(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 101719082 B
(45)授权公告日 2013.01.02

(21)申请号 200910244006.2
(22)申请日 2009.12.24
(73)专利权人 中国科学院计算技术研究所
地址 100080 北京市海淀区中关村科学院南路 6 号
(72)发明人 冯斌全 宋莹 王若倪 孙毓忠
(74)专利代理机构 北京律诚同业知识产权代理有限公司 11006
代理人 祁建国 梁挥
(51)Int.Cl.
G06F 9/50(2006.01)
G06F 9/455(2006.01)
(56)对比文件
CN 101408853 A,2009.04.15,
CN 101470634 A,2009.07.01,
CN 101540776 A,2009.09.23,
CN 101115016 A,2008.01.30,
龚梅 王鹏
吴跃 . 一种集群系统的透明动态反馈负载均衡算法 .《计算机应用》.2007, 第 27 卷 (第 11 期),
审查员 陈婕

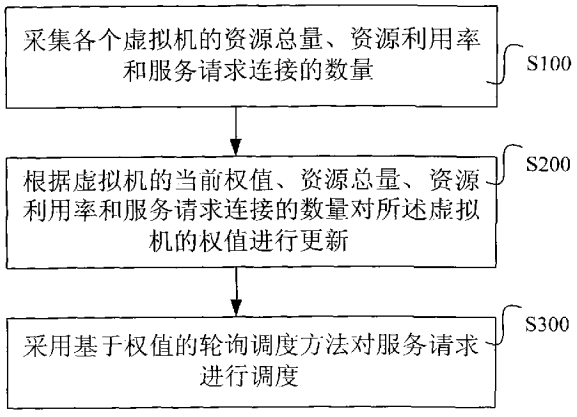
权利要求书 3 页 说明书 12 页 附图 3 页

(54)发明名称

虚拟化计算平台中应用请求调度的方法及其系统

(57)摘要

本发明涉及虚拟化计算平台中应用请求调度的方法及其系统,方法包括 :步骤 1,采集各个虚拟机的资源总量、资源利用率和服务请求连接的数量 ;步骤 2,根据虚拟机的当前权值、资源总量、资源利用率和服务请求连接的数量对所述虚拟机的权值进行更新 ;步骤 3,采用基于权值的轮询调度方法对服务请求进行调度。本发明能够在虚拟机系统中实现资源与负载匹配,提高整个虚拟化计算平台的利用效率。



CN 101719082 B

1. 一种虚拟化计算平台中应用请求调度的方法，其特征在于，包括：

步骤 1，采集各个虚拟机的资源总量、资源利用率和服务请求连接的数量；

步骤 2，根据虚拟机的当前权值、资源总量、资源利用率和服务请求连接的数量对所述虚拟机的权值进行更新；

步骤 3，采用基于权值的轮询调度方法对服务请求进行调度；

所述步骤 3 后还包括：

步骤 61，计算虚拟机预设时长的平均资源利用率，根据该平均资源利用率纠正调度偏差。

2. 如权利要求 1 所述的虚拟化计算平台中应用请求调度的方法，其特征在于，所述步骤 2 进一步为：

步骤 21，根据虚拟机的资源总量、资源利用率和服务请求连接的数量判断所述虚拟机是否超载，如果超载，则将所述虚拟机的权值记为预设最小值，否则，执行步骤 22；

步骤 22，根据虚拟机的资源利用率、服务请求连接的数量和当前的权值计算所述虚拟机新的权值，按所述新的权值对所述虚拟机进行更新。

3. 如权利要求 2 所述的虚拟化计算平台中应用请求调度的方法，其特征在于，所述步骤 21 进一步为：

步骤 31，由虚拟机的资源总量和资源利用率计算出虚拟机的可用资源量，将所述虚拟机的可用资源量和服务请求连接数量带入性能预测模型计算出所述虚拟机的最大连接数；

步骤 32，如果所述虚拟机的服务请求的连接数量大于等于所述虚拟机的最大连接数，则将所述虚拟机的权值记为预设最小值；否则，执行所述步骤 22。

4. 如权利要求 2 所述的虚拟化计算平台中应用请求调度的方法，其特征在于，所述步骤 22 进一步为：

步骤 41，根据资源利用率和服务请求连接的数量计算出综合负载值；

步骤 42，根据所述虚拟机的综合负载值和所述虚拟机的当前的权值算出所述虚拟机的新的权值；

步骤 43，如果所述虚拟机的新的权值和所述虚拟机的当前的权值的差值大于设定的阀值，则将所述虚拟机的权值更新为所述新的权值。

5. 如权利要求 4 所述的虚拟化计算平台中应用请求调度的方法，其特征在于，所述步骤 42 进一步为：

按如下公式计算所述虚拟机的新的权值，

$$w_{newi} = w_i + K \times \sqrt[3]{1 - Server\_load_i}$$，

其中，w_{newi} 为新的权值，w_i 当前权值，K 为预设的调整系数，$Server\_load_i$ 为综合负载值。

6. 如权利要求 1 所述的虚拟化计算平台中应用请求调度的方法，其特征在于，所述步骤 1 前还包括：

步骤 71，在初始时，采用加权最小连接调度算法，并为虚拟机群中虚拟机设置初始的权值。

7. 如权利要求 1 所述的虚拟化计算平台中应用请求调度的方法，其特征在于，所述步骤 1 前还包括：

步骤 81，在一个物理机器上配置不同资源密集型的服务对应的虚拟机。

8. 一种虚拟化计算平台中应用请求调度的系统，包括运行于物理节点上的虚拟机，其特征在于，系统还包括：

中控模块，用于采集各个虚拟机的资源总量，并将各个虚拟机的资源总量反馈给调度器模块；

所述调度器模块，用于采集各个虚拟机的资源利用率和服务请求连接的数量；

所述调度器模块还用于根据虚拟机的当前权值、资源总量、资源利用率和服务请求连接的数量对所述虚拟机的权值进行更新；

所述中控模块用于接收服务请求，并将所述服务请求转发给所述调度器模块；

所述调度器模块还用于采用基于权值的轮询调度方法对服务请求进行调度；

所述系统还包括偏差纠正模块，用于计算虚拟机预设时长的平均资源利用率，根据该平均资源利用率纠正调度偏差。

9. 如权利要求 8 所述的虚拟化计算平台中应用请求调度的系统，其特征在于，

所述调度器模块在更新权值时进一步用于根据虚拟机的资源总量、资源利用率和服务请求连接的数量判断所述虚拟机是否超载，如果超载，则将所述虚拟机的权值记为预设最小值，否则，根据虚拟机的资源利用率、服务请求连接的数量和当前的权值计算所述虚拟机新的权值，按所述新的权值对所述虚拟机进行更新。

10. 如权利要求 9 所述的虚拟化计算平台中应用请求调度的系统，其特征在于，

所述调度器模块在判断超载时进一步用于由虚拟机的资源总量和资源利用率计算出虚拟机的可用资源量，将所述可用资源量发送给所述中控模块，接收所述中控模块反馈的最大连接数，如果所述虚拟机的服务请求的连接数量大于等于所述虚拟机的最大连接数，则判断为超载；否则，判断为不超载；

所述中控模块还用于将所述虚拟机的可用资源量和服务请求连接数量带入性能预测模型计算出所述虚拟机的最大连接数，将所述最大连接数反馈给所述调度器模块。

11. 如权利要求 9 所述的虚拟化计算平台中应用请求调度的系统，其特征在于，

所述调度器模块在更新虚拟机的权值时进一步用于根据资源利用率和服务请求连接的数量计算出综合负载值；根据所述虚拟机的综合负载值和所述虚拟机的当前的权值算出所述虚拟机的新的权值；如果所述虚拟机的新的权值和所述虚拟机的当前的权值的差值大于设定的阀值，则将所述虚拟机的权值更新为所述新的权值。

12. 如权利要求 11 所述的虚拟化计算平台中应用请求调度的系统，其特征在于，

所述调度器模块计算综合负载值时进一步用于按如下公式计算所述虚拟机的新的权值，

$$w_{newi} = w_i + K \times \sqrt[3]{1 - Server\_load_i}\ ,$$

其中，w_{newi} 为新的权值，w_i 当前权值，K 为预设的调整系数，$Server\_load_i$ 为综合负载值。

13. 如权利要求 8 所述的虚拟化计算平台中应用请求调度的系统，其特征在于，

所述调度器模块还用于在初始时，采用加权最小连接调度算法，并为虚拟机群中虚拟

机设置初始的权值。

14. 如权利要求 8 所述的虚拟化计算平台中应用请求调度的系统，其特征在于，在一个物理机器上配置不同资源密集型的服务对应的虚拟机。

虚拟化计算平台中应用请求调度的方法及其系统

技术领域

[0001] 本发明涉及到虚拟集群领域，尤其涉及虚拟化计算平台中应用请求调度的方法及其系统。

背景技术

[0002] 现有技术中，一种提高资源利用率的实现方法是利用虚拟化技术将大量网络服务整合到一个共享的框架下，将整体资源在各个虚拟机间分配。

[0003] 现有技术中实现虚拟化技术的方法之一为 Web 集群方法。Web 集群是指由两台或者多台服务器通过网络联合起来，共同处理用户请求的网络服务站点。虽然大规模集群具有大量的服务器节点，但是它使用一台主机给所有用户提供唯一的访问接口。为了控制所有的请求到达站点并屏蔽掉后台的分布式服务器结构，Web 集群提供一个网络交换器，使用单一的虚拟 IP 地址来关联后台的实际服务器，其为网络调度器。在集群中的所有服务器都被放置在相同的物理空间中。网络请求流经过网络交换器将请求分发到各个网络服务器中。网络交换器接收客户端的服务请求，并从网络服务器池中选择一台进行后续的处理操作。在网络集群服务器中，调度方法为根据网络客户信息和服务器资源总量和资源利用率进行调度。该调度方法的缺点在于无法对动态伸缩的虚拟机进行感知。动态伸缩的虚拟机是指在虚拟化环境中虚拟机的资源可以按需进行流动分配。

[0004] 目前的虚拟化技术仅仅实现了资源在物理节点内的流动，也就是资源流动不能跨节点。对于大规模 Web 服务器虚拟集群构成的环境来说，它将导致能力流动和应用负载匹配的问题。一个应用的网络请求被调度到了分布在不同物理节点的虚拟机上。虚拟机跨节点分布，而资源流动层对其它物理节点上的虚拟机资源流动不可知，如果仅仅被动依靠分配过来的负载来自适应地按需流动本地资源，就会忽略其它物理节点上该服务的宏观调度信息，最后的结果总是会造成能力流动和应用负载失配，表现为当虚拟机资源扩展的时候负载没有分配进来或者资源收缩了以后负载还频繁调度进来。甚至因为能力流动无法感知请求调度，而使得请求在各个动态伸缩的虚拟机之间，形成负载的分配不均，不能充分利用系统资源来提高服务吞吐率。

[0005] 一个虚拟机上 CPU 资源能力流动和应用负载匹配的模式如图 1 所示。上图是线性匹配效果，下图是经过 DTW(Dynamic Time Wrap，动态时间包络）处理后的包络图。每个图中上面的曲线说明了该虚拟机的总资源流动情况，它的凹凸表示了资源的流出和流进，下面的曲线表示负载对资源的使用情况，它的凹凸间接表示了应用负载分配数量的减少和增加。从图 1 可见，需要达到的最佳效果是两条曲线的包络线之间的空间越小越好，如公式 (1) 所示。因为它表示流进来虚拟机的资源都能有应用负载能够分配进来消耗这部分资源，这就是能力流动和应用负载的匹配。

[0006] $$Min\sum_{i=1}^{N} Allocate_i - Consume_i \qquad (1)$$

[0007] 在虚拟互联网服务集群系统中，由于虚拟资源动态伸缩，如果仅仅只是通过节点

内对资源按需流动来进行管理，解决不了能力流动和应用负载匹配的问题的。该匹配有两个层面的问题，单纯从一个节点上观察这两条匹配曲线，当然是使用率越高，波峰波谷越对应越好；但是从整个调度系统上看，应该是尽量负载均衡的情况下，匹配度越高越好。因为某些节点利用率高，其它节点空闲的情况不利于充分利用整个分布式系统的可用资源。

发明内容

[0008] 为解决上述问题，本发明提供了虚拟化计算平台中应用请求调度的方法及其系统，能够在虚拟机系统中实现资源与负载匹配，提高整个虚拟化计算平台的利用效率。

[0009] 本发明公开了一种虚拟化计算平台中应用请求调度的方法，包括：

[0010] 步骤 1，采集各个虚拟机的资源总量、资源利用率和服务请求连接的数量；

[0011] 步骤 2，根据虚拟机的当前权值、资源总量、资源利用率和服务请求连接的数量对所述虚拟机的权值进行更新；

[0012] 步骤 3，采用基于权值的轮询调度方法对服务请求进行调度。

[0013] 所述步骤 2 进一步为：

[0014] 步骤 21，根据虚拟机的资源总量、资源利用率和服务请求连接的数量判断所述虚拟机是否超载，如果超载，则将所述虚拟机的权值记为预设最小值，否则，执行步骤 22；

[0015] 步骤 22，根据虚拟机的资源利用率、服务请求连接的数量和当前的权值计算所述虚拟机新的权值，按所述新的权值对所述虚拟机进行更新。

[0016] 所述步骤 21 进一步为：

[0017] 步骤 31，由虚拟机的资源总量和资源利用率计算出虚拟机的可用资源量，将所述虚拟机的可用资源量和服务请求连接数量带入性能预测模型计算出所述虚拟机的最大连接数；

[0018] 步骤 32，如果所述虚拟机的服务请求的连接数量大于等于所述虚拟机的最大连接数，则将所述虚拟机的权值记为预设最小值；否则，执行所述步骤 22。

[0019] 所述步骤 22 进一步为：

[0020] 步骤 41，根据资源利用率和服务请求连接的数量计算出综合负载值；

[0021] 步骤 42，根据所述虚拟机的综合负载值和所述虚拟机的当前的权值算出所述虚拟机的新的权值；

[0022] 步骤 43，如果所述虚拟机的新的权值和所述虚拟机的当前的权值的差值大于设定的阀值，则将所述虚拟机的权值更新为所述新的权值。

[0023] 所述步骤 42 进一步为：

[0024] 按如下公式计算所述虚拟机的新的权值，

[0025] $w_{newi} = w_i + K \times \sqrt[3]{1 - Server\_load_i}$，

[0026] 其中，w_{newi} 为新的权值，w_i 当前权值，K 为预设的调整系数，$Server\_load_i$ 为综合负载值。

[0027] 所述步骤 3 后还包括：

[0028] 步骤 61，计算虚拟机预设时长的平均资源利用率，根据该平均资源利用率纠正调度偏差。

[0029] 所述步骤 1 前还包括：

[0030] 步骤 71，在初始时，采用加权最小连接调度算法，并为虚拟机群中虚拟机设置初始的权值。

[0031] 所述步骤 1 前还包括：

[0032] 步骤 81，在一个物理机器上配置不同资源密集型的服务对应的虚拟机。

[0033] 本发明还公开了一种虚拟化计算平台中应用请求调度的系统，包括运行于物理节点上的虚拟机，系统还包括：

[0034] 中控模块，用于采集各个虚拟机的资源总量，并将各个虚拟机的资源总量反馈给调度器模块；

[0035] 所述调度器模块，用于采集各个虚拟机的资源利用率和服务请求连接的数量；

[0036] 所述调度器模块还用于根据虚拟机的当前权值、资源总量、资源利用率和服务请求连接的数量对所述虚拟机的权值进行更新；

[0037] 所述中控模块用于接收服务请求，并将所述服务请求转发给所述调度器模块；

[0038] 所述调度器模块还用于采用基于权值的轮询调度方法对服务请求进行调度。

[0039] 所述调度器模块在更新权值时进一步用于根据虚拟机的资源总量、资源利用率和服务请求连接的数量判断所述虚拟机是否超载，如果超载，则将所述虚拟机的权值记为预设最小值，否则，根据虚拟机的资源利用率、服务请求连接的数量和当前的权值计算所述虚拟机新的权值，按所述新的权值对所述虚拟机进行更新。

[0040] 所述调度器模块在判断超载时进一步用于由虚拟机的资源总量和资源利用率计算出虚拟机的可用资源量，将所述可用资源量发送给所述中控模块，接收所述中控模块反馈的最大连接数，如果所述虚拟机的服务请求的连接数量大于等于所述虚拟机的最大连接数，则判断为超载；否则，判断为不超载；

[0041] 所述中控模块还用于将所述虚拟机的可用资源量和服务请求连接数量带入性能预测模型计算出所述虚拟机的最大连接数，将所述最大连接数反馈给所述调度器模块。

[0042] 所述调度器模块在更新虚拟机的权值时进一步用于根据资源利用率和服务请求连接的数量计算出综合负载值；根据所述虚拟机的综合负载值和所述虚拟机的当前的权值算出所述虚拟机的新的权值；如果所述虚拟机的新的权值和所述虚拟机的当前的权值的差值大于设定的阀值，则将所述虚拟机的权值更新为所述新的权值。

[0043] 所述调度器模块计算综合负载值时进一步用于按如下公式计算所述虚拟机的新的权值，

[0044] $w_{newi} = w_i + K \times \sqrt[3]{1 - Server\_load_i}$，

[0045] 其中，w_{newi} 为新的权值，w_i 当前权值，K 为预设的调整系数，$Server\_load_i$ 为综合负载值。

[0046] 所述系统还包括偏差纠正模块，用于计算虚拟机预设时长的平均资源利用率，根据该平均资源利用率纠正调度偏差。

[0047] 所述调度器模块还用于在初始时，采用加权最小连接调度算法，并为虚拟机群中虚拟机设置初始的权值。

[0048] 在一个物理机器上配置不同资源密集型的服务对应的虚拟机。

[0049] 本发明的有益效果在于，根据资源的流动和负载分布动态分配请求，使得能力流动和应用负载达到匹配；通过将不同资源密集型的服务对应的虚拟机安装配置在一个物理

机器上，避免对本地资源的竞争带来的冲突；通过每一种服务对应着一个调度器，并且每个调度器有自主决策权，能够保证在采集到的资源信息准确的情况下，避免 herd 效应出现；herd 效应为所有的作业在一个决策信息更新的间隔内，被调度到相同的服务器子集中，而导致的失衡和某一资源的急剧竞争。

附图说明

[0050] 图 1 是现有技术中一个虚拟机上 CPU 资源能力流动和应用负载匹配的示意图；

[0051] 图 2 是本发明虚拟化计算平台中应用请求调度的方法的流程图；

[0052] 图 3 是本发明虚拟化计算平台中应用请求调度的方法的具体实施方式的流程图；

[0053] 图 4 是本发明虚拟化计算平台中应用请求调度的系统的结构图；

[0054] 图 5 是 DFBS 算法工作环境的示意图。

具体实施方式

[0055] 下面结合附图，对本发明做进一步的详细描述。

[0056] 在资源层，采用虚拟化技术，在每个物理服务节点上启动了多个虚拟机，在同一个物理平台上的虚拟机提供着不同的服务，调度的目的是使该些服务尽量充分利用物理平台的资源，比如，如果它们分别运行着 CPU、Memory（内存）和网络密集型的计算，则一个物理平台上的不同硬件资源就能得到充分利用，同时由于虚拟机的隔离使得几个虚拟机的执行没有冲突。

[0057] 为了避免资源使用的冲突，在服务的分发时，尽量将提供相同服务的几个镜像的虚拟机部署到不同的物理节点上，并通过调度算法将该服务的作业分发到该些镜像的虚拟机，以协调一致地完成工作。

[0058] 此外，通过服务优先级和对资源的动态需求，以及调整同一虚拟化计算平台内部资源在构建于其上的虚拟机之间分配，来对多个服务的部分镜像的虚拟机进行调整。同时，还可以在服务的分发调度层对负载进行重分配，来避免某一个物理节点过载的现象。

[0059] 本发明的调度方法如图 2 所示。

[0060] 步骤 S100，采集各个虚拟机的资源总量、资源利用率和服务请求连接的数量。

[0061] 步骤 S200，根据虚拟机的当前权值、资源总量、资源利用率和服务请求连接的数量对该虚拟机的权值进行更新。

[0062] 步骤 S300，采用基于权值的轮询调度方法对服务请求进行调度。

[0063] 本发明的调度方法的具体实施方式如图 3 所示。

[0064] 步骤 S301，初始的时候，在调度器内核中采用加权最小连接调度算法，并为虚拟机群的虚拟机设定一个初始权值。

[0065] 现有技术中，加权最小连接调度是 IPVS(IP Virtual Server，IP 虚拟服务器）实现的内核调度算法，为把新的连接请求分配到当前加权连接数最小的虚拟机，通过虚拟机当前所活跃的连接数来估计虚拟机的负载情况。在该算法中调度器需要记录各个虚拟机已建立连接的数目，当一个请求被调度到某台虚拟机，其连接数加 1；当连接中止或超时，其连接数减一。加权调度可以通过自动问询真实虚拟机的负载情况，并动态地调整其权值。各个虚拟机用相应的权值表示其处理性能。虚拟机缺省的权值为 1，可以动态地设置虚拟机的

权值。加权最小连接调度在调度新连接时尽可能使虚拟机的已建立连接数和其权值成比例具有较高的权值的虚拟机将承受较大比例的活动连接负载。

[0066] 步骤 S302，采集各个虚拟机的资源总量、资源利用率和服务请求连接的数量。

[0067] 虚拟机所在的物理节点的资源监控进程采集各个虚拟机的资源总量和资源利用率，同时调度器的应用请求监控进程，采集到达各个虚拟机的服务请求连接数量。

[0068] 步骤 S303，根据虚拟机的资源总量、资源利用率和服务请求连接的数量判断所述虚拟机是否超载，如果超载，则将所述虚拟机的权值记为预设最小值，否则，执行步骤 S304。

[0069] 中控节点获得虚拟机的资源总量、资源利用率、请求连接数量通过该中控节点发送广播请求从物理节点获得；或者由物理节点定期发送给中控节点。

[0070] 同时，中控节点还不时向虚拟机发送请求，以测量虚拟机的响应时间，用来评估当前虚拟机的真实负载，以确定当前服务质量。

[0071] 由虚拟机的资源总量和资源利用率计算出虚拟机的可用资源量，将所述虚拟机的可用资源量和服务请求连接数量带入性能预测模型计算出所述虚拟机的最大连接数，如果所述虚拟机的服务请求的连接数量大于等于所述虚拟机的最大连接数，则将所述虚拟机的权值记为预设最小值；否则，执行所述步骤 S304。

[0072] 实施例中预设最小值为 0。

[0073] 可用资源量的计算方法为，可用资源量＝（1- 资源利用率）× 资源总量

[0074] 性能预测模型通过输入的服务连接数量和可用资源量，输出虚拟机能承受的最大连接数。性能预测模型是根据服务性能和请求速率与可用资源量的曲线图，进行曲线拟合与回归分析，抽象出来的通过服务连接数和可用资源量计算出最大连接数的公式。现有技术中，基于实际的线下测试获得的结果进行的拟合，对不同的服务，如网页服务，数据库服务，办公应用服务等，根据实际数据产生对应的公式。

[0075] 在调度算法中，当虚拟机的权值为零，已建立的连接会继续得到该虚拟机的服务，而新的连接不会分配到该虚拟机。可以将一台虚拟机的权值设置为零，使得该虚拟机安静下来，当已有的连接都结束后，又可以将该虚拟机切出，对其进行维护。所以，在动态反馈负载均衡机制中要保证该功能，当虚拟机的权值为零时，不对虚拟机的权值进行调整。

[0076] 根据给定的性能目标，比如 3 秒的平均响应时间，和当前虚拟机已经分配的资源，代入性能模型计算，预测出该虚拟机能支持的最大连接数，如果当前的动态连接数超过这个最大连接数，那么就将权值就置零，保证该节点的服务质量。如果所有的虚拟机都达到最大连接数，那么必须抛弃部分的请求，进行访问接纳控制。

[0077] 步骤 S304，根据虚拟机的资源利用率、服务请求连接的数量和当前的权值计算所述虚拟机新的权值，按所述新的权值对所述虚拟机进行更新。

[0078] 根据资源利用率和服务请求连接的数量计算出综合负载值；根据所述虚拟机的综合负载值和所述虚拟机的当前的权值算出所述虚拟机的新的权值；如果所述虚拟机的新的权值和所述虚拟机的当前的权值的差值大于设定的阀值，则将所述虚拟机的权值更新为所述新的权值。

[0079] 综合负载值是根据未来的负载预测、当前的负载监控和实际性能负荷这三个值乘上相应的系数而得到的值，用于综合评估服务器的负载状况。对于不同的应用，各种负载指标的对综合负载贡献的权重不同，因此引入各个指标信息的系数来表示各个负载指标在综

合负载中比重。

[0080] 计算综合负载的方法如下所述。

[0081] 步骤 S304a，计算各个虚拟机的请求负载。

[0082] 请求负载是虚拟机收到的请求负载数量，收到的请求量越大说明虚拟机需要执行的工作量越多，负载越重。

[0083] 请求负载定义为在单位时间内虚拟机收到的新连接数与平均连接数的比例，它是在调度器上收集到的。平均连接数是从初始时间到当前时间内的平均连接数。

[0084] 该值越大说明分配的负载越多，所以服务器的需要处理的工作负载就越大。在调度器上维护有各个虚拟机收到连接数的计数器，对于虚拟机 Si，可以得到分别在时间 T1 和 T2 时的计数器值 Ci1 和 Ci2，计算出在时间间隔 T2-T1 内虚拟机 Si 收到新连接数 Ni = Ci2-Ci1。依此方法，得到一组虚拟机在时间间隔 T2-T1 内收到新连接数的集合。请求连接的数量和应用负载成正比，被分配到的连接数越大说明该虚拟机要处理的工作量越大，负载也越重。虚拟机 Si 的请求负载的计算公式为

[0085] $$Request\_load_i = \frac{N_i}{\sum_{j=1}^{n} \frac{N_j}{n}}$$

[0086] 其中，$Request\_load_i$ 为虚拟机 Si 的请求负载，N_i 为虚拟机 Si 收到的新连接数，n 为服务器总数，N_j 为虚拟机 Sj 收到的新连接数。从上面公式上看，请求负载的数据值以所有虚拟机收到的连接数平均值为分母，其落在 [0，∞）的区间内，大于 1 表示超载，等于 1 负载刚好，小于 1 说明服务器负载较轻，这个数值越大，表示应用负载越重。

[0087] 步骤 S304b，计算各个虚拟机的虚拟机负载。

[0088] 虚拟机负载是虚拟机当前执行的作业队列工作量，虚拟机负载越大说明正在执行的工作队列越长。

[0089] 虚拟机负载记录虚拟机各种资源负载信息，如虚拟机的当前 CPU 负载、当前内存利用情况等。因为网络服务主要是受到 CPU 和 Memory 影响，所以在本实施例中，该两个虚拟机负载指标，分别表示为 CPU_loadi 和 Mem_loadi。

[0090] 在本实施例中，虚拟机上实现和运行收集信息的 Monitor Daemon 是基于 Xentop 命令采集的资源信息，它反映的是当前资源利用百分比，比较的基数是虚拟机当前分配到的资源额度。在本发明中需要对动态伸缩的虚拟机的能力负载进行计算，因而对采集的资源利用百分比进行归一化处理。

[0091] 比如，虚拟机 A、B、C 分别获得总资源的额度为 4 ： 2 ： 1，当前的 CPU 使用率分别为 80％、60％和 40％，虚拟服务器 A 的 80％利用率并不意味着它比 B 和 C 负载率更高，以 A 的能力基数进行归一化，则 B 和 C 的负载率应该为 60％ ×2 和 40％ ×4，结果说明实际上 A 的负载最轻。

[0092] 所以为了准确评估该些异构虚拟机的实际真实负载，还需要从获取当前的虚拟主机分配到的资源配额，然后对动态采集到的负载做归一化处理。

[0093] 虚拟机 CPU 负载的计算公式为 ：

[0094] $$CPU\_load_i = CPU\_Useage\_rate_i \times \frac{\sum_{j-1}^{n} \frac{CPU\_Entitle_j}{n}}{CPU\_Entitle_i}$$

[0095] Entitle 表示虚拟机分配到的资源额度，该公式的含义是，我们将当前的资源使用率，向平均的资源额度这个基数进行统一，也就是调整后的资源负载都是相对同一个基点的比较结果，这样的绝对值就具有可比性。

[0096] CPU_loadi 表示虚拟机 i 的 CPU 负载，CPU_Useage_ratei 表示虚拟机 i 的 CPU 利用率，CPU_Entitlei 表示虚拟机 i 分配到的 CPU 资源额度。

[0097] 虚拟机 Memory 负载的计算公式为：

[0098] $$Mem\_load_i = Mem\_Useage\_rate_i \times \frac{\sum_{j-1}^{n} \frac{Mem\_Entitle_j}{n}}{Mem\_Entitle_i}$$

[0099] Mem_loadi 表示虚拟机 i 的 Mem 负载，Mem_Useage_ratei 表示虚拟机 i 的 Mem 利用率，Mem_Entitlei 表示虚拟机 i 分配到的 Mem 资源额度。

[0100] 在程序运行过程中，由驻留在虚拟机的守护进程 Gmond 定时地向 Monitor Daemon 报告负载信息。若虚拟机在设定的时间间隔内没有响应，Monitor Daemon 认为服务器是不可达的，将该虚拟机在调度器中的权值设置为零，不会有新的连接再被分配到该虚拟机；若在下一次服务器有响应，再对虚拟机的权值进行调整。这些虚拟机负载数据值落在 [0，∞）的区间内，服务器负载大于 1 表示超载，等于 1 负载刚好，小于 1 说明服务器负载较轻，该数值越大，表示负载越重。

[0101] 步骤 S304c，计算各个虚拟机的实际性能负载。

[0102] 实际性能负载是虚拟机当前的服务工作队列长度和处理时间。

[0103] 虚拟机提供服务的响应时间，能够反映虚拟机上请求等待队列的长度和请求的处理时间。从调度器上访问虚拟机所提供的服务，并测得每个虚拟机的响应时间。若虚拟机在设定的时间间隔内没有响应，Monitor Daemon 认为该虚拟机不可达，将虚拟机在调度器中的权值设置为零。对虚拟机的响应时间进行调整得到实际性能负载：

[0104] $$Performance\_load_i = \frac{Response\_time_i}{Desired\_response\_time}$$

[0105] 其中，Desired_response_time 是初始设置的相应时间的要求，是期望的响应时间。

[0106] 实际性能负载值落在得到 [0，∞）的区间内，大于 1 表示超载，等于 1 负载刚好，小于 1 说明服务器负载较轻。当连接数量太多，预测将要超过 Desired_response_time 的时候就不会给给服务器分配连接请求，所以一般 Performance_loadi 的值都会控制在 [0，1) 范围内。

[0107] 步骤 S304d，计算综合负载。

[0108] 三个负载指标分别从未来的负载预测，当前的负载监控和实际性能负荷三个方面综合评估服务器的负载状况。对于不同的应用，各种负载指标的对综合负载贡献的权重不同，因此引入各个指标信息的系数来表示各个负载指标在综合负载中比重。系统管理员根据不同应用的需求，可调整各个负载信息的系数。另外，系统管理员也可以设置收集负载信

息采集的时间间隔。

[0109] 一组可以动态调整的系数 Ri 来表示各个负载参数的比重，其中 Σ Ri = 1。

[0110] 综合负载的计算公式为：

[0111] Server_Loadi = R1×Request_loadi+R2×CPU_loadi+R3×Mem_loadi+R4×Performance_loadi

[0112] 按如下公式计算所述虚拟机的新的权值，

[0113] $w_{newi} = w_i + K \times \sqrt[3]{1 - Server\_load_i}$ ，

[0114] 其中，w_{newi} 为新的权值，w_i 当前权值，K 为预设的调整系数，$Server\_load_i$ 为综合负载值。

[0115] 在公式中，1 为期望利用率，K 在实施例中值为 5。当综合负载值为 1 时，虚拟机权值不变；当综合负载值大于 1 时，权值变小；当综合负载值小于 1 时，权值变大。

[0116] 如此，当综合负载值表示虚拟机比较忙时，新算出的权值会比其当前的权值要小，这样新分配到该虚拟机的请求数就会少一些。当综合负载值表示虚拟机处于低利用率时，新算出的权值会比其当前的权值要大，那就增加新分配到该虚拟机的请求数。

[0117] 若新权值和当前的权值的差值大于设定的阀值，监控进程将该虚拟机的权值设置到内核中的 IPVS 调度中。过了一定的时间间隔，如 5 秒钟，物理节点的资源监控进程 (Monitor Daemon) 再查询各个虚拟机的情况，并相应调整虚拟机的权值；如此周期性地进行。

[0118] 步骤 S305，采用基于权值的轮询调度方法对服务请求进行调度。

[0119] 具体实施方式中采用 DFBS 算法进行调度。

[0120] DFBS(Dynamic Feedback Balancing Schedule，动态反馈平衡调度）算法使用基于动态反馈负载均衡机制，来控制新连接的分配，从而控制各个服务器的负载。

[0121] DFBS 算法考虑虚拟机的实时负载和系统真实的响应情况，不断调整虚拟机间处理请求数量的比例，来避免有些虚拟机超载时依然收到大量请求，从而提高整个系统的吞吐率，以保证公平性和有效性。

[0122] 图 5 显示了该算法的工作环境，在负载调度器上运行 Monitor Daemon 进程。开始的时候，在内核中连接调度采用加权最小连接调度算法，并为虚拟机群给定一个初始的权值。同时，Monitor Daemon 用来周期性地采集各个虚拟机的负载信息，以获知虚拟资源的动态伸缩。通过这个资源的负载信息，可以算出一个虚拟机的负载指标，评估当前虚拟机负载高低，即可用资源的多少。IPVS 也周期性地给 Monitor Daemon 返回请求连接的计数值，根据该些采样的参考量，计算出请求在各个虚拟机上的分布预测作为请求负载的输入指标，并可以用性能模型评估当前的连接数是否超载。如果估算出超载，那么将该虚拟机的权值设置为零，否则根据前面计算的虚拟机负载和请求负载计算出综合负载值。Monitor Daemon 将各个虚拟机的综合负载值和当前的权值算出一组新的权值，若新权值和当前的权值的差值大于设定的阀值，Monitor Daemon 将该虚拟机的权值设置到内核中的 IPVS 加权最小连接调度中。

[0123] 按照该些权值的比例给各个虚拟服务器动态调度请求连接，使得调度总是能根据服务器的工作能力来负载均衡地分发请求任务。

[0124] 权值大小和分配的连接数成正比，即权值越大，分配的连接数量越多。

[0125] 当综合负载值表示服务器比较忙时，新算出的权值会比其当前的权值要小，使得新分配到该服务器的请求数就会少一些。当综合负载值表示服务器处于低利用率时，新算出的权值会比其当前的权值要大，则增加新分配到该服务器的请求数。若新权值和当前的权值的差值大于设定的阀值，MonitorDaemon 将该服务器的权值设置到内核中的 IPVS 调度中。经过一定的时间间隔（如 5 秒钟），Monitor Daemon 再查询各个服务器的情况，并相应调整服务器的权值；如此周期性地进行。

[0126] 步骤 S306，计算虚拟机预设时长的平均资源利用率，根据该平均资源利用率纠正调度偏差。

[0127] 平均资源利用率为一种量度资源需求和可用资源之间使用程度的指标，定义为在历史前 t 时间段内某一个虚拟机资源的使用率进行归一化后求平均。

[0128] 由于资源是动态变化的，所以有必要归一化 CPU 的使用率，通过归一化，将其对分配的资源配额的利用率转化成对总资源的使用率。从平均资源利用率的定义上看，虚拟机的平均资源利用率越小，说明虚拟机被分配资源越多，被分配的服务请求的负载越少，该虚拟机被认为综合负载值低。反之，则说明综合负载值高。对于虚拟机服务 i，对时间 t ＝ 1 到 t ＝ N 的值求和，然后除以 N。由平均资源利用率计算出新的权值。

[0129] 步骤 S306a，在 t 时间段内某一个虚拟机资源的使用率进行归一化后求平均，获得该段时间该虚拟虚拟机的资源匹配度。

[0130] 比如，虚拟机 A、B、C 分别获得总资源的额度为 4 ： 2 ： 1，当前的 CPU 使用率分别为 80％、60％和 40％，虚拟服务器 A 的 80％利用率并不意味着它比 B 和 C 负载率更高，以 A 的能力基数进行归一化，则 B 和 C 的负载率应该为 60％ ×2 和 40％ ×4，结果说明实际上 A 的负载最轻。

[0131] 平均的匹配率计算公式如下所示，对于虚拟虚拟机 i，将所有 t 时刻的资源负载乘以分配的资源额度，最后求平均。

[0132] $$Global\_load_i = \frac{\sum_{t=1}^{N}(1-\frac{consume_{it}}{Allocate_{it}})\times entitle}{N}$$

[0133] Entitle 表示虚拟机分配到的资源额度，Allocate 表示分配的资源总额，consume 表示消耗的资源总额。$\frac{consume_{it}}{Allocate_{it}}$ 表示虚拟机 i 在 t 时刻的资源负载。

[0134] 全局负载反映实际的请求和可用资源的调度后的一个综合结果，它能表示前 T 时间段内服务请求调度和资源流动的效果，其作为未来的预测

[0135] 步骤 S306b，将预测结果和当前的权值进行比对，

[0136] $$r = \frac{Global\_load_i - w_i}{\sum_{i=1}^{N} w_j}$$

[0137] $Global\_load_i$ 表示对虚拟机 i 的全局负载，w_i 表示虚拟机 i 的权值。

[0138] 如果偏差超过阈值，则将权值进行调整，更新虚拟机的权值为

[0139] $$w_i = \frac{\sum_{j=1}^{N}\frac{global\_load_j}{N_i}}{global\_load} \times w_i$$

[0140] $Global\_load_i$ 表示对虚拟机 i 的全局负载，w_i 表示虚拟机 i 的权值。

[0141] 因为根据综合负载对虚拟机真实负载的评估并不完全准确，评估指标可能没有能真实地反映真实的负载，只是对真实负载的一个量化预测指标。所以，在调度决策后，通过对真实的虚拟机负载的监控，并对调度决策的偏差进行纠正，使得系统更加能够负载均衡地进行匹配调度。

[0142] 一种虚拟化计算平台中应用请求调度的系统如图 4 所示，包括运行于物理节点上的虚拟机 100、中控模块 200、调度器模块 300。

[0143] 中控模块 200，用于采集各个虚拟机 100 的资源总量，并将各个虚拟机 100 的资源总量反馈给调度器模块 300。

[0144] 调度器模块 300，用于采集各个虚拟机 100 的资源利用率和服务请求连接的数量。

[0145] 调度器模块 300 还用于根据虚拟机的当前权值、资源总量、资源利用率和服务请求连接的数量对虚拟机 100 的权值进行更新。

[0146] 中控模块 200 还用于接收服务请求，并将该服务请求转发给调度器模块 300。

[0147] 调度器模块 300 还用于采用基于权值的轮询调度方法对服务请求进行调度。例如，采用 DFBS 算法进行调度。

[0148] 中控模块 200 负责管理系统的所有物理信息，包括所有被管理物理节点、每个物理节点上创建的虚拟机 100 和每个机器的资源分配和使用情况。通过该些管理信息，利用的性能模型计算出系统上大量异构服务的最大负载量，并将结果返回给调度器模块 300 进行请求和资源的分配。

[0149] 中控模块 200 还负责接收 Web Service 提出的服务请求。在接收到应用调度请求后，向调度器模块 300 发送调度询问请求，并传递给调度器模块 300 相应的虚拟机信息。

[0150] 调度器模块 300 通过该些接收到的信息部署虚拟机 100 配置，把相应的虚拟机 100 加入到调度列表中。

[0151] 中控模块 200 周期性地对相应的虚拟机 100 提出请求，要求各个节点返回虚拟机 100 的管理信息，包括：虚拟主机 IP、虚拟主机名称、所在物理节点等。并同时将全局资源分配计算结果传递给虚拟机 100，用来指导物理节点内的资源流动决策。中控模块 200 还与调度器模块 300 交互，一方面动态改变提供特定服务的虚拟机信息，另一方面，将最大支持连接数的预测结果反馈给调度器模块 300，修改调度权值。

[0152] 中控模块 200 利用性能模型的预测，从宏观的角度优化服务间的请求调度，以及为保证系统在高负载状态下的正常工作，优化应用层调度以实现允许控制。从而进一步优化系统的请求调度和能力流动的匹配、提高系统的稳定性和可用性。

[0153] DVMM(Distribution Virtual Machine Monitor，分布虚拟机监控器）

[0154] DVMM 是分布式的系统性能监控系统，有两个 Daemon，分别是：客户端 Dmond(Distributed Monitoring Daemon，分布监控进程）和服务端 Dmetad(Distributed Meta Daemon，分布改变进程），是一个 Linux 下监控系统运行性能的软件，监控对象包括：CPU、内存、硬盘利用率，I/O 负载、网络流量情况等。

[0155] Dmetad 负责收集分布式虚拟机 100 上 CPU、Mem 负载变化情况，当调度器模块 300 请求虚拟机群资源利用率时，返回 CPU 和 Mem 负载的全系统实时数据。

[0156] Dmond 负责在客户端监控虚拟机 100 的当前负载情况，和 Dmetad 配合使用。

[0157] 调度器模块 300 实现了动态反馈负载均衡 DFBS 调度算法，负责处理 WebServices 请求的调度。调度器模块 300 本身并不直接获取虚拟机 100 的当前资源利用率，该些信息都是通过 Dmond 获得的。在获取到当前所有虚拟机 100 的负载情况后，根据 DFBS 均衡算法和中控模块 200 提出的服务请求类型，调度器模块 300 决定某一台合适的虚拟机 100 作为调度结果。调度器模块 300 通过 Dmetad，Dmetad 收集的数据，进行计算并根据虚拟机 100 当前的负载进行权值的调整，该权值大小和分配的连接数成正比，权值越大，分配的连接数量越多。权值的计算同时考虑了虚拟机 100 的处理器能力和当前可用资源大小。

[0158] 具体实施方式如下所述。

[0159] 中控模块 200，用于采集各个虚拟机 100 的资源总量，并将各个虚拟机 100 的资源总量反馈给调度器模块 300。

[0160] 调度器模块 300，用于采集各个虚拟机 100 的资源利用率和服务请求连接的数量。

[0161] 调度器模块 300 还用于根据虚拟机的当前权值、资源总量、资源利用率和服务请求连接的数量对虚拟机 100 的权值进行更新。

[0162] 调度器模块 300 根据虚拟机 100 的资源总量、资源利用率和服务请求连接的数量判断所述虚拟机是否超载，如果超载，则将该虚拟机 100 的权值记为预设最小值，否则，根据虚拟机 100 的资源利用率、服务请求连接的数量和当前的权值计算该虚拟机新的权值，按该新的权值对所述虚拟机进行更新。

[0163] 调度器模块 300 在判断超载时，由虚拟机的资源总量和资源利用率计算出虚拟机 100 的可用资源量，将该可用资源量发送给中控模块 200。

[0164] 中控模块 200 将虚拟机 100 的可用资源量和服务请求连接数量带入性能预测模型计算出该虚拟机 100 的最大连接数，将所述最大连接数反馈给调度器模块 300。调度器模块 300 接收中控模块 200 反馈的最大连接数，如果虚拟机 100 的服务请求的连接数量大于等于所述虚拟机的最大连接数，则判断为超载；否则，判断为不超载。

[0165] 调度器模块 300 在更新虚拟机的权值时，根据资源利用率和服务请求连接的数量计算出综合负载值；根据所述虚拟机的综合负载值和所述虚拟机的当前的权值算出所述虚拟机的新的权值；如果所述虚拟机的新的权值和所述虚拟机的当前的权值的差值大于设定的阀值，则将所述虚拟机的权值更新为所述新的权值。

[0166] 调度器模块 300 计算综合负载值时进一步用于按如下公式计算所述虚拟机的新的权值，

[0167] $w_{newi} = w_i + K \times \sqrt[3]{1 - Server\_load_i}$ ，

[0168] 其中，w_{newi} 为新的权值，w_i 当前权值，K 为预设的调整系数，$Server\_load_i$ 为综合负载值。

[0169] 所述系统还包括偏差纠正模块，用于计算虚拟机预设时长的平均资源利用率，根据该平均资源利用率纠正调度偏差。

[0170] 使用一个平均资源利用率定义调度偏差，调度偏差为一种量度资源需求和可用资源之间使用程度的指标。定义为在历史前 t 时间段内某一个虚拟机资源的使用率进行归一化后求平均，即为该段时间该虚拟机的资源匹配度。由于资源是动态变化的，所以有必要归一化 CPU 的使用率，将其标准化成将其对分配的资源配额的利用率转化成对总资源的使用率。从平均资源利用率上看，平均的匹配度越小，说明分配的资源越多，请求负载分配越少，

则认为该虚拟机综合负载越低。请求负载分配越多，则认为该虚拟机综合负载越高。对于虚拟机服务 i，将所有 t 时刻的资源负载乘以分配的资源额度，最后求平均。

[0171] 调度器模块 300 还用于在初始时，采用加权最小连接调度算法，并为虚拟机群中虚拟机设置初始的权值。

[0172] 在一个物理机器上配置不同资源密集型的服务对应的虚拟机。

[0173] 虚拟机上包括本地管理模块，负责维护物理节点上运行的虚拟机之间的能力分配信息管理。本地管理模块从本地采集虚拟资源利用率，并获得当前每个虚拟机分配的资源配额，即资源流动的动态决策结果；能采集到资源流动算法的动态参数，包括流动阈值和 activity 的权值。通过该些信息，调度器模块 300 可以停止向即将达到流动阈值的服务节点分配请求，避免资源频繁流动带来的系统开销。

[0174] 本领域的技术人员在不脱离权利要求书确定的本发明的精神和范围的条件下，还可以对以上内容进行各种各样的修改。因此本发明的范围并不仅限于以上的说明，而是由权利要求书的范围来确定的。

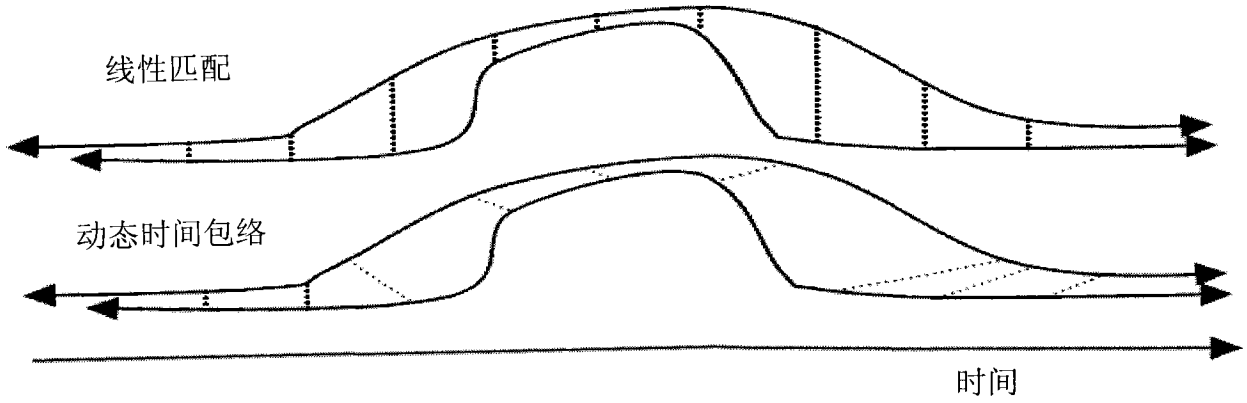

线性匹配
动态时间包络
时间

图 1

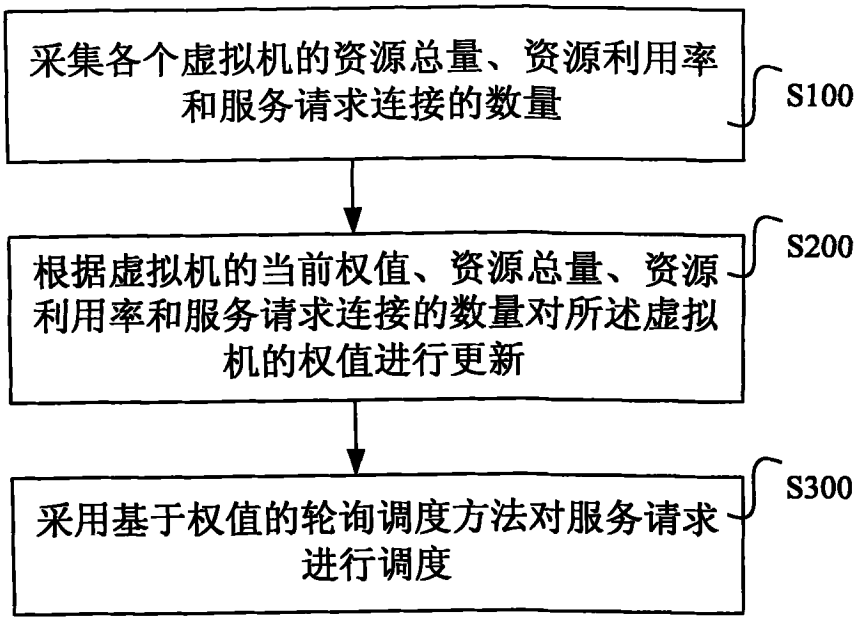

采集各个虚拟机的资源总量、资源利用率和服务请求连接的数量
S100
根据虚拟机的当前权值、资源总量、资源利用率和服务请求连接的数量对所述虚拟机的权值进行更新
S200
采用基于权值的轮询调度方法对服务请求进行调度
S300

图 2

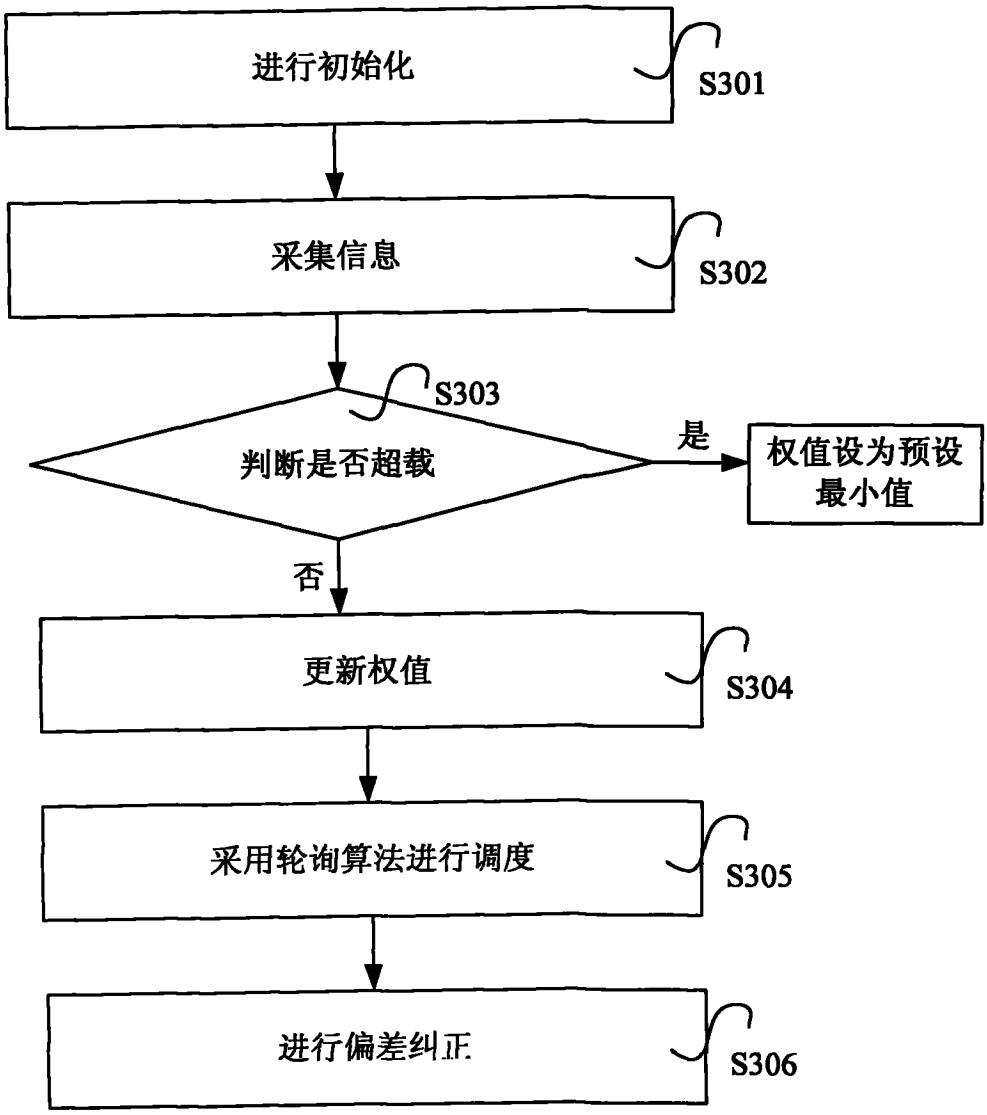

进行初始化
S301
采集信息
S302
S303
判断是否超载
是
权值设为预设最小值
否
更新权值
S304
采用轮询算法进行调度
S305
进行偏差纠正
S306

图 3

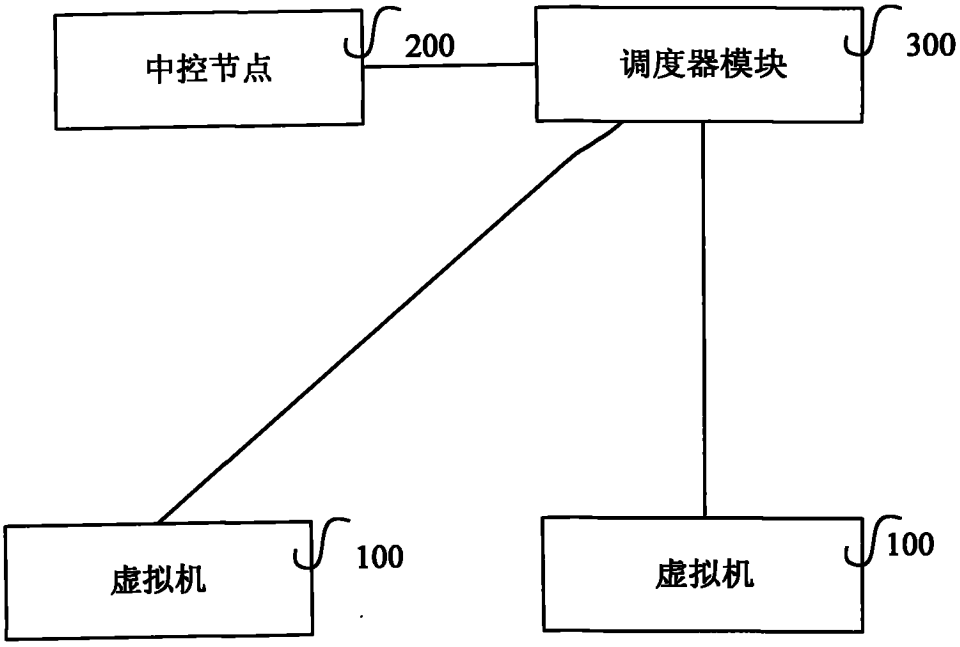

中控节点
200
调度器模块
300
虚拟机
100
虚拟机
100

图 4

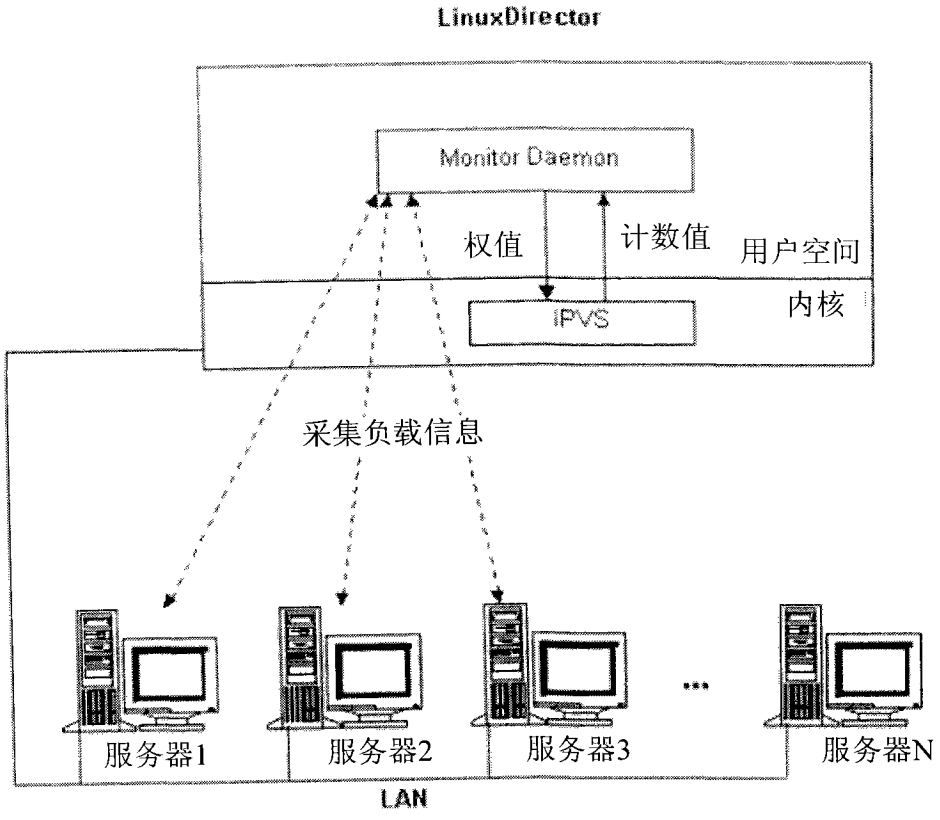

LinuxDirector
Monitor Daemon
权值
计数值
用户空间
内核
IPVS
采集负载信息
服务器1
服务器2
服务器3
...
服务器N
LAN

图 5