第 39 卷第 1 期 东 北 电 力 大 学 学 报 Vol.39,No.1
2019 年 2 月 Journal Of Northeast Electric Power University Feb,2019

DOI: 10.19718/j.issn.1005-2992.2019-01-0088-08

基于负载均衡度的云计算任务调度算法

叶 波

(国网新疆电力有限公司电力科学研究院,乌鲁木齐 830011)

摘 要: 针对当前云计算环境中用户群与数据量庞大的特点,如何设计高效的负载均衡调度算法是云计算领域一直探索的重要课题.提出一种基于负载均衡度的云计算任务调度算法(TS-CCLB),该算法首先依据空间案投影分析计算了集群的负载均衡度,以此给出调度决策变量,并依据任务的执行代价完成时限赋予任务不同的优先级别.任务调度时将任务按优先级调度到最大决策变量值所对应的虚拟机上.实验结果表明,该算法可有效提高云计算集群的负载均衡性,缩短总任务的完成时间,尤其当任务数与节点规模较大时,优势更为明显.

关 键 词: 云计算; 任务调度; 负载均衡度; 优先级

中图分类号: TP391.9 文献标识码: A

当前,传统的云任务调度算法在缩短任务的完成时间、提高云计算集群利用率方面都有了比较理想的效果,但是在当系统处理器数量较多,任务量庞大时,云计算集群就容易出现负载失衡问题[1~3].经典的 Min-Min 与 Max-Min 算法对于计算量过小或过大的任务赋予更高的优先级,这样做避免了某些用等待时间过长,也缩短了总任务完成时间,但是算法的负载均衡性能较差[4~5].文献[6] Min-Min 算法中引入优先级思想,任务优先级与任务等待时间呈增长关系,这样能够有效缩短整个任务的完成时间.文献[7] 将任务分组,且使同组内任务优先级相等,组内截止时间短的任务优先得到调度.Min-Min 与 Max-Min 算法都实现了较高的任务调度效率,但是没有出现负载不平衡问题.文献[8] 提出了一种动态优先级实时任务调度算法,该算法实现了任务的抢占次数减少,截止期错失率低的效果.文献[9] 提出了一种依据动态优先级进行任务调度的 TS-PFB 算法,每一轮重新调度时都会重新计算任务优先级,以保证任务调度时的公平和效率.

优秀的任务调度算法应该既能有效减少总任务完成时间,也能够较好的实现系统的负载均衡,提高云计算资源利用率,显然,计算资源的负载均衡性越好,越能提高云计算集群的资源服务能力.文献[10] 将基于认知可信模型设计的动态优先级调度算法,使任务调度时的资源稳定性得到改善,从而使任务调度的成功率得到极大提高.文献[11] 比较了蜂群算法、动态聚集算法以及随机抽样算法的负载均衡性,综合分析了 3 种算法中节点数和节点性能对集群性能的影响.文献[12] 提出了一种基于蜜蜂觅食行为的动态负载均衡算法 HBB-LB,该算法适用于处理云环境下非抢占式独立任务的调度,并实现了较好的负载均衡,还考虑了节点中等待序列内任务优先级.文献[13] 提出了一种应用于网格计算的路由负载均衡准则,模仿网络中的路由方式确定邻节点,最后选取最合适的计算节点作为任务分配节点,将并行应用任务平均分配上去,该算法在任务重新分配过程中需要付出较多的通信成本.文献[14] 将负载约束考虑进 Max-Min 算法中.文献[15] 是对 Min-Min 算法的改进,把虚拟机上过饱和任务中的最小任务调出,

收稿日期: 2018-06-05
基金项目: 吉林省自然科学基金项目(20150101002JC)
第一作者: 叶 波(1990-),男,硕士,工程师,主要研究方向: 智能电网与云计算
电子邮箱: bby90031@163.com(叶波)

将其调度到负载较小且计算能力大的虚拟机上执行,缩短了总任务的完成时间.

文献[6]提出相空间分析方法,将集群中每个服务器的资源占用率参数(如 CPU 占用率、内存占用率、带宽等)投影到以这些参数为坐标轴的相空间中,将服务器参数的变化看做相空间中点的运动.相空间方法适用于设计大规模云计算集群调度算法,同时也是分析云计算系统的工作状态的有效工具.文献[7]在云计算相空间分析方法的基础上,提出了负载均衡量度概念,设计了最小负载均衡优先调度算法并对该算法的基本理论进行了研究和分析.

本文在文献[6]、文献[7]所提出的云计算相空间分析方法的基础上,经过分析论证,建立最小负载均衡优先调度分析方法.将负载均衡度作为系统负载均衡的量度,并作为任务调度决策变量的一个重要参数,保证了任务调度系统的负载均衡性.

综合考虑任务价值与截止时间对优先级的影响,同时兼顾系统的负载均衡,基于负载均衡度的云计算任务调度算法(TS-CCLB),该算法依据任务价值度与执行紧急度构建了动态优先级模型,然后按任务优先级高低依次调度.比较任务调度过程中任务与资源匹配的过程,与人工萤火虫算法在优化最优解过程中的相似性,通过模仿萤火虫行为,将虚拟机和待调度任务作为萤火虫,依据吸引度、荧光亮度和负载均衡度,计算出任务调度的决策变量 λ,将其作为个体的适应度值.调度过程即是将待调度任务调度到最大 λ 值所对应的虚拟机上.

1　系统负载均衡度的定义

1.1　计算节点到空间的投影

对于云计算集群节点中的服务器,其工作状态(如 CPU 使用率 0.6,内存占用率 0.3 等)可用一个参数向量来表示,如 $a=(a_1,a_2,\cdots,a_n)$.当节点接收任务进行处理时,其各个参数值将会发生变化.这样,参数向量的变化值即反映了服务器的当前工作状态.这里选取服务器的 3 个参数建立 3 维空间,称云计算集群的节点投影空间.然后将计算节点按参数值投影到空间.节点投影空间描述了云计算集群在某一时刻整体工作状态.显然,对于一个良好的调度系统,计算节点在空间中的投影点应该聚集在重心点周围.

假设云计算集群中服务器的个数为 m,服务器参数个数为 n.任务调度时根据服务器的各个参数的状态进行合理的任务分配.n 个参数在调度时依据不同任务的需求,其重要性并不相同.因此假设各个参数在当前系统中进行调度时的权重分别为 $a_1,a_2,\cdots,a_n$,令 $a_1+a_2+\cdots+a_n=1$.则节点向 n 维空间投影的点集为

$$U=\{x_{i1},x_{i2}\cdots x_{in}\},1\leqslant i\leqslant m,0\leqslant x_{i1}\leqslant a_1,\cdots,0\leqslant x_{in}\leqslant a_n.$$

设节点投影点的重心位置为 $G(X_1,X_2,\cdots,X_n)$,计算重心的公式为

$$X_1=\frac{\sum_{i=1}^{m}x_{i1}}{m},\quad X_2=\frac{\sum_{i=1}^{m}x_{i2}}{m},\cdots,\quad X_n=\frac{\sum_{i=1}^{m}x_{in}}{m}.$$

重心位置显示了集群资源的总体负载情况,也是指导设计任务调度策略的重要考量参数.通过将资源节点按参数值映射到投影空间,使得研究云计算集群的负载时状况更加直观:如果各计算节点聚集于重心的周围,则集群负载均衡性较好.如果节点投影点在空间中的映射分散,则集群负载不均衡,负载均衡状况比较差.

1.2　系统负载均衡度的定义

为评估集群的负载均衡状况,可用一个参数 LB 进行衡量,定义其为负载均衡度.令云计算集群计算节点个数为 m,参数个数为 n,计算节点在 n 维参数空间中的投影重心为 $G(X_1,X_2,\cdots,X_n)$.则各节点投影到重心距离的平均值为

$$\frac{\sum_{i=1}^{m}\sqrt{\sum_{j=1}^{n}(x_{ij}-X_j)^2}}{m}, \tag{1}$$

显然，当云计算集群中负载为空与满负载的节点各占一半时，此时集群的负载均衡性最差.

计算节点参数按权重投影在参数空间中的重心位置 G 的坐标为$\frac{a_1}{2},\frac{a_2}{2},\cdots,\frac{a_n}{2}$，各节点投影点到重心位置的距离均为$\frac{\sqrt{a_1^2+a_2^2+\cdots+a_n^2}}{2}$，它是各节点到重心距离的最大值. 计算中各节点投影点到重心距离的平均值，并作归一化处理，将其定义为节点投影空间负载均衡度 LB_{ij}，表达式为

$$LB_{ij}=\frac{\sum_{i=1}^{m}\sqrt{\sum_{j=1}^{n}(x_{ij}-X_j)^2}}{m\times\frac{\sqrt{a_1^2+a_2^2+\cdots+a_n^2}}{2}}, \tag{2}$$

即任务 i 分配到节点 j 上时，系统负载均衡度的大小.

负载均衡度是衡量云计算集群负载均衡状况的重要指标，量化地给出集群的负载均衡程度. 在理想情况下，某一时刻的所有计算节点负载相同，它们的投影点聚集为一点，此时集群的负载均衡度为 0，云计算集群处于理想负载均衡状态，而 LB_{ij} 的最大取值为 1. 因此，$LB_{ij}\in[0,1]$，其值越小，表明当前云计算集群负载均衡性越好.

2 任务调度模型

本文将 n 个相互独立的计算子任务分配到 m 个虚拟机计算节点上，令 $m<n$. n 个任务的集合 $T(n)=(t_1,t_2,\cdots,t_n)$，$t_i$ 为第 i 个子任务. 任务 t_i 可由其参数表示，即

$$t_i=(t_{i\_money},t_{i\_m},t_{i\_memory},t_{i_{bw}}), \tag{3}$$

公式中：t_{i_money} 为用户期望费用；t_{i_m} 为任务大小；t_{i_memory} 为任务处理时所需内存大小；$t_{i_{bw}}$ 为任务处理所需的带宽要求.

m 个虚拟机资源可表示为 $VM(m)=(vm_1,vm_2,\cdots,vm_m)$，$vm_j$ 表示第 j 个虚拟机，其属性向量可对应表示为

$$vm_j=(vm_{j\_money},vm_{j\_capacity},vm_{j\_memory},vm_{j\_bw}), \tag{4}$$

公式中：vm_{j_money} 为虚拟机处理单位大小任务的费用；$vm_{j\_capacity}$ 为虚拟机资源的处理能力；vm_{j_memory} 为虚拟机的内存大小；vm_{j_bw} 为虚拟机能提供的传输带宽.

任务在虚拟机上的期望处理时间为

$$ET_{ij}=\frac{t_{i\_m}}{vm_{j\_capacity}}. \tag{5}$$

任务 t_i 在虚拟机 j 上起始执行时间为 st_j，期望完成时间为 RT_{ij}，即

$$RT_{ij}=st_j+ET_{ij}. \tag{6}$$

任务集合中所有任务完成时间表示为

$$RT = \max\{RT_{ij}\} \quad . \tag{7}$$

任务调度的目标函数与约束条件则为

$$\min\{RT\} \quad . \tag{8}$$

$$\begin{cases} t_{i\_memory} \leqslant vm_{j\_memory} & i = 1,2,\cdots,n, \\ t_{i\_bw} \leqslant vm_{j\_bw} & j = 1,2,\cdots,m. \end{cases} \tag{9}$$

3　构建动态优先级模型

任务的优先级决定了任务执行的优先权大小. 为任务设置优先级能够实现任务调度时的公平和效率,并能提供与用户期望费用预算、截止时间、服务质量等相称的服务. 因此,本文综合考虑任务的价值度和紧急度,构建动态优先级.

(1) 任务的价值度是实际费用与用户预算的比值,显然,它的理想值应该小于 1,用 Q_{budget} 表示,即

$$Q_{i\_budget} = \frac{t_{i\_m} \times vm_{j\_money}}{t_{i\_money}} , \tag{10}$$

显然,Q_{budget} 越小,任务优先级越高.

(2) 任务紧急度就是用户要求的任务完成时限. 本文在尽量满足任务时间约束的条件下,定义任务紧急度为任务等待时间与任务期望最早完成时间的比值,用 Q_{i_time} 表示任务紧急度,

$$Q_{i\_time} = \frac{t_{i\_wait}}{t_{i\_wait} + t_{i\_left}} , \tag{11}$$

公式中: t_{i_wait} 为任务等待的时间,$t_{i\_wait} = t_{i\_current} - t_{i\_submit}$,$t_{i\_current}$ 为当前时间,t_{i_submit} 为任务提交的时间; t_{i_left} 为任务的剩余时间,$t_{i\_left} = t_{i\_submit} + t_{i\_deadline} - t_{i\_current}$,$t_{i\_left} \geqslant 0$. 显然,随着等待时间 t_{i_wait} 的增加,剩余时间 t_{i_left} 相应地会减小,Q_{i_time} 会迅速增加,显现出任务优先级的动态特性.

为防止任务剩余时间 t_{i_left} 过短,导致任务处理超时. 需要对公式(9) 进行修正. 定义 $\overline{ET_{ij}} = \frac{1}{n}\sum_{j=1}^{n} ET_{ij}$ 为任务 t_i 在虚拟机上的平均处理时间. 当剩余时间 $t_{i\_left} \leqslant \overline{ET_{ij}}$ 时,令此时任务紧急度为 1,给与任务最高紧急度,使任务能够优先处理. 修正后的任务紧急度表示为

$$Q_{i\_time} = \begin{cases} \dfrac{t_{i\_wait}}{t_{i\_wait} + t_{i\_left}} & t_{i\_left} > \overline{ET_{ij}}, \\ 1 & t_{i\_left} \leqslant \overline{ET_{ij}}. \end{cases} \tag{12}$$

构建任务动态优先级,需要对 Q_{i_budget}、Q_{i_time} 进行标准化处理. 用 ξ_{ik} 表示任务 t_i 的第 k 个优先级指标标准化结果,表示为 $\xi_{ik} = (d_{ik} - \overline{d_{ik}}) / \delta_k$,其中: $k = 1,2$; d_{ik} 属于矩阵

$$d = \begin{bmatrix} d_{11} & d_{12} \\ \vdots & \vdots \\ d_{n1} & d_{n2} \end{bmatrix} = \begin{bmatrix} Q_{1\_budget} & Q_{1\_time} \\ \vdots & \vdots \\ Q_{n\_budget} & Q_{n\_time} \end{bmatrix} , \tag{13}$$

公式中: $\overline{d_{ik}}$ 为矩阵 d 第 k 列的平均值,$\overline{d_{ik}} = \frac{1}{n}\sum_{i=1}^{n} d_{ik}$; δ_k 为全部任务第 k 个指标的标准差,

$$\delta_k = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_{ik} - \overline{d_{ik}})} \quad . \tag{14}$$

用 $F(t_i)$ 表示任务 t_i 的动态优先级，其计算公式为

$$F(t_i) = \omega_1 \times \xi_{i1} + \omega_2 \times \xi_{i2} \ , \tag{15}$$

公式中：$\omega_1, \omega_2 \in [0,1]$ 是权重因子，且满足 $\omega_1 + \omega_2 = 1$.

4 模仿萤火虫行为的任务分配模型

人工萤火虫算法中，萤火虫依据自身亮度和吸引度彼此吸引. 而任务调度时，任务和虚拟机的匹配看做是一个相互吸引的过程. 在该过程中，可以将用户期望最早完成时间作为吸引度，任务期望处理时间作为亮度，模拟萤火虫亮度吸引的行为进行任务调度. 此外，本文中加入负载均衡度作为衡量任务与虚拟机的匹配度变量，以使任务调度时确保相虚拟机资源负载均衡.

吸引度：用户期望最早完成时间 RT_{ij} 作为吸引度，RT_{ij} 越小，吸引度越大.

荧光亮度：任务在虚拟机上的期望处理时间 PT_j 与 1 的和作为荧光亮度 LI_j. 初始时刻亮度最大；$PT_j = 0, LI_j = 1$ 节点增加负载时，$PT_j > 0, LI_j$ 值变大，亮度降低.

假设分配给虚拟机 vm_j 的任务的总长度为 L_j，则

$$PT_j = \frac{L_j}{vm_{j\_capacity}} . \tag{16}$$

荧光亮度计算公式为

$$LI_j = 1 + PT_j . \tag{17}$$

最后，综合考虑吸引度、荧光亮度以及负载均衡度，得到调度决策变量

$$\lambda_{ij} = \frac{1}{ET_{ij}} \times \frac{1}{LI_j} \times \frac{1}{LB_{ij}} . \tag{18}$$

显然，ET_{ij}、LI_j 和 LB_{ij} 越小时，λ_{ij} 越大. 此时，将任务 t_i 调度到最大 λ_{ij} 值所对应的节点上即可.

5 TS - DPSD 算法描述

本文设计的云计算中任务调度算法 TS - CCLB 过程如下：

(1) 首先由公式(2) 计算出所有任务的优先级，并对任务按优先级大小进行排序；

(2) 计算出任务在各节点上的最早完成时间 ET_{ij}，以及任务添加到各计算节点上时系统的负载均衡度 LB_{ij}，依据任务带宽要求($t_{i\_bw} \leqslant vm_{j\_bw}$) 和内存限制($t_{i\_memory} \leqslant vm_{j\_memory}$)，预先建立任务与可行虚拟机组的组合；

(3) 计算 RT_j 和 LI_j，结合步骤(2) 中的 ET_{ij} 和 LB_{ij}，计算出任务与其可行虚拟机组内的各虚拟机的调度决策变量 λ_{ij}；

(4) 找到任务 t_i 与最大 λ_{ij} 值对应的 vm_j，调度相应任务至该虚拟机上；

(5) 更新 st_j 与 LI_j，并将任务队列中已调度的任务删除掉；

(6) 返回步骤(1) 开始下一个任务的调度，直到任务调度完毕为止.

6　仿真实验结果与分析

为分析本文提出的 TS-CCLB 算法的性能，本文的模拟实验在 CloudSimToolkit 环境中进行的，改写 DatacenterBroker 类中的 bindCloudletToVm 方法，实现 TS-CCLB 调度算法，并且使用 Ant 工具将 TS-CCLB 调度算法加入到模拟平台的任务调度单元中，进行模拟实验.同理，在相同的环境配置下实现 Min-Min，Max-Min 和文献 [8] 中的 TS-PFB 算法，以及文献 [12] 中的 HBB-LB 算法.实验比较 4 种算法的总任务平均完成时间 RT、负载均衡度 LB_{ij}和截止时间内成功完成率.

本仿真实验中在戴尔 Xeon E5-2609 v3 服务器上创建了 40 个虚拟机作为计算节点，用于 50～450 个独立任务的调度.虚拟机到主机的映射分配由 Cloudsim 自带的 Time-Shared 算法实现.计算节点的属性，如表 1 所示.任务的属性表，如表 2 所示.

表 1　虚拟机节点属性

节点属性	数量	处理能力	存储能力	带宽范围
属性值	40	500～1 000	1 000～2 000	10～20

表 2　任务属性

任务属性	数量	长度	存储需求	带宽需求	截止时间
属性值	50～450	10 000～20 000	800～1 800	8～15	50～800

优先级权重因子设置为 $\omega_1=\omega_2=0.5$.

实验 1: TS-CCLB 负载均衡前后最大完成时间的比较

TS-CCLB 算法在调度决策变量加入负载均衡度前后所有任务最大完成时间比较结果，如图 1 所示.实验结果表明: TS-CCLB 算法在决策变量中加入负载均衡度后，随着任务数量的增大，最大完成时间相比于负载均衡前显著减少.

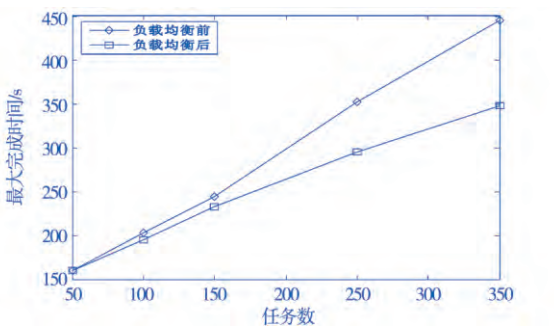

图 1　负载均衡前后的最大完成时间

实验 2: 总任务平均完成时间 RT 测试

图 2 给予 TS-PFB、Min-Min、Max-Min、HBB-LB 和本文的 TS-CCLB 五种算法在 6 组不同任务数，测试它们平均完成时间 RT.从图 2 中可以看出任务量较小时，TS-CCLB 算法略优于 Max-Min 算法，随着任务量的增大，TS-CCLB 算法的平均完成时间 RT 明显优于其他 3 种算法，实验表明 TS-CCLB 算法能有效地减小总任务的完成时间，尤其是当云计算集群规模较大时.这里用调度长度比率作为对调度策略的性能评价指标，调度长度比率为

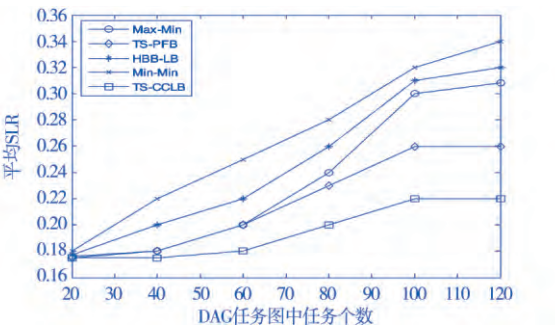

图 2　总任务平均完成时间

$$SLR=\frac{\max(ft(v_n,m_k))}{CPCC},\qquad(19)$$

公式中: $\max(ft(v_n,m_k))$ 为每组最后一个任务完成时间; $CPCC$ 为所有每组任务的用户规定完成时限之和.显然，$SLR\leqslant 1$，且 SLR 越小，说明当前调度策略的 DAG 任务图完成时间越短，其调度性能越好.

实验 3: 负载均衡性测试

随机选取 3 个虚拟机和 350 个子任务，分配到虚拟机 vm_j 上的任务总长度记为 L_j，虚拟机 vm_j 上载有任务期望处理时间 $PT_j = \dfrac{L_j}{vm_{j\_capacity}}$，全部虚拟机的期望处理时间的平均值为，全部虚拟机的期望处理时间的平均值为 $\overline{PT} = \dfrac{1}{m}\sum_{j=1}^{m} PT_j$. 本文用期望处理时间的标准差 δ 来指示虚拟机集群的负载均衡性. 计算为

$$\delta = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(PT_j - \overline{PT})^2}\ . \tag{20}$$

由公式(20) 计算出集群的负载均衡性实验结果，如图 3 所示.

随着任务数的增大，Min - Min 算法负载均衡性逐渐变差; 而 TS - CCLB 算法呈现出良好的负载均衡性，δ 值小于其他 3 种算法，且随着任务数的增大，δ 值趋于稳定. 实验结果表明 TS - CCLB 算法具有很好的负载均衡特性，并且适用于当前较大规模云计算集群的任务调度，具有良好的应用前景.

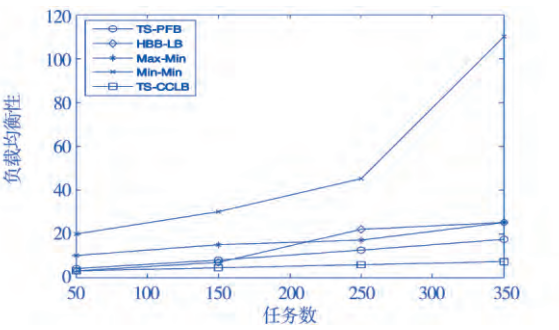

图 3 负载均衡性测试

实验 4: 任务截止时间内任务完成率测试

测试结果如表 3 所示. 由表 3 的测试结果，TS - CCLB 算法在任务截止时间内的任务完成率最高，这是在任务优先级中考虑了任务紧急度，并且设定了当 $t_{i\_left} \leqslant \overline{ET_{ij}}$ 时，令任务紧急度为 1，使截止时间短的任务具有较高的优先级，从而得到更快处理.

表 3 任务完成率测试

算法\任务数	50	150	250	350	450
TS-PFB 完成率/%	100	100	93	92	90
HBB-LB 完成率/%	100	92	90	87	86
Max-Min 完成率/%	98	91	87	85	85
Min-Min 完成率/%	96	83	83	78	78
TS-DPSD 完成率/%	100	100	98	98	98

7 结束语

本文将动态优先级与调度决策变量结合，并加入负载均衡度，将提出了一种基于负载均衡度的云任务调度算法(TS-CCLB). 实验表明，该算法在有较少总任务完成时间，同时实现了良好的负载均衡特性. 尤其是适用于较大规模云计算集群的任务调度. 本文算法将负载均衡度加入到决策变量中，使算法实现了良好的负载均衡特性，该算法能很好地满足电力用户多 QoS 需求，有效提高电力数据中心运行的效率，达到更加优越的负载均衡效果.

任务优先级作为主要的 QoS 参数，以后的工作是引入其他 QoS 参数，如用户安全，任务类别等. 同时，该算法仅模仿了萤火虫行为，没有进行全局寻优过程. 今后将进一步优化该算法的调度效率，使其能够应用于未来具有海量计算节点和任务的云计算环境.

参 考 文 献

[1] 张春艳，刘清林，孟珂. 基于蚁群优化算法的计算任务分配[J]. 计算机应用 2012，32(5)：1418-1420.
[2] 朱宗斌，杜中军. 基于改进 GA 的云计算任务调度算法[J]. 计算机工程与应用，2013，49(5)：77-80.
[3] 孙佳佳，王兴伟，高程希，等. 云环境下基于神经网络和群搜索优化的资源分配机制[J]. 软件学报，2014，25(8)：1858-1873.
[4] 魏 赟，陈元元. 基于改进蚁群算法的云计算任务调度模型[J]. 计算机工程，2015，41(2)：13-15.

[5] 李建锋,彭舰.云计算环境下基于改进遗传算法的任务调度算法[J].计算机应用,2011,31(1):185-186.
[6] Li Qi,Ba Wei.A group priority earliest deadline first scheduling algorithm[J].Frontiers of Computer Science,2012,6(5):560-567.
[7] Liu Gang,Li Jing,Xu Jian-chao,et al.An improved Min-Min algorithm in cloud computing[C]//Proc of International Conference of Modern Computer Science and Applications.Berlin: Springer,2013: 47-52.
[8] 夏家莉,陈辉,杨兵,等.一种动态优先级实时任务调度算法[J].计算机学报,2012,35(12):2685-2695.
[9] 刘亚秋,赵青华,景维鹏,等.基于动态优先级和萤火虫行为的云任务调度算法[J].计算机应用研究,2015,32(4):1041-1043.
[10] C.A.Bohna,G.B.Lamont.Load balancing for heterogenous clusters of PCs[J].Future Generation Computer System,2012,18(3):389-400.
[11] Dong Bin,Li Xiuqiao,Wu Qimeng,et al.A dynamic and adaptive load balancing strategy for parallel file with large-scale I/O servers[J].Journal of Parallel and Distributed Computing,2012,72(10):1254-1268.
[12] L.D.D.Babu,P.V.Kirshna.Honey bee behavior inspired load balancing of tasks in cloud computing environments[J].Applied Soft Computing,2013,13(5):2292-2303.
[13] G.N.Reddy,S.P.Kumar.Review of load balancing techniques in cloud computing environment: challenges and algorithms[J].International Journal of Advanced Research in Computer Science,2014,5(4):124-133.
[14] Mao Yingchi,Chen Xi,Li Xie,et al.Max-Min task scheduling algorithm for load balance in cloud computing[C]//Proc of International Conference on Computer Science and Information Technology[M].Springer,2015: 457-465.
[15] Chen Huankai,F.Wang,N.Helian,et al.User-priority guided Min-Min scheduling algorithm for load balancing in cloud computing[C]//Proc of National Conference on Parallel Computing Technologies.Piscataway: IEEE Press,2013: 1-8.
[16] Wang Peng,ZhangLei,Ren Chao,et al.The analytical model and simulation research in phase space of cloud computing[J].Chinese Journal of Computers,2013,36(2):286-296.
[17] 王鹏,黄焱,李坤,等.云计算集群相空间负载均衡度优先调度算法研究[J].计算机研究与发展,2014,51(5):1095-1107.

Task Scheduling Algorithm for Cloud Computing Based on Load Balance Degree

Ye Bo

(Electric Power Research Institute,State Grid Xinjiang Electric Power Company,Urumqi Xinjiang 830011)

Abstract: According to the characteristics of huge amount of users and data in current cloud computing environment,how to design an efficient load balancing scheduling algorithm is an important topic in the field of cloud computing.This paper presents a Task Scheduling Algorithm for Cloud Computing Based on Load Balance (TS-CCLB),Firstly,the load balance of the cluster is calculated based on the projection analysis,and give the scheduling decision variable,and the priority level is given according to the execution cost and the completion time of the task.The experimental results show that the algorithm can effectively improve the load balancing of the cloud computing cluster and shorten the total task completion time,especially when the number of tasks and the node size is larger,the advantage is more obvious.

Key words: Cloud computing; Task Scheduling; Load Balancing Degree; Priority