

(19) 中华人民共和国国家知识产权局



(12) 发明专利申请

(10) 申请公布号 CN 105009084 A

(43) 申请公布日 2015. 10. 28

(21) 申请号 201380061634. 1

(51) Int. Cl.

(22) 申请日 2013. 12. 02

G06F 9/50(2006. 01)

H04L 29/08(2006. 01)

(30) 优先权数据

13/705,039 2012. 12. 04 US

(85) PCT国际申请进入国家阶段日

2015. 05. 26

(86) PCT国际申请的申请数据

PCT/US2013/072676 2013. 12. 02

(87) PCT国际申请的公布数据

W02014/088967 EN 2014. 06. 12

(71) 申请人 微软技术许可有限责任公司

地址 美国华盛顿州

(72) 发明人 T·莫希布罗达 Z·钱

M·E·鲁斯诺维奇 X·于 J·张

F·赵

(74) 专利代理机构 上海专利商标事务所有限公司 31100

代理人 陈斌

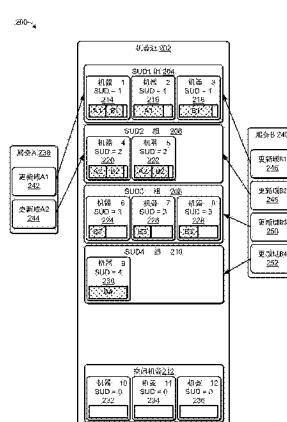
权利要求书2页 说明书13页 附图11页

(54) 发明名称

分布式计算平台中的服务分配

(57) 摘要

本文所述的技术和安排提供了在满足更新域约束的同时，更新服务、主机操作系统以及其他应用。在一些示例中，一个或多个控制器模块可维护一包括多个服务器更新域的数据结构，每个服务器更新域包括分布式计算系统的多个机器中可被同时更新的一组机器。所述一个或多个控制器模块可向所述多个机器分配多个实例，使得服务器更新域的数量最小化。



1. 一种计算机实现的方法,包括 :

在配置有可执行指令的一个或多个处理器的控制下,

将多个服务中的每一个服务的多个实例分配给分布式计算系统的多个机器,每个服务具有多个相应的域,每个实例被指派给相应服务的多个域之一,其中将实例分配给所述多个机器受到约束,使得所述多个机器中的每个机器不能主持来自同一服务的不同域的实例,且所述分布式计算系统的操作受到约束使得主持同一服务的不同域的实例的不同机器不能同时不可用;以及

维护一数据结构,所述数据结构包括多个服务器域,每个服务器域包括所述多个机器中可同时不可用的一组机器。

2. 如权利要求 1 所述的计算机实现的方法,其特征在于,所述多个机器中主持所述多个服务的一实例的每个机器被指派给所述多个服务器域之一。

3. 如权利要求 1 至 2 中任一项所述的计算机实现的方法,其特征在于,所述分配被执行使得通过对至少一个实例确定所述多个机器中的至少一个机器来将服务器域的数量最小化,所述至少一个机器如果被分配了所述至少一个实例则可基于服务器域的有序标识系统而被指派给最低服务器域。

4. 如权利要求 1 至 3 中任一项所述的计算机实现的方法,其特征在于,进一步包括:

对于所述多个服务中的至少一个服务,维护从该服务的每个域到至少一个服务器域的映射。

5. 存储计算机可执行指令的一个或多个计算机可读存储介质,所述计算机可执行指令在一个或多个处理器上被执行时执行包括以下的动作:

将多个服务中的每一个服务的多个实例分配给分布式计算系统的多个机器,每个服务具有多个相应的更新域,每个实例被指派给相应服务的多个更新域之一,其中将实例分配给所述多个机器受到约束,使得所述多个机器中的每个机器不能主持来自同一服务的不同更新域的实例,且所述分布式计算系统的更新受到约束使得主持同一服务的不同更新域的实例的不同机器不能同时被更新;以及

其中所述分配被执行使得为更新所述多个机器而可被同时更新的不同机器的分组的数量最小化。

6. 如权利要求 5 所述的一个或多个计算机可读存储介质,其特征在于,所述分配被执行来通过以下来使得分组的数量最小化,在第一实例要被分配给所述多个机器中的一机器时:

如果可用于主持所述第一实例的第一机器也是主持所述服务的与要被分配的所述实例相同的更新域的至少一个第二实例的分组的一部分,则将要被分配的所述实例分配给所述第一机器。

7. 如权利要求 5 至 6 中任一项所述的一个或多个计算机可读存储介质,其特征在于,所述分配被执行来通过以下来使得分组的数量最小化,在第一实例要被分配给所述多个机器中的一机器时:

如果不存在可用于主持所述第一实例且也是主持所述服务的与所述第一实例相同的更新域的至少一个第二实例的分组的一部分的第一机器,且存在所述多个机器中当前不在主持所述多个服务的实例的第二机器,则将所述第一实例分配给所述第二机器。

8. 如权利要求5至7中任一项所述的一个或多个计算机可读存储介质，其特征在于，所述分配被执行来通过以下来使得分组的数量最小化，在第一实例要被分配给所述多个机器中的一机器时：

如果不存在可用于主持所述第一实例且也是主持所述服务的与所述第一实例相同的更新域的至少一个第二实例的分组的一部分的第一机器，不存在所述多个机器中当前不在主持所述多个服务的实例的第二机器，且存在可主持所述第一实例而不造成更新域约束违反的第三机器，则将所述第一实例分配给所述第三机器。

9. 如权利要求5至8中任一项所述的一个或多个计算机可读存储介质，其特征在于，所述分配被执行来通过以下来使得分组的数量最小化，在第一实例要被分配给所述多个机器中的一机器时：

如果不存在可用于主持所述第一实例且也是主持所述服务的与所述第一实例相同的更新域的至少一个第二实例的分组的一部分的第一机器，不存在所述多个机器中当前不在主持所述多个服务的实例的第二机器，且存在可主持所述第一实例而不造成更新域约束违反的多个第三机器，则确定至少一个第三机器，该至少一个第三机器如果被分配了所述第一实例则可基于分组的有序标识系统而被指派给最低分组标识符；以及

将所述第一实例分配给所确定的机器。

10. 一种分布式计算系统，包括：

存储了控制器模块的计算机可执行指令的一个或多个计算机可读存储介质，所述计算机可执行指令由一个或多个处理器执行来：

维护一数据结构，所述数据结构包括多个服务器更新域，每个服务器更新域包括多个机器中可被同时更新的一组机器，所述多个机器主持多个服务中每个服务的多个实例，每个服务具有多个相应的更新域，每个实例被指派给相应服务的多个更新域之一，其中将实例分配给所述多个机器受到约束，使得所述多个机器中的每个机器不能主持来自同一服务的不同更新域的实例，且所述分布式计算系统的更新受到约束使得主持同一服务的不同更新域的实例的不同机器不能被同时更新；以及

将所述多个实例分配给所述多个机器，所述控制器模块将所述实例分配给所述多个机器使得通过对至少一个实例确定所述多个机器中的至少一个机器来将服务器更新域的数量最小化，所述至少一个机器如果被分配了所述至少一个实例则可基于服务器更新域的有序标识系统而被指派给最低服务器更新域。

分布式计算平台中的服务分配

背景技术

[0001] 使得在大规模计算群集上对应用和用户提供的服务进行构建、主持和缩放成为可能的云计算平台的持续增长导致了用于管理这些平台的系统和方法的增加。云操作系统的功能之一是在某一新应用被部署时决定构成该应用的实例应被分配到哪些机器上。该决定受不同约束的支配，且对于服务的性能、可缩放性以及容错性有影响。一个考虑因素是对机器进行实例分配造成了机器更新约束，即哪些机器能或不能在同时被重启。

发明内容

[0002] 提供本概述以便以简化形式介绍将在以下详细描述中进一步描述的一些概念。本概述并不旨在标识所要求保护主题的关键特征或必要特征，也不旨在用于限制所要求保护主题的范围。

[0003] 某些实现提供了用于在满足更新域约束的同时，更新服务、主机操作系统以及其他应用的技术和安排。在一些示例中，一个或多个控制器模块可维护一包括多个服务器更新域的数据结构，每个服务器更新域包括分布式计算系统的多个机器中可被同时更新的一组机器。所述一个或多个控制器模块可向所述多个机器分配多个实例，使得服务器更新域的数量最小化。

附图说明

[0004] 参考附图来描述详细描述。在附图中，附图标记最左边的数字标识该附图标记首次出现的附图。在不同的附图中使用相同的附图标记指示类似或相同的项。

[0005] 图 1 是根据一些实现的说明性环境的示意图。

[0006] 图 2 是根据一些实现的对服务实例的说明性分配的示例逻辑图。

[0007] 图 3 是根据一些实现的对服务实例的说明性分配的示例逻辑图。

[0008] 图 4 是根据一些实现的执行对服务实例的分配的说明性过程的流程图。

[0009] 图 5 是根据一些实现的为某一机器执行对服务器更新域的维护的说明性过程的流程图。

[0010] 图 6 是根据一些实现的对服务实例的说明性分配的示例逻辑图。

[0011] 图 7 是根据一些实现的对服务实例的说明性分配的示例逻辑图。

[0012] 图 8 是根据一些实现的执行对服务实例的分配的说明性过程的流程图。

[0013] 图 9 是根据一些实现的为某一机器执行对服务器更新域的维护的说明性过程的流程图。

[0014] 图 10 是根据一些实现的执行对多个机器的更新的说明性过程的流程图。

[0015] 图 11 例示出某些实现可以在其中操作的示例计算系统。

具体实施方式

[0016] 本文的某些实现提供了可使得机器（计算设备或服务器）群集能够在满足更新域

(UD) 约束的同时将更新服务、主机操作系统和其他应用所使用的时间和 / 或复杂性最小化的技术和安排。一般来说,某些实现可提供用于在任何类型的可用性约束的情况下分配服务实例的技术和安排,可用性约束要求例如在计划停机的情况下满足服务的最小级别可用性。在某些实现中,可用性约束可提出即使在面对更新、重启、计划停机时间、故障等情况下特定一组或特定数量的服务实例可用。如此,尽管下面的讨论是在更新服务器(例如更新服务器群集中的所有服务器或某一子集的服务器)的上下文中的,但是各实现不限于该上下文且可被应用于其他类型的可用性约束以及与之相关的域。

[0017] 更新域约束可由运行在群集上的每个服务(如每个应用)产生,为该服务指定更新域的数量。在某些实现中,更新域是声明主持相同服务的不同更新域的实例的机器不能被同时重启或离线的规则。这意味着来自相同服务的不同更新域的实例不能被分配给同一机器。在此,向某一机器分配某一服务的实例可涉及指令该机器执行或运行该服务的该实例。群集上运行的各种服务的实例跨群集中的机器的实际部署以及基于该部署而引起的更新域可确定该群集的总体更新域约束。群集的总体更新域约束可确定群集的“可更新性”,即群集中的节点可被更新的速度和容易性。

[0018] 如上所述,本公开包括可采用服务器更新域来使得在满足更新域约束的同时更新群集中的该组服务器所使用的时间和 / 或复杂性最小化的技术和安排。在某些这样的实现中,对于某一群集(或其他机器分组)中的每个机器,服务器更新域(SUD)被指派和维护。可按以下方式来维护SUD,即在任何时间,具有相同SUD的所有机器可在同一时间被重启(没有任何更新域冲突),即具有相同SUD的机器可形成机器的“重启类”。因此,在某些实现中,计划更新(如群集中所有服务器的主机OS更新)可仅仅由重启属于同一SUD的服务器组成,一次一个SUD。在一些实现中,用来向机器分配服务实例的技术用来使得SUD的数量最小化(如果可能的话)。在这些实现中,与涉及更大数量的SUD的情形相比,将SUD的数量最小化可导致更新过程的减少的时间利用和复杂性。

[0019] 本文讨论的两个示例的基于SUD的实例分配和维护技术是全局服务器更新域(G-SUD)技术和每服务服务器更新域(PS-SUD)技术。这些分配和维护技术在如何对服务实例指派和维护SUD的方式上不同,但并不意味着是互斥的或对可被采用的分配和维护技术的限制。简要来说,在G-SUD中,存在全局 $UD \rightarrow SUD$ 映射。每当一新实例被部署到机器之一时,该机器的SUD就可被调整, $UD \rightarrow SUD$ 映射可被调整,且以最小化不同SUD的数量的方式来进行该分配。在PS-SUD中,不是全局 $UD \rightarrow SUD$ 映射,而是在每服务(每应用)的基础上来维护映射。在某些实现中,这可允许更多的灵活性以及减少SUD的数量。在此,服务器更新域,即SUD,指的是机器被指派给的域,而更新域(UD)(这里也称为服务更新域)指的是服务器更新域可被映射到的某一服务内的实例的更新域。

[0020] 本文所述的分配和维护功能可在计算系统的软件和硬件中的各种级别被实现。这些级别包括操作系统(OS)级(如在带有或不带有应用支持的OS中)、应用级(要么与OS分开(即独立)要么作为到OS的插件或者作为到另一应用的插件)等等。例如,在某些实现中,分配和维护功能可由构造控制器(fabric controller)来实现。这种构造控制器可被实现为分布式计算服务的机器内的某一服务。具体来说,执行构造控制器的功能的机器是构造控制器正向其分配服务实例的那些机器。在其他实现中,构造控制器可被实现于一组一个或多个机器上,这些机器可专门用于执行构造控制器的功能性。出于简化,本文

的讨论将假设分配和维护功能性是由在分布式计算服务的机器上而不是专用机器上运行的控制器服务来执行的。这一实现的示例在下文关于图 1 和 11 被讨论。

[0021] 此外,尽管本文的讨论可涉及或讨论在群集和云计算的上下文中的实现,但各实现不限于群集或云计算。同样,尽管可作为应用或在应用的上下文中来讨论服务,但是这仅仅是服务的示例,而不应被视为限制。

[0022] 在此,更新域、服务器更新域等在本文的讨论中是通过数字标识符或服务字母和更新域号的组合(如更新域 A1)的方式来被标识或标记的。各实现不限于此,标识符可以是数字、字母、颜色、或其他标记系统的任何组合。在某些实现中,标记系统可以是诸如本文所讨论的数字系统之类的有序标识系统。然而,在其他实现中,标记系统可缺少严格排序(如基于颜色的标记系统)。而且,尽管下面关于附图讨论的各实现讨论了由分布式计算服务主持的两个服务,但是应理解各实现不限于此,且在某些实现中,更多的服务可同时由一分布式计算服务主持。

[0023] 出于可读性,本文可将模块间的交互描述为信号、命令或数据项的传递,但本领域的普通技术人员会理解,这样的交互可以以各种方式来实现,例如通过各种程序模块之间的功能调用来实现。

[0024] 示例实现

[0025] 图 1 例示出根据一些实现的系统的示例环境 100。系统 100 包括分布式计算服务 102、一个或多个服务所有者 104、以及使用用户设备 108 的一个或多个用户 106,它们全部都通过网络 110 通信。

[0026] 以此深度来观察系统 100 的操作,服务所有者 104-1 至 104-N 可向分布式计算服务 102,诸如向机器 102-1 的分布式控制器模块 112,作出提供服务 114(如应用)的请求,每个服务具有若干更新域。分布式计算服务 102 在机器 102-1、102-2 等直至 102-N 上分配和维护所请求的服务 114 的实例。然后使得这些服务 114 可由用户 106 通过用户设备 108 访问。当然,这仅仅是所述技术和安排能够在其中操作的一个示例环境,且被提供来允许对下文的阐述提供示例上下文。所述技术和安排当然不限于该环境。

[0027] 图 2 例示出在机器池 202 的机器(如计算设备或服务器)之间分配服务实例的示例逻辑图 200。具体来说,机器池 202 的机器被例示为被指派给四个 SUD 组之一:SUD1 组 204、SUD2 组 206、SUD3 组 208、SUD4 组 210,或指派给一组空闲机器 212。具体来说,如图 2 所示,机器 1 214、机器 2 216 以及机器 3 218 被指派给 SUD1 组 204;机器 4 220 和机器 5 222 被指派给 SUD2 组 206;机器 6 224、机器 7 226 以及机器 8 228 被指派给 SUD4 组 208;机器 9 230 被指派给 SUD4 组 210;以及机器 10 232、机器 11 234 和机器 12 236 是当前空闲机器 212,在图 2 中被指示为具有 SUD = 0。当然,其他实现可使用其他 SUD 标识符用于空闲机器组,或可将空闲机器标识为不具有 SUD 标识符的一组机器。出于简化和一致性,在这里使用 SUD 号 0。

[0028] 图 2 例示出根据 G-SUD 技术,服务 A 238 和服务 B 240 的实例的示例分配,每个服务包括相应的更新域。服务 A 238 包括两个更新域:更新域 A1 242 和更新域 A2 244。服务 B 240 包括四个更新域:更新域 B1 246、更新域 B2 248、更新域 B3 250 以及更新域 B4 252。在所例示的 G-SUD 技术中,在资源允许的情况下,来自每个更新域号的实例被分配给具有相同号的服务器更新域组。这是通过从更新域 242-252 至 SUD 204-210 的箭头来例示出的

以便于可视化。例如,属于更新域 A1 242 和更新域 B1 246 的实例被分配给具有 SUD = 1 的 SUD1 组 204; 属于更新域 A2 244 和更新域 B2 248 的实例被分配给具有 SUD = 2 的 SUD2 组 206; 属于更新域 B3 250 的实例被分配给具有 SUD = 3 的 SUD3 组; 而属于更新域 B4 252 的实例被分配给具有 SUD = 4 的 SUD4 组 210。尽管为了便于例示而未在图中示出,但是应注意到某一服务的某一更新域的许多实例可被分配或部署到某一机器。

[0029] 图 2 中也例示出了向个体机器分配实例。具体来说,图 2 指示出更新域 A1 242 的实例被分配给机器 1 214 以及机器 2 216,而更新域 B1 246 的实例被分配给机器 1 214 和机器 3 218。如所例示的,机器 1 214、机器 2 216 和机器 3 218 各自具有由它们相应的参考标号下的框反映出的某一量的计算资源。正被使用的资源的量由用所主持的实例的更新域标识符以及阴影线标记的子框来指示。空闲计算资源被指示为资源框内的空白。关于 SUD 组 206–210 例示出了类似的分配。

[0030] 在图 2 中所例示的时刻,机器 10 232、机器 11 234 和机器 12 236 是空闲机器 212,它们各自可被分配给 SUD 组 204–210 之一或被分配给额外的 SUD 组,例如在某一服务请求多于四个更新域的情况下或由于例如机器池 202 的高利用率而造成的将更新域分段 (fragmentation) 成其他服务器更新域的情况下。这一分段情形的处理参考图 3 在下文描述。

[0031] 图 3 例示出继图 2 中所示的状态之后机器池 202 的一种可能状态,其中服务 A 238 和服务 B 240 已经请求了额外的资源。为了便于例示,机器 11 234 和机器 12 236 没有被包括在图 3 中。

[0032] 在图 3 中所示的状态之前,更新域 B4 252 请求了额外的实例应被分配,结果是机器 10 232 被指派给了 SUD4 组 210 且更新域 B4 252 的实例在其上被实例化。如此,没有空闲机器 212 可用。

[0033] 在图 3 中所例示的时刻,更新域 A2 244 已经请求了一额外实例的实例化。然而,机器 4 220 和机器 5 222 没有足够的空闲计算资源来主持所请求的额外实例。由于没有空闲机器 212 可用,分配和维护机制搜索来定位具有所希望的计算资源且在被分配了来自更新域 A2 244 的该额外实例的情况下不会违反更新域约束的机器。在图 3 中,当前被指派给 SUD = 4 的机器 10 232 具有所希望的空闲计算资源来主持更新域 A2 244 的该额外实例,且并没有已经主持服务 A238 的一实例(已经主持服务 A 238 的实例会在被分配了更新域 A2 244 的额外实例的情况下违反更新域约束)。

[0034] 一旦确定机器 10 232 能够主持该额外实例,则分配机制将机器 10 232 移动到新形成的 SUD5 组 302,并将更新域 A2 244 的该额外实例分配给机器 10 232。这在图 3 中由虚线示出。在某些实现中,分配机制将该“分段的”机器移动到具有最小 SUD 号的 SUD,使得不会违反任何更新域约束。

[0035] 在某些实现中可提供其他附加特征。例如,某些实现可提供附加的维护机制。例如,某些实现可提供一种机制以在某一实例被移除时确定机器是否仍然是分段的。如果该机器仍然是分段的,即该机器仍然被分配了具有不同更新域号的实例,则该机制操作来确定是否有该机器可被转移到的具有更小号的 SUD 而不引起对更新域约束的违反。如果机器变为未分段,即仍然分配给该机器的实例是具有相同号的更新域的实例,则该机器可被指派到其号等于该机器正在主持的实例的更新域的号的 SUD 中。如果该机器被转移出其当前

SUD 且该机器是具有该 SUD 的最后一个机器，则该机制可将具有比现在为空的 SUD 更高 SUD 号的所有机器的 SUD 向下转移一个。

[0036] 图 4 例示出根据某些实现的示例过程流 400。在该具体情况下，该过程流例示出在 G-SUD 实现中向机器和服务器更新域分配实例的过程。在 4 的流程图中，每一个框表示可以用硬件、软件、或其组合实现的一个或多个操作。在软件的上下文中，各个框表示在由一个或多个处理器执行时使处理器执行既定操作的计算机可执行指令。一般而言，计算机可执行指令包括执行特定功能或实现特定抽象数据类型的例程、程序、对象、模块、组件、数据结构等。描述各个框的次序并不旨在被解释为限制，并且任何数量的所述操作可以按任何次序和 / 或并行地组合以实现各过程。出于讨论的目的，过程流 400 是参考上述的系统 100 和 200 来描述的，但其他模型、框架、系统和环境可实现所例示的过程。除了过程 400 之外，贯穿本公开所描述的其他过程（如图 5 和 8-10）也应相应地被解释。

[0037] 在某些实现中，在进行过程流 400 之前，系统可确定满足主持额外实例的最小级别适用性的一组候选机器。例如，该系统可将满足利用率约束、盘空间约束、以及故障域约束的一组机器确定为该组候选机器。

[0038] 在框 402，分配和维护机制接收为服务的某一更新域实例化一额外实例的请求。分配机制然后确定具有等于所请求的额外实例的更新域号的服务器更新域号的服务器更新域中的某一机器是否可用于主持该额外实例。如果存在具有等于所请求的额外实例的更新域号的 SUD 号的一可用机器，则该过程继续到框 404。否则，该过程继续到框 406。

[0039] 在框 404，分配和维护机制将该额外实例分配给具有等于该额外实例的更新域号的服务器更新域号的该可用机器。这就完成了该过程。

[0040] 在框 406，分配和维护机制确定是否有机器空闲以被分配给具有等于该额外实例的更新域号的 SUD 号的服务器更新域。如果是，则过程继续至框 408。否则，该过程继续到框 410。

[0041] 在框 408，分配和维护机制将该空闲机器指派给具有等于该额外实例的更新域号的服务器更新域号的服务器更新域，以及将该额外实例分配给该新指派的机器。这就完成了该过程。

[0042] 在框 410，分配和维护机制确定是否存在具有不等于该额外实例的 UD 号的 SUD 号的 SUD 中的、但如果被分配了该额外实例将不会违反更新域约束的机器。如果否，则过程继续至框 412。如果是，则过程继续至框 414。

[0043] 在框 412，分配和维护机制报告其不能实例化一额外实例。过程随后完成。

[0044] 在框 414，分配和维护机制将该额外实例分配给框 410 中所确定的机器。结果，该机器被分段，即该机器具有来自具有不同更新域号的两个或更多个服务的实例。在框 416，分配和维护机制将分配了该额外实例的该被分段的机器移动到其中不会发生更新域违反的最低编号的服务器更新域。在某些实现中，分配和维护机制可将被分段的机器移动到具有大于号 N 的号的服务器更新域。例如，某些实现可将被分段的机器移动到具有大于 N 的号的服务器更新域，其中 N 等于机器池 202 的机器上运行的任何服务的更新域的最大号。当然，这并非是限制性的，其他实现可使用不同的号或对 N 不同的确定。这就完成了该过程。

[0045] 对图 4 中所示的过程流 400 的上述讨论提供了在确定向哪个机器分配额外实例以及向所确定的机器分配该额外实例之后的操作的指示。然而，在某些实现中，这些后续操作

可由一不同的过程或一不同的控制器机制来执行,如下文关于图 5 所述的维护机制。

[0046] 图 5 例示出根据某些实现的示例过程流 500。在该具体情况中,该过程流示出在 G-SUD 实现中对某一机器的服务器更新域指派的维护。这一维护过程可被周期性地进行,或者该过程可人工地发起或甚至基于触发事件来被发起。可能的触发事件可包括添加实例、移除实例、或可能影响机器可被指派给的 SUD 组的任何其他事件。

[0047] 在框 502,分配和维护机制确定是否该机器是空闲的。如果是,则过程继续至框 504。否则,该过程继续到框 506。

[0048] 在框 504,分配和维护机制将该机器的服务器更新域号设为 0,从而将该机器移动到空闲机器组 212。这就完成了该过程。如上所述,出于简化和一致性,在这里使用 SUD 号 0。其他实现可使用其他 SUD 标识符用于空闲机器组,或可将空闲机器标识为不具有 SUD 标识符的一组机器。

[0049] 在框 506,分配和维护机制确定该机器是否被分段。如果是,则过程继续至框 508。否则,该过程继续到框 510。

[0050] 在 508,分配和维护机制确定并向该被分段的机器指派 SUD 号,该 SUD 号是比 N 大的最小号,使得不会导致对更新域约束的违反。用于该确定的 N 可以是上面关于图 4 的框 416 所讨论的。这就完成了该过程。

[0051] 在框 510,分配和维护机制确定是否某一实例已被移除(在此之前该机器曾是被分段的(但现在该机器是未分段的)),以及这是否是具有当前 SUD 号的最后的机器。如果否,则过程继续至框 512。如果都是,则过程继续至框 514。

[0052] 在框 512,分配和维护机制将该机器的 SUD 号设置为运行在该机器上的实例的 UD 号。由于该机器是未分段的,因此该机器上运行的所有实例都应具有相同的 UD 号。这就完成了该过程。

[0053] 在框 514,分配和维护机制将该机器的 SUD 号设置为运行在该机器上的实例的 UD 号。如上所述,由于该机器是未分段的,因此该机器上运行的所有实例都应具有相同的 UD 号。因为该机器曾是具有前一 SUD 号的最后的机器,分配和维护机制还可将具有高于现在为空的 SUD 号的全部机器的 SUD 号下移一个。这就完成了该过程。

[0054] 图 4 和 5 中所示的过程流 400 和 500 可用本领域技术人员鉴于本文所提供的公开而显而易见的许多变型来实现。可改变的一个方面是所述步骤是顺序执行还是并行执行。另一变型可涉及各种确定的顺序和 / 或在分配和维护确定时使用哪些规则或测试。鉴于本文的公开,关于 G-SUD 实现中分配和维护机制的实现的这些和其他变型对于本领域技术人员来说将显而易见。

[0055] 图 6 例示出在机器池 202 的机器之间分配服务实例的另一示例逻辑图。具体来说,在图 6-7 中,使用每服务更新域(PS-SUD)分配和维护机制来将服务 A 238 和服务 B 240 的实例分配给机器池 202 的机器。如上所述,PS-SUD 分配和维护机制不必采用所有应用的所有实例对具有等于每个实例的 UD 的 SUD 号的 SUD 的全局分配(在可能的情况下)。相反,PS-SUD 分配和维护机制维护从服务的更新域的每一个到机器池 202 的一个或多个服务器更新域的“每服务”映射。在图 6-7 中,这是通过在更新域框中包含该映射来指示出的。例如,在图 6 中,标记为“更新域 A2 244”的框包括映射“更新域 A2 → {3, 5}”以指示出更新域 A2 映射到服务器更新域三(3)和五(5),即 SUD3 组 208 和 SUD5 组 302。出于以下讨论的目

的,应注意图 6 中所示的更新域 A1 242 和更新域 B2 248 的映射来与图 7 比较。具体来说,如图 6 所示,更新域 A1 242 映射到 SUD1 组 204,而更新域 B2 248 映射到 SUD2 组 206。

[0056] 图 7 例示出继图 6 中所示的状态之后的机器池 202 的可能状态,其中服务 A 238 和服务 B 240 已经请求了额外的资源。具体来说,已对更新域 A1 242 然后对更新域 B2 248 请求了额外实例。

[0057] 在操作中,在接收到对更新域 A1 242 的一个或多个额外实例的请求之际,分配和维护机制确定在图 6 中所示的时刻更新域 A1 被映射到的 SUD1 组不具有可用于主持更新域 A1 242 的该一个或多个额外实例的资源。分配和维护机制然后确定没有空闲机器 212 可用。在发现没有空闲机器 212 可用之际,分配和维护机制确定是否有指派给没有被映射到另一服务 A 更新域实例的 SUD 组且具有满足所述请求的资源的至少一个机器。在图 6 所例示的状态中,分配和维护机制确定 SUD2 组 206 当前没有被映射到服务 A 238 以及 SUD2 组 206 的机器 4 220 和机器 5 222 具有满足该请求的可用资源。分配和维护机制然后将 SUD2 组 206 映射到更新域 A1 242,以及将更新域 A1 的所请求的一个或多个实例分配给机器 4 220 和机器 5 222。

[0058] 在将 SUD2 组 206 映射到更新域 A1 242 之后,分配和维护机制从服务 B 接收对更新域 B2 240 的额外实例的请求。如图 7 所示,更新域 B2 所映射到的 SUD2 组 206 的机器不具有空闲资源来使得额外实例能被主持。具体来说,映射到更新域 A1 242 的额外实例的资源已经消耗了机器 4 220 和机器 5 222 上可用的所有资源。如此,分配和维护机制再次确定是否有任何空闲机器 212 或是否有分配给一 SUD 组的、不在主持另一服务 B 更新域的实例且具有满足该请求的资源的至少一个机器。当然,在同时对某一更新域请求了多个额外实例的情况下,同一或不同 SUD 组的多个机器可被使用,且可给予具有可用于主持所有被请求的额外实例的资源的机器优选。在图 6 所例示的状态中,分配和维护机制确定 SUD5 组 302 当前没有被映射到服务 B 240 以及 SUD5 组 302 的机器 11 234 和机器 12 236 具有满足该请求的可用资源。因此,分配和维护机制将更新域 B2 的一个或多个实例分配给机器 11 234 和机器 12 236。

[0059] 图 8 例示出根据某些实现的示例过程流 800。在该具体情况中,该过程流例示出在 PS-SUD 实现中向服务器更新域分配实例的过程。在某些实现中,在进行过程流 800 之前,系统确定满足主持该实例的最小级别适用性的一组候选机器。例如,该系统可将满足利用率约束、盘空间约束、以及故障域约束的一组机器确定为该组候选机器。

[0060] 在框 802,分配和维护机制确定该额外实例是否是该额外实例的服务更新域的第一实例。如果是,则过程流继续至框 804。否则,该过程继续到框 806。

[0061] 在框 804,分配和维护机制选择可用的最低非冲突 SUD 中的一候选机器。过程随后完成。

[0062] 在框 806,分配和维护机制确定是否在映射到该额外实例的服务更新域的某一 SUD 中已经存在具有可用资源的机器。如果是,则过程流继续至框 808。否则,该过程继续到框 810。

[0063] 在框 808,分配和维护机制将该额外实例分配给所确定的机器。过程随后完成。

[0064] 在框 810,分配和维护机制确定是否有空闲机器。如果是,则过程流继续至框 812。否则,该过程继续到框 814。

[0065] 在框 812, 分配和维护机制将该额外实例分配给一空闲机器。分配和维护机制然后将该空闲机器指派给例如映射到该额外实例的更新域的最低 SUD。过程随后完成。

[0066] 在框 814, 分配和维护机制确定是否有可用的机器, 该机器如果被分配了该额外实例则将不会在该机器本身内具有更新域约束违反。如果是, 则过程流继续至框 816。否则, 该过程继续到框 818。

[0067] 在框 816, 分配和维护机制确定一机器, 该机器如果被分配了该额外实例则允许该机器被指派给最低 SUD 而不会在该 SUD 内造成更新域约束违反。分配和维护机制然后将该额外实例分配给所确定的机器, 且在某些情况下将所确定的机器指派给所确定的最低 SUD。过程随后完成。

[0068] 在框 818, 分配和维护机制报告其不能实例化该额外实例。过程随后完成。

[0069] 尽管上述讨论以及图 8 中所示的过程流提供了除了确定向哪个机器分配额外实例以及将该额外实例分配给所确定的机器之外的操作的指示, 但是这些后续操作可由一不同的过程或一不同的控制器机制来执行, 如下文关于图 9 所述的维护机制。

[0070] 图 9 例示出根据某些实现的示例过程流 900。在该具体情况中, 该过程流例示出在 PS-SUD 实现中维护某一机器的服务器更新域号的过程。这一维护过程可被周期性地进行, 或者该过程可人工地发起或甚至基于触发事件来被发起。可能的触发事件可包括添加实例、移除实例、或可能影响机器可被指派给的 SUD 组的任何其他事件。

[0071] 在 902, 分配和维护机制确定是否一实例已被添加到该机器或从该机器移除。如果某一实例已被移除, 则该过程继续到框 904。如果某一实例已被添加, 则该过程继续到框 906。

[0072] 在框 904, 分配和维护机制确定该机器是否现在空闲。如果是, 则过程继续至框 908。否则, 该过程继续到框 910。

[0073] 在框 908, 分配和维护机制将该现在空闲的机器指派给空闲机器 SUD 组。这就完成了该过程。

[0074] 在框 910, 分配和维护机制将该机器指派给最低可能的非冲突 SUD。这就完成了该过程。

[0075] 返回到框 906, 分配和维护机制确定该额外实例是否是该额外实例的服务更新域中的第一实例。如果是, 则过程继续至框 912。否则, 该过程继续到框 914。

[0076] 在框 912, 分配和维护机制将该机器指派给最低非冲突 SUD, 以及将所确定的最低非冲突 SUD 映射到该额外实例的服务更新域。这就完成了该过程。

[0077] 在框 914, 分配和维护机制确定该机器的 SUD 是否已经被映射到该额外实例的服务更新域。如果是, 则过程继续至框 916。否则, 该过程继续到框 918。

[0078] 在框 916, 分配和维护机制保持该机器的 SUD 不变。这就完成了该过程。

[0079] 在框 918, 分配和维护机制确定在分配该额外实例之前该机器是否曾是空闲的。如果是, 则过程继续至框 920。否则, 该过程继续到框 922。

[0080] 在框 920, 分配和维护机制将该机器指派给已经映射到该额外实例的服务更新域的 SUD, 该 SUD 包含该额外实例的服务的大多数实例。这就完成了该过程。

[0081] 在框 922, 分配和维护机制确定是否有映射到该机器所主持的所有实例的服务更新域的一公共 SUD。如果是, 则过程继续至框 924。否则, 该过程继续到框 910, 框 910 如上

所述完成该过程。

[0082] 在框 924, 分配和维护机制将该机器指派给被映射到由该机器所主持的实例的服务更新域的最低公共 SUD。这就完成了该过程。

[0083] 一般来说, 上面讨论的基于 SUD 的分配和维护技术可基于特定的实现来被修改或重新组织。如此, 上面讨论的技术不应被视为限制。通过上面的讨论, 在任何给定的基于 SUD 的分配和维护技术中可使用 (但不要求使用) 若干分配和维护规则。以下概述这些规则中的一些。

[0084] 一个可能的规则, 在本文中称为现有 SUD 规则, 可选择其 SUD 已经在额外实例的服务更新域 SUD 映射中的机器。作为第二选择, 该现有 SUD 规则还选择一空闲机器, 而不是已经主持实例、不在该额外实例的服务更新域 SUD 映射中的一机器。

[0085] 另一可能的规则, 在本文中称为公共 UD 规则, 可选择正主持具有与额外实例相同 UD 号的实例的机器, 而不管所主持的实例属于什么服务。该公共 UD 规则的一个实现在上面讨论的全局 SUD 分配技术中被使用。在那里, 该公共 UD 规则可选择具有等于该额外实例的 UD 的 SUD 的机器, 这相当于选择主持具有与该额外实例相同 UD 的其他实例的机器。在某些实现中, 作为第二选择, 该公共 UD 规则可选择主持具有更少不同 UD 号的实例的 SUD 中的机器。

[0086] 第三个可能的规则, 在本文中称为最低 SUD 规则, 可选择能由该实现所使用的 SUD 分配和维护算法指派给最低“结果 SUD”的机器。在此, 最低“结果 SUD”指的是如果该额外实例被分配给机器则该机器将可被指派给的最低 SUD。如果存在对机器的等价选择, 则最低 SUD 规则可选择将任何 SUD 上移最小化的机器, 即机器的 SUD 号的增加被最小化。最低 SUD 规则的实现在上面关于 G-SUD 和 PS-SUD 分配和维护机制两者进行了讨论。例如, 框 816 采用了某种形式的最低 SUD 规则, 即选择机器, 该机器如果被分配该额外实例则能够被指派最低 SUD 而没有更新域约束冲突。

[0087] 第四个可能的规则, 在本文中称为“批规则”, 可选择能主持要对某一服务 (例如新部署的服务) 的某一更新域分配的一组额外实例中的全部实例或最大数量的实例的机器,。

[0088] 上面讨论的可能的规则可以或可以不在任何特定的基于 SUD 的分配和维护实现中被使用, 且因此不应被视为限制。此外, 规则被应用于一组候选机器的顺序可基于实现的细节来改变。从而, 某些实现可执行现有 SUD 规则以及使用批规则作为“附加赛”, 而其他实现可相反进行。当然, 如上所述, 某些实现可采用现有 SUD 规则和批规则两者, 一个规则或另一个规则, 或两者都不采用。这对上述讨论的规则的任何组合都适用。鉴于本文的公开, 关于所述分配和维护技术的细节的实现的这些和其他变型对于本领域技术人员来说将显而易见。

[0089] 已经建立了服务器更新域 (SUD) 的系统可调用采用这些 SUD 的更新过程。这一更新过程在图 10 中被例示出。

[0090] 图 10 例示出用于向分布式计算服务 102 的机器 102-1、102-2 至 102-N 提供更新机制的示例过程流 1000。具体来说, 过程流 1000 使用上述关于图 2-9 所述的分配和维护机制所分配的 SUD。每当计划停机发生且机器要被重启 (由于服务更新或者由于主机 OS 更新) 时, 该更新过程可被调用。在后一情况中, 可按所有服务的 UD 约束都被满足的方式来

重启所有机器。

[0091] 在框 1002，更新机制确定最高编号的 SUD 是否已被更新。如果否，则过程流继续至框 1004。否则，该过程流继续到框 1006。

[0092] 在框 1004，更新机制按数字顺序执行下一 SUD 中的机器的更新。取决于给定实现的细节，其他 SUD 中的其他机器可被同时更新或者 SUD 的其他排序可被使用。在完成了当前 SUD 中的机器的更新之际，过程流返回到框 1002。

[0093] 在框 1006，更新机制已经完成了对机器的更新，并准备该更新的结果的报告。过程随后完成。应注意，取决于实现，对更新的结果的报告的准备可不被执行、可贯穿更新过程被执行、或可被部分地执行。

[0094] 图 10 中阐述的过程属于本文中称为“纯基于 SUD 的”更新过程的类型。如上所述，各实现不限于这一过程。下面，讨论某些替代更新过程。鉴于本文的公开，关于所述更新技术的细节的实现的这些和其他变型对于本领域技术人员来说将显而易见。

[0095] 在某些实现中，可采用更激进的基于 SUD 的更新机制。该机制包含两个步骤。首先，类似于图 10 的纯基于 SUD 的更新算法，它将具有当前被选择用于更新的 SUD 的所有机器添加到准备更新列表。然而，该机制还向该列表添加来自其他 SUD 组的任何额外的非冲突机器。具体来说，该机制以给定顺序（如随机地或按 SUD）来查看具有不同 SUD 的所有剩余机器，并将任何非冲突机器添加到准备更新列表。

[0096] 在其他实现中，可使用介于纯基于 SUD 的更新机制和激进的基于 SUD 的更新机制之间的流水线化的基于 SUD 的更新机制。该机制如下工作。。当 SUD = 1 被选择来更新时，不是试图将来自所有其他 SUD 组的额外的机器添加到准备更新列表（但激进的基于 SUD 的过程这么做），流水线化的基于 SUD 的过程可试图将 SUD = 2 机器添加到准备更新列表，即来自要被更新的下一 SUD 的非冲突机器。

[0097] 又一些实现可使用基于冲突的试探法来排序要被更新的机器。在某些实现中，就例如效率而言，机器被考虑来被更新（或添加到准备更新列表）的顺序是重要的。两个示例是“最冲突的”机器最先方法以及“最不冲突的”机器最先方法。如名称所暗示的，这些方法基于机器具有的与其他 SUD 中的机器的更新域冲突的量来排序机器进行更新（或机器被指派给的 SUD）。

[0098] 尽管出于讨论的目的已经例示出若干示例，但是可使用许多其他配置，且因此本文的各实现不限于任何特定的配置或安排。例如，本文的讨论涉及从服务更新域到服务器更新域的映射被用于服务的实例到机器的分配。这不应被视为限制，因为这些映射不必被包括在内。例如，分配和维护机制可维护一数据结构，该数据结构包括机器被指派给的服务器更新域，但分配技术可仅考虑最低 SUD 规则，通过该规则实例被分配给可被指派给最低 SUD 而不会导致更新域约束违反的可用机器。换言之，该最低 SUD 规则可被使用而无需从服务 UD 到 SUD 的映射（全局或每服务）。鉴于本文提供的公开，各种实现的逻辑和实际结构和框架中的其他变型将对于本领域的技术人员显而易见。

[0099] 本文所讨论的过程仅仅是为讨论的目的而提供的示例。鉴于本文的公开，许多其他变型将对于本领域的技术人员显而易见。此外，尽管本文的公开阐述了用于执行本文的技术和过程的合适的框架、架构和环境的若干示例，但是本文的实现不限于所示和所讨论的具体示例。本文所例示的过程被示为逻辑流程图中的操作的集合，这表示一系列操作，这

些操作中的某些或全部可用硬件、软件或其组合来实现。在软件的上下文中，这些框表示存储在一个或多个计算机可读介质上的计算机可执行指令，这些指令在由一个或多个处理器执行时执行既定操作。一般而言，计算机可执行指令包括执行特定功能或实现特定抽象数据类型的例程、程序、对象、组件、数据结构等等。描述操作的次序并不旨在被解释为限制。任何数量的所述框可以任何顺序被组合和 / 或并行来实现所述过程，且不是所有的框都需要被执行。

[0100] 示例计算设备

[0101] 图 11 例示出可用于实现本文所描述的模块和功能的计算设备 1100 和环境的示例配置。计算设备 1100 可包括至少一个处理器 1102、存储器 1104、通信接口 1106、显示设备 1108（例如，触摸屏显示器或其他显示器）、其他输入 / 输出（I/O）设备 1110（例如，触摸屏显示器或鼠标和键盘）、以及一个或多个大容量存储设备 1112，它们能够诸如经由系统总线 1114 或其他合适的连接而彼此通信。作为示例，计算设备 1100 可以是分布式计算服务的机器或节点（如机器 102-1）。

[0102] 处理器 1102 可以是单个处理单元或数个处理单元，它们都可包括单个或多个计算单元或多个核。处理器 1102 可被实现为一个或多个微处理器、微型计算机、微控制器、数字信号处理器、中央处理单元、状态机、逻辑电路、和 / 或基于操作指令来操纵信号的任何器件。除其他能力之外，处理器 1102 可被配置成取出并执行存储在存储器 1104、大容量存储设备 1112 或其他计算机可读介质中的计算机可读指令。

[0103] 存储器 1104 和大容量存储设备 1112 是用于存储由处理器 1102 执行来执行上述各种功能的指令的计算机存储介质的示例。例如，存储器 1104 一般包括易失性存储器和非易失性存储器（例如，RAM、ROM 等）。此外，大容量存储设备 1112 一般可包括硬盘驱动器、固态驱动器、包括外部和可移动驱动器在内的可移动介质、存储卡、闪存、软盘、光盘（例如，CD、DVD）、存储阵列、网络附连存储、存储区域网络等等。存储器 1104 和大容量存储设备 1112 在本文中可被统称为存储器或计算机存储介质，并可能够按照计算机程序代码存储计算机可读、处理器可执行程序指令，计算机程序代码可由处理器 1102 执行作为被配置来执行在本文的实现中所描述的操作和功能的特定机器。

[0104] 计算设备 1100 还可包括用于诸如经由网络、直接连接等与其他设备交换数据的一个或多个通信接口 1106，如以上所讨论的。通信接口 1106 可便于各种各样网络和协议类型内的通信，包括有线网络（例如，LAN、电缆等）和无线网络（例如，WLAN、蜂窝、卫星等）、因特网等等。通信接口 1106 也可提供与诸如存储阵列、网络附连存储、存储区域网络等中的外部存储（未示出）的通信。

[0105] 诸如触摸屏显示器或其他显示设备之类的显示设备 1108 可被包括在某些实现中。其他 I/O 设备 1110 可以是从用户接收各种输入并向用户提供各种输出的设备，并且可包括触摸屏、键盘、遥控器、鼠标、打印机、音频和 / 或话音输入 / 输出设备等等。

[0106] 存储器 1104 可包括根据本文所讨论的实现的用于计算设备 1100 的模块和组件。在所例示的示例中，存储器 1104 可包括控制器模块 1116，控制器模块 1116 可包括分配模块 1118、维护模块 1120 以及更新模块 1122。具体来说，该示例计算设备 1100 是先前所述的类型，它运行控制器功能性作为分布式计算服务 102 机器（如 102-1、102-2、102-N、或机器 214-236）的至少某一些上运行的分布式服务。如此，该示例计算设备 1100 包括用于

执行上面讨论的分配机制、维护机制和更新机制的功能的模块（即，它分别包括分配模块 1118、维护模块 1120 和更新模块 1122）。存储器 1104 可进一步包括服务 A 模块 1124、服务 B 模块 1126 以及一个或多个操作系统 1128。服务 A 模块 1124 和服务 B 模块 1126 是由机器 214-236 执行来提供服务 A 和服务 B 的模块。这些模块可被存储在计算设备 1100 上以允许计算设备执行服务的实例。在其他实现中，在向机器分配了一实例之后，这些模块可被添加到机器。一个或多个操作系统 1128 可被包括来向例如服务 A 模块 1124 和服务 B 模块 1126 提供操作系统支持。如先前所述，尽管本文所讨论的实现讨论了两个服务被主持（服务 A 和服务 B），但是应理解各实现不限于此，且可由分布式计算服务 102 为许多服务所有者 104 同时主持多得多的服务。

[0107] 存储器 1104 可进一步包括一个或多个其他模块 1130，如驱动器、应用软件、通信软件、其他服务模块等等。存储器 1104 还可包括其他数据 1132，如在执行上述功能的同时存储的数据以及其他模块 1130 所使用的数据。存储器 1104 还可包括本文描述或提及的其他数据和数据结构。

[0108] 本文所描述的示例系统和计算设备仅是适用于某些实现的示例，并且不旨在对可实现本文所描述的过程、组件和特征的环境、架构和框架的使用范围或功能性范围提出任何限制。因此，本文的实现可用于众多环境或架构，并且可以在通用或专用计算系统或具有处理能力的其他设备中实现。一般而言，参考附图描述的任何功能都可使用软件、硬件（例如，固定逻辑电路）或这些实现的组合来实现。本文所使用的术语“模块”、“机制”、或“组件”一般表示可被配置成实现规定功能的软件、硬件或软件和硬件的组合。例如，在软件实现的情况下，术语“模块”、“机制”或“组件”可表示当在一个或多个处理设备（例如，CPU 或处理器）上执行时执行指定任务或操作的程序代码（和 / 或声明型指令）。程序代码可被存储在一个或多个计算机可读存储器设备或其他计算机存储设备中。由此，本文所描述的过程、组件和模块可由计算机程序产品来实现。

[0109] 虽然在图 11 中被例示为存储在计算设备 1100 的存储器 1104 中，但模块 1116-1126 或其各部分可以使用可由计算设备 1100 访问的任何形式的计算机可读介质来实现。如本文所使用的，“计算机可读介质”包括至少两种类型的计算机可读介质，即计算机存储介质和通信介质。

[0110] 计算机存储介质包括以存储如计算机可读指令、数据结构、程序模块或其他数据等信息的任何方法或技术实现的易失性和非易失性、可移动和不可移动介质。计算机存储介质包括但不限于：RAM、ROM、EEPROM、闪存或其他存储器技术、CD-ROM、数字多功能盘（DVD）或其他光存储、磁带盒、磁带、磁盘存储或其他磁存储设备，或者可用于存储信息以供计算设备访问的任何其他非传输介质。

[0111] 相反，通信介质可在诸如载波之类的已调制数据信号或其他传输机制中体现计算机可读指令、数据结构、程序模块或其他数据。如本文所定义的，计算机存储介质不包括通信介质。

[0112] 此外，尽管各模块在图 11 中被例示为存储在计算设备 1100 的存储器 1104 中，但是在其他实现中，模块或其部分可被实现为专用集成电路（ACIS）或其他形式的专用计算设备，以及与计算设备 1100 的其他硬件和软件组件集成。

[0113] 此外，本公开提供了如在附图中描述和例示出的各种示例实现。然而，本公开不限

于此处所描述并例示出的实现，而可扩展到其他实现，如本领域技术人员已知或将变得已知的那样。说明书中对“一个实现”、“该实现”、“这些实现”或“某些实现”的引用意味着所描述的特定特征、结构或特性被包括在至少一个实现中，并且这些短语在说明书各处的出现不一定都指代同一实现。

[0114] 结语

[0115] 尽管用结构特征和 / 或方法动作专用的语言描述了该主题，但所附权利要求书中定义的主题不限于上述具体特征或动作。更确切而言，上述具体特征和动作是作为实现权利要求的示例形式而被公开的。本公开旨在覆盖所公开的实现的任一和所有改编或变型，并且所附权利要求书不应被解释为限于说明书中所公开的具体实现。相反，本文档的范围完全由所附权利要求书以及这些权利要求所拥有的等效技术方案的完整范围来确定。

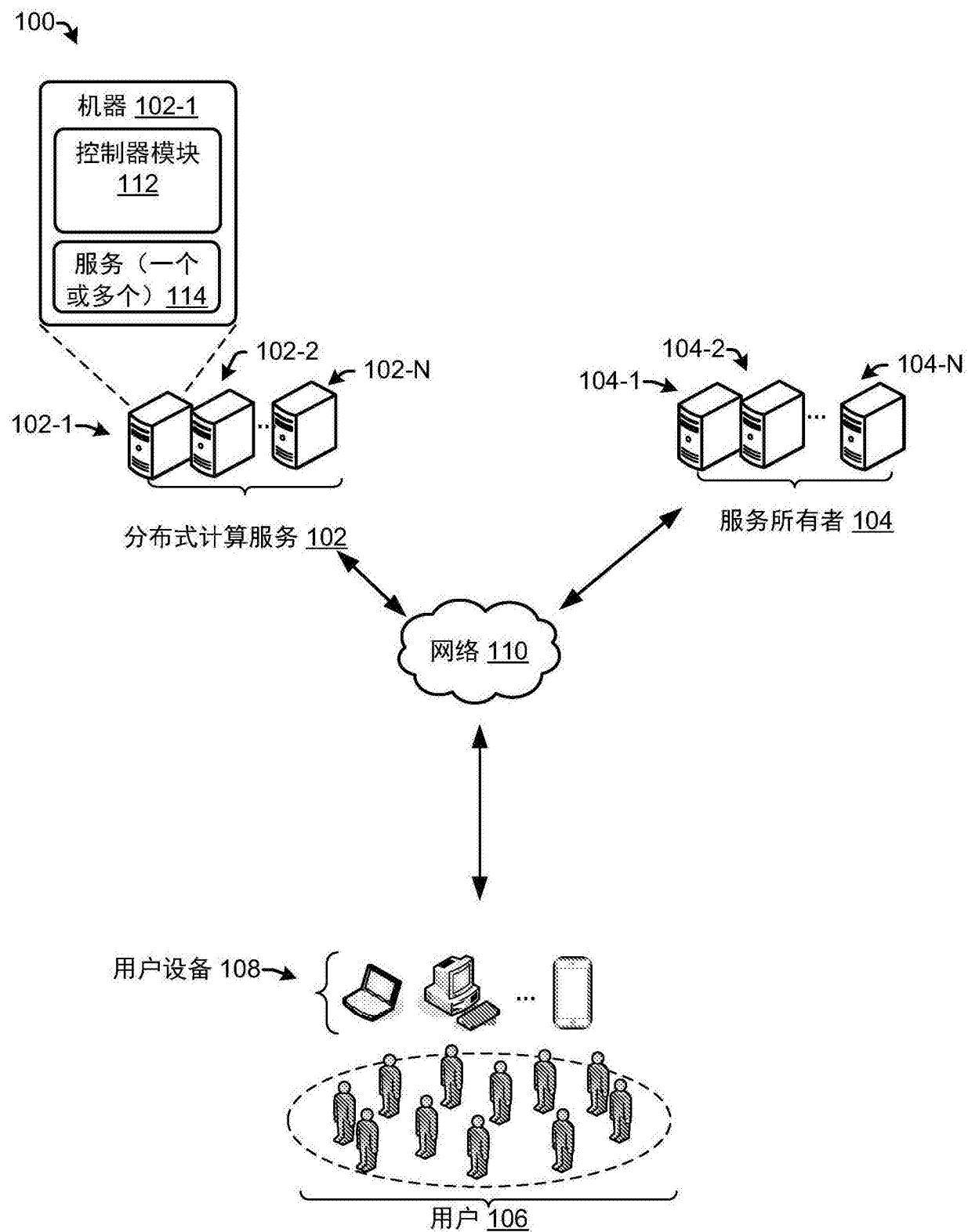


图 1

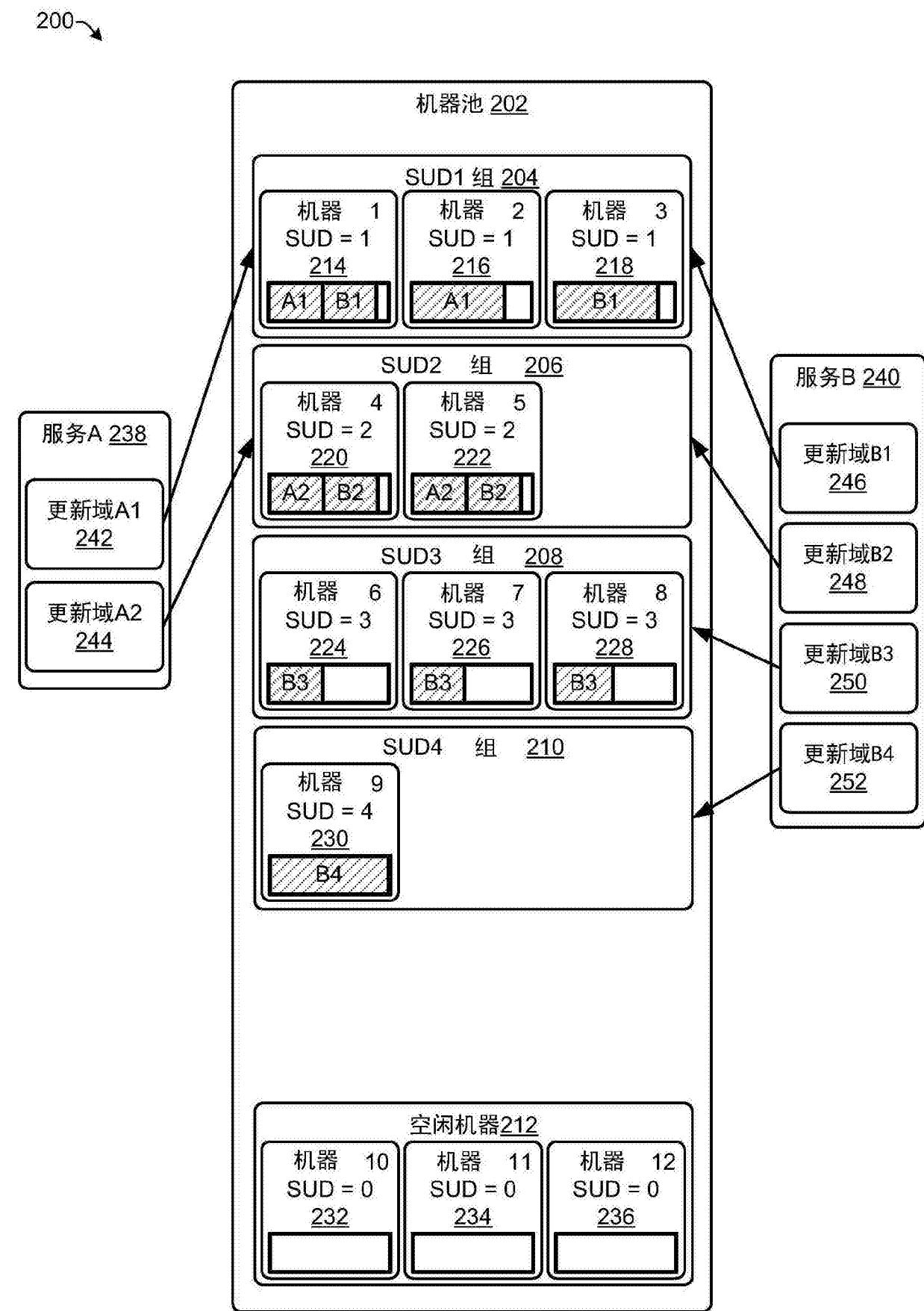


图 2

300 ↘

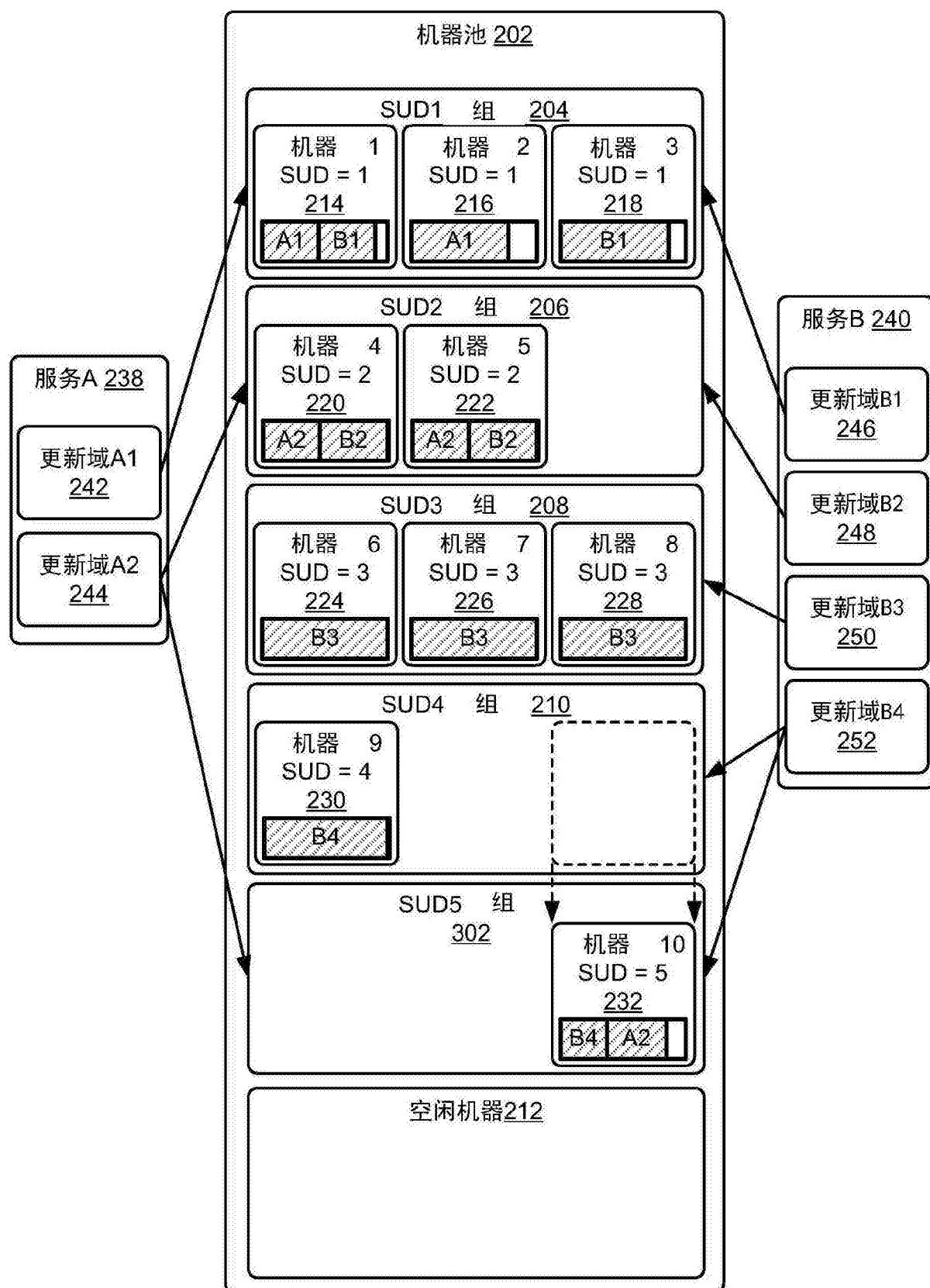


图 3

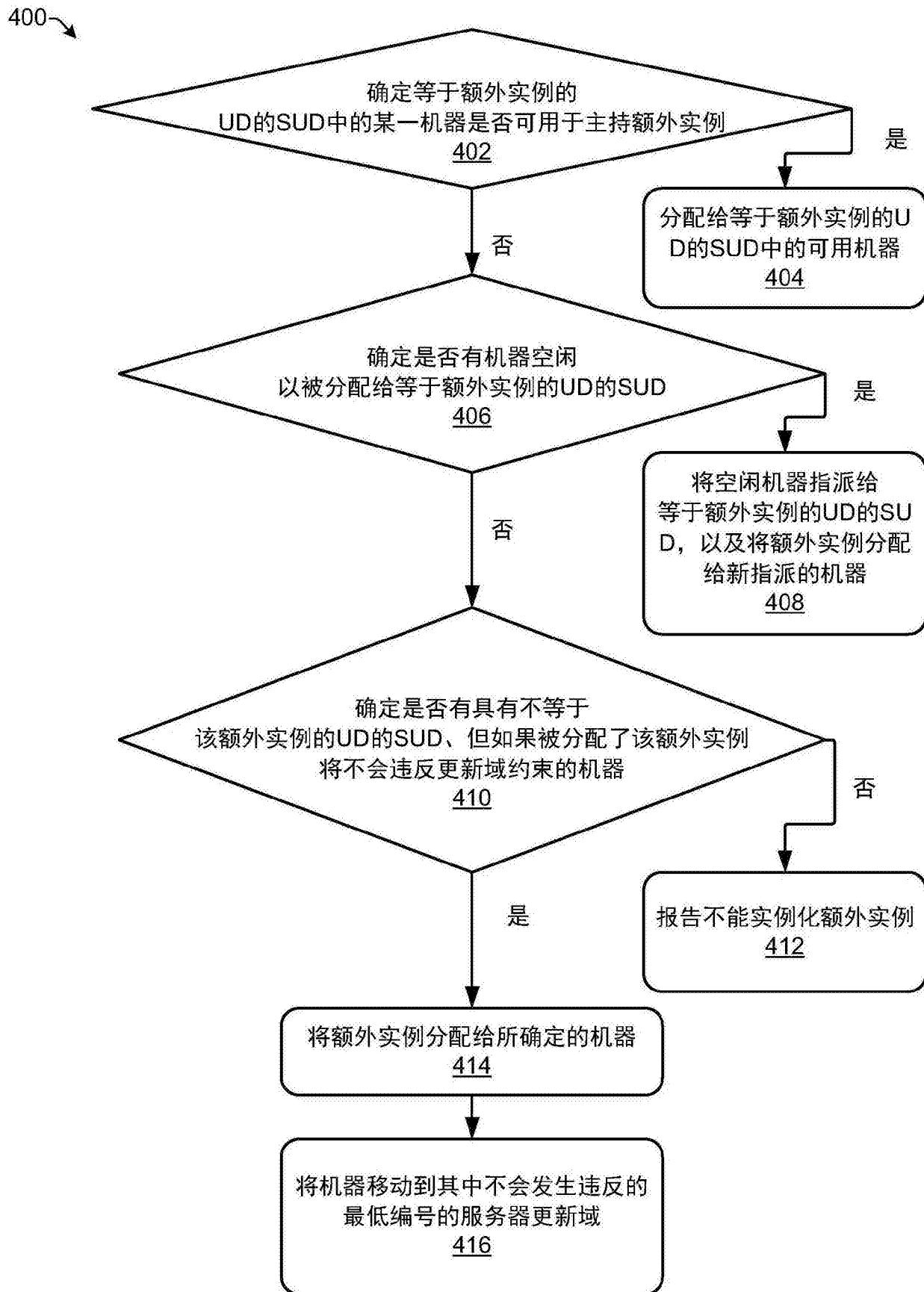


图 4

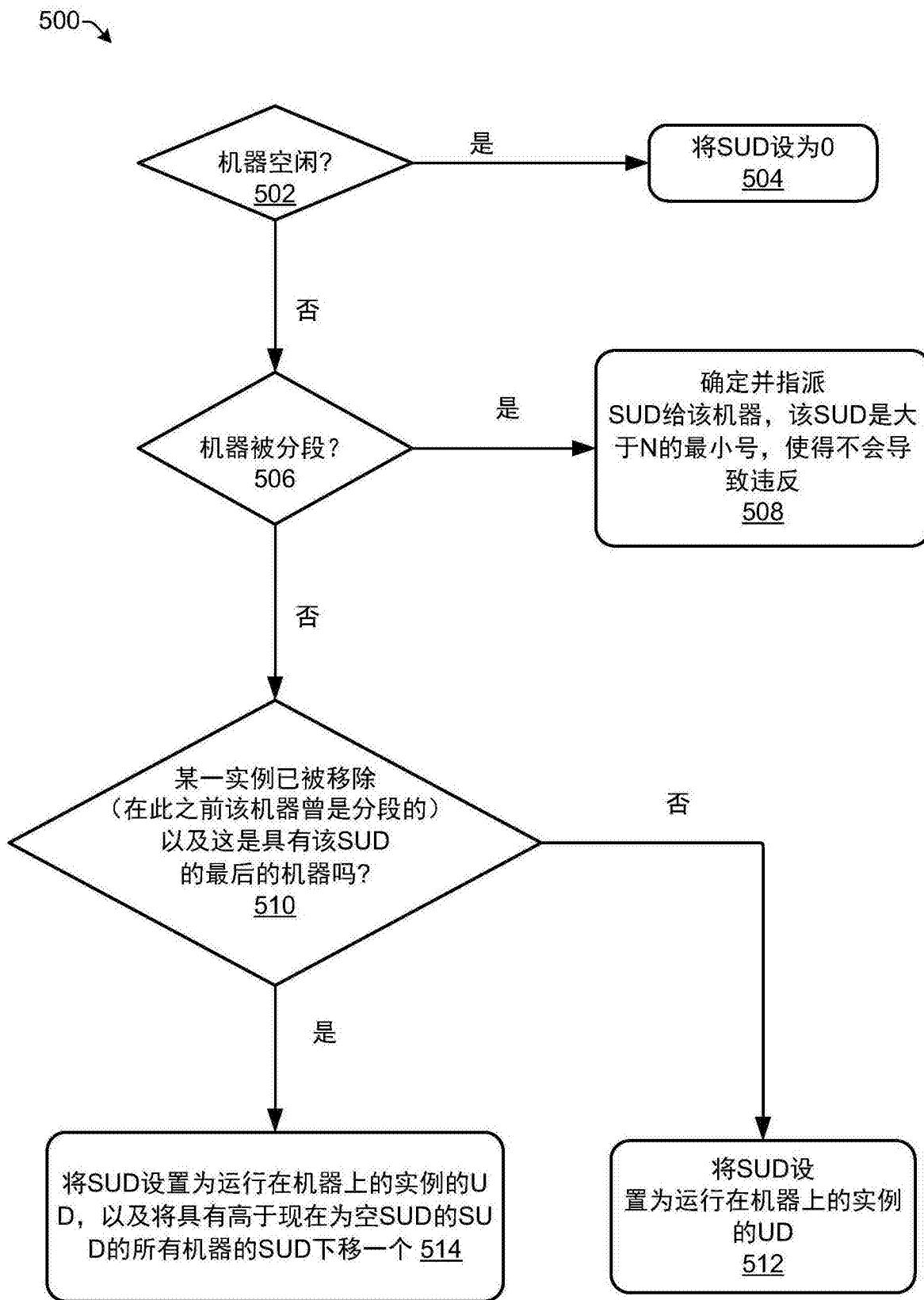


图 5

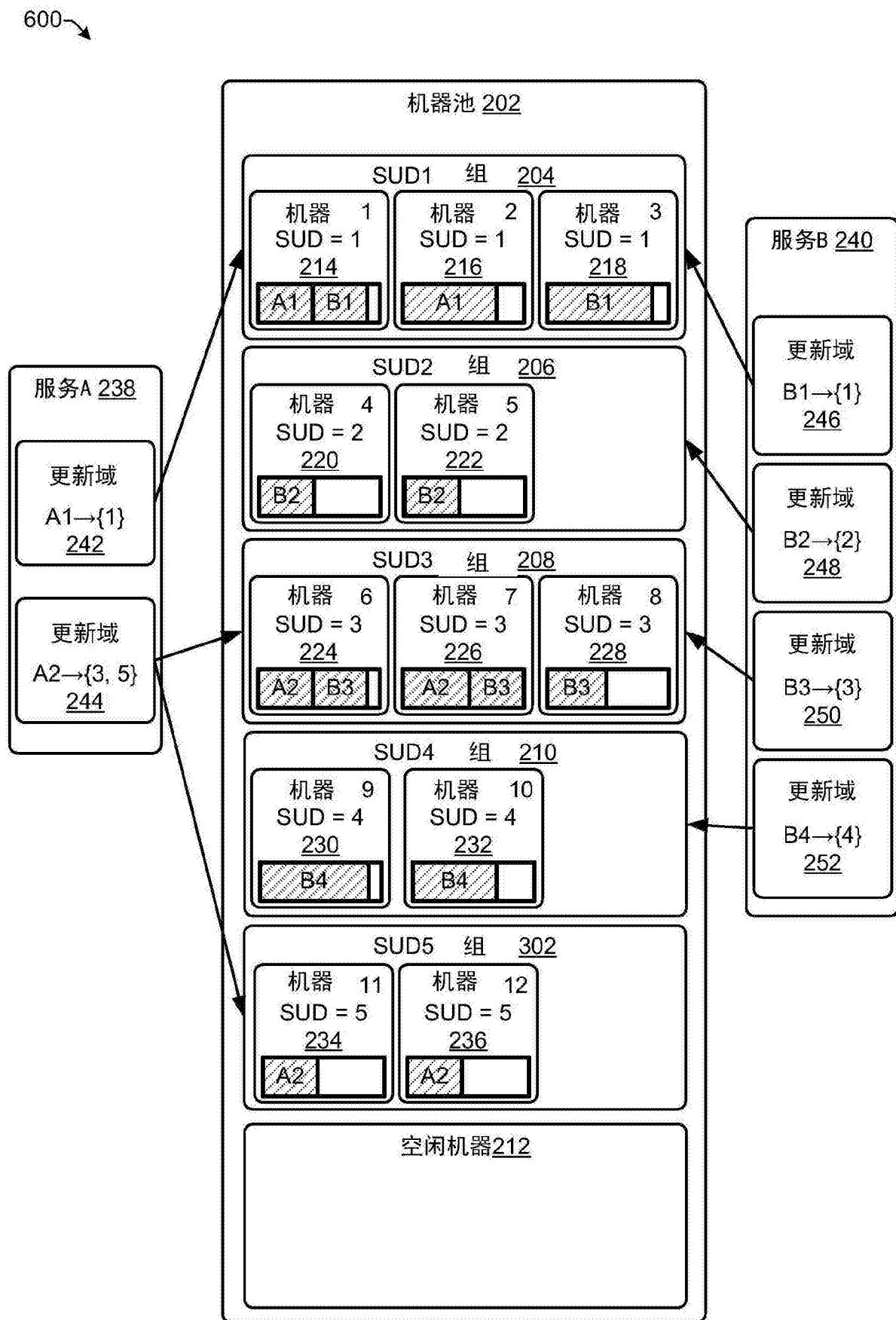


图 6

700 ↘

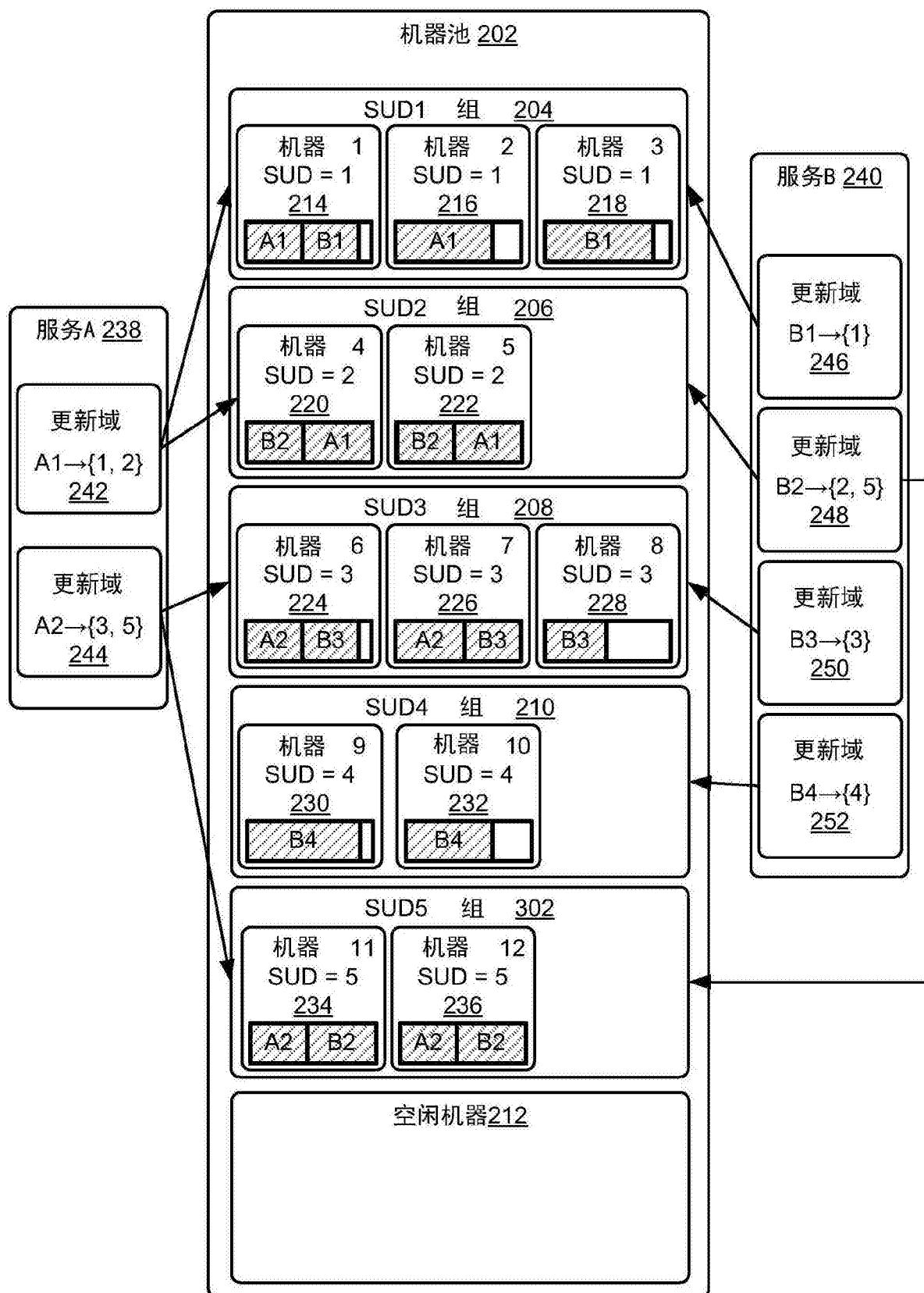


图 7

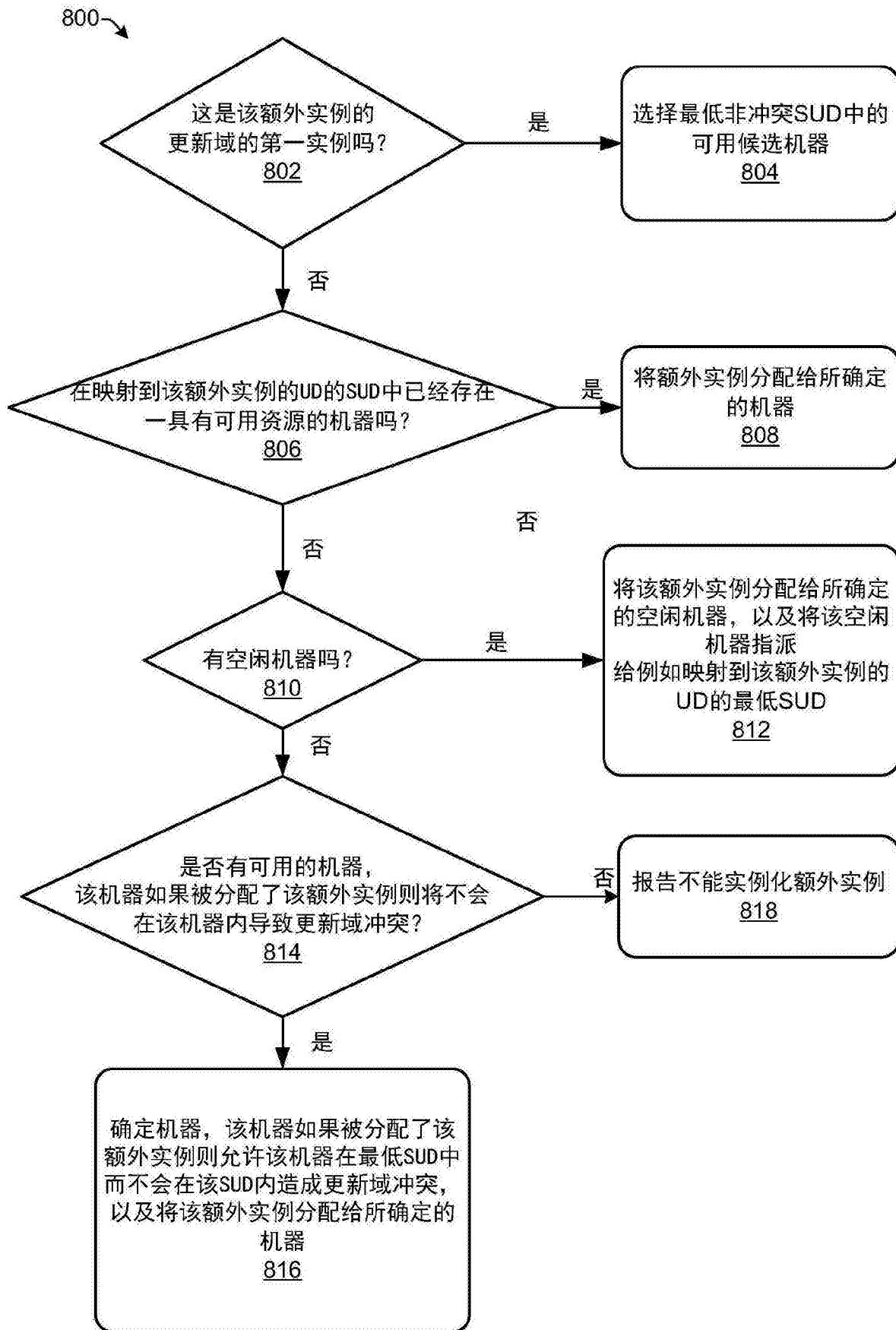


图 8

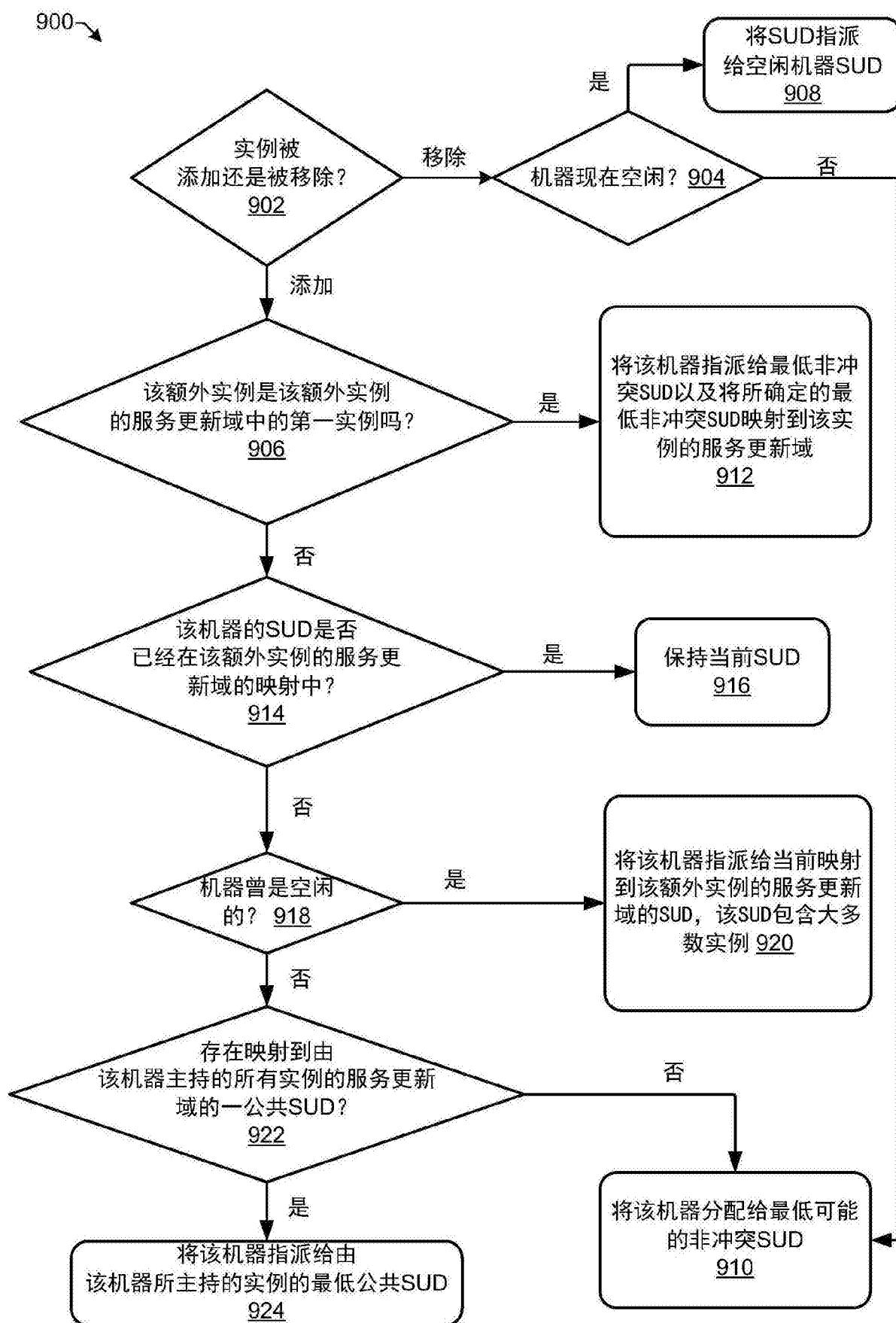


图 9

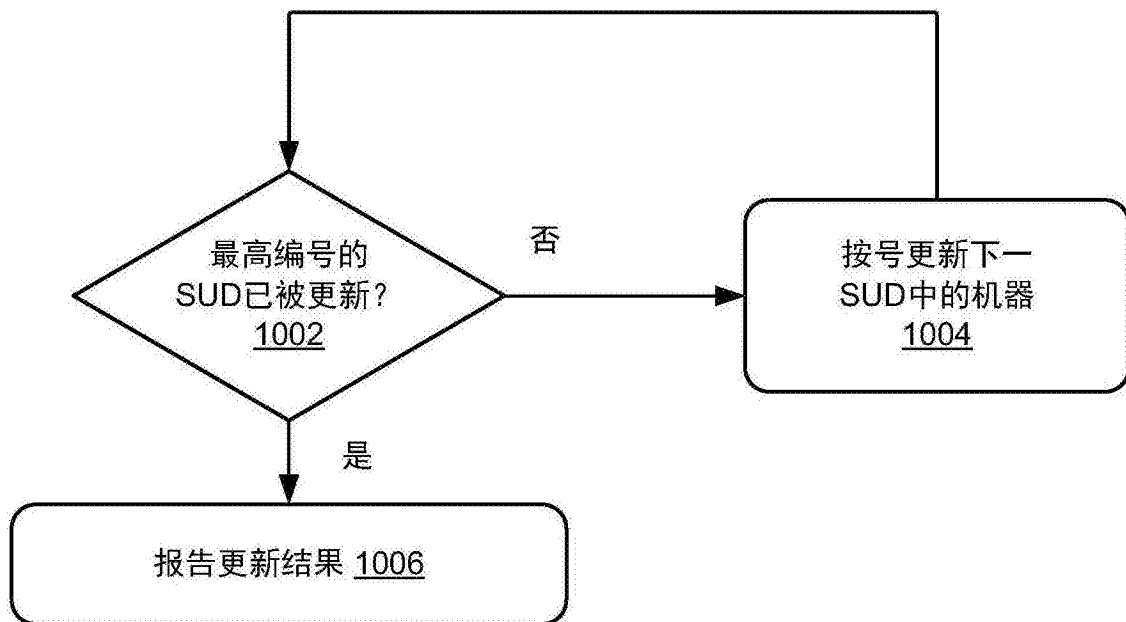


图 10

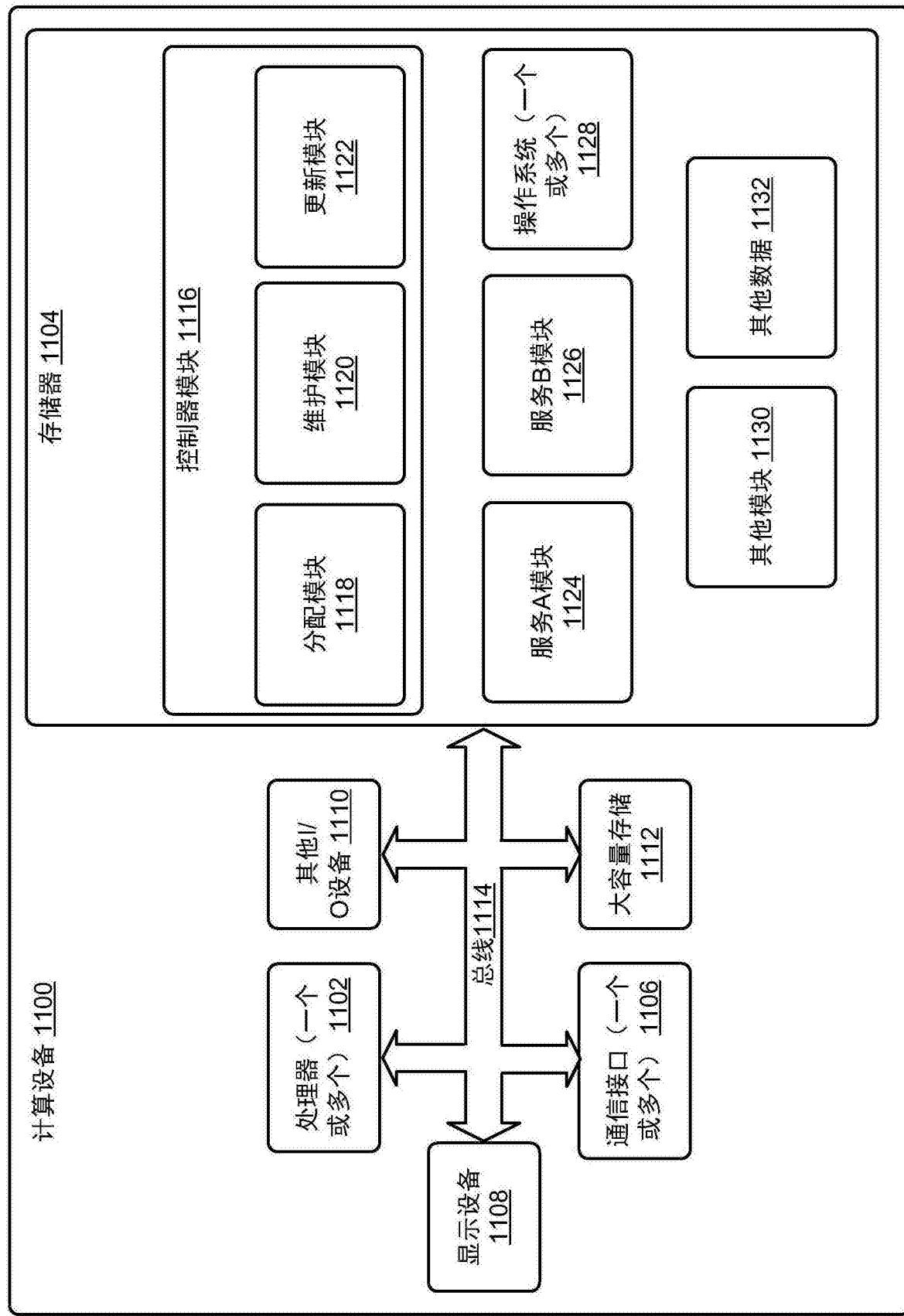


图 11