(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 103677990 A
(43)申请公布日 2014.03.26

(21)申请号 201310684535.0

(22)申请日 2013.12.13

(71)申请人 清华大学
地址 100084 北京市海淀区 100084-82 信箱

(72)发明人 郑纬民 武永卫 姜进磊 赵勋

(74)专利代理机构 北京清亦华知识产权代理事务所(普通合伙) 11201
代理人 张大威

(51)Int.Cl.
G06F 9/48(2006.01)
G06F 9/455(2006.01)

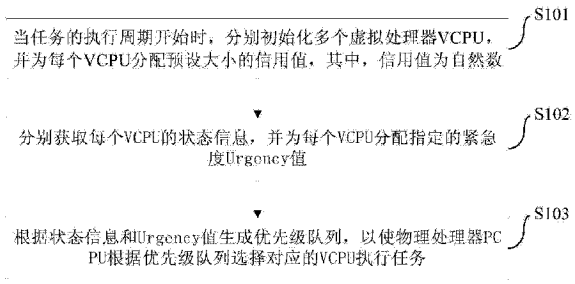
权利要求书4页 说明书12页 附图5页

(54)发明名称

虚拟机实时任务的调度方法、装置和虚拟机

(57)摘要

本发明提出一种虚拟机实时任务的调度方法、装置和虚拟机。其中,所述方法包括以下步骤:当任务的执行周期开始时,分别初始化多个虚拟处理器VCPU,并为每个VCPU分配预设大小的信用值,其中,信用值为自然数;分别为每个VCPU分配指定的紧急度Urgency值;以及分别获取每个VCPU的状态信息,并根据状态信息和Urgency值生成优先级队列,以使物理处理器PCPU根据优先级队列选择对应的VCPU执行任务。本发明实施例方法,会在每个周期优先切换具有urgent状态的VCPU,由此,实时任务可以被及时响应。实验使用ping延迟与web服务器的吞吐量表明这种调度方法可以及时处理实时任务,并提高了响应时间的稳定性。



CN 103677990 A

1. 一种虚拟机实时任务的调度方法，其特征在于，包括以下步骤：

当任务的执行周期开始时，分别初始化多个虚拟处理器 VCPU，并为每个 VCPU 分配预设大小的信用值，其中，所述信用值为自然数；

分别获取每个所述 VCPU 的状态信息，并为每个所述 VCPU 分配指定的紧急度 Urgency 值；以及

根据所述状态信息和所述 Urgency 值生成优先级队列，以使物理处理器 PCPU 根据所述优先级队列选择对应的所述 VCPU 执行任务。

2. 如权利要求 1 所述的方法，其特征在于，所述根据状态信息和所述 Urgency 值生成优先级队列具体包括：

根据所述每个 VCPU 状态信息的优先级对所述 VCPU 进行排序；以及

根据排序结果生成优先级队列。

3. 如权利要求 1 所述的方法，其特征在于，所述根据状态信息和所述 Urgency 值生成优先级队列具体包括：

当多个所述 VCPU 的状态信息相同时，根据所述多个 VCPU 的所述 Urgency 值对所述多个 VCPU 进行排序；以及

根据排序结果生成优先级队列。

4. 如权利要求 1 所述的方法，其特征在于，所述根据状态信息和所述 Urgency 值生成优先级队列具体包括：

当多个所述 VCPU 的状态信息以及所述状态信息对应的 Urgency 值相同时，根据先进先出的规则对所述多个 VCPU 进行排序；以及

根据排序结果生成优先级队列。

5. 如权利要求 1-4 任意一项所述的方法，其特征在于，所述 VCPU 的状态信息包括：boost 状态、urgent 状态、under 状态、over 状态和 idle 状态，优先级为 boost 状态的优先级 >urgent 状态的优先级 >under 状态的优先级 >over 状态的优先级 >idle 状态的优先级。

6. 如权利要求 5 所述的方法，其特征在于，所述 Urgency 值为小于等于 10 的自然数。

7. 如权利要求 1 所述的方法，其特征在于，还包括：

当所述 VCPU 执行任务完成后，从所述 VCPU 分配的所述预设大小的信用值中减去对应的信用值，并根据剩余信用值和所述 VCPU 的状态信息获取所述 VCPU 更新后的状态信息。

8. 如权利要求 7 所述的方法，其特征在于，所述根据剩余信用值和所述 VCPU 的状态信息获取所述 VCPU 更新后的状态信息具体包括：

如果所述 VCPU 处于 boost 状态，且所述 VCPU 的剩余信用值大于 0，且所述 VCPU 的所述 Urgency 值等于 0，则将所述 VCPU 的状态信息更新为 under 状态；

如果所述 VCPU 处于 boost 状态，且所述 VCPU 的剩余信用值大于 0，且所述 VCPU 的所述 Urgency 值大于 0，则将所述 VCPU 的状态信息更新为 urgent 状态；以及

如果所述 VCPU 处于 boost 状态，且所述 VCPU 的剩余信用值等于 0，则将所述 VCPU 的状态信息更新为 over 状态。

9. 如权利要求 7 所述的方法，其特征在于，所述根据剩余信用值和所述 VCPU 的状态信息获取所述 VCPU 更新后的状态信息具体包括：

如果所述 VCPU 处于 urgent 状态，且所述 VCPU 接收到 I/O 事件，则将所述 VCPU 的状态

信息更新为 boost 状态；

如果所述 VCPU 处于 urgent 状态，且所述 VCPU 的剩余信用值等于 0，则将所述 VCPU 的状态信息更新为 over 状态；以及

如果所述 VCPU 处于 urgent 状态，且所述 VCPU 的 I/O 阻塞，则将所述 VCPU 的状态信息更新为 idle 状态。

10. 如权利要求 7 所述的方法，其特征在于，所述根据剩余信用值和所述 VCPU 的状态信息获取所述 VCPU 更新后的状态信息具体包括：

如果所述 VCPU 处于 under 状态，且所述 VCPU 的剩余信用值等于 0，则将所述 VCPU 的状态信息更新为 over 状态；以及

如果所述 VCPU 处于 under 状态，且所述 VCPU 的 I/O 阻塞，则将所述 VCPU 的状态信息更新为 idle 状态。

11. 如权利要求 7 所述的方法，其特征在于，所述根据剩余信用值和所述 VCPU 的状态信息获取所述 VCPU 更新后的状态信息具体包括：

如果所述 VCPU 处于 over 状态，且所述 VCPU 重新分配信用值，且所述 VCPU 的所述 Urgency 值等于 0，则将所述 VCPU 的状态信息更新为 under 状态；以及

如果所述 VCPU 处于 over 状态，且所述 VCPU 重新分配信用值，且所述 VCPU 的所述 Urgency 值大于 0，则将所述 VCPU 的状态信息更新为 urgent 状态。

12. 如权利要求 7 所述的方法，其特征在于，所述根据剩余信用值和所述 VCPU 的状态信息获取所述 VCPU 更新后的状态信息具体包括：

如果所述 VCPU 处于 idle 状态，且所述 VCPU 接收到 I/O 事件，则将所述 VCPU 的状态信息更新为 boost 状态。

13. 如权利要求 7 所述的方法，其特征在于，还包括：

计算所有的所述 VCPU 的剩余信用值的总和；以及

当所述 VCPU 的剩余信用值的总和等于 0 时，重新为每个 VCPU 分配预设大小的信用值。

14. 一种虚拟机实时任务的调度装置，其特征在于，包括：

第一分配模块，用于当任务的执行周期开始时，分别初始化多个虚拟处理器 VCPU，并为每个 VCPU 分配预设大小的信用值，其中，所述信用值为自然数；

获取与分配模块，用于分别获取每个所述 VCPU 的状态信息，并为每个所述 VCPU 分配指定的紧急度 Urgency 值；以及

生成模块，用于根据所述状态信息和所述 Urgency 值生成优先级队列，以使物理处理器 PCPU 根据所述优先级队列选择对应的所述 VCPU 执行任务。

15. 如权利要求 14 所述的装置，其特征在于，所述生成模块具体包括：

第一排序单元，用于根据所述每个 VCPU 状态信息的优先级对所述 VCPU 进行排序；以及

生成单元，用于根据排序结果生成优先级队列。

16. 如权利要求 14 所述的装置，其特征在于，所述生成模块具体包括：

第二排序单元，用于当多个所述 VCPU 的状态信息相同时，根据所述多个 VCPU 的所述 Urgency 值对所述多个 VCPU 进行排序，

所述生成单元还用于根据排序结果生成优先级队列。

17. 如权利要求 14 所述的装置，其特征在于，所述生成模块具体包括：

第三排序单元，用于当多个所述 VCPU 的状态信息以及所述状态信息对应的 Urgency 相同时，根据先进先出的规则对所述多个 VCPU 进行排序，

所述生成单元还用于根据排序结果生成优先级队列。

18. 如权利要求 14-17 任意一项所述的装置，其特征在于，所述 VCPU 的状态信息包括：boost 状态、urgent 状态、under 状态、over 状态和 idle 状态，优先级为 boost 状态的优先级 >urgent 状态的优先级 >under 状态的优先级 >over 状态的优先级 >idle 状态的优先级。

19. 如权利要求 18 所述的装置，其特征在于，所述 Urgency 值为小于等于 10 的自然数。

20. 如权利要求 14 所述的装置，其特征在于，还包括：

更新模块，用于当所述 VCPU 执行任务完成后，从所述 VCPU 分配的所述预设大小的信用值中减去对应的信用值，并根据剩余信用值和所述 VCPU 的状态信息获取所述 VCPU 更新后的状态信息。

21. 如权利要求 20 所述的装置，其特征在于，所述更新模块具体包括：

第一更新单元，用于在所述 VCPU 处于 boost 状态，且所述 VCPU 的剩余信用值大于 0，且所述 VCPU 的所述 Urgency 值等于 0 时，将所述 VCPU 的状态信息更新为 under 状态；

第二更新单元，用于在所述 VCPU 处于 boost 状态，且所述 VCPU 的剩余信用值大于 0，且所述 VCPU 的所述 Urgency 值大于 0 时，将所述 VCPU 的状态信息更新为 urgent 状态；以及

第三更新单元，用于在所述 VCPU 处于 boost 状态，且所述 VCPU 的剩余信用值等于 0 时，将所述 VCPU 的状态信息更新为 over 状态。

22. 如权利要求 20 所述的装置，其特征在于，所述更新模块具体包括：

第四更新单元，用于在所述 VCPU 处于 urgent 状态，且所述 VCPU 接收到 I/O 事件时，将所述 VCPU 的状态信息更新为 boost 状态；

第五更新单元，用于在所述 VCPU 处于 urgent 状态，且所述 VCPU 的剩余信用值等于 0 时，将所述 VCPU 的状态信息更新为 over 状态；以及

第六更新单元，用于在所述 VCPU 处于 urgent 状态，且所述 VCPU 的 I/O 阻塞时，将所述 VCPU 的状态信息更新为 idle 状态。

23. 如权利要求 20 所述的装置，其特征在于，所述更新模块具体包括：

第七更新单元，用于在所述 VCPU 处于 under 状态，且所述 VCPU 的剩余信用值等于 0 时，将所述 VCPU 的状态信息更新为 over 状态；以及

第八更新单元，用于在所述 VCPU 处于 under 状态，且所述 VCPU 的 I/O 阻塞时，将所述 VCPU 的状态信息更新为 idle 状态。

24. 如权利要求 20 所述的装置，其特征在于，所述更新模块具体包括：

第九更新单元，用于在所述 VCPU 处于 over 状态，且所述 VCPU 重新分配信用值，且所述 VCPU 的所述 Urgency 值等于 0 时，将所述 VCPU 的状态信息更新为 under 状态；以及

第十更新单元，用于在所述 VCPU 处于 over 状态，且所述 VCPU 重新分配信用值，且所述 VCPU 的所述 Urgency 值大于 0 时，将所述 VCPU 的状态信息更新为 urgent 状态。

25. 如权利要求 20 所述的装置，其特征在于，所述更新模块具体包括：

第十一更新单元，用于在所述 VCPU 处于 idle 状态，且所述 VCPU 接收到 I/O 事件时，将所述 VCPU 的状态信息更新为 boost 状态。

26. 如权利要求 20 所述的装置，其特征在于，还包括：

计算模块，用于计算所有的所述 VCPU 的剩余信用值的总和；以及

第二分配模块，用于当所述 VCPU 的剩余信用值的总和等于 0 时，重新为每个 VCPU 分配预设大小的信用值。

27. 一种虚拟机，其特征在于，包括权利要求 14-26 任一项所述的虚拟机实时任务的调度装置。

虚拟机实时任务的调度方法、装置和虚拟机

技术领域

[0001] 本发明涉及虚拟机调度领域，尤其涉及一种虚拟机实时任务的调度方法、装置和虚拟机。

背景技术

[0002] 随着云计算的快速发展，业内公司提供了越来越多的基于虚拟化技术的服务。它们逐步使用虚拟机代替了传统的物理机，这种方式能够充分利用物理资源，并且提供更好的服务，成为了一种主要发展趋势。

[0003] 目前，最常被使用到的虚拟机是 VMware ESX 和 Xen(VMware ESX 是由 VMware 公司研发的软件产品，Xen 则是剑桥大学创建的开放源代码虚拟机监视器)。Xen 作为虚拟机监控程序，使得多个操作系统可以同时运行在单个计算机上，并且支持半虚拟化和虚拟机的在线迁移。在 Xen 虚拟机中，管理程序会提供区域来运行虚拟机。Dom0 是特殊的驱动区域，类似底层物理硬件与其他区域之间的抽象层。因此，Dom0 将处理所有 I/O（input/output，输入 / 输出端口）请求。当某个虚拟机产生 I/O 请求，就会产生虚拟中断，然后将该请求提交给 Dom0。类似于进程的上下文切换，在调度器切换虚拟机时，也需要完成相应的上下文切换(例如，运行中虚拟机的状态将被保存，而其它虚拟机的状态将被恢复)。由于上下文切换会带来额外的开销，因此需要根据不同的应用属性做出良好的决策，避免不合理的切换，从而更好的利用物理机。

[0004] 为此，Xen 虚拟机监控程序提供了三种 CPU 调度算法，Borrowed Virtual Time 调度算法、Simple Earliest Deadline First 调度算法以及信用调度算法。例如，在信用调度算法中，由于在 MapReduce 程序中有很多 I/O 操作，Kang 等推迟 Dom0 来聚集更多的 I/O 事件一并处理，这大大减少了虚拟处理器 VCPU 上下文切换的频率。其中还引入了 group_id 与 group_credit 的定义，并提供了一种组调度策略，使得在虚拟集群中能更好的运行 MapReduce（映射规约编程模型，一种用于大规模数据集的并行运算的编程模型）程序。基于 Xen 信用调度器、Hu 等针对多核机器将核进行分类以处理不同类型的任务(驱动，I/O 密集型，计算密集型)，而减少上下文切换的开销。

[0005] 然而目前存在的问题是，上述的两种方法均将 VCPU 上下文切换开销作为主要考虑因素，从而加速了计算密集型任务的运行。但是不同于物理处理器 PCPU，VCPU 并不会一直在线，这将带来大量的自旋锁 spinlock 阻塞。虽然 Weng 等使用 VCRD 衡量 VCPU 之间的关系，并且使用机器学习的方法来预测未来使用情形，但是对于某些实时任务来说，由于实时任务的响应时间需要被限制在毫秒甚至微秒级别，VCPU 在执行实时程序时，需要及时反馈信息。然而，Xen 的调度策略并不单独标识运行实时任务的 VCPU，这不仅导致了切换策略的结果不准确性，也推迟了实时计算的运行。

发明内容

[0006] 本发明旨在至少解决上述技术问题之一。

[0007] 为此，本发明的第一个目的在于提出一种虚拟机实时任务的调度方法。该方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0008] 本发明的第二个目的在于提出一种虚拟机实时任务的调度装置。

[0009] 本发明的第三个目的在于提出一种虚拟机。

[0010] 为了实现上述目的，本发明第一方面实施例的虚拟机实时任务的调度方法，包括以下步骤：当任务的执行周期开始时，分别初始化多个虚拟处理器 VCPU，并为每个 VCPU 分配预设大小的信用值，其中，所述信用值为自然数；分别为每个所述 VCPU 分配指定的紧急度 Urgency 值；以及分别获取每个所述 VCPU 的状态信息，并根据所述状态信息和所述 Urgency 值生成优先级队列，以使物理处理器 PCPU 根据所述优先级队列选择对应的所述 VCPU 执行任务。

[0011] 本发明实施例的虚拟机实时任务的调度方法，添加了新的 VCPU 的状态信息 urgent 状态用以标识实时任务，使其在单一的 PCPU 的运行队列中优先级高于 under 状态。因此，调度方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0012] 为了实现上述目的，本发明第二方面实施例的虚拟机实时任务的调度装置，包括：第一分配模块，用于当任务的执行周期开始时，分别初始化多个虚拟处理器 VCPU，并为每个 VCPU 分配预设大小的信用值，其中，所述信用值为自然数；获取与分配模块，用于分别获取每个所述 VCPU 的状态信息，并为每个所述 VCPU 分配指定的紧急度 Urgency 值；以及生成模块，用于根据所述状态信息和所述 Urgency 值生成优先级队列，以使物理处理器 PCPU 根据所述优先级队列选择对应的所述 VCPU 执行任务。

[0013] 本发明实施例的虚拟机实时任务的调度装置，添加了新的 VCPU 的状态信息 urgent 状态用以标识实时任务，使其在单一的 PCPU 的运行队列中优先级高于 under 状态。因此，调度方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0014] 为了实现上述目的，本发明第三方面实施例的虚拟机，包括本发明第二方面实施例的虚拟机实时任务的调度装置。

[0015] 本发明实施例的虚拟机，添加了新的 VCPU 的状态信息 urgent 状态用以标识实时任务，使其在单一的 PCPU 的运行队列中优先级高于 under 状态。因此，调度方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0016] 本发明附加的方面和优点将在下面的描述中部分给出，部分将从下面的描述中变得明显，或通过本发明的实践了解到。

附图说明

[0017] 本发明上述的和 / 或附加的方面和优点从下面结合附图对实施例的描述中将变得明显和容易理解，其中，

[0018] 图 1 是根据本发明一个实施例的虚拟机实时任务的调度方法的流程图；

[0019] 图 2 是根据本发明一个具体实施例的虚拟机实时任务的调度方法的流程图；

[0020] 图 3 是根据本发明一个具体实施例的 VCPU 的状态信息转换的示意图；

[0021] 图 4 是根据本发明一个具体实施例的 ping 延迟测试结果的示意图；

[0022] 图 5 是根据本发明一个具体实施例的 web 服务器的吞吐量的示意图；

[0023] 图 6 是根据本发明一个实施例的虚拟机实时任务的调度装置的结构示意图；

[0024] 图 7 是根据本发明一个具体实施例的虚拟机实时任务的调度装置的结构示意图；

[0025] 图 8 是根据本发明另一个具体实施例的虚拟机实时任务的调度装置的结构示意图；

[0026] 图 9 是根据本发明又一个具体实施例的虚拟机实时任务的调度装置的结构示意图。

具体实施方式

[0027] 下面详细描述本发明的实施例，所述实施例的示例在附图中示出，其中自始至终相同或类似的标号表示相同或类似的元件或具有相同或类似功能的元件。下面通过参考附图描述的实施例是示例性的，仅用于解释本发明，而不能理解为对本发明的限制。相反，本发明的实施例包括落入所附加权利要求书的精神和内涵范围内的所有变化、修改和等同物。

[0028] 在本发明的描述中，需要理解的是，术语“第一”、“第二”等仅用于描述目的，而不能理解为指示或暗示相对重要性。在本发明的描述中，需要说明的是，除非另有明确的规定和限定，术语“相连”、“连接”应做广义理解，例如，可以是固定连接，也可以是可拆卸连接，或一体地连接；可以是机械连接，也可以是电连接；可以是直接相连，也可以通过中间媒介间接相连。对于本领域的普通技术人员而言，可以具体情况理解上述术语在本发明中的具体含义。此外，在本发明的描述中，除非另有说明，“多个”的含义是两个或两个以上。

[0029] 流程图中或在此以其他方式描述的任何过程或方法描述可以被理解为，表示包括一个或更多个用于实现特定逻辑功能或过程的步骤的可执行指令的代码的模块、片段或部分，并且本发明的优选实施方式的范围包括另外的实现，其中可以不按所示出或讨论的顺序，包括根据所涉及的功能按基本同时的方式或按相反的顺序，来执行功能，这应被本发明的实施例所属技术领域的技术人员所理解。

[0030] 下面参考附图描述根据本发明实施例的虚拟机实时任务的调度方法、装置和虚拟机。

[0031] 实时任务是一类需要在严格时间内得到反馈的任务，然而目前的虚拟机调度方法中并没有提供对实时任务的充分支持，所以实时任务的运行总是被低效率的虚拟 I/O 以及调度延迟所限制。如果虚拟机调度器能够通过合理的切换与迁移虚拟机，从而实现整体系统的高性能。为此，本发明提出了一种虚拟机实时任务的调度方法，包括以下步骤：当任务的执行周期开始时，分别初始化多个虚拟处理器 VCPU，并为每个 VCPU 分配预设大小的信用值，其中，信用值为自然数；分别为每个 VCPU 分配指定的紧急度 Urgency 值；以及分别获

取每个 VCPU 的状态信息，并根据状态信息和 Urgency 值生成优先级队列，以使物理处理器 PCPU 根据优先级队列选择对应的 VCPU 执行任务。

[0032] 图 1 是根据本发明一个实施例的虚拟机实时任务的调度方法的流程图。如图 1 所示，该虚拟机实时任务的调度方法包括以下步骤。

[0033] S101，当任务的执行周期开始时，分别初始化多个虚拟处理器 VCPU，并为每个 VCPU 分配预设大小的信用值，其中，信用值为自然数。

[0034] S102，分别获取每个 VCPU 的状态信息，并为每个 VCPU 分配指定的紧急度 Urgency 值。

[0035] 在本发明的实施例中，VCPU 的状态信息包括 boost 状态、urgent 状态、under 状态、over 状态和 idle 状态。具体地，基于信用调度算法，本发明中引入了一种新的 VCPU 的状态信息，即 urgent 状态。当某个 VCPU 仍然具有剩余的信用值，且该 VCPU 运行了实时任务时，则该 VCPU 的状态会被指派为 urgent 状态。此外，对于 VCPU 的其它状态信息而言，当某个 VCPU 仍然具有剩余的信用值时，则该 VCPU 的状态会被指派为 under 状态；当某个 VCPU 的信用值已用完时，则该 VCPU 的状态会被指派为 over 状态；当某个 VCPU 此时不需要 PCPU 时，则该 VCPU 的状态会被指派为 idle 状态；当某个 VCPU 处于 idle 状态，且该 VCPU 接收到了 I/O 事件时，则该 VCPU 的状态会被指派为 boost 状态；当某个 VCPU 处于 idle 状态，且该 VCPU 接收到了 I/O 事件时，则该 VCPU 的状态会被指派为 boost 状态。

[0036] 进一步而言，为了保证 PCPU 及时响应实时任务，为每个 VCPU 分配指定的紧急度 Urgency 值。其中，Urgency 值为小于等于 10 的自然数。本发明的方法和现有的信用调度算法不同的是，处于 urgent 状态的 VCPU 在 PCPU 的运行队列中不是以先进先出的方式排列的。在本发明中是通过 Urgency 值来衡量这些 VCPU 的实时任务的紧急度，Urgency 值越大，VCPU 实时任务的紧急度越高。应当理解的是，Urgency 值的默认值是 0，也就是说，如果某个 VCPU 处于 boost 状态，或者 under 状态，或者 over 状态，或者 idle 状态时，为该 VCPU 分配指定的 Urgency 值为 0，这意味着该 VCPU 上的任务并不是实时的，可以推迟处理。其中，VCPU 的 Urgency 值可以是用户设定的，或者是任务中被指定的。

[0037] S103，根据状态信息和 Urgency 值生成优先级队列，以使物理处理器 PCPU 根据优先级队列选择对应的 VCPU 执行任务。

[0038] 具体而言，可以根据每个 VCPU 状态信息的优先级对 VCPU 进行排序，并根据排序结果按照 VCPU 状态信息的优先级从高至低的顺序生成优先级队列。其中，VCPU 的状态信息的优先级为 boost 状态的优先级 >urgent 状态的优先级 >under 状态的优先级 >over 状态的优先级 >idle 状态的优先级。然后，PCPU 可根据优先级队列，选择排在优先级队列中队首的 VCPU 执行任务。此外，如果其中某些 VCPU 的具有相同的状态信息，则根据 VCPU 被分配指定的 Urgency 值对这些 VCPU 进行排序，并根据排序结果按照 VCPU 的 Urgency 值从高至低的顺序生成优先级队列。换言之，本发明实施例中的方法是通过参考 VCPU 状态信息和 VCPU 的 Urgency 值相结合的方式，对 VCPU 的任务的执行顺序进行排序。应当理解的是，当多个 VCPU 的状态信息以及状态信息对应的 Urgency 相同时，根据先进先出的规则对多个 VCPU 进行排序，并根据排序结果生成优先级队列。也就是说，只有具有相同 Urgency 值，即具有相同紧急度的处于 urgent 状态的 VCPU，PCPU 才使用先进先出的方式选择 VCPU 执行任务，由此，可以保证紧急度高的实时任务先被执行。

[0039] 本发明实施例的虚拟机实时任务的调度方法，添加了新的 VCPU 的状态信息 urgent 状态用以标识实时任务，使其在单一的 PCPU 的运行队列中优先级高于 under 状态。因此，调度方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0040] 图 2 是根据本发明一个具体实施例的虚拟机实时任务的调度方法的流程图。如图 2 所示，该虚拟机实时任务的调度方法包括以下步骤。

[0041] S201，当任务的执行周期开始时，分别初始化多个虚拟处理器 VCPU，并为每个 VCPU 分配预设大小的信用值，其中，信用值为自然数。

[0042] S202，分别获取每个 VCPU 的状态信息，并为每个 VCPU 分配指定的紧急度 Urgency 值。

[0043] S203，根据状态信息和 Urgency 值生成优先级队列，以使物理处理器 PCPU 根据优先级队列选择对应的 VCPU 执行任务。

[0044] S204，当 VCPU 执行任务完成后，从 VCPU 分配的预设大小的信用值中减去对应的信用值，并根据剩余信用值和 VCPU 的状态信息获取 VCPU 更新后的状态信息。

[0045] 如图 3 所示，优选地可以根据以下方法定义 VCPU 的状态信息的转换过程以及相应的转换条：

[0046] 1、如果 VCPU 处于 boost 状态，且 VCPU 的剩余信用值大于 0，且 VCPU 的 Urgency 值等于 0，则将 VCPU 的状态信息更新为 under 状态；

[0047] 2、如果 VCPU 处于 boost 状态，且 VCPU 的剩余信用值大于 0，且 VCPU 的 Urgency 值大于 0，则将 VCPU 的状态信息更新为 urgent 状态；

[0048] 3、如果 VCPU 处于 boost 状态，且 VCPU 的剩余信用值等于 0，则将 VCPU 的状态信息更新为 over 状态；

[0049] 4、如果 VCPU 处于 urgent 状态，且 VCPU 接收到 I/O 事件，则将 VCPU 的状态信息更新为 boost 状态；

[0050] 5、如果 VCPU 处于 urgent 状态，且 VCPU 的剩余信用值等于 0，则将 VCPU 的状态信息更新为 over 状态；

[0051] 6、如果 VCPU 处于 urgent 状态，且 VCPU 的 I/O 阻塞，则将 VCPU 的状态信息更新为 idle 状态；

[0052] 7、如果 VCPU 处于 under 状态，且 VCPU 的剩余信用值等于 0，则将 VCPU 的状态信息更新为 over 状态；

[0053] 8、如果 VCPU 处于 under 状态，且 VCPU 的 I/O 阻塞，则将 VCPU 的状态信息更新为 idle 状态；

[0054] 9、如果 VCPU 处于 over 状态，且 VCPU 重新分配信用值，且 VCPU 的 Urgency 值等于 0，则将 VCPU 的状态信息更新为 under 状态；

[0055] 10、如果 VCPU 处于 over 状态，且 VCPU 重新分配信用值，且 VCPU 的 Urgency 值大于 0，则将 VCPU 的状态信息更新为 urgent 状态；

[0056] 11、如果 VCPU 处于 idle 状态，且 VCPU 接收到 I/O 事件，则将 VCPU 的状态信息更新为 boost 状态。

[0057] S205，计算所有的 VCPU 的剩余信用值的总和。

[0058] S206，当 VCPU 的剩余信用值的总和等于 0 时，重新为每个 VCPU 分配预设大小的信用值。

[0059] 本发明实施例的虚拟机实时任务的调度方法，添加了新的 VCPU 的状态信息 urgent 状态用以标识实时任务，使其在单一的 PCPU 的运行队列中优先级高于 under 状态。因此，调度方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0060] 下面结合图 4 和图 5 详细说明一下本发明实施例中的虚拟机实时任务的调度方法。

[0061] 类似于 Linux 和 Xen 中的调度模型，我们的模型在每个时钟周期均被调用。在每个时钟周期中，运行于每个 PCPU 的调度模型将决定下一个执行的 VCPU。调度模型同时也决定时间片的长度，其中，时间片即 CPU 分配给各个任务程序的时间。根据任务的不同种类，设计者和程序员可以设计不同的调度方法，以执行不同的调度策略。

[0062] Xen 虚拟机使用结构体来表示不同的调度方法。针对于每个调度方法，我们需要定义该结构体中的参数与函数。类似于 Xen 信用调度算法的结构体，sched_urgent_def 调度方法的具体定义如下：

[0063]

```
const struct scheduler sched_urgent_def = {
    .name           = "Urgent Scheduler",
    .opt_name       = "Urgent",
    .sched_id       = XEN_SCHEDULER_URGENT,
    .init_domain    = usched_dom_init,
    .destroy_domain = usched_dom_destroy,
    .init_vcpu      = usched_vcpu_init,
    .destroy_vcpu   = usched_vcpu_destroy,
    .sleep          = usched_vcpu_sleep,
    .wake           = usched_vcpu_wake,
    .adjust         = usched_dom_cntl,
    .pick_cpu       = usched_cpu_pick,
    .do_schedule    = usched_schedule,
    .dump_cpu_state = usched_dump_pcpu,
```

[0064]

```
    .dump_settings    = usched_dump,
    .init             = usched_init,
    .tick_suspend     = ucsched_tick_suspend,
    .tick_resume      = usched_tick_resume,
};
```

[0065] 基于信用调度算法，sched_urgent_def 调度方法增加了名为 urgent 的新状态优先级，于是 VCPU 的所有优先级如下所示：

[0066] #define USCHED_PRI_TS_BOOST 0

[0067] #define USCHED_PRI_TS_URGENT -1

[0068] #define USCHED_PRI_TS_UNDER -2

[0069] #define USCHED_PRI_TS_OVER -3

[0070] #define USCHED_PRI_IDLE -64

[0071] 我们还在每个虚拟机中提供了代理，这些代理可以记录虚拟机的状态信息，例如 CPU 状态信息、内存状态信息和硬盘状态信息等。在这里它们被用于提供每个虚拟机的紧急度。同时，Xen 的 VCPU 结构体需要被添加参数 Urgency 值来记录每个 VCPU 的紧急度。

[0072] 由于 Xen 并没有为添加新的调度机制提供注册函数，所以当我们添加注册函数时，则需要修改调度方法数组。紧急调度方法 sched_urgent_def 在 Xen 中实现如下：

[0073]

```
extern const struct scheduler sched_sedf_def;
extern const struct scheduler sched_credit_def;
extern const struct scheduler sched_credit_def;
static const struct scheduler *schedulers[] = {
    &sched_sedf_def,
    &sched_credit_def,
    &sched_urgent_def,
    NULL
};
```

[0074] 当调度组件需要调度虚拟机时，调度组件将会请求目前的调度器，然后，调度方法将返回 task_slice 结构体，该结构体包含了虚拟机的下一步调度操作。紧急调度实施的大体框架如下所示：

[0075]

```
/*
 * de-schedule the current domain
```

[0076]

```
 * pick a new domain (scheduler dependent).
 */
static void schedule(void)
{
        struct vcpu    *prev = current, *next = NULL;
        struct task_slice    next_slice;
        //get urgent-specific decision on scheduling
        next_slice = ops.do_schedule(now);
        //the next time slide and virtual machine
      r_time = next_slice.time;
next = next_slice.task;
        //fix time
        set_timer(&sd->s_timer, now + r_time);

        //to do context switching, it includes preservation and restoration of the scene
context_switch(prev, next);
      set_current(next);
      __context_switch();
            //complete context switching
            schedule_tail(next);
}
```

[0077] 处于 PCPU 运行队列中的 VCPU 根据优先级从高到低的顺序排列，其中具有 urgent 状态的 VCPU 则根据紧急度 Urgency 值的大小排列。调度核心将选择队列中排在首位的 VCPU 运行，因此实时任务将会被及时的处理。

[0078] 我们进行了若干实验来测试我们的紧急调度器，并将测试结果与信用调度模型进行比较。实验系统的硬件配置如下：两个四核 Intel(R)Xeon(R)E5620CPU（每个 CPU 具有四个核共享 L2 缓存），DDR3 内存 16GB，DELL PERC6/I500GB IDE 磁盘，网卡是 NetXtreme II BCM5709Gigabit Ethernet。虚拟机使用 Xen4.0.1，所有的 domU 运行 Ubuntu12.04i386 系统，内核版本是 Linux3.2.0。为了更好的进行实验，每个虚拟机只被分配了一个 VCPU。

[0079] 我们启动了三个客户操作系统，通过修改 Xen 的配置文件，使得所有的 domU 运行在一个 PCPU 上，因为我们的目标是测试单个 PCPU 运行队列中 VCPU 之间的切换。我们在每个虚拟机中运行 SEPC2006 标准测试程序，对其中的一个 VCPU 赋予优先级 urgent，然后记录这些虚拟机对 Ping 命令的响应时间。我们比较了与信用调度器的实验结果。

[0080] 如图 4 所示，图 4 中展示了 15 个连续 Ping 命令的延迟。其中圆形物理主机线条

显示了物理主机的 Ping 延迟以作为参照。方形信用调度模型线条表示信用调度模型的情况，其平均延迟时间大约是 0.605ms，并且某些峰值的延迟超过了 0.8ms。实验结果中的大多数峰值是由于 Ping 的虚拟机位于 PCPU 运行队列的后端，需要等待之前的 VCPU 完成任务。相反的，三角紧急调度器线条显示了使用紧急调度器的情况，平均的 Ping 延迟降低至 0.208ms，并且消除了大多数的峰值。这是因为 Ping 虚拟机被赋予了 urgent 优先级，在测试的过程中，它将处于 PCPU 运行队列的前端，因此降低了由于调度过程而造成的 ping 延迟。正如前面所说，在使用新调度器后，延迟的稳定性也有明显的提高，其标准差从 0.406 减少为 0.026。这类测试证明了紧急调度器可以快速响应实时任务，并且提高了响应时间的稳定性。

[0081] 然后，我们在虚拟机中部署了 Apache2 服务器，并使用 Httperf 测量 Web 服务器的吞吐量。Httperf 是一种测量 Web 服务器性能的工具，它利用反馈时间来计算 Web 服务器的吞吐量。如图 5 所示，相对于信用调度模型，Web 服务器的吞吐量在请求率为 40req/sec（请求 / 秒）与 70req/sec 的情形下，分别增长了 28.6% 与 31.5%。这是因为处于紧急区域的 Web 服务器的响应时间降低了，所以吞吐量就提高了。

[0082] 为了实现上述实施例，本发明还提出一种虚拟机实时任务的调度装置。

[0083] 图 6 是根据本发明一个实施例的虚拟机实时任务的调度装置的结构示意图。如图 6 所示，虚拟机实时任务的调度装置包括第一分配模块 100、获取与分配模块 200 和生成模块 300。

[0084] 具体地，分配模块 100 用于当任务的执行周期开始时，分别初始化多个虚拟处理器 VCPU，并为每个 VCPU 分配预设大小的信用值，其中，信用值为自然数。

[0085] 获取与分配模块 200 用于分别获取每个 VCPU 的状态信息，并为每个 VCPU 分配指定的紧急度 Urgency 值。更具体地，VCPU 的状态信息包括 boost 状态、urgent 状态、under 状态、over 状态和 idle 状态。具体地，基于信用调度算法，本发明中引入了一种新的 VCPU 的状态信息，即 urgent 状态。当某个 VCPU 仍然具有剩余的信用值，且该 VCPU 运行了实时任务时，则该 VCPU 的状态会被指派为 urgent 状态。此外，对于 VCPU 的其它状态信息而言，当某个 VCPU 仍然具有剩余的信用值时，则该 VCPU 的状态会被指派为 under 状态；当某个 VCPU 的信用值已用完时，则该 VCPU 的状态会被指派为 over 状态；当某个 VCPU 此时不需要 PCPU 时，则该 VCPU 的状态会被指派为 idle 状态；当某个 VCPU 处于 idle 状态，且该 VCPU 接收到了 I/O 事件时，则该 VCPU 的状态会被指派为 boost 状态；当某个 VCPU 处于 idle 状态，且该 VCPU 接收到了 I/O 事件时，则该 VCPU 的状态会被指派为 boost 状态。

[0086] 进一步而言，为了保证 PCPU 及时响应实时任务，获取与分配模块 200 为每个 VCPU 分配指定的紧急度 Urgency 值。其中，Urgency 值为小于等于 10 的自然数。本发明的方法和现有的信用调度算法不同的是，处于 urgent 状态的 VCPU 在 PCPU 的运行队列中不是以先进先出的方式排列的。在本发明中是通过 Urgency 值来衡量这些 VCPU 的实时任务的紧急度，Urgency 值越大，VCPU 实时任务的紧急度越高。应当理解的是，Urgency 值的默认值是 0，也就是说，如果某个 VCPU 处于 boost 状态，或者 under 状态，或者 over 状态，或者 idle 状态时，获取与分配模块 200 为该 VCPU 分配指定的 Urgency 值为 0，这意味着该 VCPU 上的任务并不是实时的，可以推迟处理。其中，VCPU 的 Urgency 值可以是用户设定的，或者是任务中被指定的。

[0087] 生成模块 300 用于根据状态信息和 Urgency 值生成优先级队列，以使物理处理器 PCPU 根据优先级队列选择对应的 VCPU 执行任务。

[0088] 本发明实施例的虚拟机实时任务的调度装置，添加了新的 VCPU 的状态信息 urgent 状态用以标识实时任务，使其在单一的 PCPU 的运行队列中优先级高于 under 状态。因此，调度方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0089] 图 7 是根据本发明一个具体实施例的虚拟机实时任务的调度装置的结构示意图。如图 7 所示，虚拟机实时任务的调度装置包括第一分配模块 100、获取与分配模块 200 和生成模块 300，其中，生成模块 300 包括 :第一排序单元 310、第二排序单元 330、第三排序单元 340 和生成单元 320。

[0090] 具体地，第一排序单元 310 用于根据每个 VCPU 状态信息的优先级对 VCPU 进行排序。更具体地，第一排序单元 310 可以根据每个 VCPU 状态信息的优先级对 VCPU 进行排序，其中，VCPU 的状态信息的优先级为 boost 状态的优先级 >urgent 状态的优先级 >under 状态的优先级 >over 状态的优先级 >idle 状态的优先级。

[0091] 生成单元 320 用于根据排序结果生成优先级队列。更具体地，生成单元 320 根据排序结果按照 VCPU 状态信息的优先级从高至低的顺序生成优先级队列。然后，PCPU 可根据优先级队列，选择排在优先级队列中队首的 VCPU 执行任务。

[0092] 第二排序单元 330 用于当多个 VCPU 的状态信息相同时，根据多个 VCPU 的 Urgency 值对多个 VCPU 进行排序，生成单元 320 还用于根据排序结果生成优先级队列。更具体地，如果其中某些 VCPU 的具有相同的状态信息，则第二排序单元 330 根据 VCPU 被分配指定的 Urgency 值对这些 VCPU 进行排序，生成单元 320 根据排序结果按照 VCPU 的 Urgency 值从高至低的顺序生成优先级队列。

[0093] 第三排序单元 340 用于当多个 VCPU 的状态信息以及状态信息对应的 Urgency 相同时，根据先进先出的规则对多个 VCPU 进行排序，生成单元 320 还用于根据排序结果生成优先级队列。也就是说，只有具有相同 Urgency 值，即具有相同紧急度的处于 urgent 状态的 VCPU，PCPU 才使用先进先出的方式选择 VCPU 执行任务，由此，可以保证紧急度高的实时任务先被执行。

[0094] 本发明实施例的虚拟机实时任务的调度装置，添加了新的 VCPU 的状态信息 urgent 状态用以标识实时任务，使其在单一的 PCPU 的运行队列中优先级高于 under 状态。因此，调度方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0095] 图 8 是根据本发明另一个具体实施例的虚拟机实时任务的调度装置的结构示意图。如图 8 所示，虚拟机实时任务的调度装置包括第一分配模块 100、获取与分配模块 200、生成模块 300、更新模块 400、计算模块 500 和第二分配模块 600，其中，生成模块 300 包括 :第一排序单元 310、第二排序单元 330、第三排序单元 340 和生成单元 320。

[0096] 具体地，更新模块 400 用于当所述 VCPU 执行任务完成后，从所述 VCPU 分配的所述预设大小的信用值中减去对应的信用值，并根据剩余信用值和所述 VCPU 的状态信息获取

所述 VCPU 更新后的状态信息。

[0097] 计算模块 500 用于计算所有的所述 VCPU 的剩余信用值的总和。

[0098] 第二分配模块 600 用于当所述 VCPU 的剩余信用值的总和等于 0 时，重新为每个 VCPU 分配预设大小的信用值。

[0099] 本发明实施例的虚拟机实时任务的调度装置，添加了新的 VCPU 的状态信息 urgent 状态用以标识实时任务，使其在单一的 PCPU 的运行队列中优先级高于 under 状态。因此，调度方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0100] 图 9 是根据本发明又一个具体实施例的虚拟机实时任务的调度装置的结构示意图。如图 9 所示，虚拟机实时任务的调度装置包括第一分配模块 100、获取与分配模块 200、生成模块 300、更新模块 400、计算模块 500 和第二分配模块 600，其中，生成模块 300 包括：第一排序单元 310、第二排序单元 330、第三排序单元 340 和生成单元 320，更新模块 400 包括：第一更新单元 410、第二更新单元 420、第三更新单元 430、第四更新单元 440、第五更新单元 450、第六更新单元 460、第七更新单元 470、第八更新单元 480、第九更新单元 490、第十更新单元 4100 和第十一更新单元 4110。

[0101] 具体地，第一更新单元 410 用于在 VCPU 处于 boost 状态，且 VCPU 的剩余信用值大于 0，且 VCPU 的 Urgency 值等于 0 时，将 VCPU 的状态信息更新为 under 状态。

[0102] 第二更新单元 420 用于在 VCPU 处于 boost 状态，且 VCPU 的剩余信用值大于 0，且 VCPU 的 Urgency 值大于 0 时，将 VCPU 的状态信息更新为 urgent 状态 .

[0103] 第三更新单元 430 用于在 VCPU 处于 boost 状态，且 VCPU 的剩余信用值等于 0 时，将 VCPU 的状态信息更新为 over 状态。

[0104] 第四更新单元 440 用于在 VCPU 处于 urgent 状态，且 VCPU 接收到 I/O 事件时，将 VCPU 的状态信息更新为 boost 状态。

[0105] 第五更新单元 450 用于在 VCPU 处于 urgent 状态，且 VCPU 的剩余信用值等于 0 时，将 VCPU 的状态信息更新为 over 状态。

[0106] 第六更新单元 460 用于在 VCPU 处于 urgent 状态，且 VCPU 的 I/O 阻塞时，将 VCPU 的状态信息更新为 idle 状态。

[0107] 第七更新单元 470 用于在 VCPU 处于 under 状态，且 VCPU 的剩余信用值等于 0 时，将 VCPU 的状态信息更新为 over 状态。

[0108] 第八更新单元 480 用于在 VCPU 处于 under 状态，且 VCPU 的 I/O 阻塞时，将 VCPU 的状态信息更新为 idle 状态。

[0109] 第九更新单元 490 用于在 VCPU 处于 over 状态，且 VCPU 重新分配信用值，且 VCPU 的 Urgency 值等于 0 时，将 VCPU 的状态信息更新为 under 状态。

[0110] 第十更新单元 4100 用于在 VCPU 处于 over 状态，且 VCPU 重新分配信用值，且 VCPU 的 Urgency 值大于 0 时，将 VCPU 的状态信息更新为 urgent 状态。

[0111] 第十一更新单元 4110 用于在 VCPU 处于 idle 状态，且 VCPU 接收到 I/O 事件时，将 VCPU 的状态信息更新为 boost 状态。

[0112] 本发明实施例的虚拟机实时任务的调度装置，添加了新的 VCPU 的状态信息

urgent 状态用以标识实时任务，使其在单一的 PCPU 的运行队列中优先级高于 under 状态。因此，调度方法会在每个周期优先切换具有 urgent 状态的 VCPU，由此，实时任务可以被及时响应。实验使用 ping 延迟与 web 服务器的吞吐量表明这种调度方法可以及时处理实时任务，并提高了响应时间的稳定性。

[0113] 为了实现上述实施例，本发明还提出一种虚拟机。

[0114] 一种虚拟机包括本发明任一项实施例所述的虚拟机实时任务的调度装置。

[0115] 应当理解，本发明的各部分可以用硬件、软件、固件或它们的组合来实现。在上述实施方式中，多个步骤或方法可以用存储在存储器中且由合适的指令执行系统执行的软件或固件来实现。例如，如果用硬件来实现，和在另一实施方式中一样，可用本领域公知的下列技术中的任一项或他们的组合来实现：具有用于对数据信号实现逻辑功能的逻辑门电路的离散逻辑电路，具有合适的组合逻辑门电路的专用集成电路，可编程门阵列(PGA)，现场可编程门阵列(FPGA)等。

[0116] 在本说明书的描述中，参考术语“一个实施例”、“一些实施例”、“示例”、“具体示例”、或“一些示例”等的描述意指结合该实施例或示例描述的具体特征、结构、材料或者特点包含于本发明的至少一个实施例或示例中。在本说明书中，对上述术语的示意性表述不一定指的是相同的实施例或示例。而且，描述的具体特征、结构、材料或者特点可以在任何的一个或多个实施例或示例中以合适的方式结合。

[0117] 尽管已经示出和描述了本发明的实施例，本领域的普通技术人员可以理解：在不脱离本发明的原理和宗旨的情况下可以对这些实施例进行多种变化、修改、替换和变型，本发明的范围由权利要求及其等同物限定。

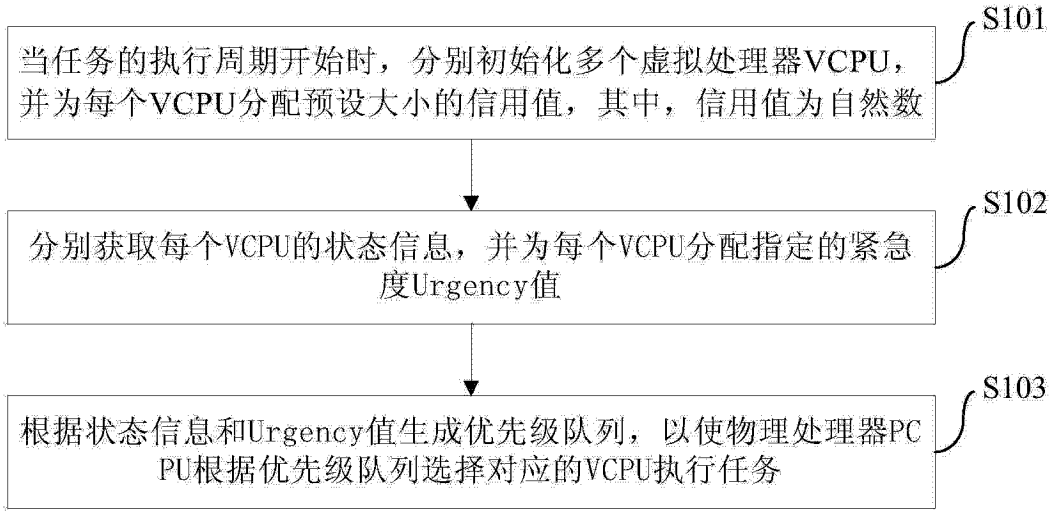

S101
当任务的执行周期开始时，分别初始化多个虚拟处理器VCPU，并为每个VCPU分配预设大小的信用值，其中，信用值为自然数
S102
分别获取每个VCPU的状态信息，并为每个VCPU分配指定的紧急度Urgency值
S103
根据状态信息和Urgency值生成优先级队列，以使物理处理器PCPU根据优先级队列选择对应的VCPU执行任务

图 1

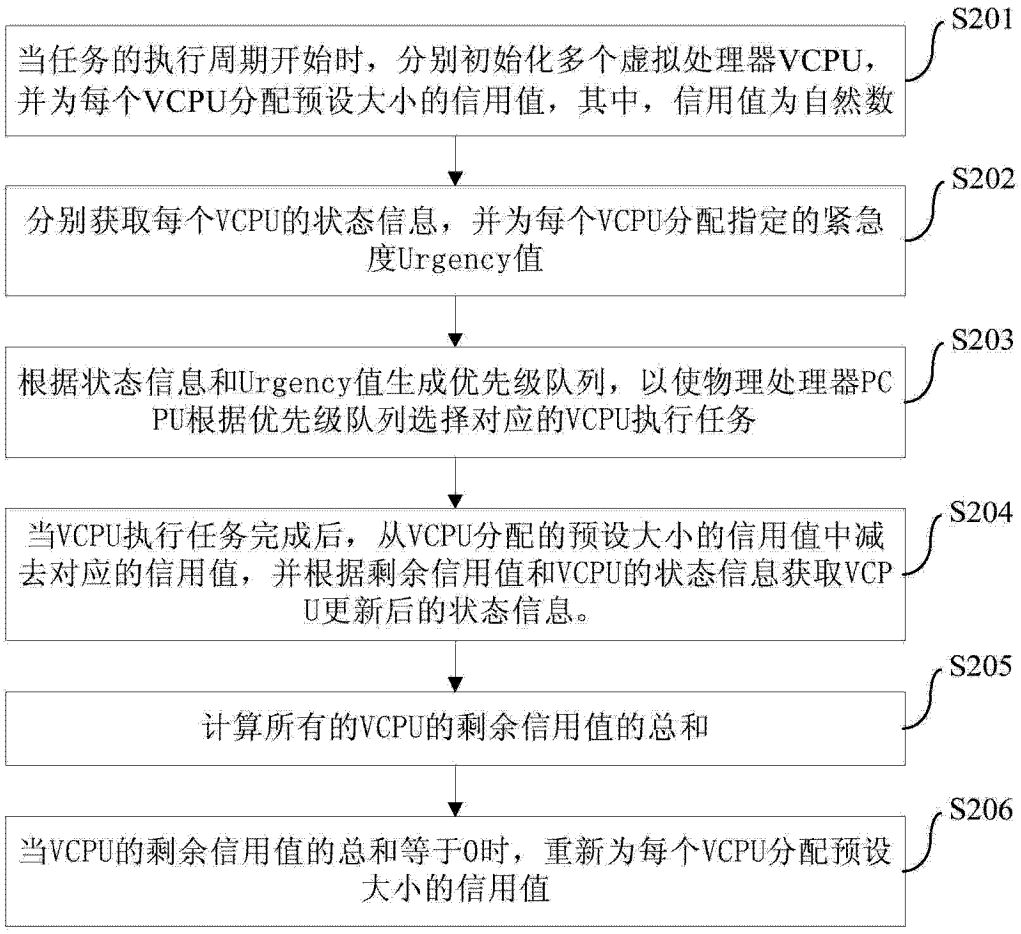

S201
当任务的执行周期开始时，分别初始化多个虚拟处理器VCPU，并为每个VCPU分配预设大小的信用值，其中，信用值为自然数
S202
分别获取每个VCPU的状态信息，并为每个VCPU分配指定的紧急度Urgency值
S203
根据状态信息和Urgency值生成优先级队列，以使物理处理器PCPU根据优先级队列选择对应的VCPU执行任务
S204
当VCPU执行任务完成后，从VCPU分配的预设大小的信用值中减去对应的信用值，并根据剩余信用值和VCPU的状态信息获取VCPU更新后的状态信息。
S205
计算所有的VCPU的剩余信用值的总和
S206
当VCPU的剩余信用值的总和等于0时，重新为每个VCPU分配预设大小的信用值

图 2

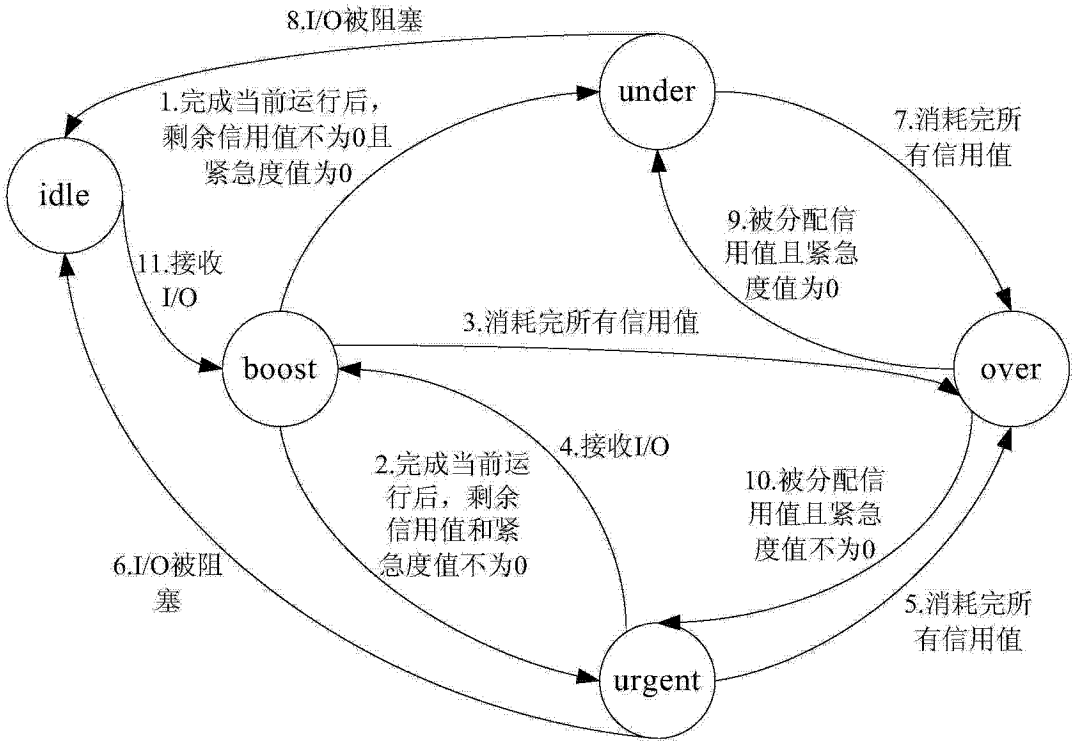

8.I/O被阻塞
1.完成当前运行后，剩余信用值不为0且紧急度值为0
under
7.消耗完所有信用值
idle
9.被分配信用值且紧急度值为0
11.接收I/O
3.消耗完所有信用值
boost
over
4.接收I/O
2.完成当前运行后，剩余信用值和紧急度值不为0
10.被分配信用值且紧急度值不为0
6.I/O被阻塞
5.消耗完所有信用值
urgent

图 3

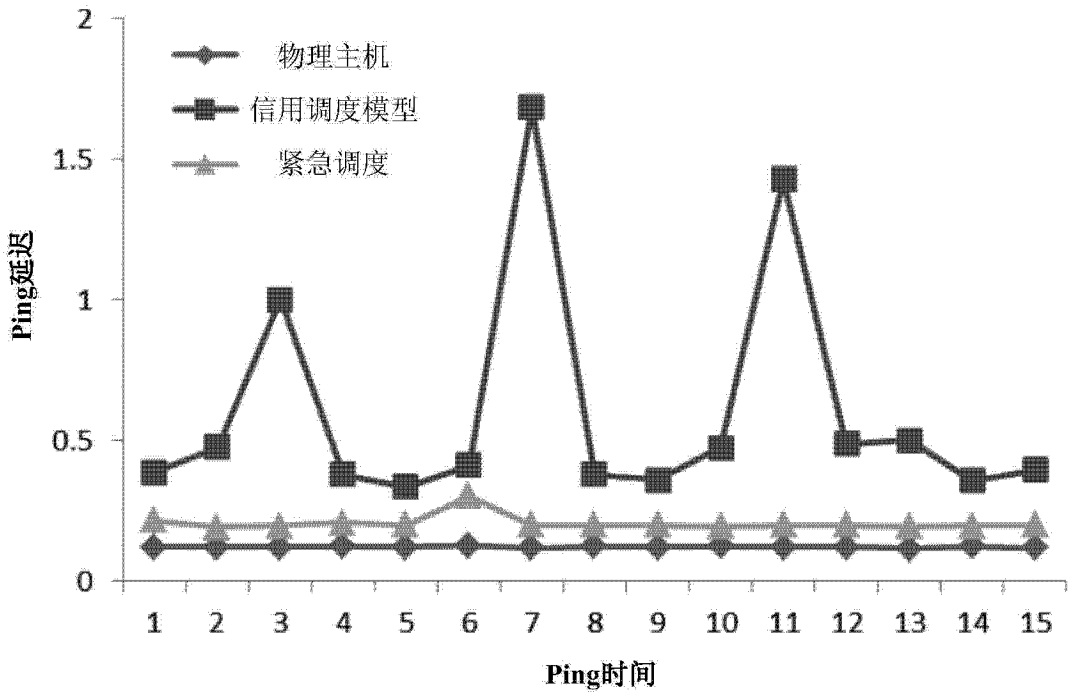

物理主机
信用调度模型
紧急调度
Ping延迟
Ping时间
0
0.5
1
1.5
2
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

图 4

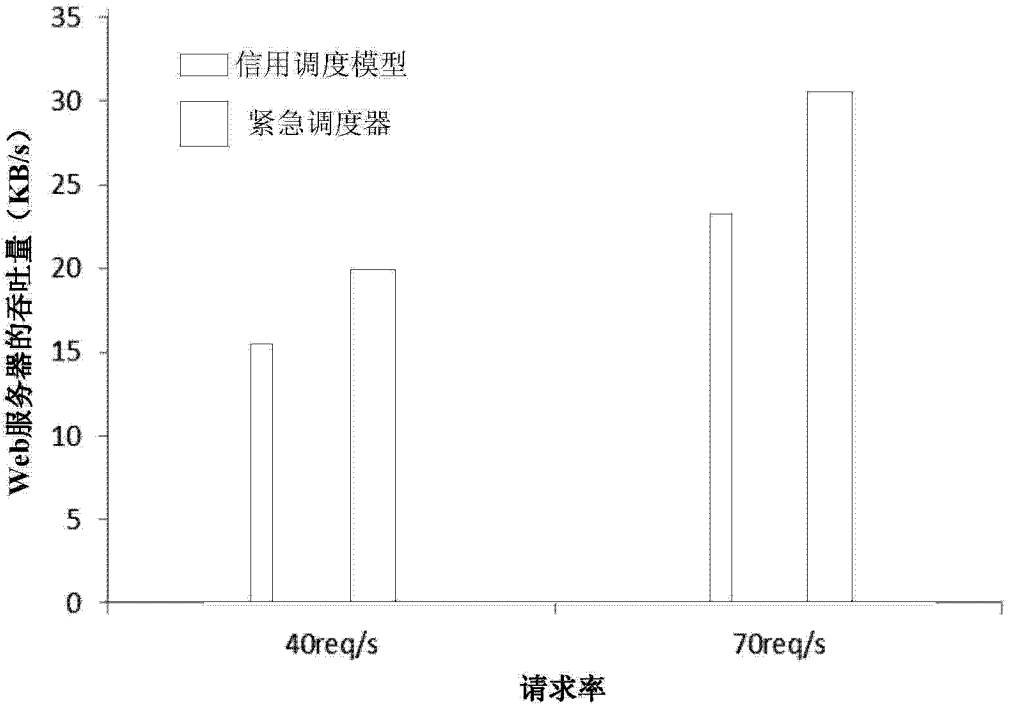

35
30
25
20
15
10
5
0
信用调度模型
紧急调度器
Web服务器的吞吐量（KB/s）
40req/s
70req/s
请求率

图 5


第一分配模块100
获取与分配模块200
生成模块300

图 6

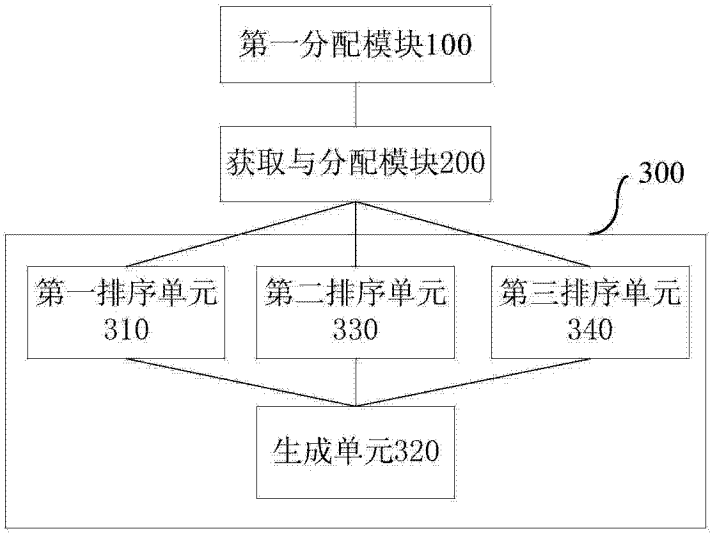

第一分配模块100
获取与分配模块200
300
第一排序单元
310
第二排序单元
330
第三排序单元
340
生成单元320

图 7

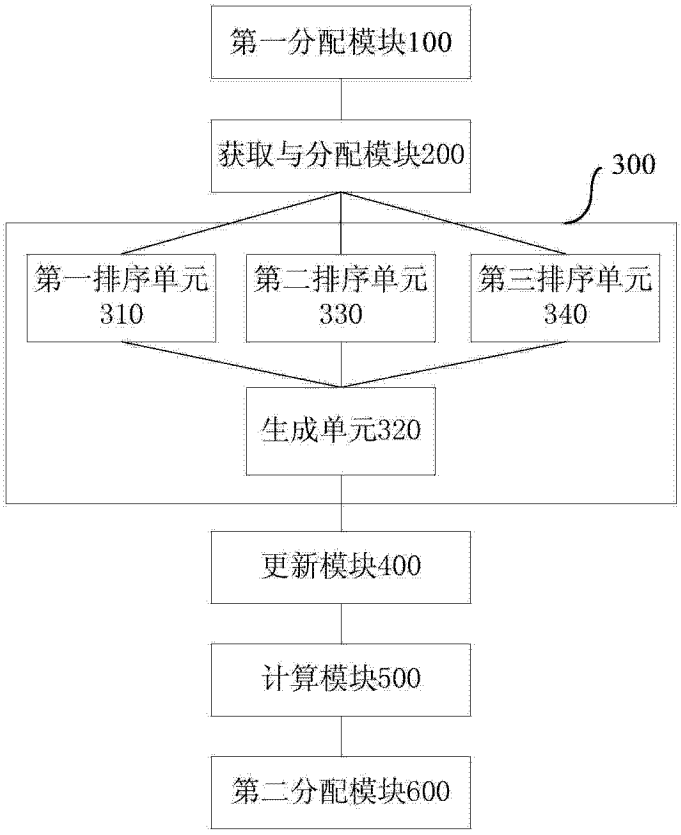

第一分配模块100
获取与分配模块200
300
第一排序单元
310
第二排序单元
330
第三排序单元
340
生成单元320
更新模块400
计算模块500
第二分配模块600

图 8

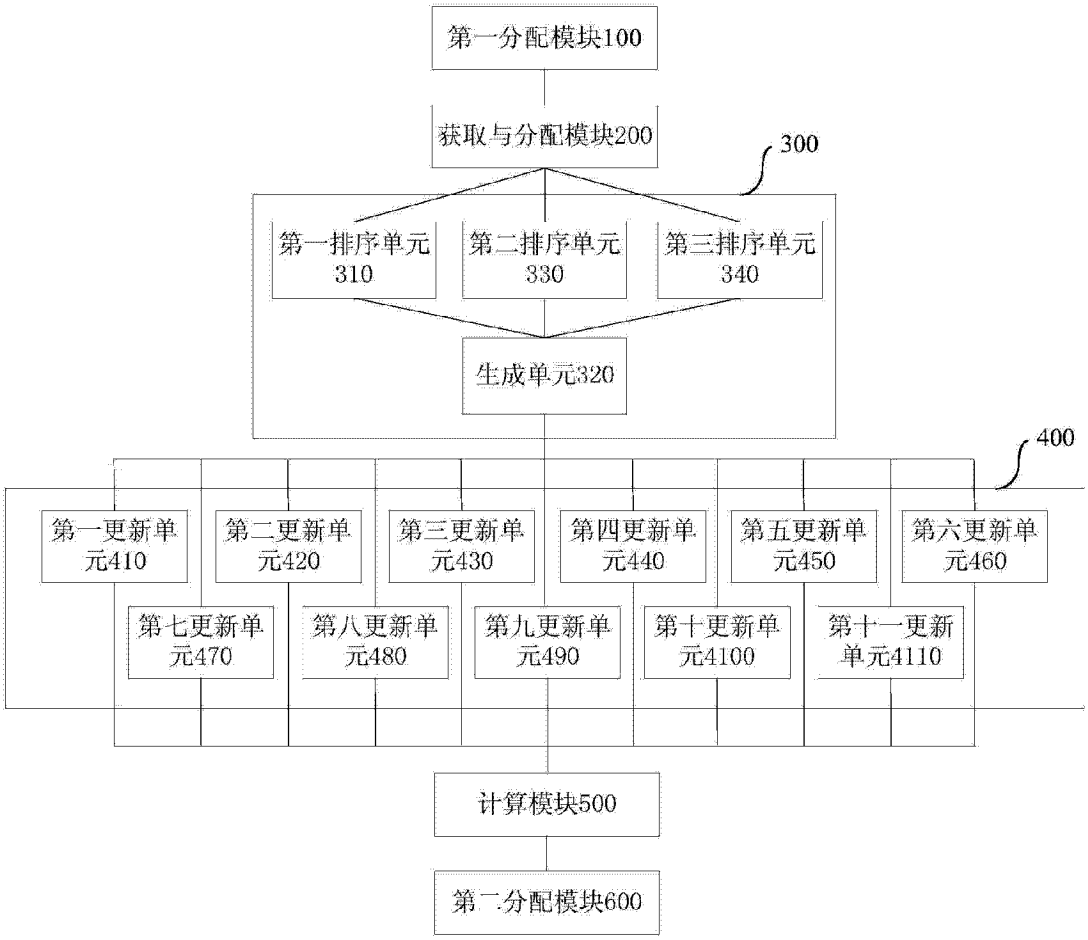

第一分配模块100
获取与分配模块200
300
第一排序单元310
第二排序单元330
第三排序单元340
生成单元320
400
第一更新单元410
第二更新单元420
第三更新单元430
第四更新单元440
第五更新单元450
第六更新单元460
第七更新单元470
第八更新单元480
第九更新单元490
第十更新单元4100
第十一更新单元4110
计算模块500
第二分配模块600

图 9