存操作的格式(例如,压缩格式或非压缩格式)来确定是否需要解压缩。检查收到的键值缓存操作的格式还可以指示压缩(如果已完成)是使用静态压缩还是动态压缩执行的。例如,如果压缩是动态执行的,则可以使用提供关于压缩的信息的压缩元数据来格式化压缩键值缓存操作。如果压缩是使用静态且已知的压缩算法执行的,则可以有压缩元数据。

[0108] 如果压缩是使用静态且已知的压缩算法执行的(例如,使用方法400中的步骤405),则CSS 230可以使用相应已知解压缩算法来执行解压缩。

[0109] 如果压缩是动态执行的(例如,使用方法400中的步骤406),则CSS 230可以执行步骤606和608以确定要使用的适当解压缩算法并解压缩键值缓存操作。

[0110] 在可选的606处,CSS 230根据与压缩键值缓存操作相关联的压缩元数据来确定一个或多个压缩属性。CSS 230可以知道压缩元数据的格式并且能够解析压缩元数据以确定该信息。压缩属性可以包括所使用的压缩算法的指示、数据的原始大小和数据的压缩大小等信息。使用该信息,CSS 230可以选择要使用的相应解压缩算法。压缩属性还可以提供信息以使CSS 230能够验证解压缩是否正确执行(例如,通过检查解压缩的大小是否与压缩元数据中指示的原始大小相匹配)。

[0111] 在可选的608处,CSS 230使用选定的解压缩算法执行解压缩。CSS 230可以执行验证以检查解压缩是否正确执行。

[0112] 在610处,CSS 230对其区域210中的缓存服务集群220重放(例如,对缓存服务集群220中的每个主实例222重放)键值缓存操作。

[0113] 图7是示出了可以由CRDT模块执行的示例性方法700的流程图,CRDT模块作为其功能的一部分由缓存服务实例(例如,缓存服务集群220中的主实例222)实现。方法700可以由缓存服务实例(其正在运行CRDT模块)执行以使缓存数据能够被写入其本地缓存(例如,缓存数据库)。

[0114] 在702处,给定区域210中的缓存服务实例从同区域210中的实体(例如CSS 230)接收键值缓存操作。应当注意,尽管在从CSS 230接收键值缓存操作的场景下描述了方法700,但是在从同区域中的客户端接收键值缓存操作的情况下,可以由缓存服务实例服务器执行类似的方法。

[0115] 在704处,生成冲突解决元数据。冲突解决元数据可以包括可以用于解决任何可能冲突的任何合适的信息。例如,冲突解决元数据可以包括版本向量、时间戳或任何其它合适的信息。

[0116] 可选地,在706中,如果收到的键值缓存操作是非幂等键值缓存操作,则将键值缓存操作转换为幂等键值缓存操作。将键值缓存操作转换为幂等键值缓存操作可以包括为键值缓存操作生成基于状态的元数据。下面将进一步讨论执行步骤706的示例性细节。

[0117] 在708处,将缓存数据和任何相关元数据存储到相应数据类型。示例性数据类型包括计数器模块数据类型、非计数器模块数据类型或复合模块数据类型(例如,集合和列表类型)等。下文描述了可以从现有数据类型扩展的示例性模块数据类型。下面将进一步描述本发明提供的用于将缓存数据和相关元数据以模块数据类型一起存储的示例性数据结构。

[0118] 在710处,缓存服务实例确定缓存数据是否会与缓存数据库中的任何其它键值缓存操作冲突。该确定可以通过识别数据库中是否存在与缓存数据具有相同键值的任何条目来执行。如果存在具有相同键值的条目,则可以通过将缓存数据的冲突解决元数据与条目的冲突解决元数据进行比较来确定可能的冲突。例如,如果冲突解决元数据包括版本向量,则可以通过将缓存数据的版本向量与条目的版本向量进行比较来确定可能的冲突。可以使用确定可能的冲突的其它方式,以及不同类型的冲突解决元数据。如果确定了可能的冲突,则可以执行可选步骤712来解决冲突。如果确定没有冲突,则方法700继续执行步骤714。

[0119] 在可选的步骤712处,执行冲突解决。可以根据冲突解决元数据的比较来执行冲突解决。考虑要写入数据库的新缓存数据具有第一组冲突解决元数据,并且现有数据库条目(与新缓存数据具有相同的键值)具有第二组冲突解决元数据的情况。根据第一冲突解决元数据和第二冲突解决元数据的比较,可以发现一组元数据优于另一组。例如,如果冲突解决元数据包括时间戳,则较晚的时间戳可以优于较早的时间戳(遵循最后写入者获胜的解决方法)。在另一个示例中,如果冲突解决元数据包括版本向量,那么如果给定版本向量中的每个条目都等于或大于另一个版本向量中的相应条目,并且给定版本向量中至少有一个条目严格大于另一个版本向量中的相应条目,则给定版本向量可以优于(或支配)另一个版本向量。如果发现第一冲突解决元数据优于第二冲突解决元数据,则将新的缓存数据写入缓存数据库;否则,丢弃新的缓存数据。

[0120] 在一些示例中,第一冲突解决元数据和第二冲突解决元数据可能不可比较。如果第一冲突解决元数据和第二冲突解决元数据具有不同格式或包含不同信息(例如,第一冲突解决元数据仅具有时间戳,而第二冲突解决元数据仅具有版本向量,或者第一冲突解决元数据和第二冲突解决元数据具有不同的向量长度),则可能是这种情况。如果第一冲突解决元数据和第二冲突解决元数据具有相同的格式但比较结果是不确定的,则它们也可能不可比较。例如,如果给定版本向量有一些条目大于另一个版本向量中的相应条目,但给定版本向量也有一些条目小于另一个版本向量中的相应条目,则可以认为这两个版本向量不可比较。在冲突解决元数据不可比较(或者比较给出不确定的结果)的情况下,缓存服务可以

使用预定义CRDT规则来解决冲突。用于冲突解决的一些示例性预定义CRDT规则包括：基于区域ID、可用区域ID或集群ID的字母数字顺序的方案；或依据区域、可用区域或集群之间的预定义优先级顺序的方案。

[0121] 通常，在将新操作写入缓存数据库(或丢弃新操作)之前，可以针对具有相同键值的现有数据库条目对每个收到的新操作执行冲突解决。这也可以称为将新收到的键值操作与具有相同键值的现有数据库条目合并。例如，为了合并字符串数据类型的数据和元数据，具有较大版本向量的键值操作将覆盖具有较小版本向量的键值操作。如果两个版本向量不可比较，则具有较大(即较晚)时间戳的操作将覆盖具有较小时间戳的操作。根据预定义CRDT规则，不同的数据类型具有不同的合并算法。

[0122] 在714处，转发缓存数据(如果已成功写入缓存数据库)以复制到同区域210中的CRS 240。例如，主实例缓存服务实例将缓存数据复制到其从实例缓存服务实例，所述从实例缓存服务实例继而将缓存数据复制到CRS 240。

[0123] 根据图2所示的关系，方法400、600和700可以由缓存服务集群220中的CRS 240、CSS 230和主实例222一起使用。因此，可以在系统200的所有区域210处理和复制来自客户端205的用户操作(即，来自通过客户端访问缓存数据库的缓存服务或应用的用户的操作)。

[0124] 在一些方面，本发明描述了用于将非幂等键值缓存操作转换为幂等键值缓存操作的示例性方法。这类方法可以用作方法700的一部分(例如，在步骤706中)。通常，通过将状态元数据附加到缓存数据，可以将非幂等键值缓存操作转换为幂等键值缓存操作，从而使键值缓存操作变为基于状态的操作(因此是幂等操作)。有多种方法可以生成状态元数据并将其附加到缓存数据。下面描述一个示例。

[0125] 图8是示出了可以由CRDT模块执行的示例性方法800的流程图，CRDT模块作为其功能的一部分由缓存服务实例(例如，缓存服务集群220中的主实例222)实现。可以执行方法800以将非幂等键值缓存操作转换为幂等键值缓存操作(例如，对于从同区域客户端接收的或从同区域CSS 230接收的非幂等键值缓存操作)。方法800可以由实现缓存服务实例(其正在运行CRDT模块)的处理设备执行，例如通过执行存储在存储器中的指令。方法800可以作为方法700的一部分执行，例如执行步骤706。

[0126] 在802处，缓存服务服务器确定缓存数据是否已在缓存数据库中具有现有条目。对于键值缓存数据库，可以通过确定是否已经存在具有相同键值的条目来执行该确定。

[0127] 如果缓存数据库中没有相应现有条目，则在804处为缓存数据生成新的状态元数据。生成状态元数据可以包括将元数据值初始化为默认值(例如，将计数器设置为零)。然后方法800继续执行步骤808。

[0128] 如果缓存数据库中有相应现有条目，则在806中，获取该相应条目的状态元数据。然后方法800继续执行步骤808。

[0129] 通常，与缓存数据相关联的状态元数据可以包括区域标识符(identifier, ID)和/或集群ID等来源信息，其可以用于标识缓存数据的来源区域和/或集群。给定区域中的实体可以使用来源信息来确定缓存数据来源于同一给定区域还是另一个区域。如果缓存数据来源于另一个区域，则不应将缓存数据广播到其它区域，以避免循环数据复制。冲突解决元数据可以包括在状态元数据中(例如，状态元数据可以包括用于冲突解决的信息，比如版本向量和/或时间戳)。在其它示例中，冲突解决元数据可以与状态元数据分开。

[0130] 状态元数据根据键值缓存操作的数据类型进行更新。下面讨论计数器模块数据类型和非计数器(或复合)模块数据类型。可能的非计数器数据类型包括集合、散列和列表等。可以针对不同的数据类型以各种适当的方式更新状态元数据。还应当注意,不同的数据类型可以具有不同类型或不同格式的状态元数据。

[0131] 在808处,确定键值缓存操作是否是计数器模块数据类型。

[0132] 如果确定是计数器模块数据类型,则在810处,使用缓存数据中指定的增量和/或减量的数量来更新状态元数据。然后方法800结束。

[0133] 如果确定不是计数器模块数据类型,则在812处确定操作是否是散列模块数据类型。

[0134] 如果确定是散列模块数据类型,则在814处确定字段数据类型是否是计数器字段。如果确定是计数器字段,则方法800继续执行步骤810以使用指定的增量和/或减量的数量来更新状态元数据。

[0135] 如果在814处确定不是计数器字段,或者如果在812处确定不是散列模块数据类型,则方法800继续执行步骤816。

[0136] 在816处,更新状态元数据可以不需要任何操作,并且方法800结束。

[0137] 可以针对从同区域中的客户端收到的用户操作中的每个写操作来执行方法800。由于从CSS 230收到的键值缓存操作可能(在跨区域发送到CSS 230之前)已经在来源区域中转换为幂等键值缓存操作,因此可以不需要将从CSS 230收到的缓存数据转换为幂等键值缓存操作。

[0138] 上述方法800包括根据某些数据类型将非幂等键值缓存操作转换为幂等键值缓存操作的示例性步骤。尽管上文讨论了计数器数据类型和散列数据类型(具有计数器字段),但应当理解,对于其它数据类型,可以类似地执行从非幂等键值缓存操作到幂等键值缓存操作的转换。

[0139] 图9是示出了由运行CRDT模块的缓存服务实例执行的示例性进程的简化框图。该框图是图2所示的系统图的简化,以便集中于CRDT模块的操作细节。尽管仅示出了区域1 210a中的一些实体的细节,但应当理解,可以在系统200的其它区域210中找到类似的细节。

[0140] 详细示出了一个主缓存服务实例222a(或简称为主实例222a),包括在主实例222a上实现的CRDT模块225a。主实例222a还实现缓存服务核心228a(例如,现有数据存储(例如缓存数据库,比如Redis)的核心代码)。作为缓存服务核心228a的附加组件的CRDT模块225a可以在主实例222a首次启动时加载和初始化。CRDT模块225a提供CRDT操作226a,其包括CRDT专用SET命令,在Redis作为核心缓存数据存储的环境中,所述命令可以使用新的Redis命令CRDT.SET来实现。CRDT模块225a还可以提供本地操作227a,其包括缓存服务核心228a的原始SET命令,在Redis环境中,所述命令可以使用新的Redis命令NATIVE.SET来实现。

[0141] 区域1 210a中的客户端205向同区域210a中的缓存服务集群发送用户操作(例如,SET命令)。SET命令将设置缓存数据的键值。在缓存服务集群中,命令由主实例222a使用缓存服务核心228a处理。应当注意,由CSS 230a对主实例222a重放的键值缓存操作也首先在主实例222a的缓存服务核心228a处进行处理。缓存服务核心228a将键值缓存操作转发到CRDT模块225a。

[0142] 在CRDT模块225a处,首先使用CRDT操作226a处理键值缓存操作。CRDT操作226a包

括:为缓存数据生成冲突解决元数据;为任何非幂等键值缓存操作添加状态元数据;将缓存数据及其相关元数据以模块数据类型一起存储。例如,CRDT操作226a可以包括图7的步骤704至708。

[0143] 在CRDT操作226a之后,将缓存数据转发(例如,通过CRDT.SET命令)到本地操作227a。本地操作227a包括:将缓存数据(包括CRDT元数据)写入缓存数据库(并执行任何所需的冲突解决);将缓存数据复制到从实例224a(所述从实例继而将缓存数据复制到同区域CRS 240a)。本地操作227a可以包括:执行用于将缓存数据写入缓存数据库的NATIVE.SET命令。例如,本地操作227a可以包括图7的步骤710至714。

[0144] 在成功复制缓存数据之后,本地操作227a可以将OK消息转发到CRDT操作226a,以指示缓存数据已成功写入缓存数据库并已复制。OK消息可以由CRDT操作226a转发到缓存服务核心228a,然后再转发到客户端205。

[0145] 将缓存数据跨区域复制到区域2 210b的CSS 230b(以及系统200中的其它区域210的其它CSS 230)由CRS 240a执行。类似地,将跨区域缓存数据从区域2 210b的CRS 240b(和系统200中其它区域210的其它CRS 240)复制到区域1 210a的CSS 230a。

[0146] 在一些方面,本发明描述了用于将冲突解决元数据和状态元数据与相关缓存数据一起放置的示例性数据结构。公开了可以用于将缓存数据与其元数据一起存储的示例性模块数据类型。这种模块数据类型通过将缓存数据与其元数据一起存储,可以实现更好的数据移动性并降低缓存数据与其元数据分离的风险。

[0147] 图10示出了用于键值缓存操作的示例性数据结构定义1000,它通过(例如)将缓存数据附加到其相关元数据来将缓存数据与其相关元数据一起放置。可以在缓存服务的可扩展模块中(例如,在作为Redis的扩展的CRDT模块中)将该定义1000注册为从其相应原始数据类型(例如,计数数据类型、非计数数据类型、复合数据类型等)扩展的新模块数据类型。

[0148] 如图10所示,数据结构根据缓存数据的原始数据类型定义数据字段1002,用于包含缓存数据。数据结构还定义了元数据字段1004,用于包含冲突解决和状态元数据。数据字段的所有缓存键值操作类型都是从原始数据类型派生的,且所有键值缓存操作类型都是针对元数据字段实现的。在示例性定义1000中,键值缓存操作类型:

[0149] `stringMetadata*moduleTypeStringGetMetadata();`

[0150] `void moduleTypeStringMerge();`

[0151] `stringMetadata*moduleTypeStringSet();`

[0152] 已针对元数据字段1004实现。其它的派生自原始数据类型并针对数据字段1002实现。

[0153] 使用要用于CRDT模块的这种模块数据类型,缓存数据和相关元数据一起包含在一个数据结构中。因为所公开的模块数据类型建立在原始数据类型上并且用于CRDT模块,所以该模块数据类型可以对当前数据结构和实现方式影响很小或没有影响。图10示出了示例性实现方式,然而可以将不同的数据结构定义为类似地同时包含缓存数据和相关元数据。

[0154] 在各示例中,本发明描述了能够在云计算中的缓存服务的键值缓存服务实例之间实现缓存数据的跨区域双活数据复制的系统和方法。本发明已经描述了各个方面。在一些方面,已经描述了一种用于将缓存键值操作复制到其它区域的模块(例如,CRS),用作同区域中多个源的从实例。在一些方面,已经描述了一种用于将键值缓存操作从其它区域同步

到同区域中的多个目标的模块(例如,CSS),用作同区域目标的客户端。

[0155] 在一些方面,已经描述了一种用于在将键值缓存操作发送到其它缓存服务实例之前动态地压缩键值缓存操作(例如,根据有效负载大小和类型,使用不同压缩算法或者完全省略压缩)的方法。根据有效负载大小和/或类型,从不同的可用压缩算法中选择压缩算法以动态压缩键值缓存服务操作。压缩属性与压缩数据一起存储在压缩元数据中。然后可以使用缓存服务的合适格式(例如用于Redis的RESP)将压缩操作与压缩元数据一起写入缓存操作日志/从缓存操作日志读取。

[0156] 在一些方面,已经描述了一种用于将缓存服务的非幂等键值缓存操作转换为幂等键值缓存操作的方法。可以通过添加状态元数据来将键值缓存服务的非幂等键值缓存操作转换为幂等键值缓存操作。然后可以使用基于状态的CRDT来解决任何冲突的操作。

[0157] 在一些方面,已经描述了一种用于解决冲突缓存操作的内置冲突解决模块(例如,CRDT模块)。可以通过可扩展模块添加缓存操作的内置冲突解决机制,以便从缓存实例可以从多个主实例复制操作。

[0158] 在一些方面,已经描述了一种用于将冲突解决元数据(和其它元数据,例如状态元数据)与相关缓存数据一起存储的数据结构。所述数据结构使用键值缓存服务的可扩展模块数据类型来将元数据与相关缓存数据一起放置。这可以帮助确保将数据和相关元数据作为一个原子事务来迁移。

[0159] 本发明的各示例可以帮助为最终用户和客户端提供具有全局可靠高性能云应用的跨区域双活键值缓存服务。

[0160] 在一些示例中,根据有效负载中的操作的大小和/或类型使用动态压缩,缓存操作的跨区域传输可以更快和/或更可靠。

[0161] 在一些示例中,缓存操作的冲突解决可以更快和/或更可靠,因为可以将非幂等键值缓存操作转换为幂等键值缓存操作并使用基于状态的CRDT解决冲突。

[0162] 在一些示例中,通过使用通过缓存服务的可扩展模块实现的内置冲突解决机制,从多个主从对复制缓存操作可以更快和/或更可靠。

[0163] 在一些示例中,通过将缓存服务的可扩展模块数据类型中的冲突解决元数据与相关缓存数据一起放置,缓存操作的冲突解决可以保持数据移动性。

[0164] 尽管本发明描述了由某些组件和物理实体执行的功能,但应当理解,在分布式系统中,过程中的一些或全部可以分布在多个组件和实体之间,并且过程的多个实例可以在分布式系统上执行。

[0165] 尽管本发明利用呈某种顺序的步骤描述方法和过程,但所述方法和过程的一个或多个步骤可视情况省略或更改。一个或多个步骤视情况可以按除了所描述顺序以外的顺序进行。

[0166] 尽管就方法而言至少部分地描述了本发明,但本领域普通技术人员将理解,本发明还涉及用于执行所描述的方法的至少一些方面和特征的各种组件,无论是硬件组件、软件或两者的任何组合。因此,本发明的技术方案可以以软件产品的形式体现。合适的软件产品可以存储在预先记录的存储设备或其它类似的非易失性或非瞬时性计算机可读介质中,包括DVD、CD-ROM、USB闪存盘、移动硬盘或其它存储介质等。所述软件产品包括有形地存储在其中的指令,这些指令使处理设备(例如,个人计算机、服务器或网络设备)能够执行本文

中所公开的方法示例。

[0167] 本发明可以在不脱离权利要求书的主题的情况下以其它特定形式体现。所描述的示例性实施例在所有方面均被视为仅是说明性的而非限制性的。可以对上述一个或多个实施例中的选定特征进行组合,以创建未明确描述的替代实施例,适合此类组合的特征均理解为落入本发明的范围内。

[0168] 还公开了所公开范围内的所有值和子范围。此外,虽然本文所公开和示出的系统、设备和过程可以包括特定数量的元件/组件,但是可以修改这些系统、设备和组件以包括更多或更少此类元件/组件。例如,尽管所公开的任何元件/组件可以为单个数量,但是可以修改本文所公开的实施例以包括多个此类元件/组件。本文所述的主题意在涵盖和包括技术上的所有合适更改。

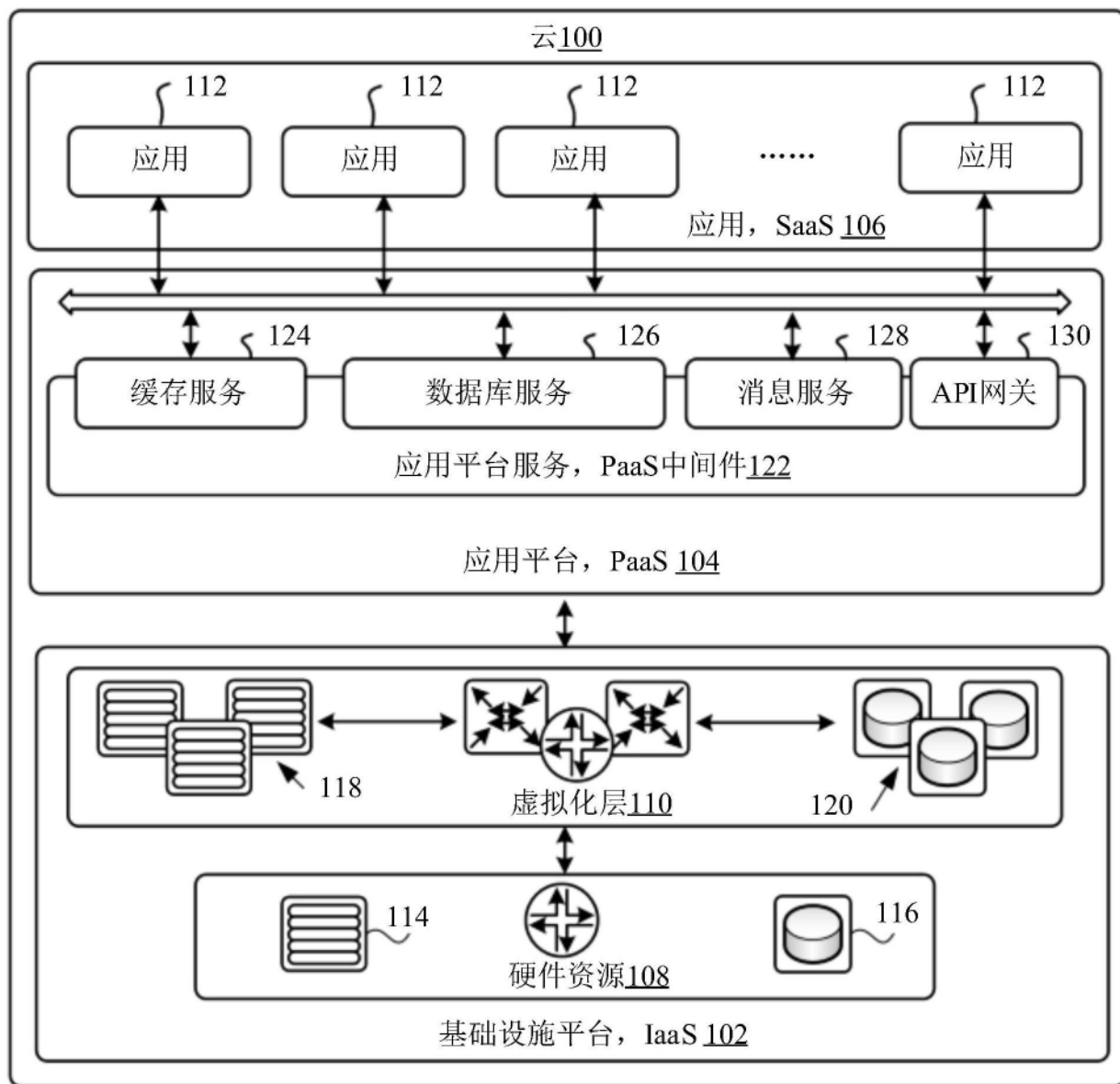


图1

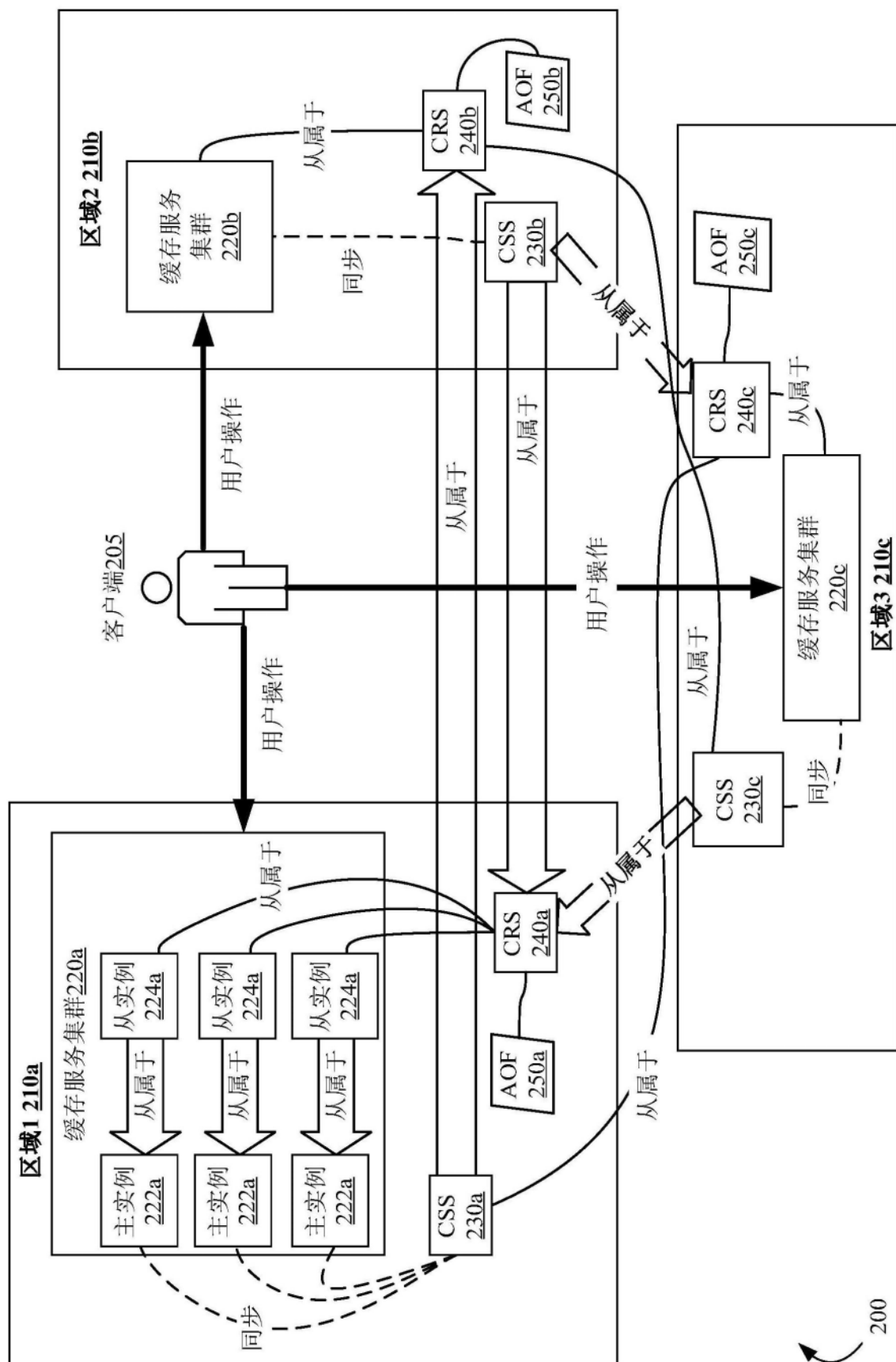


图2

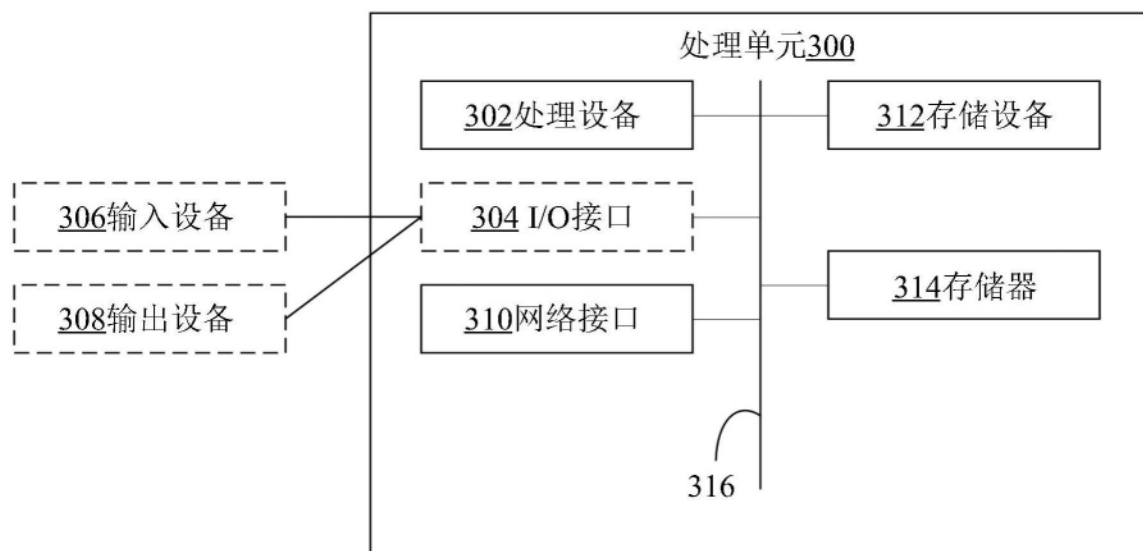


图3

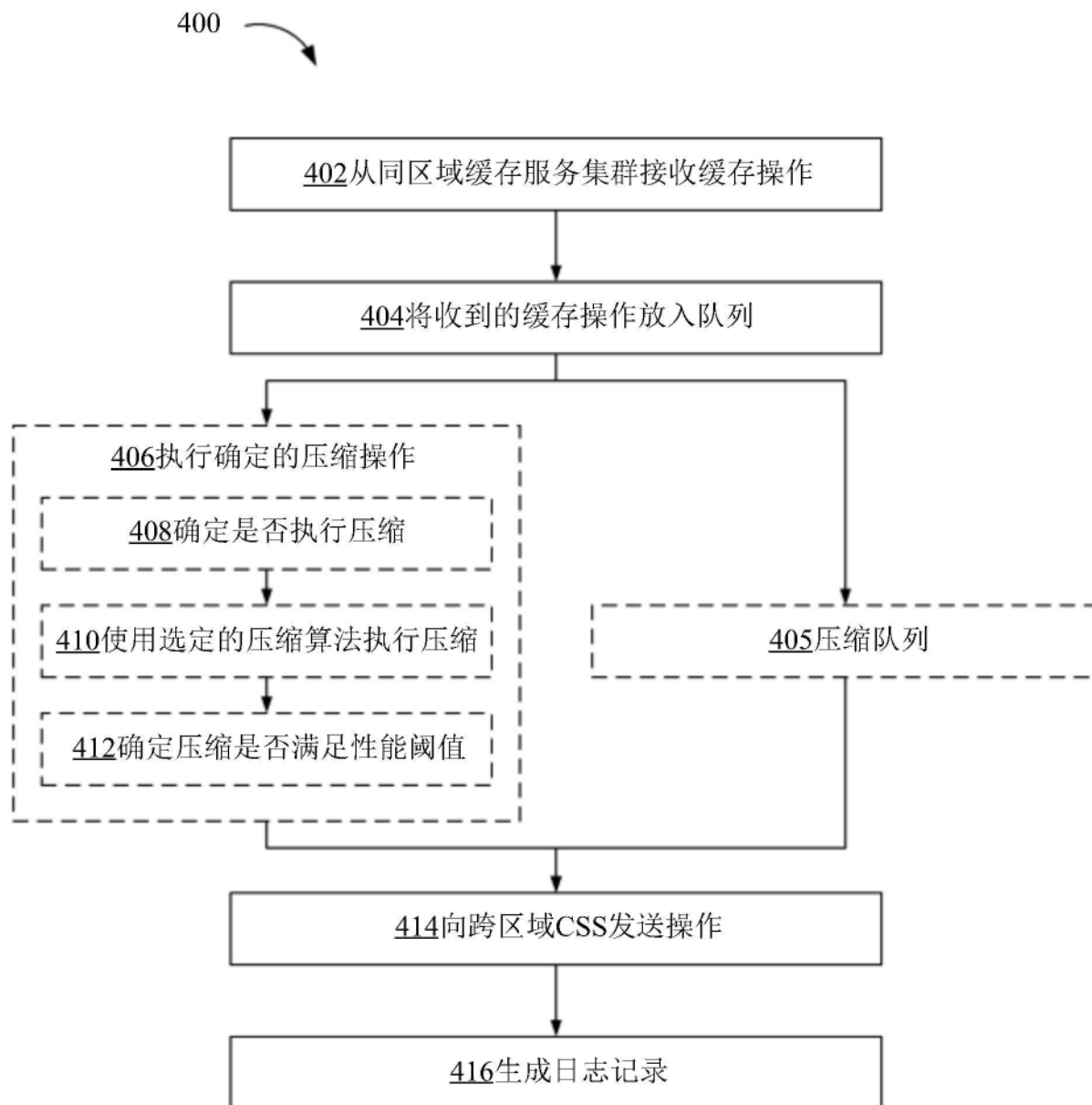


图4

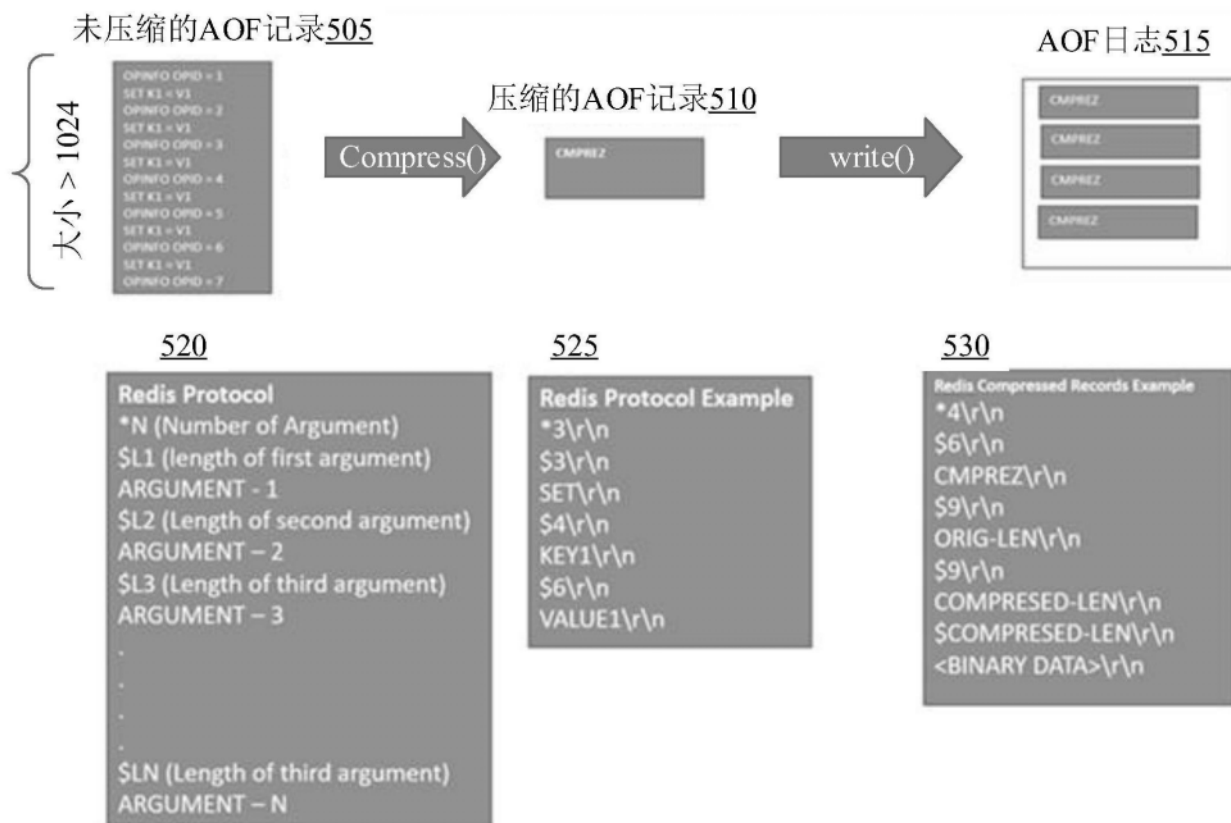


图5

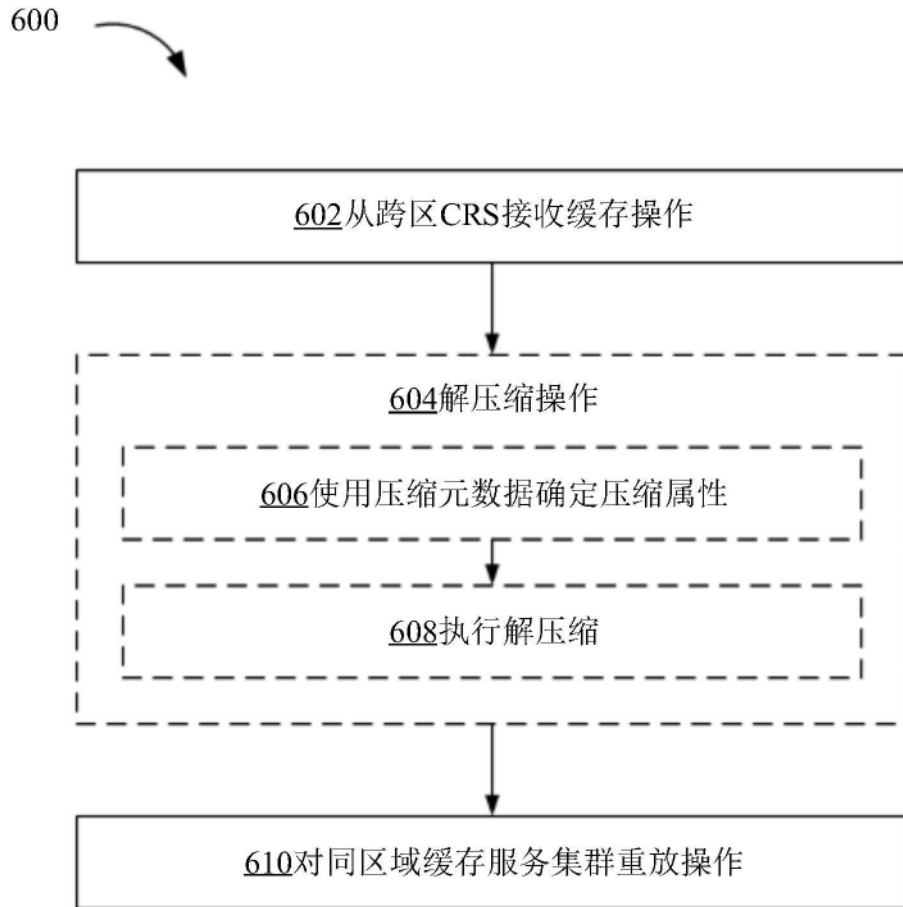


图6

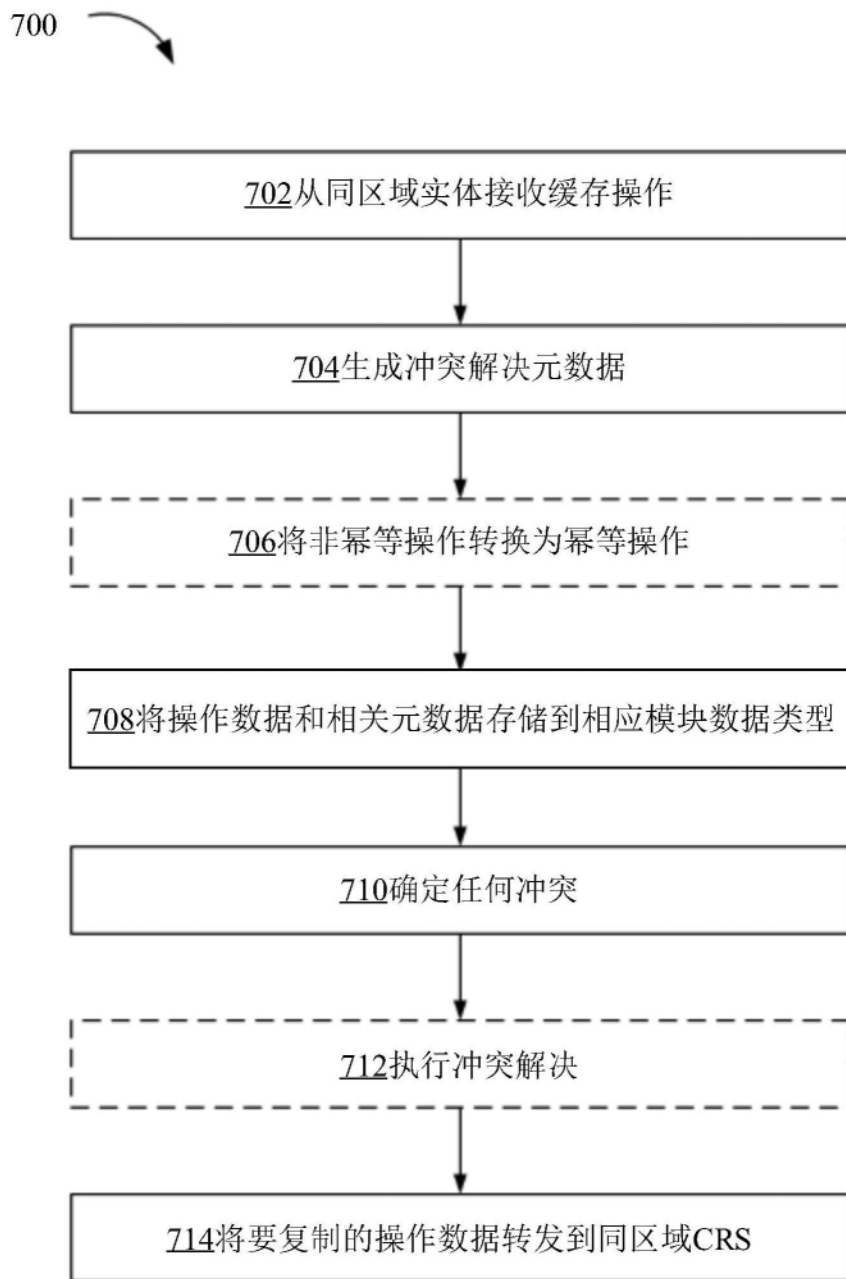


图7

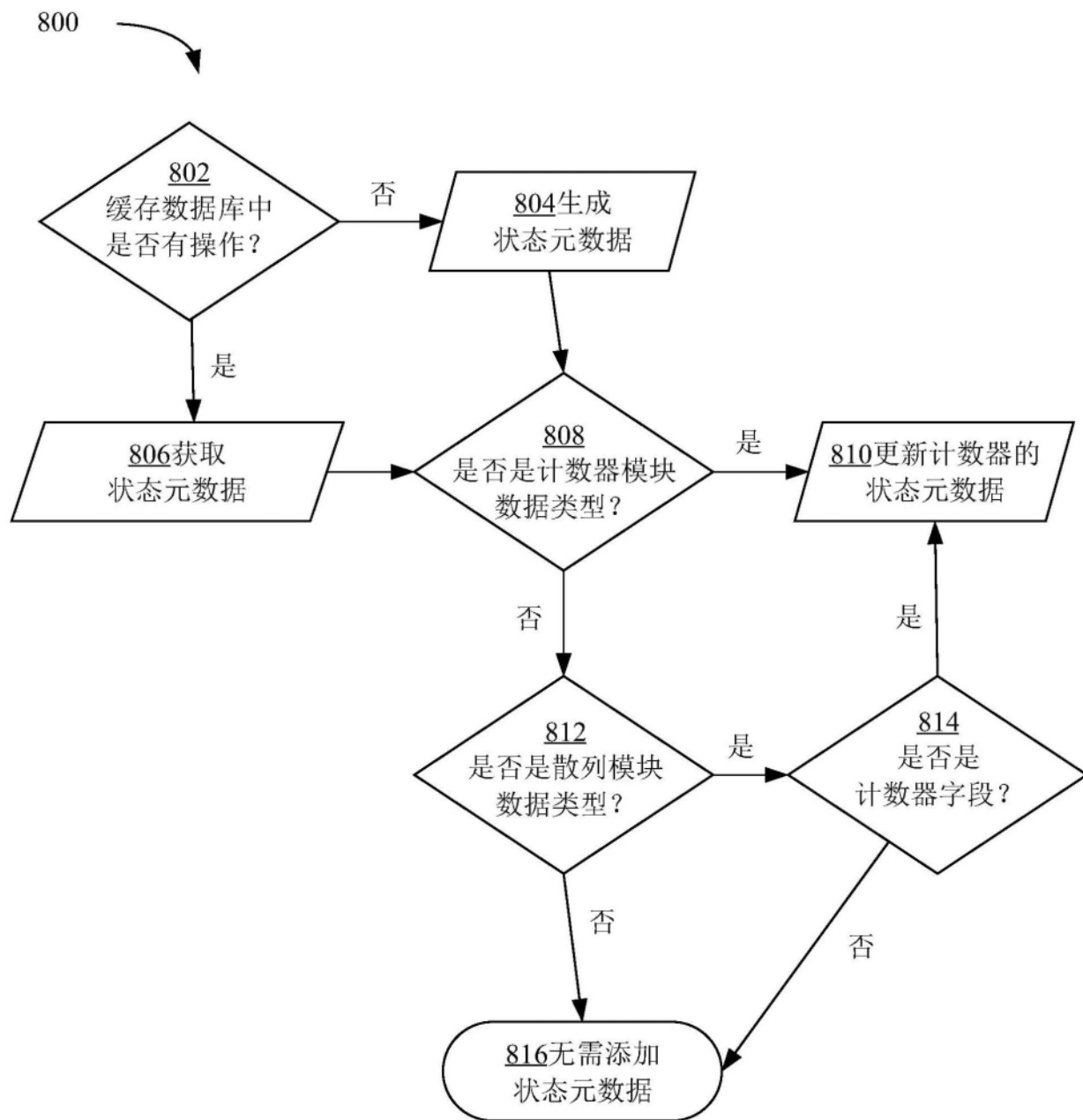


图8

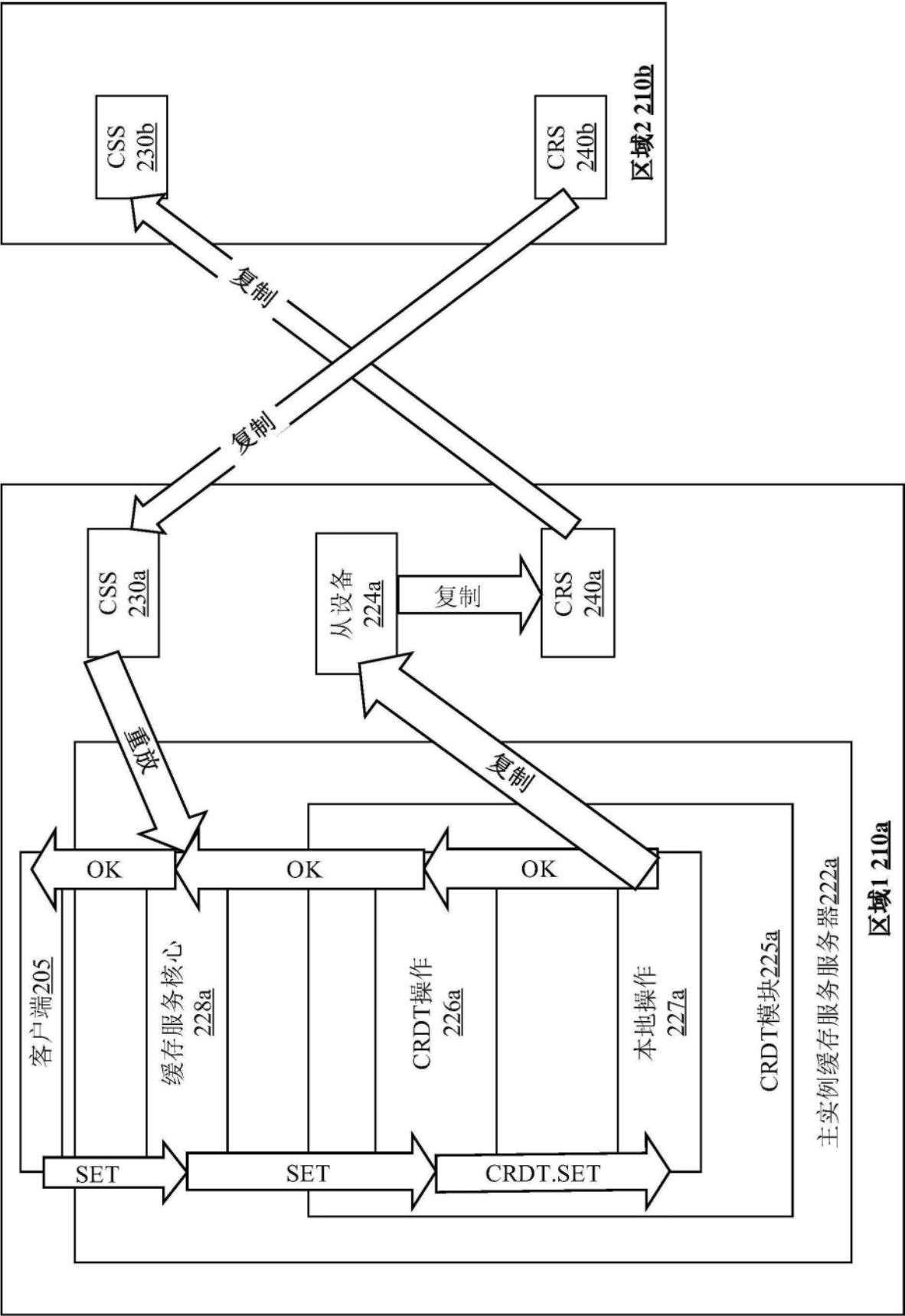


图9

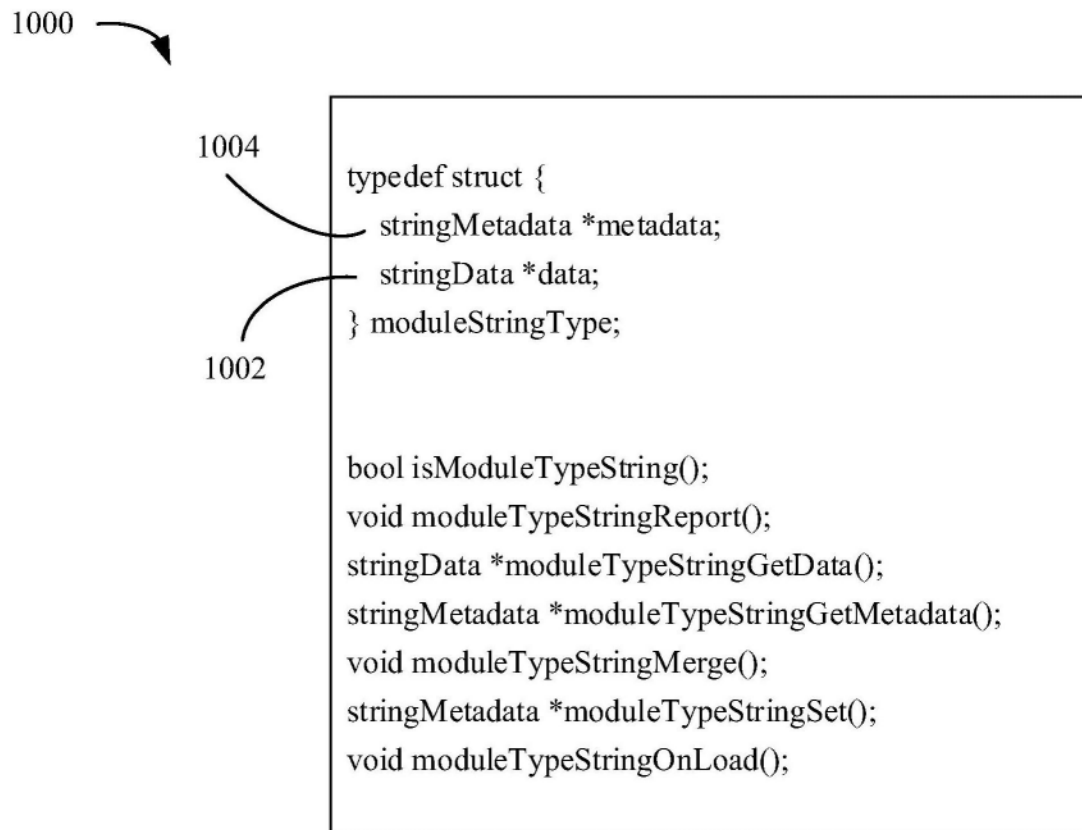


图10