面向数据中心能效优化的虚拟机迁移调度方法

阮顺领[1] 卢才武[1] 邢 雪[1,2]
[1](西安建筑科技大学管理学院 陕西 西安 710055)
[2](商洛学院信息管理系 陕西 商洛 726000)

摘 要 由于服务器资源利用率偏低且资源负载不均衡,使得数据中心能耗浪费严重。针对上述情况,提出基于虚拟机迁移的数据中心节能调度方法。该方法通过选择合适的迁移时机、迁移对象和目标主机,完成虚拟机迁移前的准备工作,然后基于迭代-停止迁移方法对服务器进行动态迁移和整合,从而减少服务器的运行数量,以此最小化数据中心能耗。实验结果表明,该方法能有效提高服务器资源利用率,减少服务器的冗余数量,提高数据中心整体能效。

关键词 数据中心 能效优化 虚拟机迁移 服务器整合

中图分类号 TP302.7 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2016.01.004

VIRTUAL MACHINE MIGRATION SCHEDULING METHOD OPTIMISING ENERGY-EFFICIENCY OF DATA CENTRE

Ruan Shunling[1] Lu Caiwu[1] Xing Xue[1,2]
[1](*School of Management, Xi' an University of Architecture and Technology, Xi' an 710055, Shaanxi, China*)
[2](*School of Information Management, Shangluo University, Shangluo 726000, Shaanxi, China*)

Abstract Since the server resources utilisation is low and the resources load is unbalanced, the energy waste of data centre is serious. In view of this, we proposed a virtual machine migration-based data centre energy-saving scheduling method. Through selecting proper migration time, migrate object and target host, this method completes the preparatory work before the migration of virtual machine. Then, it dynamically migrates and integrates the servers based on the iteration-stopping migration method, so that minimises the energy consumption of data centres by reducing the running number of the servers. Experimental results showed that this method could improve the server resources utilisation effectively, reduce the redundant number of servers, and improve the overall energy efficiency of data centre.

Keywords Data centre Energy-efficiency optimisation Virtual machine migration Server integration

0 引 言

近年来,随着企业信息化和云计算[1]应用的迅速发展,促使数据中心的规模和复杂性日益增大。据统计,大规模的基础设施和 IT 设备导致数据中心的能耗成本以平均每年 20% 的速度增长。其中服务器设备的能耗占据了数据中心总能耗的 40%[2]以上。然而 IDC 调查[3]显示,大部分的服务器能耗是由闲置的设备所消耗,许多数据中心的服务器平均利用率只有 30% 左右。虚拟化迁移技术可实现对服务器资源的动态分配,将若干低负载的服务器迁移整合到较少服务器上[4],有效降低服务器能耗。文献[5]提出一种基于虚拟机迁移的节能方法,可有效优化在服务器负载分布严重不平衡情况下的能耗。文献[6]提出一种面向云计算能效优化的网络结构和规则,并基于该架构提出了数据中心节能管理的资源调度算法。文献[7]提出了基于粒子群算法的数据中心负荷分配方法,通过流体力学模拟对数据中心进行优化配置,从而减少服务器的能耗。文献[8]提出了基于最优的启发式算法,该算法可以应用在固定的时间间隔,改变的 CPU 核心功率,以适应不同的任务需求和节能策略,保持数据中心运行能效最佳。

虽然已有研究成果能够取得一定的节能效果,但在虚拟机的迁移时机,迁移对象和目标服务器选择,以及迁移方法等方面还有待优化。特别是当虚拟机资源利用率不稳定时,不合理的迁移会导致虚拟机迁移颠簸现象发生。针对上述问题,提出基于虚拟化迁移的数据中心节能调度方法,以此最小化数据中心能耗。

1 节能调度模型

在虚拟机迁移调度模型中,如图 1 所示。根据设备的物理位置划分虚拟设备组 VDG(Virtual Device Group),其中包括服务器,空调(AC)和 UPS 等设备,迁移调度包括以下 5 个过程:

(1) 迁移触发策略的选择,设所有服务器的集合为 $S=\{s_1,s_2,\cdots,s_n\}$,依据迁移选择约束条件,在最佳的时机从集合 S 中选择合适的服务器 S_i 触发迁移,从而保证迁移的合理性。

收稿日期:2014-07-21。陕西省科技计划项目(2009K08-25)。**阮顺领**,博士生,主研领域:数据中心节能。**卢才武**,教授。**邢雪**,讲师。

(2) 迁移虚拟机的选择:当确定了被迁移服务器 S_i 后,在服务器的虚拟机集合 $VM = \{vm_1, vm_2, \cdots, vm_i\}$ 中,选择 1 个迁移开销小且释放资源更多的虚拟机 vm_j 进行迁移。

(3) 目标服务器的选择,为了给被迁移虚拟机 vm_j 寻找 1 个最佳的目标服务器,因此需要在目标服务器集合 $D = \{d_1, d_2, \cdots, d_{n-1}\}$ 中,$D \in S$ 且 $S_i \notin D$,依据目标服务器选择算法选择最佳目标服务器 D_k。

(4) 虚拟机迁移调度,当完成被迁移虚拟机 vm_j 的选择和目标服务器 D_k 的选择后,则开始虚拟机迁移工作。

(5) 联动节能调度,当完成服务器的迁移整合后,重新计算数据中心对空调系统的制冷需求,并根据节能调度策略对制冷供电设备进行调度,从而减少数据中心能耗浪费。

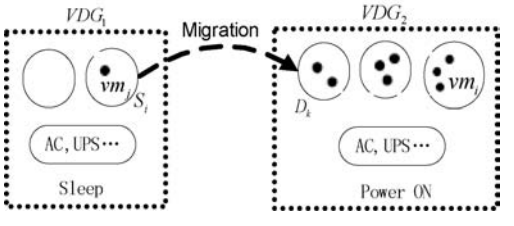

图 1 虚拟机迁移调度模型

由于篇幅所限,本文将重点描述(1)-(4)过程中服务器资源的虚拟化迁移调度,暂不对动环设备的联动调度策略及算法进行描述。

2 迁移选择方法

在进行虚拟机调度迁移前,应首先完成迁移时机的选择,迁移对象的选择和目标服务器的选择,从而为后续的动态迁移做好准备,本节将主要对上述三者的选择方法和规则进行说明。

2.1 迁移时机的选择

2.1.1 规则定义

定义 1 资源负载阈值:服务器资源利用率的上限和下限要求,主要指服务器的 CPU 利用率和内存利用率。其中:RC_{Min} 指 CPU 的下限负载阈值;RC_{Max} 指 CPU 的上限负载阈值;RM_{Min} 指内存的下限负载阈值;RM_{Max} 指内存的上限负载阈值。

定义 2 双时间阈值:为了避免因资源负荷度的骤变而引起迁移颠簸,采取双时间阈值对峰值进行滤波。其中,T_{Max1} 表示资源负载达到峰值后持续的第一个时间阈值。T_{Max2} 表示资源负载达到峰值后持续的第二个时间阈值。其中 $T_{\text{Max2}} > T_{\text{Max1}}$,两者间的差值 $T_{\text{Offset}} = T_{\text{Max2}} - T_{\text{Max1}}$。

定义 3 阈值回差:指负载率超过或低于阈值后可以接受的变化范围。其中定义:C_{Offset} 目标服务器 CPU 负载率超过或低于阈值的合理范围;M_{Offset} 目标服务器内存负载率超过或低于阈值的合理范围。通常,需要分别定义阈值的上限回差和下限回差。如:下限回差:$C_{\text{MinOffset}}$ 和上限回差:$C_{\text{MaxOffset}}$。

定义 4 迁移紧急度:当有多个服务器同时有迁移需求时,为了避免虚拟机群体迁移引起网络风暴等问题,系统根据服务器的资源情况和触发原因对待迁移对象划分紧急等级,然后根据迁移紧急度进行分批迁移。迁移紧急度分为 3 个等级,等级越高迁移的紧急度越高。迁移紧急度定义如表 1 所示。

表 1 迁移紧急度定义

紧急度	触发原因
3 级	由于服务器资源严重不足,或设备故障预警,或数据中心环境严重异常等原因触发的迁移; 如:$C_{\text{utili}} \geqslant C_{\text{RMax}} + 2C_{Offset}$ 或 $M_{\text{utili}} \geqslant M_{\text{RMax}} + 2M_{Offset}$
2 级	由于服务器资源负荷过高,或数据中心环境异常预警等原因触发的迁移; 如:$C_{\text{utili}} \geqslant C_{\text{RMax}} + C_{Offset}$ 或 $C_{\text{utili}} \leqslant C_{\text{RMin}} - C_{Offset}$
1 级	由于服务器资源利用率超过负载阈值等原因触发的迁移; 如:$C_{\text{utili}} \geqslant C_{\text{RMax}}$ 或 $C_{\text{utili}} \leqslant C_{\text{RMin}}$

另外,还可以根据服务器历史同期的负载统计情况,预测服务器的负载变化趋势,从而决定是否触发迁移。还可以设置服务器最小运行数量约束条件,保证服务器运行的数量,以应对计算需求骤增的情况。

2.1.2 迁移触发策略

策略 1 当迁移紧急度为 3 级且达到第一个时间阈值 T_{Max1} 时,则触发虚拟机迁移。

策略 2 当迁移紧急度小于 3 级且持续时间达到 T_{Max1} 时,则计算 T_{Max2} 时刻的负载预测值和负载经验值,若负载预测值或负载经验值达到负载阈值,则触发虚拟机迁移。

策略 3 当迁移紧急度小于 3 级,采集值超过负载阈值且持续时间达到 T_{Max2},则触发虚拟机迁移。

2.2 迁移对象的选择

当某个服务器被触发迁移时,如果服务器上运行多个虚拟机,则需要在该服务器上选择一个能够释放所需的资源,又能最小化迁移开销的虚拟机对象进行迁移,因此需要对虚拟机群进行综合评估,从而找出最佳的迁移对象。假设虚拟机的迁移优先级范围为 $P \in [1,10]$,优先级越大则虚拟机被迁移的优先权越高。当 $P = 10$ 时,意味着需要立即迁移该虚拟机对象。虚拟机调度优先级评价可通过以下步骤完成:

步骤 1 获得服务器及其运行虚拟机的状态信息,并根据成员函数对状态信息模糊化处理。虚拟机状态信息包括:虚拟机所占 CPU、内存和磁盘等资源大小,虚拟机最近迁移时间,触发迁移的需求信息等。模糊化成员函数可表示为:

$$P_{1,i} = F_i(x_i) \tag{1}$$

其中,x_i 为输入的虚拟机状态信息。如:根据虚拟机所占 CPU 和内存资源的大小,将其分布到不同的区间,然后转化为模糊输出,作为评价该虚拟机优先级的依据之一。

步骤 2 根据隶属度函数计算虚拟机状态信息和调度规则的关联度,计算值和关联度成正比。可表示为:

$$P_{2,i} = R_i = F_{A_i}(x_i)F(y_j) \tag{2}$$

其中,$F(y_j)$ 为调度规则成员函数。

关联度原则包括:

(1) 资源匹配原则。根据迁移的资源需求满足度进行计算,虚拟机能释放的资源大小与触发迁移所需资源大小越接近,则其关联度就越大。如:当因内存资源不足而触发虚拟机迁移,所需内存资源为 M_1,虚拟机迁移后能释放的内存为 S_1。若

$S_1 - M_1 < 0$,则虚拟机关联度较小。若$S_1 - M_1 \geq 0$,则差值越小关联度越大。如果所有虚拟机 S_1 与 M_1 的差值都小于 0,则需要同时迁移多个虚拟机。

(2) 迁移开销最小化原则。由于在进行虚拟机迁移时,主要的能耗是内存迁移拷贝,为了降低迁移的能耗成本,因此当有多个虚拟机同时满足迁移条件时,即:$S_1 - M_1 \geq 0$,还需综合考虑内存、CPU 和带宽等资源的使用率,内存使用率越低关联度越大。如:虚拟机 V_1 的内存使用率为 M_1,CPU 的使用率为 C_1,带宽的使用率为 L_1,则需根据 M_1、C_1、L_1、$M_1 \times C_1$ 和 M_1/C_1 调整虚拟机的关联度。

(3) 分层管理原则。根据虚拟机在服务器所驻的时长对虚拟机进行垂直分层管理,分为稳定层、中间层和活跃层。在进行虚拟机优先级计算时,活跃层的虚拟机关联度最高,其次为中间层和稳定层。随着服务器运行时间的延伸,虚拟机的迁移部署将更加均衡和趋于稳定。

(4) 后进优先出原则。按照虚拟机在服务器上已运行的时长进行排序,对于最后被迁入的虚拟机优先被迁出。该规则有利于分层管理的形成,对于经常被迁移的虚拟机将会被分配到活跃层。

步骤 3　根据虚拟机迁移触发条件,计算每个调度规则的关联度加权输出。其计算函数可表示为:

$$P_{3,i} = R_i \omega_i \tag{3}$$

如:因内存因素触发虚拟机迁移,则内存因素的权重大于 CPU,带宽等其他因素。

步骤 4　通过精确化计算实现对关联度去模糊化,将推理得出的模糊输出转化为虚拟机的调度优先级 P_x。其输出函数可表示为:

$$P_{4,i} = P_{x_i} = \sum R_i \omega_i \tag{4}$$

步骤 5　根据计算所得的优先级对虚拟机进行排列,并依据该优先级进行迁移调度。

2.3　目标服务器的选择

2.3.1　选择规则

规则 1　资源匹配规则,对于服务器资源必须同时满足以下 4 个公式条件。如果对带宽等其他资源有要求时,可以把其他资源的要求加到限制条件中。

$$C_{\text{Requir}} \leq Min(C_{\text{Usable}}, C_{\text{Usable}+i}) \tag{5}$$

$$M_{\text{Requir}} \leq Min(M_{\text{Usable}}, M_{\text{Usable}+i}) \tag{6}$$

$$RC_{\text{Min}} \geq (C_{\text{Usable}} - C_{\text{Requir}} - C_{MaxOffset}) \tag{7}$$

$$RM_{\text{Min}} \geq (M_{\text{Usable}} - M_{\text{Requir}} - M_{MaxOffset}) \tag{8}$$

其中,C_{Usable} 目标服务器可用的 CPU 空间;

$C_{\text{Usable}+i}$ 预测该目标服务器第 i 时刻可用的 CPU 空间;

M_{Usable} 目标服务器可用的内存空间;

$M_{\text{Usable}+i}$ 预测该目标服务器第 i 时刻可用的内存空间;

C_{Requir}　被迁移虚拟的 CPU 空间要求;

M_{Requir} 被迁移虚拟的内存空间要求;

当有多个服务器 $S_i = (S_1, S_2, \cdots, S_n)$ 同时满足条件时,为了提高服务器的利用率,选择 S_{C_i} 最小的作为目标服务器,如果 S_{C_i} 大小一样,则选择其中 S_{M_i} 较小者作为目标服务器。

$$S_{C_i} = S_{CPU} - (C_{\text{Usable}} + C_{\text{Requir}} + 3C_{MaxOffset}) \tag{9}$$

$$S_{M_i} = S_{Memory} - (M_{\text{Usable}} + M_{\text{Requir}} + 3M_{MaxOffset}) \tag{10}$$

规则 2　反竞争规则,当多个虚拟机间有相同的特殊资源需求或者服务器间某些安全策略有冲突时,则不能将这类虚拟机部署到同一个服务器上。如:当虚拟机服务的通信端口冲突时,则需要分开部署。

规则 3　最大迁入数规则,如果一个服务器同时被多个虚拟机对象选中,则可能会导致该目标服务器的负载骤增及性能骤降,同时也会影响制冷系统的负载平衡,因而限制服务器同时迁入的虚拟机最大个数。该约束规则可描述为:$\sum V_i \leq S_{Max}$,V_i 为适合该服务器的虚拟机,S_{Max} 为服务器同时迁入虚拟机数限制。

规则 4　组内优先规则,在选择目标服务器时,首先遍历被迁移服务器所在虚拟组内的服务器集合,如果虚拟群组内的服务器都无法满足迁移要求,则遍历其他虚拟组的目标服务器。

2.3.2　选择算法

算法主要步骤:

```
1. Begin Format(VM Information, Scheduling rules,…)
2. Do {                                              //Step (1)
3.   If (Match the Requirements)
4.   Compute C_{Usable+i} And M_{Usable+i} ,
5.   If ((C_Requir ≤ C_{Usable+i}) && (M_Requir ≤ M_{Usable+i}))
6.   Push S_i into the Server Candidate List;
7.     End if
8.   End if
9.   } until (Iterates through all the servers of S_i )
10. For each (Server Candidate List) Do              //Step (2)
11.   Get the resources information of servers and server group
12.   For each (Resource Require List) Do
13. If (resources conflict OR infrastructure less resources
OR immigration limit)
14.   Remove S_i form Server Candidate List;
15.         Else
16.   Update the list of servers that to be migrate;
17.         End if
18.      End for
19. End for
20. If (Length (Server Candidate List) = = 1)
21.       Return D_k , D_k ∈ S                       //Step (3)
22. Else If (Length (Server Candidate List) > 1)
23.      Return S_i which selected by Rule5
24.   Else       // Length (Server Candidate List) < 1
25.   If (Have dormant server)
26.      Select a server of D_k , D_k ∈ S
27.   Else
28.      Repeat (1) - (3) step, try to choose a suitable server;
29.   End if
30. End if
```

3　虚拟机拷贝迁移方法

虚拟机的动态迁移分为两个阶段,第一个阶段通过虚拟机迁移选择方法完成了迁移前的准备工作;第二个阶段的迁移拷贝过程是整个迁移过程的核心阶段。虚拟机动态迁移方法分为以下四个阶段:

(1) 预拷贝阶段:该阶段迭代拷贝所有的状态页,将原服务器的状态数据拷贝到目标服务器上。

(2) 迭代拷贝阶段:完成预拷贝之后进入迭代拷贝阶段,每轮迭代只拷贝在上一轮迭代过程中修改过的页面,以此类推,第

n 轮拷贝的是在第 $n-1$ 轮迭代过程中修改过的页面。当出现内存剧烈动荡时,需要通过迭代停止条件停止迭代,以缩短迁移的总时长。当满足迭代终止条件时,对原服务器进行冻结并进入第三阶段即停止-拷贝阶段。

在迭代拷贝过程中,为了减少重复拷贝的数据量以提高拷贝效率,采用内存页位图对迭代过程中的脏页面进行标记。本文分别用 $Data_{last}$、$Data_{current}$ 和 $Data_{fix}$ 进行标记。其中,$Data_{last}$ 标记在上次迭代过程中所产生的脏页,$Data_{current}$ 标记在本次的迭代过程中所产生的脏页,$Data_{fix}$ 标记需在停止拷贝阶段进行迁移的脏页。脏页面标记方法用图 2 表示。

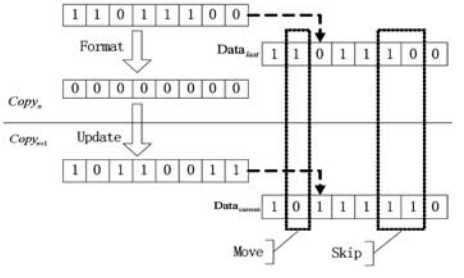

图 2 脏页面标记示意图

在迁移拷贝算法中,只有当页位图中 $lastData = l$ 且 $currentData = 0$ 的情况下进行脏页面迁移,其他情况下则认为脏页面未被修改过或还未完成数据更改,而不进行页表迁移。

(3) 停止-拷贝阶段:在此阶段,将最后一轮迭代过程中产生的脏页面和状态信息,一次性拷贝到目标服务器,从而完成所有虚拟机数据的迁移。

(4) 服务转移阶段:完成脏页面拷贝之后,激活目标服务器的虚拟机,完成虚拟机服务的迁移。此时,如果原服务器为空载状态,则关闭原服务器设备,从而完成服务器的整合工作。

4 实验与分析

本文选择一个微模块数据中心 MDC(Modular Data Center)进行虚拟化实验。该微模块数据中心配置了 4 台精密空调,2 个配电柜,42 个温湿度传感器和 128 台 4 核服务器。服务器的 CPU 主频为 2.3 GHz,16 GB 内存和千兆网卡。软件环境为:Centos5.6 系统和 Xen4.0 虚拟化软件。系统参数如下:

$RC_{\text{Min}} = 2\%$ $RC_{\text{Max}} = 98\%$

$RM_{\text{Min}} = 40\%$ $RM_{\text{Max}} = 85\%$

$T_{\text{Max1}} = 2\text{minute}$ $T_{\text{Max2}} = 5\text{minute}$

$C_{MaxOffset} = 3\%$ $C_{MinOffset} = 2\%$

$C_{Offset} = 10\%$ $M_{Offset} = 10\%$

(1) 服务器利用率方面

设定当服务器 CPU 平均利用率 $AVG.\ C_{\text{utili}} \leqslant 2\%$,并且内存平均利用率 $AVG.\ M_{\text{utili}} \leqslant 5\%$,且持续时间达到 T_{Max2} 时,标记服务器为空闲状态(Idle Server),否则标记为重要状态(Primary Server)。

通过对 MDC 中服务器的运行状态统计分析,可以得到实施虚拟化调度前后服务器的利用率趋势图,如图 3 和图 4 所示。从实验统计结果可知,采取静态管理方式会导致约 30% 的服务器处于空闲状态。采用虚拟化调度整合后,服务器空闲率下降至 3% 左右,约 25% 的服务器被停机或休眠。同时,采用静态管理时服务器运行的数量基本不会发生变化,而采用动态调度管理后,服务任务的变化将会影响服务器运行的数量。

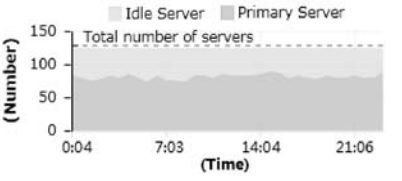

图 3 虚拟化整合前服务器利用率统计

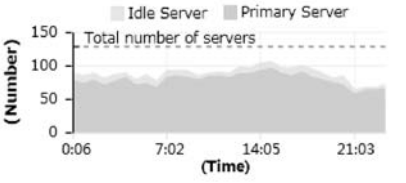

图 4 虚拟化整合后服务器利用率统计

从服务器资源平均利用率统计来看,采用虚拟化调度管理能明显提升服务器的 CPU 和内存利用率,单台服务器所承担的计算任务也有明显增加,从而提升了服务器的整体利用率。统计信息如表 2 所示。

表 2 服务器资源利用情况

	平均任务数	CPU 平均利用率	内存平均利用率
优化前	6 个	8.85%	12.63%
优化后	11 个	39.51%	53.72%

(2) 能效利用率方面

为了准确评估虚拟化整合调度的节能效果,本实验采用数据中心通常的能耗有效性 PUE(Power Usage Effectiveness)作为定量评估指标[9]。其计算方法如下:

$$PUE = \frac{\text{数据中心总能耗}}{IT\text{ 设备能耗}} = \frac{IT\text{ 设备能耗} + \text{空调能耗} + UPS\text{ 能耗} + \text{其他能耗}}{IT\text{ 设备能耗}} \quad (11)$$

采取静态管理方式时,MDC 某天的 PUE 趋势统计如图 5 所示。从统计结果可知,由于服务器群基本都处于运行状态,且有近 30% 的服务器处于空闲运行状态,致使 MDC 全天的 PUE 平均值高达 1.71,超过了 MDC 的行业平均能耗利用率 1.5 左右。

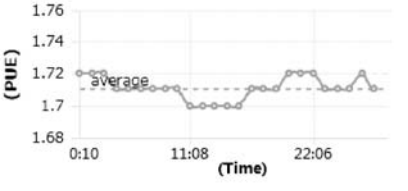

图 5 虚拟化整合前 PUE 统计图

采用虚拟化调度管理后,由于通过虚拟化整合减少了服务器的运行数量,并提高了服务器的利用率,使得 MDC 能耗利用率明显改善,其平均 PUE 为 1.46,如图 6 所示。按服务器额定功耗计算,其平均额定能耗为 34.51 千瓦/时,而实际平均能耗为 21.83 千瓦/时,MDC 的节约能耗近达 36.74%。

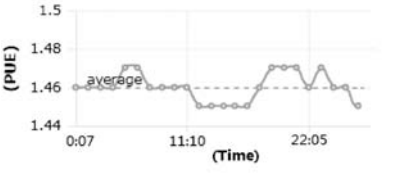

图 6 虚拟化整合后 PUE 统计图

综上实验分析可知,通过对服务器的虚拟化迁移调度,可有效减少服务器的能耗,从而节约微模块数据中心整体的能耗成本。

5 结　语

针对数据中心服务器资源利用率偏低,能耗浪费严重的问题,提出了基于虚拟机迁移的服务器设备节能调度方法。该方法通过感知服务器资源的利用情况,选择合适的迁移时机、虚拟机迁移对象和合适的目标服务器,依据服务器节能调度策略完成对服务器的动态迁移和整合,以此减少服务器运行数量,降低数据中心整体能耗。通过实验应用,该方法可以实现对虚拟机动态均衡调度,提高服务器的利用率,更好地实现了数据中心的节能。

参　考　文　献

[1] 姚宏宇,田溯宁. 云计算:大数据时代的系统工程[M]. 北京:电子工业出版社,2013.

[2] 王克勇,王丽,徐靖文,等. 绿色数据中心空调节能技术研究[J]. 能源研究与利用,2012(2):29-31.

[3] Lamia Lafdil, Marcus Torchia, Jill Feblowitz. Worldwide Utility Industry IT Spending Guide[R]. July,2012.

[4] Weiwei Fang, Xiangmin Liang, Shengxin Li, et al. VMPlanner: Optimizing virtual machine placement and traffic flow routingto reduce network power costs in cloud data centers[J]. Computer Networks, 2013(57):179-196.

[5] 胡志刚,欧阳晟,阎朝坤. 云环境下面向能耗降低的资源负载均衡方法[J]. 计算机工程,2012,38(5):53-55.

[6] Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing [J]. Future Generation Computer Systems, 2012, 28(5):755-768.

[7] Jayantha Siriwardana, Saman K Halgamuge, Thomas Scherer, et al. Minimizing the thermal impact of computing equipment upgrades in data centers[J]. Energy and Buildings, 2012, 50(7):81-92.

[8] Reinaldo A Bergamaschi, Leonardo Piga, Sandro Rigo, et al. Data Center Power and Performance Optimization through Global Selection of P-States and Utilization Rates[J]. Informatics and Systems, 2012, 2(4): 198-208.

[9] Dan Azevedo, Mark Blackburn, Jud Cooley. Data Center Efficiency Metrics[R]. Green Grid:Tech Rep, 2011.

(上接第 3 页)

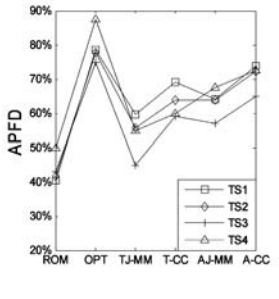

图 6　各组测试集的 APFD 值对比图

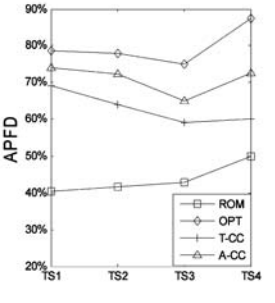

图 7　四种排序技术的 APFD 值对比图

针对问题(2),图 8 显示了应用全覆盖排序算法的优先排序技术对比图。图 9 显示了应用递增覆盖排序算法的优先排序技术对比图。

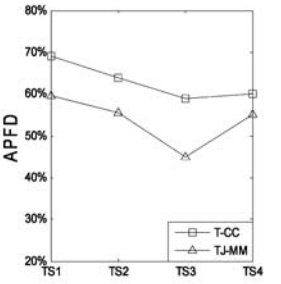

图 8　应用全覆盖排序算法排序技术的 APFD 值对比图

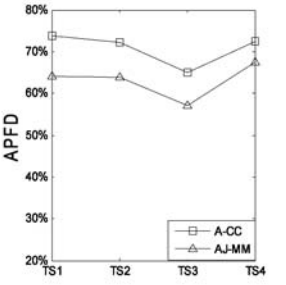

图 9　应用递增覆盖排序算法排序技术的 APFD 对比图

由图 8 可知,对于应用全覆盖排序算法的排序技术,在四组测试集中,T-CC 都要比 TJ-MM 检错效率高,证明了基于圈复杂度的排序技术在应用全覆盖排序算法的高效性。

类似应用全覆盖排序算法的排序技术,由图 9 可知,对于应用递增覆盖排序算法的排序技术, A-CC 都要比 AJ-MM 的检错效率高,证明了基于圈复杂度的排序技术在应用递增覆盖排序算法的高效性。

4 结　语

本文通过对已有的基于方法覆盖的排序技术进行扩展,主要从两个方面研究了基于圈复杂度的测试用例排序方法:(1)分析测试用例覆盖方法的的圈复杂度、全覆盖排序算法和递增覆盖排序算法。(2) 设计新的排序算法,将圈复杂度应用于两种排序算法中,以方法的圈复杂度取代方法覆盖率作为测试用例的排序依据。通过实验,验证了基于圈复杂度的测试用例排序方法的有效性和高效性。本文的不足在于实际工业程序的错误分布不均衡,使用圈复杂度来评估程序代码质量存在一定的局限性,如何更精确地确定圈复杂度与排序技术之间的关系,在今后还需进一步研究。

参　考　文　献

[1] Mei Hong, Hao Dan, Zhang Lingming, et al. A static approach to prioritizing junit test cases[J]. IEEE Transactions on Software Engineering, 2012, 38(6):1258-1275.

[2] Srikanth H, Williams L. On the economics of requirements based test case prioritization[C]//Proceedings of the International WorkShop on Economics-driven Software Engineering Research, 2005:1-3.

[3] Rothermel G, Untch R H, Chu C, et al. Prioritiz-ing test cases for regression testing[J]. IEEE Transactions on Software Engineering, 2001, 27(10):929-948.

[4] 屈波,聂长海,徐宝文. 基于测试用例设计信息的回归测试优先级算法[J]. 计算机学报,2008,31(3):431-439.

[5] 高建华,殷定媛. 回归测试中路径覆盖生成方法研究及其应用[J]. 小型微型计算机系统,2009,30(3):398-404.

[6] 肖自乾,王弗雄,陈经优. 基本路径测试方法之圈复杂度计算[J]. 软件导刊,2009,8(7):10-12.

[7] 司艳艳. 通过软件测试提高软件可靠性研究[J]. 软件导刊,2013,12(9):43-46.

[8] 杨广华,包阳,李东红,等. 基于需求的测试用例优先级排序[J]. 计算机工程与设计,2011,32(8):2724-2728.

[9] 李华莹,胡兢玉. 回归测试用例优先级排序技术研究[J]. 计算机仿真,2013,30(10):298-301.