(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 101122872 B
(45)授权公告日 2010.06.02

(21)申请号 200710128640.0

(22)申请日 2007.07.09

(30)优先权数据
11/462,843 2006.08.07 US

(73)专利权人 国际商业机器公司
地址 美国纽约

(72)发明人 格里特·休伊曾加

(74)专利代理机构 中国国际贸易促进委员会专利商标事务所 11038
代理人 党建华

(51)Int.Cl.
G06F 9/50(2006.01)

(56)对比文件
US 20030061262 A1,2003.03.27, 全文.
US 20020053011 A1,2002.05.02, 说明书第0012段,00013段,0025段,0027段,0033-0035段,0047-0059段、图1,2,4A,4B.

审查员 程琼

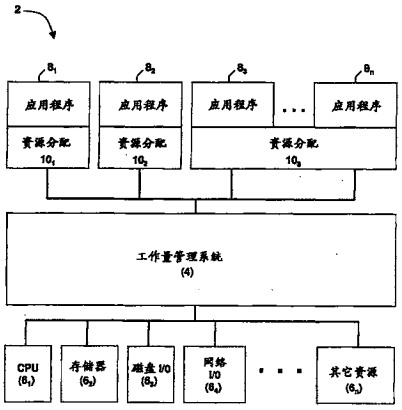
权利要求书 2 页 说明书 10 页 附图 6 页

(54)发明名称

用于管理应用程序工作量的方法和系统

(57)摘要

本发明公开一种涉及管理数据处理系统中的应用程序工作量的方法,包括:建立数据处理资源的保留资源分配,以供具有利用该资源的需求的数据处理应用程序使用;共享部分资源,该部分资源不是保留资源分配的部分,或相反由所述应用程序与其它应用程序利用;当应用程序要求增加其对资源的利用时,允许该应用程序消耗保留资源分配;和,当该应用程序消耗保留资源分配时,从可共享的一部分资源处补充保留资源分配;优点是,为减少应用程序延迟而不使其它应用程序丧失使用资源的能力,可以有效地管理保留资源分配,以使数据处理资源对该应用程序是可用的。



CN 101122872 B

1. 一种用于管理数据处理系统中的应用程序工作量的方法，包括：

建立数据处理资源的保留资源分配，以供具有利用所述数据处理资源的需求的数据处理应用程序使用，并且其它应用程序不能利用该保留资源分配；

共享部分的所述数据处理资源，该部分的所述数据处理资源不是所述保留资源分配的部分，并可由所述应用程序以及其它应用程序利用；

当所述应用程序要求增加其对所述数据处理资源的利用时，允许所述应用程序消耗所述保留资源分配；和

当所述应用程序消耗所述保留资源分配时，从所述可共享的一部分所述数据处理资源处补充所述保留资源分配。

2. 如权利要求 1 所述的方法，其中，通过从所述其它应用程序处收回数据处理资源来补充所述保留资源分配。

3. 如权利要求 1 所述的方法，其中，所述保留资源分配的尺寸是固定的。

4. 如权利要求 1 所述的方法，其中，所述保留资源分配的尺寸是可变的，并且，所述尺寸是根据所述应用程序对所述数据处理资源的利用水平而变动的。

5. 如权利要求 1 所述的方法，其中，根据预定的时间量进度表来管理所述保留资源分配。

6. 如权利要求 1 所述的方法，其中，根据所述应用程序的运行状态来管理所述保留资源分配。

7. 如权利要求 1 所述的方法，其中，根据规定的策略来管理所述保留资源分配。

8. 如权利要求 1 所述的方法，其中，使用自治的自调节技术来管理所述保留资源分配。

9. 如权利要求 1 所述的方法，还包括向所述应用程序指派最小资源分配和最大资源分配。

10. 一种用于管理数据处理系统中的应用程序工作量的系统，包括：

用于建立所述数据处理资源的保留资源分配，以供具有利用所述数据处理资源的需求的数据处理应用程序使用的装置，其中其它应用程序不能利用该保留资源分配；

用于共享部分的所述数据处理资源的装置，该部分的所述数据处理资源不是所述保留资源分配的部分，并由所述应用程序以及其它应用程序利用；

用于当所述应用程序要求增加其对所述数据处理资源的利用时，允许所述应用程序消耗所述保留资源分配的装置；和

用于当所述应用程序消耗所述保留资源分配时，从所述可共享的一部分所述数据处理资源处补充所述保留资源分配的装置。

11. 如权利要求 10 所述的系统，其中，通过从所述其它应用程序处收回数据处理资源来补充所述保留资源分配。

12. 如权利要求 10 所述的系统，其中，所述保留资源分配的尺寸是固定的。

13. 如权利要求 10 所述的系统，其中，所述保留资源分配的尺寸是可变的，并且，所述系统适于根据所述应用程序对所述数据处理资源的利用水平来变动所述尺寸。

14. 如权利要求 10 所述的系统，其中，根据预定的时间量进度表来管理所述保留资源分配。

15. 如权利要求 10 所述的系统，其中，根据所述应用程序的运行状态来管理所述保留

资源分配。

16. 如权利要求 10 所述的系统，其中，根据在所述数据处理系统中规定的策略来管理所述保留资源分配。

17. 如权利要求 10 所述的系统，其中，使用自治的自调节技术来管理所述保留资源分配，该自治的自调节技术由在所述数据处理系统中的遗传调节引擎实现。

18. 如权利要求 10 所述的系统，其中，所述系统适于为所述应用程序建立最小资源分配和最大资源分配。

19. 一种在数据处理平台中用以支持应用程序工作量管理的方法，包括：

提供用于管理数据处理资源的界面，以使它可被分配，以供在数据处理系统中执行的数据处理应用程序使用；

经由所述界面接收第一资源分配参数，其指定对数据处理应用程序可用的且可与其它应用程序共享的资源池；

经由所述界面接收第二资源分配参数，其指定所述资源池中的所述资源的最小分配，该最小分配将专用于所述应用程序且不与所述其它应用程序共享；

经由所述界面接收第三资源分配参数，其指定所述资源池中的保留资源分配，以供所述应用程序使用；

使用所述第三资源分配参数来确定所述保留资源分配的尺寸，该尺寸可以是固定的，也可以根据所述应用程序对所述资源的利用水平而变动；

共享一部分所述数据处理资源，该部分不是所述保留资源分配的一部分，由所述应用程序以及其它应用程序利用；

当所述应用程序要求增加其对所述数据处理资源的利用时，允许所述应用程序消耗所述保留资源分配；和

当所述应用程序消耗所述保留资源分配时，从所述可共享的一部分所述数据处理资源处补充所述保留资源分配。

20. 如权利要求 19 所述的方法，还适于对数据处理平台编程，以根据规定的策略或自治的自调节技术之一来管理所述保留资源分配。

21. 一种在数据处理平台中用以支持应用程序工作量管理的方法，包括：

定义数据处理资源池，该资源池对利用所述资源的数据处理应用程序是可用的，并且可与其它应用程序共享；

在所述资源池中建立所述资源的保留分配，以供所述应用程序使用，并且其它应用程序不能利用该保留资源分配；

共享部分的所述数据处理资源，该部分的所述数据处理资源不是所述保留资源分配的部分，并可由所述应用程序以及其它应用程序利用；

当所述应用程序要求增加其对所述数据处理资源的利用时，允许所述应用程序消耗所述保留资源分配；

在改变所述应用程序的资源利用期间，将所述保留资源分配动态地保持在所要求的尺寸。

用于管理应用程序工作量的方法和系统

技术领域

[0001] 本发明涉及单个处理器或共享的多处理器的计算环境中的应用程序工作量管理。更特殊地，本发明关注对在并发执行的工作量中所共享的数据处理资源进行分配的技术。

背景技术

[0002] 作为背景技术，单个处理器或共享的多处理器的系统中的工作量管理涉及多个应用程序中的数据处理资源的智能共享，而没有任何一个应用程序支配资源的使用，并因此降低同等应用程序的性能。可以以此方式管理的示例资源包括 CPU 时间、存储器、磁盘 I/O 带宽以及网络带宽。在某些工作量管理系统中，（典型地由系统管理员）确定所有将要利用一特定资源的应用程序所需要的该特定资源的数量。接着，分配该资源至不同的专用资源池，每一个资源池都代表该资源的一部分。指派由一个应用程序或由一系列相关的应用程序专用的每一个专用资源池。为方便起见，术语“应用程序”在下文中应理解为表示一个或多个应用程序。

[0003] 固定的分配方案保证应用程序响应时间的可预测性，因为每一个专用资源池对其应用程序来说总是可用的。然而，应用程序的实际的资源需求会经常少于由专用资源池所提供的量。因此，资源的利用并非是最佳的，因为其中的一部分可能是专用的而未被使用。为纠正这个问题，一些工作量管理系统允许在竞争的应用程序中共享资源。代替被指派一固定的资源分配，在非专用的资源池内部给予应用程序最小资源保证，该非专用的资源池在所有的应用程序间是另外可共享的。这个工作量管理的方法的目的是通过下列保证来提升资源分配的公平性：一方面，保证应用程序的资源利用水平不降到最小保证的水平之下，而另一方面，保证允许与其它应用程序共享资源的不分配部分，其中，该其它应用程序可以具有超过其自身的最小保证的临时需求。也可以制定最大资源分配，以规定将使应用程序得到保证的资源的最大利用。在某些情况下，最大资源分配是固定的。在其它情况下，它仅仅代表对应用程序可用的资源的总数与由其它应用程序使用的实际资源的总数之间的差别。

[0004] 资源共享方案的缺点是，最初给予应用程序的资源保证可能少于资源使用的高峰期的实际需求。应用程序可以获得更多所需资源的唯一办法是变成与其它应用程序共享的被指派的资源池的一部分。这意味着，在资源使用中的尖峰期，应用程序响应性可能是不可预计的。特别地，如果 100％地利用了共享的资源池，并且应用程序具有增加其对共享池的使用的许可和要求，则在可以将另外的分配给予提出请求的应用程序之前，负责资源分配的软件必须从其它应用程序处收回资源。因此，从共享池向提出请求的应用程序分配另外的资源可能要花费时间。举例说来，如果该资源是存储器，则应用程序将要经受延迟，在延迟中，数据传输至磁盘而存储器被释放或相反地被收回。资源分配中的这样的延迟会影响提出请求的应用程序的性能和响应时间。

[0005] 因此，尽管一些工作量管理解决方案由于创建超尺寸的专用资源池而浪费了资源，但其它的解决方案却因为对共享的资源池分配不足以及当获得多个资源时引入延迟而

影响了应用程序的性能。本发明关注于后一类型的应用程序工作量管理的改进，以便避免资源共享中所固有的问题。特别地，需要的是一种特殊技术，凭借该技术可以共享数据处理资源，而不必使可能对该共享资源具有增长的即刻需求的应用程序处于不利。

发明内容

[0006] 通过用于管理数据处理系统中的应用程序工作量的新方法、系统和计算机程序产品，解决前述问题并获得在此领域中的进步，其中，使用了数据处理资源的可动态保持的保留资源分配的概念。本发明技术包括建立由具有资源利用需求的数据处理应用程序所使用的保留资源分配、共享部分资源（该资源不是保留资源分配的一部分，或相反由该应用程序与其它应用程序所利用）、在应用程序要求增加它的资源利用的时候允许该应用程序消耗保留资源分配、以及当应用程序消耗保留资源分配时从资源的可共享的部分处补充保留资源分配。优点是，为了减少应用程序延迟而不使其它应用程序丧失使用资源的能力，可以有效地管理保留资源分配，使得数据处理资源对应用程序是可用的。

[0007] 根据示例实施例，保留资源分配可以是固定尺寸或可变尺寸的。如果是后者，则保留资源分配的尺寸可以根据应用程序对资源的利用的水平而变化。如果需要的话，可以根据预定的时间量进度表来管理保留资源分配。或者，可以根据应用程序的运行状态来管理保留资源分配。可以根据规定的策略或使用自治的自调节技术来执行对保留资源分配的管理。除了保留资源分配，应用程序还可以被指派最小资源分配和最大资源分配。

[0008] 根据本发明的一个方面，实现了一种用于管理数据处理系统中的应用程序工作量的方法，包括：建立数据处理资源的保留资源分配，以供具有利用所述数据处理资源的需求的数据处理应用程序使用，并且其它应用程序不能利用该保留资源分配；共享部分的所述数据处理资源，该部分的所述数据处理资源不是所述保留资源分配的部分，并可由所述应用程序以及其它应用程序利用；当所述应用程序要求增加其对所述数据处理资源的利用时，允许所述应用程序消耗所述保留资源分配；和当所述应用程序消耗所述保留资源分配时，从所述可共享的一部分所述数据处理资源处补充所述保留资源分配。

[0009] 根据本发明的另一方面，实现了一种用于管理数据处理系统中的应用程序工作量的系统，包括：用于建立所述数据处理资源的保留资源分配，以供具有利用所述数据处理资源的需求的数据处理应用程序使用的装置，其中其它应用程序不能利用该保留资源分配；用于共享部分的所述数据处理资源的装置，该部分的所述数据处理资源不是所述保留资源分配的部分，并由所述应用程序以及其它应用程序利用；用于当所述应用程序要求增加其对所述数据处理资源的利用时，允许所述应用程序消耗所述保留资源分配的装置；和用于当所述应用程序消耗所述保留资源分配时，从所述可共享的一部分所述数据处理资源处补充所述保留资源分配的装置。

[0010] 根据本发明的另一方面，实现了一种在数据处理平台中用以支持应用程序工作量管理的方法，包括：提供用于管理数据处理资源的界面，以使它可被分配，以供在数据处理系统中执行的数据处理应用程序使用；经由所述界面接收第一资源分配参数，其指定对数据处理应用程序可用的且可与其它应用程序共享的资源池；经由所述界面接收第二资源分配参数，其指定所述资源池中的所述资源的最小分配，该最小分配将专用于所述应用程序且不与所述其它应用程序共享；经由所述界面接收第三资源分配参数，其指定所述资源池

中的保留资源分配，以供所述应用程序使用；使用所述第三资源分配参数来确定所述保留资源分配的尺寸，该尺寸可以是固定的，也可以根据所述应用程序对所述资源的利用水平而变动；共享一部分所述数据处理资源，该部分不是所述保留资源分配的一部分，由所述应用程序以及其它应用程序利用；当所述应用程序要求增加其对所述数据处理资源的利用时，允许所述应用程序消耗所述保留资源分配；和当所述应用程序消耗所述保留资源分配时，从所述可共享的一部分所述数据处理资源处补充所述保留资源分配。

[0011] 根据本发明的另一方面，实现了一种在数据处理平台中用以支持应用程序工作量管理的方法，包括：定义数据处理资源池，该资源池对利用所述资源的数据处理应用程序是可用的，并且可与其它应用程序共享；在所述资源池中建立所述资源的保留分配，以供所述应用程序使用，并且其它应用程序不能利用该保留资源分配；共享部分的所述数据处理资源，该部分的所述数据处理资源不是所述保留资源分配的部分，并可由所述应用程序以及其它应用程序利用；当所述应用程序要求增加其对所述数据处理资源的利用时，允许所述应用程序消耗所述保留资源分配；在改变所述应用程序的资源利用期间，将所述保留资源分配动态地保持在所要求的尺寸。

附图说明

[0012] 如附图所示，从下面对本发明的示例实施例的更为详细的描述中，将可以清楚看到本发明的前述的和其它的特征及优点，附图中：

[0013] 图 1 是示出根据本发明适于执行应用程序工作量管理的数据处理系统的功能框图；

[0014] 图 2 是示出当要求收回资源时与分配给应用程序的资源的增长相关的基本延迟的图形表示；

[0015] 图 3 是示出根据本发明当提供保留资源分配时与分配给应用程序的资源的增长相关的减少的延迟的图形表示；

[0016] 图 4 是示出当应用程序对资源的实际利用随时间变化时对保留资源分配的维护的图形表示；

[0017] 图 5 是示出在消耗保留资源分配的应用程序的实际资源利用增加之后，经由资源收回对保留资源分配进行维护的图形表示；

[0018] 图 6 是用于应用程序的一系列资源分配的图形表示，该一系列资源分配包括根据本发明的保留资源分配；

[0019] 图 7A 是示出根据本发明工作量管理系统的第一示例实施例的方块图；

[0020] 图 7B 是示出图 7A 的第一示例实施例的修改的方块图；

[0021] 图 8A 是示出根据本发明的工作量管理系统的第二示例实施例的方块图；

[0022] 图 8B 是示出图 8A 的第二实施例的修改的方块图；和

[0023] 图 9 是示出根据本发明可用于提供计算机程序产品以实现工作量管理系统的物理介质的示意图。

具体实施方式

[0024] 现在来参看附图，在所有这些视图中，相似的标号代表相似的组件，图 1 示出装有

工作量管理系统 4 的示例数据处理系统 2。数据处理系统 2 可以是单或多处理器（如 SMP 或 NUMA）机器、机器集群或任何其它的所需计算机体系结构的实现。工作量管理系统 4 管理一系列的数据处理资源 6_1-6_n，以供多个应用程序 8_1-8_n 使用，其中数字“n”对两者都是任意的。工作量管理系统 4 的工作是通过创建一系列的资源分配 10 而在应用程序 8_1-8_n 中共享资源 6_1-6_n，资源分配 10 中的每一个都代表全部的可用资源 6_1-6_n 的某部分。可以给予资源分配 10 单个的应用程序或一系列应用程序。为示例说明，应用程序 8_1 和 8_2 被显示为具有各自的资源分配 10_1 和 10_2，但是，应用程序 8_3-8_n 具有共同的资源分配 10_3，资源分配 10_3 作为一个组被指派给该一系列的应用程序。应用程序 8_1-8_n 中的每一个都代表传统形式的执行环境，例如处理、任务、线程等等，其运行由数据处理系统 2 管理。

[0025] 可以使用传统的工作量管理软件来实现工作量管理系统 4，其中传统的工作量管理软件已经被修改以提供本发明的增强的工作量管理特征。这样的传统的工作量管理解决方案包括**IBM® ALX®** 5L 工作量管理器以及**Linux®** CKRM（基于类的内核资源管理器）框架。两个工作量管理系统都支持基于应用程序分组的操作系统资源分配，并且都包括允许由系统管理员手动控制的或经由工作量管理中间设备自动控制的用户界面（如，**Linux®** CKRM 的资源控制文件系统 (RCFS) 界面）。提供操作系统内核机制，该机制允许策略驱动的区分服务访问诸如 CPU 时间、存储页面、硬盘 I/O 和网络带宽之类的资源。举例说来，为每种分配资源类型实现资源控制器（调动程序），以监视资源使用和保证每一个应用程序接收其特定的资源共享。还提供报告机制，用于经由用户界面向系统管理员报告资源使用和其它性能信息。允许系统管理员为每一个应用程序提供资源份额。应用程序可以包括一组具有共同的性能目标和共同的资源要求的一个或多个任务（如，处理和线程等等）。可以使用任何适当的技术提供用户界面，包括系统或库调用、套接字、HAL（硬件抽象层）调用以及信息传递 API 等等。也可以使用资源控制文件系统的范例来提供用户界面。资源控制文件系统包括树结构的名空间，其目录代表应用程序。用户可以通过创建新的目录来创建新的应用程序资源分配。每一个目录包含虚拟文件，该虚拟文件指定应用程序将要使用的一个或多个资源的资源份额。这个文件可以包括多个参数，例如，用于每个资源的最小和最大资源分配的参数。经由用户界面，可以开发自动的工作量管理控制器，该工作量管理控制器自动地设置应用程序的资源份额，所依据的是它们工作的重要性。这样的自适应控制器可以把由工作量管理系统的报告机制所提供的资源使用信息作为产生资源指派策略的反馈来使用。用户界面也允许由系统管理员手动地创建资源分配策略。

[0026] 作为使用基于核的解决方案的替代，工作量管理系统 4 可以利用用户级的应用程序软件而实现，例如用户端口监控程序 (userdaemons)、共享库等等，该用户水平的应用程序软件利用传统的操作系统调用来修改和监视那些控制应用程序访问正被管理的资源的操作系统参数。在这个实现中，不需要提供内核资源管理系统，例如**IBM® AIX®** 5L 工作量管理器和**Linux®** CKRM 框架。可以使用标准的非自觉工作量管理操作系统 (non-workload management-aware operatingsystem)。另一种实现选择是把工作量管理系统 4 设计成为包括操作系统和用户级应用程序功能的混合系统。

[0027] 资源 6_1-6_n 可以代表可由应用程序 8_1-8_n 利用的任意资源。仅通过例子，图 1 示出了四个已熟知的资源，即 CPU 资源 6_1、存储器资源 6_2、磁盘 I/O 带宽资源 6_3 和网络带宽资源

6_4。也可以根据管理的需要和偏好来管理任何数量的其它资源（如，直到 6_n）。这些资源可以包括可以在应用程序 8_1-8_n 中共享的任意所需的硬件、固件或软件实体。CPU 资源 6_1 可以代表 CPU 时间、每个应用程序的时隙以及调度延迟（如，用于实时应用程序）等等。相对于具有较小的 CPU 资源分配的应用程序来说，具有大的 CPU 资源分配的应用程序将被安排更多的 CPU 时间，或将被分配更多的时隙，或将具有较少的调度延迟，等等；反之亦然。在工作量管理系统 4 中，可以通过改变一个应用程序相对于其它应用程序的调度优先级来控制 CPU 的利用。存储器资源 6_2 可以代表物理存储地址空间。相对于具有较小的存储分配的应用程序来说，具有大的存储资源分配的应用程序将被指派更多的物理存储；反之亦然。在工作量管理系统 4 中，可以通过改变分配给一个应用程序相对于其它应用程序的物理的或逻辑的页面数目来控制物理存储。磁盘 I/O 带宽资源 6_3 可以代表随时间的磁盘块传输。相对于具有较低的磁盘 I/O 带宽分配的应用程序来说，具有大的磁盘 I/O 带宽的资源分配的应用程序将使其磁盘 I/O 传输执行得更快；反之亦然。在工作量管理系统 4 中，可以例如通过调整预读缓存、缓冲、磁盘写出缓存和 / 或多路径 I/O 子系统中的可用路径的数目，来区分一个应用程序相对于其它应用程序的磁盘 I/O 请求的优先次序，从而控制磁盘 I/O 带宽。网络 I/O 带宽资源 6_4 可以代表随时间的网络吞吐量、延迟、优先级或其它网络连接速率的节流。相对于具有较低的网络 I/O 带宽分配的应用程序来说，具有大的网络 I/O 带宽的资源分配的应用程序将具有较高的网络 I/O 传输速率；反之亦然。在工作量管理系统 4 中，可以通过限制一个应用程序相对于其它应用程序的套接字的分配速率、限制一个应用程序相对于其它应用程序的由套接字处理连接的速率、管理一个应用程序相对于其它应用程序的 IP 优先级、调整一个应用程序相对于其它应用程序的 SCTP 包 / 拥塞控制来控制网络 I/O 吞吐量。

[0028] 如上述背景技术所述，传统的实现资源共享的工作量管理系统对于特定资源为应用程序指派最小和最大资源限制。可以与其它应用程序一起共享资源中超过最小资源分配（直到最大资源分配）的部分。这个方案的缺点是应用程序的实际的资源利用通常将是在最小资源分配与最大资源分配之间。无论何时资源要求开始超过最小资源分配，应用程序只能在资源要求增长期间，通过变成与其它应用程序共享的被指派的资源池的一部分，来增加其自身的资源分配。如果需要收回资源，例如，为释放另外的内存而将数据导入硬盘，则可能导致应用程序延迟。

[0029] 图 2 是示例。它显示在应用程序请求另外的资源分配的时候会出现的延迟，该另外的资源分配必须从其它应用程序处收回，因为该资源是可共享池的部分。通过引入保留资源分配的概念，可以解决这个问题。对任何特定的资源来说，为了接受应用程序资源使用中的尖峰期而没有明显的延迟，可以使得保留资源分配对应用程序可用。如图 3 所示，由于并不与其它应用程序共享保留资源分配，因此，相对于图 2 中所示的传统的延迟来说，与资源利用的增长相关的延迟大大减少了。特别地，与变成其自身的保留资源分配的应用程序相关的减少的延迟，应该比得上与应用程序对其自身的最小资源分配的利用相关的延迟。

[0030] 现在参照图 4，保留资源分配是两个分配组件之一，该两个分配组件位于应用程序的当前的资源利用（可能随时间而变化）与它的最大资源分配之间。其它组件是对其它应用程序可用的可共享分配。该可共享的分配代表特定资源的一部分，该特定资源还没有分配给拥有保留资源分配的应用程序，并且该特定资源不是保留资源分配的部分。这个资源

部分的全部，直到最大资源分配，对共享来说都是可用的。保留资源分配毫无疑问是不可共享的。它给予应用程序在没有明显延迟的条件下增加其资源利用的能力，并且，因此可被认为当必要时在保证的最小资源分配中提供递增。然而，与最小资源分配不同，保留资源分配可以在下层资源的利用水平（由应用程序和/或整个系统利用）的基础上以及在考虑应用程序需求和资源可用性的策略的基础上被动态地指派、去除、增加、下降或相反被保持。

[0031] 图 4 显示了示例应用程序的资源利用，其中 T0 与 T1 之间的时间段代表一周期，在该周期中具有尺寸 MIN 的保证的最小资源分配，但没有指派的保留资源分配。在 T1 之前未指派保留资源分配。在这点，保留资源分配的尺寸是资源利用水平 R1 与最小资源分配 MIN 之间的差 (R1-MIN)。在时间 T2，实际的资源利用增加至水平 R1。在耗尽所有保留的情况下，通过变成保留资源分配而获得另外的资源且没有明显的延迟。为了保持保留资源的容量，工作量管理系统 4 可以从资源池的可共享部分处补充保留资源分配，如果需要的话，从其它应用程序处收回所需数量的资源。

[0032] 又如图 5 中所示，这个资源补充过程动态地将保留资源分配的尺寸保持在资源利用水平 R2 与应用程序当前利用的资源水平 R1 之间的差。取决于主要的保留资源管理策略（见下面），时间 T2 之后的保留资源分配的尺寸可以小于、等于或大于 T2 之前的保留资源分配的尺寸。如果因为资源池的剩余部分实际上正被共享而不仅仅是可共享而要求收回资源，则该补充过程会导致从时间 T2 到 T2’的延迟（如图 5 中所示）。然而，通常这个延迟是可接受的，因为应用程序已经被保证它要求的资源增加，并且将不大可能立即要求另外的增加。这个方案的优点是工作量管理系统 4 可以启动资源补充，但要求该资源增加的应用程序将不被封锁，只要它的资源需求没有超过当前的保留资源水平。这样，只要保留资源分配始终被指派给应用程序，则该应用程序将总是具有增加它的资源利用而没有明显的延迟的能力。

[0033] 还如图 4 所示，实际资源利用中的第二次增加出现在时间 T3，而第三次增加出现在时间 T4。在这两种情况下，应用程序变成它的保留资源分配而没有明显的延迟。同时，由于消耗了保留资源分配，为补充保留资源，工作量管理系统 4 收回资源，从而将保留资源容量保持在所需水平。在时间 T3，把保留资源分配的尺寸增加至资源利用水平 R3 与 R2 之间的差。在时间 T4，把保留资源分配的尺寸增加至资源利用水平 R4 与 R3 之间的差。时间 T5 和 T6 代表应用程序减少它的资源利用的时刻。作为响应，工作量管理系统 4 可以把保留资源分配继续保持在所需的水平。在时间 T7，由工作量管理系统 4 把保留资源分配从应用程序中去除。

[0034] 在每一个时间周期 T1-T7，其它的应用程序将具有共享对应用程序可用的最大资源分配的剩余资源的能力。此外，只要保留资源分配没有被指派，如在时间 T0 和 T7，则位于最小资源分配之上且没有被应用程序正在使用的所有的最大资源分配对共享来说将都是可用的。这样，应用程序的可以没有明显的延迟而访问另外资源的要求与其它应用程序的为同一资源进行竞争的能力之间的动态平衡受到冲击。

[0035] 图 6 图示一方法，在该方法中，保留资源分配可以在逻辑上被描绘成为工作量管理系统 4 中的另外的工作量管理参数。图 6 中显示了用于图 1 的应用程序 8_1 的资源分配 10_1。对于每一个 CPU、存储器、磁盘 I/O 带宽、网络 I/O 带宽和“其它”资源来说，都有资源池，其最大尺寸是由最大资源分配参数 MAX 来指定的。最小资源分配参数 MIN 指定最小资

源分配的尺寸。可以由工作量管理系统 4 来另外规定保留资源分配参数 RES，以指定保留资源分配的尺寸。

[0036] 可以以各种方式使用 RES 参数来指定保留资源分配的尺寸，这取决于由保留资源管理策略所指定的保留资源管理需求。举例说来，RES 参数可以为一数值，其代表最大资源分配 MAX 与应用程序的当前资源利用之差的百分比（即，剩余资源容量的百分比）。因此，如果应用程序当前正使用资源池的总体的 20％，则余下的可用的和其它的可共享的资源空间的尺寸应为 80％。把保留资源参数 RES 设置为 50％将创建 80％可共享部分的 50％的保留资源，或者说资源池总体的 40％。因此，当保留资源分配生效时，资源池总体的 40％不能被其它应用程序共享。对共享来说，仅资源池剩下的 40％的非保留部分才是可用的。如果接着应用程序需要把它的实际资源利用从资源池总体的 20％提高至资源池总体的 60％，则可以通过变成全部的保留资源分配来迅速获得所要求的 40％的增量。此后，为通过把保留资源分配保持在的剩余容量的 50％的水平来满足 50％这个 RES 参数，将指派资源池总体的 20％（即，资源池的 40％的剩余的 50％）为新的保留资源分配。保留资源分配的尺寸因此将从资源池总体的 40％下降到池的 20％。对其它应用程序可用的可共享的部分也将下降到资源池总体的 20％。可以理解，随着应用程序向资源池总体的 100％增长，这个保持资源保留的方法将渐近地减少保留资源分配的尺寸。

[0037] 实现保留资源分配参数“RES”的另一个技术，将是使它规定固定量的资源，该固定量的资源必须对应用程序的增加的资源利用保持可用。这可以表示成资源池总体的百分比。因此，RES 参数值为 20％意味着，应该保留地保持资源池总体的 20％多于应用程序的当前的资源利用。举例说来，不考虑应用程序当前是否正利用资源池总体的 30％、50％或 70％，资源池总体的 20％将被作为保留资源分配的部分而保留地保持（假定池中留有足够的资源容量）。这个类型的 RES 参数也可以指定必须对应用程序保持可用的实际的物理资源单元。举例说来，用于图 1 的存储器资源 6_2 的保留资源分配可以是“x”兆字节的 RAM，其中 0 ＜ x ＜＝最大的可用存储。可以理解，至少直到达到上限（即，池中没有能够把保留资源分配保持在指定尺寸的足够的资源容量），这个保持资源保留的方法将一直保持保留资源分配的尺寸，而不考虑应用程序的实际的资源利用。

[0038] 实现保留资源分配参数“RES”的另一个技术将是把它表示为数学的阶梯函数，该阶梯函数确定保留资源分配的尺寸。该阶梯函数的输入可以包括应用程序的当前实际的资源利用、资源利用的增长速率、最小保证分配 MIN、最大分配 MAX 以及资源的可共享部分的总体可用性（即，并非正在被实际共享的部分）。举例说来，如果应用程序正在以飞快的速率增加它的资源利用，则保留资源分配可以指数地增长，直到应用程序达到了指定的资源利用水平或减缓了资源利用的增长速率。

[0039] 可以通过如下策略建立在任意给定的时间对应用程序可用的保留资源分配的实际尺寸，该策略在应用程序或作为整体的系统 2 的整个运行条件的基础上，反映用于管理保留资源分配的动态方案。举例说来，可以有适于为应用程序提供可靠的保留资源分配（如，由于高优先级的应用程序需求）的某些时间，以及应该减少甚至完全解除保留资源分配（如，当其它应得应用程序对资源有过分的争夺）的其它时间。

[0040] 有许多保留资源管理策略，可由工作量管理系统 4 使用以随时间调整保留资源分配。一个示例方案是将保留资源管理策略建立在同步的时间量进度表的基础上。举例说来，

在当预计出现资源使用峰值的 24 小时周期中的预定的时间，例如，一天两次，从上午 9:00 到上午 10:00 以及从下午 1:00 下午 5:00，可以由工作量管理系统 4 将保留资源分配指派给应用程序。注意，保留资源分配的尺寸可以取决于指派的时间而改变。举例说来，示例同步的时间量的方案可能是使用应用程序开启之后（例如系统启动后 20 分钟时开始）的应用程序的初始保留策略，之后当应用程序进入稳态模式时恢复至稳定状态保留策略。初始策略可能给予应用程序非常大的保留资源分配，该保留资源分配允许应用程序在初始过程迅速增长以加速系统开启。接着工作量管理系统 4 要回到稳定状态策略，该稳定状态策略将保留资源分配限定在小得多的数量。如果应用程序从稳定模式切换到备份模式，则保留资源分配可以被切换不同的数量，或被完全禁止等等。第三方案可以是识别计算机系统白天被用于进行数据处理以及晚上被用于进行批处理。一些关键应用程序因此可能使它们的保留资源分配增加，以改进晚上批处理的吞吐量，同时收缩或者甚至禁止用于白天的交互式处理或数据处理的保留。

[0041] 工作量管理系统 4 也可以实现事件驱动的保留资源管理策略，该策略与它的最大可能配置相比，考虑应用程序的实际的资源利用。例如，另一个示例方案是采用一策略，该策略在探测到的应用程序的运行状态的基础上，不同步地指派保留资源分配，其中，该保留资源分配被指派给所述应用程序。作为例子，该策略可以规定，如果应用程序达到了预定的资源使用阈值，例如，它的最小分配之上 10％或全部资源池的 50％，则保留资源分配将被指派。其它的用户规定的应用程序状态事件也可以用来指派保留资源分配。其后，可以使用后来的触发状态来减少或去除保留资源分配。

[0042] 现在参看图 7A 和图 7B，它们分别显示工作量管理系统 4 的示例实施例 4A 和 4B，其中静态地规定了用于动态地管理对应用程序的保留资源分配的策略。为实现这样的规定，以用户界面 12 来配置工作量管理系统 4A 和工作量管理系统 4B，并且，它们或是存储一系列的保留资源管理策略 14（系统 4A），或是与自身存储策略集 14 的工作量管理中间设备系统 16 交互（系统 4B）。如前所述，工作量管理系统 4A 和工作量管理系统 4B 可以作为部分的操作系统内核、作为用户级的应用程序或者作为包括操作系统和用户应用程序功能性两者的混合系统来运行。

[0043] 可以使用任何适当的技术，例如系统或库调用、套接字、HAL（硬件抽象层）调用、信息传递 API 或者**IBM® AIX®** 5L 工作量管理器以及**Linux®** CKRM（基于类的内核资源管理器）框架（如上所述）中所建立的文件系统范例，来实现工作量管理系统 4A 和工作量管理系统 4B 两者的用户界面 12。策略集 14 包含支配应用程序的保留资源分配的利用的策略。举例说来，一个这样的策略可以规定，响应于应用程序的实际的资源利用的改变或响应于其它运行条件的改变（如上所述），保留资源分配的尺寸应该怎样增加或减小（或保持不变）。另一个策略可以规定进度表或条件，在该条件下保留资源分配特征应该代表应用程序被激活。

[0044] 因为工作量管理系统 4A 存储策略集 14，所以系统管理者可以使用这个系统的界面 12，以向工作量管理系统输入策略。该策略输入可以采取任何适当的形式，例如，保留资源分配 RES 参数的说明，或作为策略规则的说明，如随时间或基于主要的系统条件对保留资源分配进行控制的策略规则。工作量管理系统 4A 中的策略处理引擎 18 可用来处理策略集 14。这个处理将动态地设置保留资源分配的尺寸和管理保留资源分配特征的激活状态。

[0045] 在工作量管理系统 4B 中，策略集 14 被存储在工作量管理中间设备系统 16 中。中间设备系统 16 也包括处理策略规则的策略处理引擎 18。经由界面 12 将这个策略处理的输出提供给工作量管理系统 4B。界面 12 也被工作量管理系统 4B 用来向策略处理引擎 18 提供运行反馈。

[0046] 现在参看图 8A 和图 8B，它们分别显示工作量管理系统 4 的示例实施例 4C 和 4D，其中，使用自调节技术自治地优化用于向应用程序指派保留资源分配的策略。为实现这样的动态决定，以用户界面 12 来配置工作量管理系统 4C 和工作量管理系统 4D，并且，它们或是实现遗传调节引擎 22（系统 4C），或是与实现遗传调节引擎 22 的工作量管理中间设备系统 24 交互（系统 4D）。另外，如上所述，工作量管理系统 4C 和工作量管理系统 4D 可以作为部分的操作系统内核、作为用户级的应用程序或者作为包括操作系统和用户应用程序功能性两者的混合系统来运行。

[0047] 工作量管理系统 4C 和工作量管理系统 4D 的遗传调节引擎 22 接收监视输入的应用程序，该输入指示利用正被管理的资源的那些应用程序的性能。这样的输入可以从应用程序自身、从下层操作系统或从以上两者接收。遗传调节引擎 22 也接收代表当前的资源分配和利用数据的资源分配输入。这样的输入是从工作量管理系统 4C 或工作量管理系统 4D 自身接收的。

[0048] 监视输入的应用程序可以包括一个或多个应用程序的应用程序性能和工作量诊断。资源分配输入可以包括这样的信息，如最小资源分配 MIN、最大资源分配 MAX、正被利用的资源的当前水平以及保留资源分配 RES。监视的应用程序和资源分配输入被传递至遗传调节引擎 22 以用于处理。使用传统的遗传技术，遗传调节引擎 22 可以被编程以执行多代的调节周期。对每一代来说，遗传调节引擎 22 可以 (1) 选择支配应用程序的保留资源分配的一系列参数，(2) 变动一个或多个参数并相应地调节保留资源分配，(3) 评估调节保留资源分配后监视所接收的输入的应用程序，和 (4) 评定相对于来自上一代的结果的有利性。新一代的保留资源分配参数可以是基于最有利的结果而创建，并且，可以为该新一代而重复前述的过程。这样，可以为应用程序达到保留资源分配参数的最佳集合。

[0049] 因此，这里公开了用于管理数据处理系统中的应用程序工作量的技术，其中，该数据处理系统使用可临时指派的保留资源分配。可以理解，前述的概念在数据处理系统、机器实现的方法、计算机程序产品（在该计算机程序产品中，通过一个或多个机器可读介质来提供编程逻辑，以用于控制数据处理系统去执行所要求的功能）中的任何一个中可以有不同的实施。图 9 中利用附图标号 100 显示了用于提供这种编程逻辑的示例的机器可读介质。介质 100 被显示为传统上用于商业软件销售的便携式光学存储盘，例如，只读光盘 (CD-ROM)、可擦写光盘 (CD-R/W) 以及多功能数码光盘 (DVD)。这样的介质可以存储本发明的编程逻辑，或是单独或是与其它集成有所要求的功能性的软件产品相结合。编程逻辑也可以由便携式的磁介质（例如，软盘和闪存记忆棒等）提供，或由结合有驱动系统的磁介质（例如，硬盘驱动）提供，或由集成有数据处理平台的介质提供，例如随机存取存储器 (RAM)、只读存储器 (ROM) 或者其它的半导体或固态存储器。更宽泛地，该介质可以包括任何电子的、磁的、光学的、电磁的、红外的、半导体系统或装置或设备，可以包括传输或传播信号的介质或承载信号的介质（例如网络），还可以包括可包括存储、通信、传播或传送该编程逻辑的其它实体，该实体供数据处理系统、计算机或其它指令执行系统、装置或设备使

用，或供它们结合使用。

[0050] 虽然已经描述了本发明的各种实施例，但显然根据本发明可以实现许多变动或可选的实施例。因此，可以理解，除了根据权利要求的范围及其等价物外，本发明不应以任何方式受到限制。

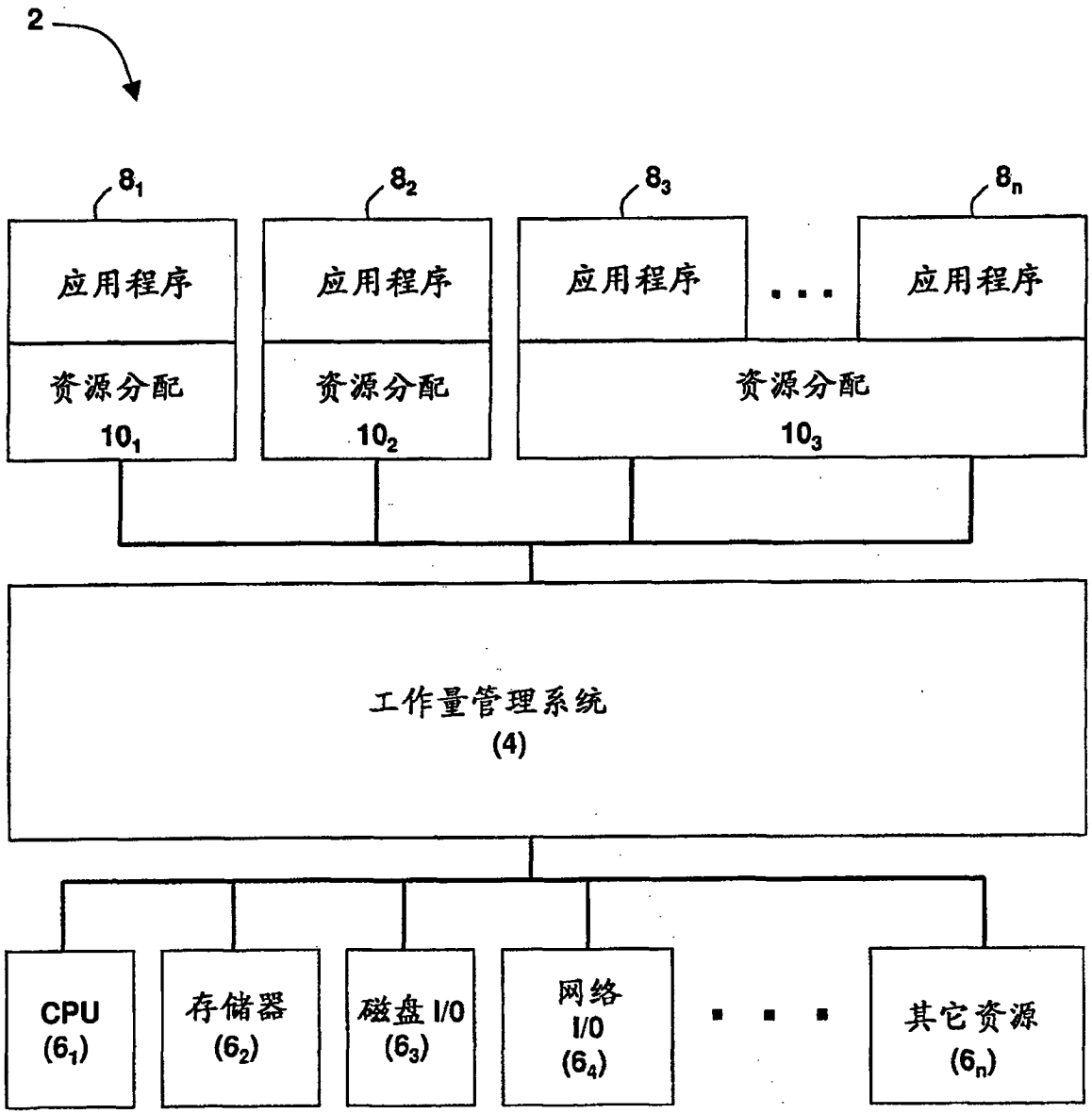

2
8_1
8_2
8_3
8_n
应用程序
应用程序
应用程序
应用程序
资源分配
10_1
资源分配
10_2
资源分配
10_3
工作量管理系统
(4)
CPU
(6_1)
存储器
(6_2)
磁盘 I/O
(6_3)
网络
I/O
(6_4)
其它资源
(6_n)

图 1

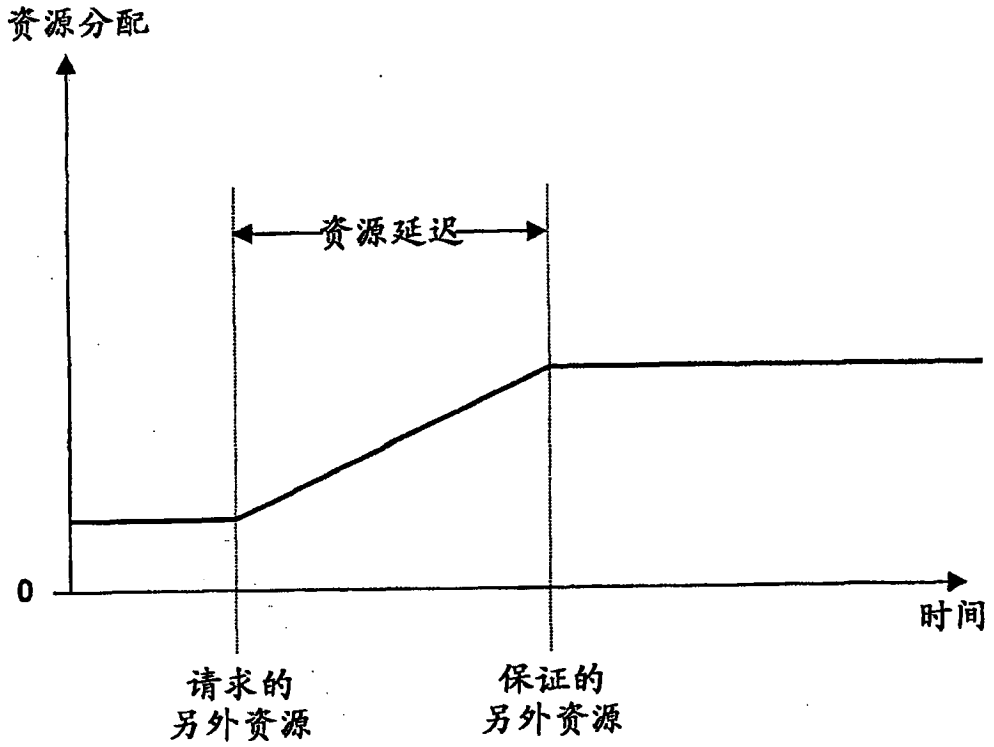

资源分配
资源延迟
0
时间
请求的
另外资源
保证的
另外资源

图 2(现有技术)

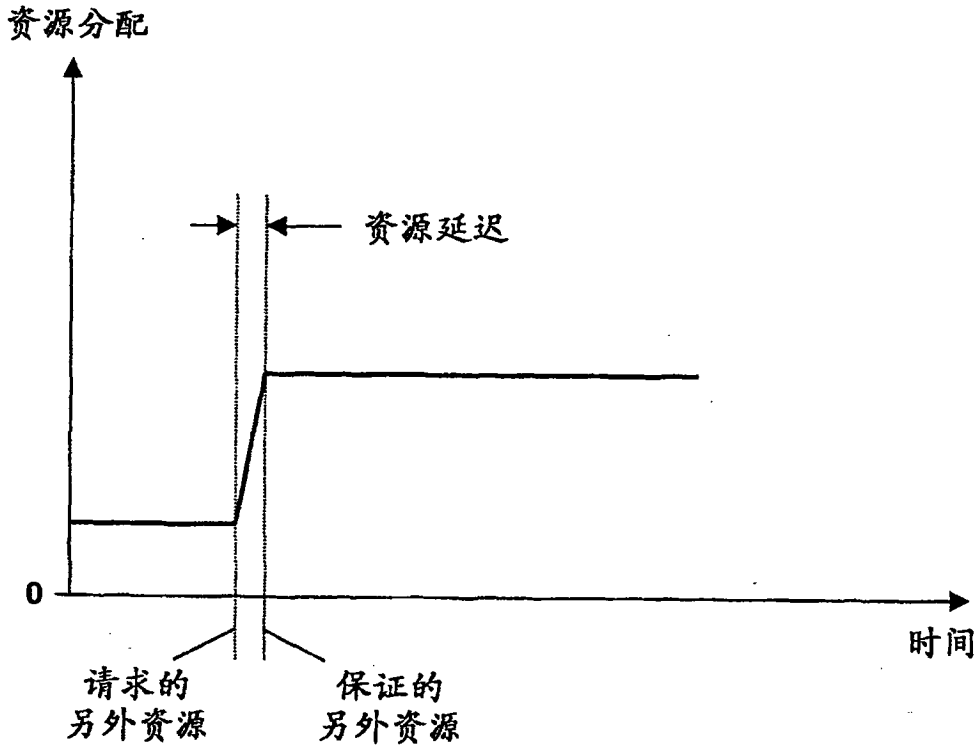

资源分配
资源延迟
0
时间
请求的
另外资源
保证的
另外资源

图 3

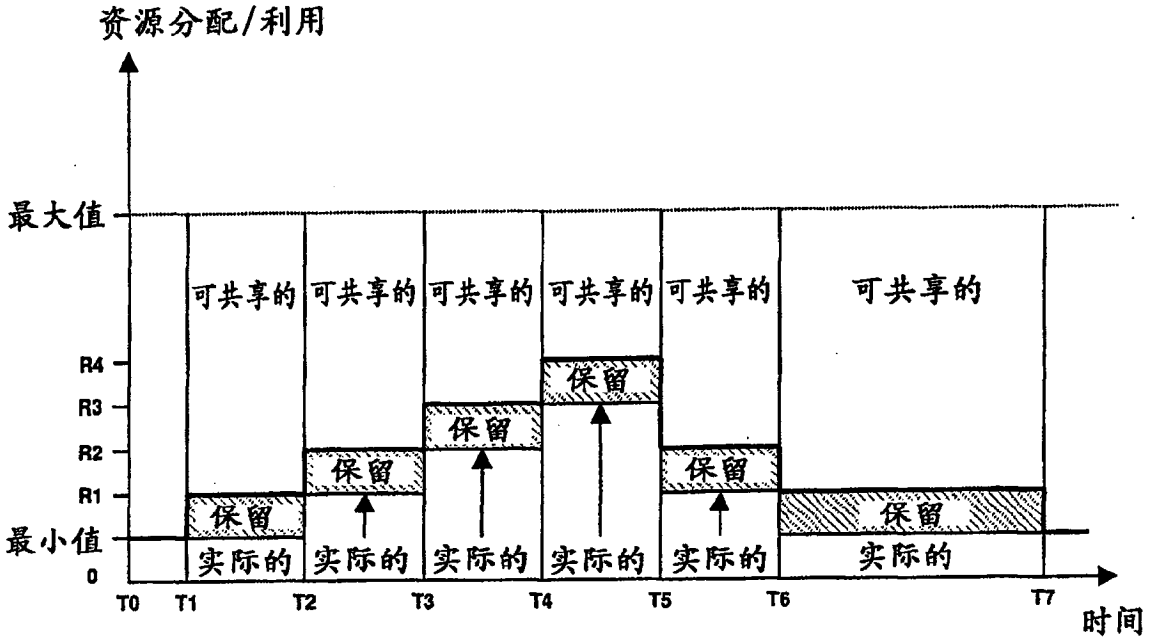

资源分配/利用
最大值
可共享的
可共享的
可共享的
可共享的
可共享的
可共享的
R4
R3
R2
R1
最小值
0
保留
保留
保留
保留
保留
保留
实际的
实际的
实际的
实际的
实际的
实际的
T0
T1
T2
T3
T4
T5
T6
T7
时间

图 4

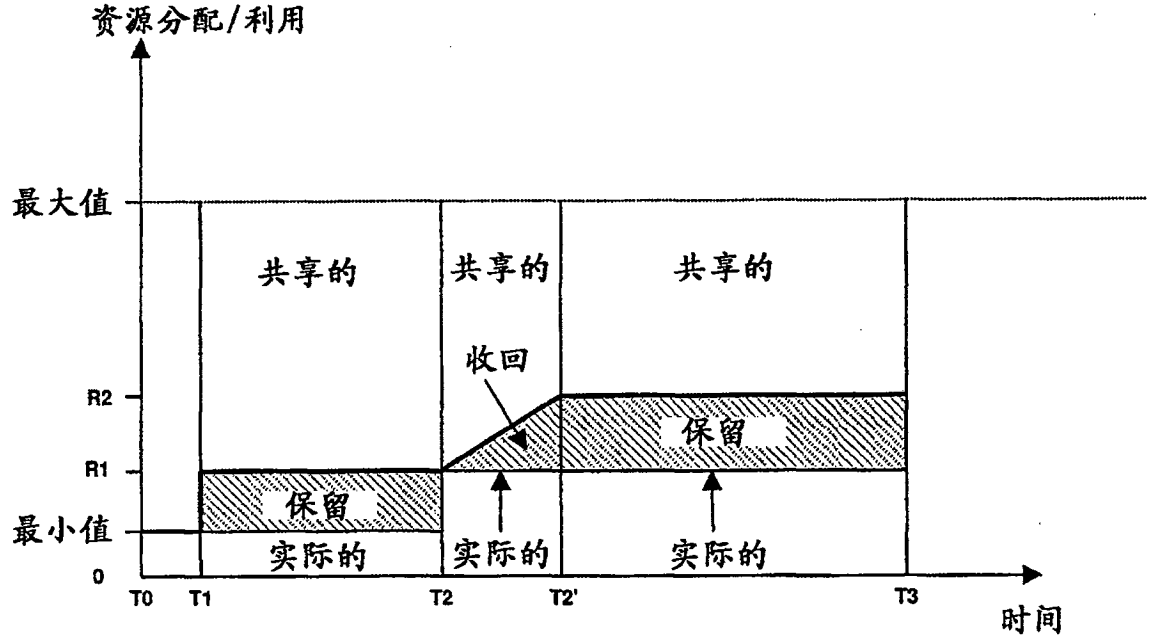

资源分配/利用
最大值
共享的
共享的
共享的
收回
R2
R1
最小值
0
保留
保留
实际的
实际的
实际的
T0
T1
T2
T2'
T3
时间

图 5

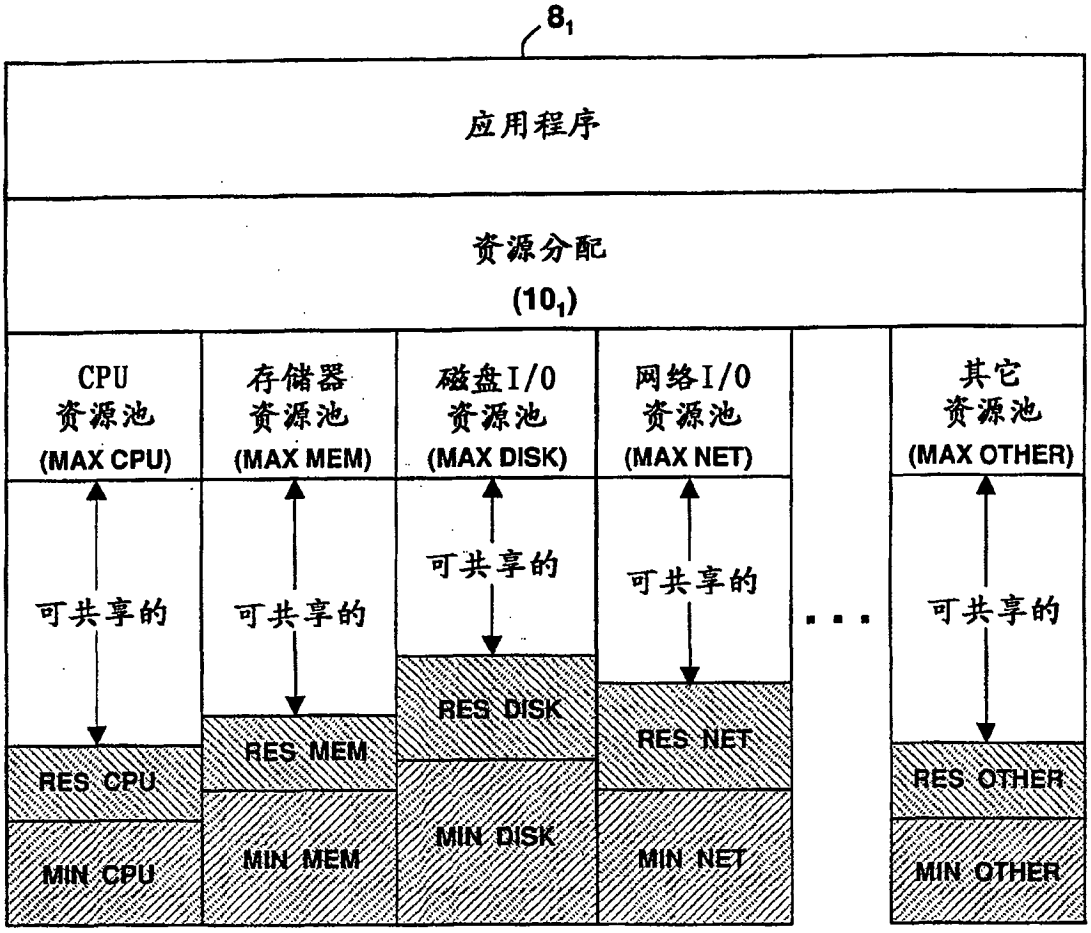

8_1
应用程序
资源分配
(10_1)
CPU
资源池
(MAX CPU)
存储器
资源池
(MAX MEM)
磁盘I/O
资源池
(MAX DISK)
网络I/O
资源池
(MAX NET)
其它
资源池
(MAX OTHER)
可共享的
可共享的
可共享的
可共享的
可共享的
RES CPU
RES MEM
RES DISK
RES NET
RES OTHER
MIN CPU
MIN MEM
MIN DISK
MIN NET
MIN OTHER

图 6

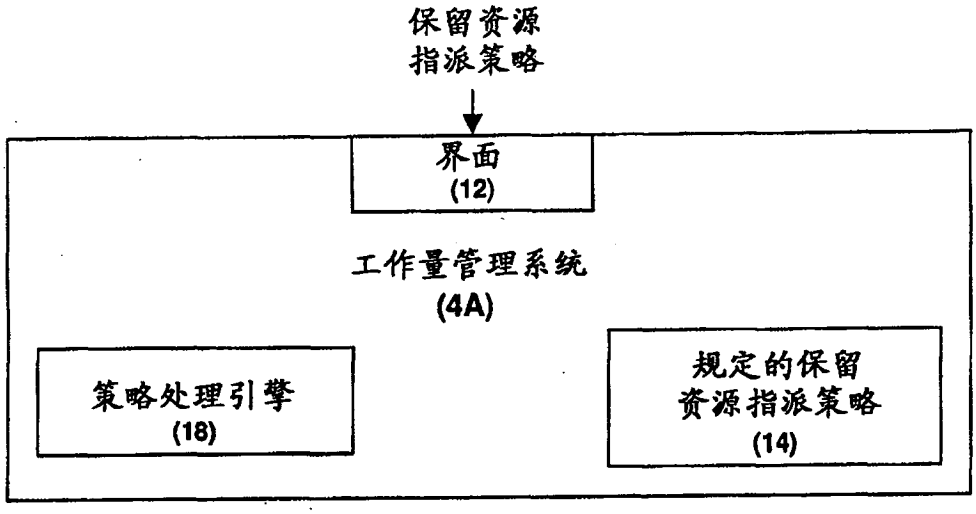

保留资源
指派策略
界面
(12)
工作量管理系统
(4A)
策略处理引擎
(18)
规定的保留
资源指派策略
(14)

图 7A

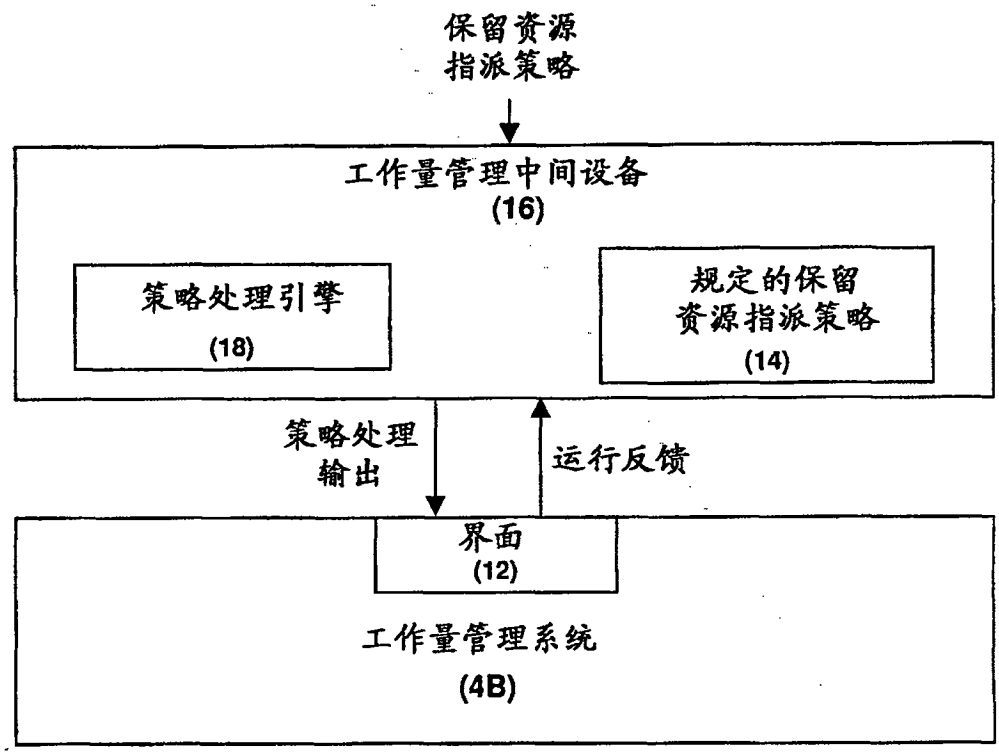

保留资源
指派策略
工作量管理中间设备
(16)
策略处理引擎
(18)
规定的保留
资源指派策略
(14)
策略处理
输出
运行反馈
界面
(12)
工作量管理系统
(4B)

图 7B

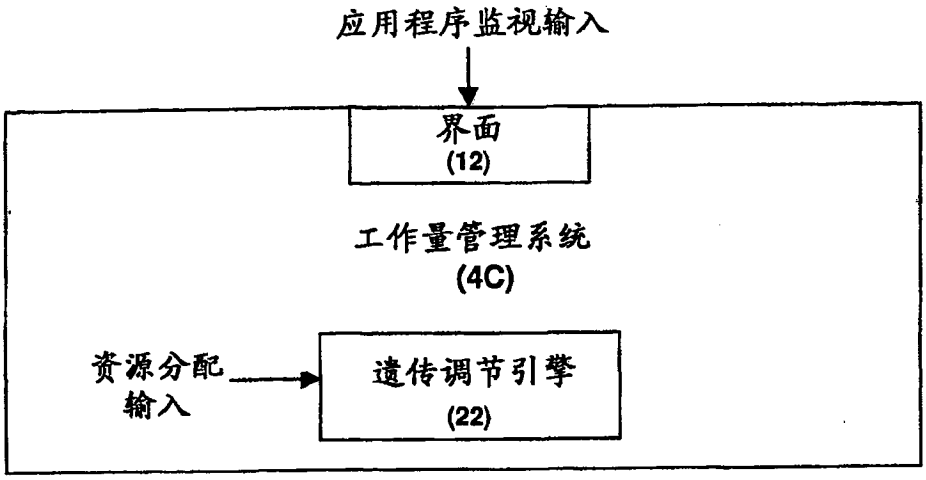

应用程序监视输入
界面
(12)
工作量管理系统
(4C)
资源分配
输入
遗传调节引擎
(22)

图 8A

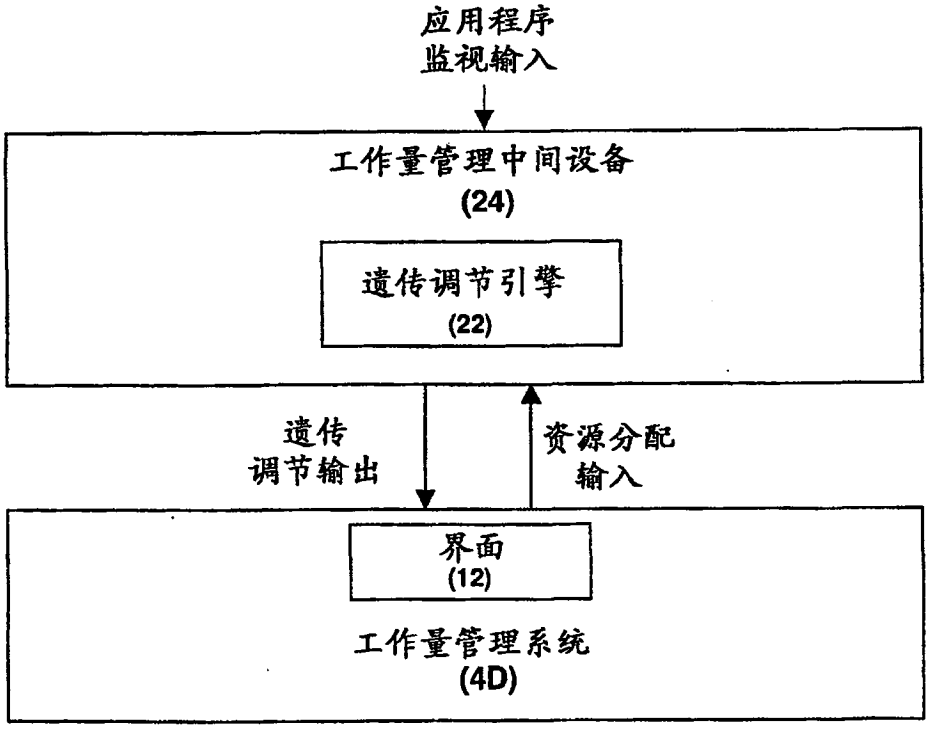

应用程序
监视输入
工作量管理中间设备
(24)
遗传调节引擎
(22)
遗传
调节输出
资源分配
输入
界面
(12)
工作量管理系统
(4D)

图 8B

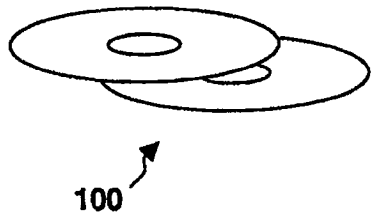

100

图 9