Journal of Computer Applications
计算机应用,2013,33(8):2140-2142,2166
ISSN 1001-9081
CODEN JYIIDU
2013-08-01
http://www.joca.cn

文章编号:1001-9081(2013)08-2140-03
doi:10.11772/j.issn.1001-9081.2013.08.2140

异构云中面向集群负载均衡的任务调度策略

刘卫宁[1,2],高　龙[1,2*]
(1.重庆大学 计算机学院,重庆 400044;　2.信息物理社会可信服务计算教育部重点实验室(重庆大学),重庆 400044)
(*通信作者电子邮箱 menglong036620@163.com)

摘　要:负载均衡是提高资源利用率和系统稳定性的重要手段。基于改进的自适应变异粒子群算法,提出了一种异构环境下面向集群负载均衡的任务调度策略。在调度策略的设计中,融入了经济学"二八"定律,通过把握用户对集群节点安全性和可靠性的偏好程度并预估任务的负载信息,在保证系统负载尽量均衡的前提下,最小化任务执行时间的同时提高大客户满意度。仿真实验显示,改进的自适应变异粒子群算法比未改进的自适应变异粒子群算法和基本粒子群算法在收敛速度和跳出局部最优两个方面都有更好的表现。结果表明,改进的自适应变异粒子群算法在保证集群负载均衡的同时可以更好地提高云服务提供商的利润空间。

关键词:负载均衡;任务调度;"二八"定律;异构;自适应变异粒子群
中图分类号: TP393.027;TP18　　**文献标志码:**A

Task scheduling strategy based on load balance of cluster in heterogeneous cloud environment

LIU Weining[1,2], GAO Long[1,2*]
(1. *College of Computer Science, Chongqing University, Chongqing* 400044, *China*;
2. *Key Laboratory of Dependable Service Computing in Cyber Physical Society, Ministry of Education (Chongqing University), Chongqing* 400044, *China*)

Abstract: Load balancing is an important means to improve resource utilization and system stability. Based on Adaptive Mutation Particle Swarm Optimization (AMPSO) algorithm, a new task scheduling model and strategy about load balancing for cluster in heterogeneous cloud environment were proposed. In order to maximize customer satisfaction degree and reduce the total execution time of a collection of tasks under ensuring the system load as much balanced as possible, a concept of user bias degree on cluster node performance such as safety and reliability and a method of grasping the degree of preference on security and reliability of cluster nodes and estimating the load information of the tasks were added into the design of scheduling policy. The simulation shows that the improved AMPSO algorithm performs better than the original AMPSO algorithm and the basic Particle Swarm Optimization (PSO) algorithm at convergence speed and the capacity of jumping out the local optimum. The results prove that the improved AMPSO can better improve the profit margins of the cloud service provider while ensuring the load balancing of the cluster.

Key words: load balancing; task scheduling; law two-eight; isomerism; Adaptive Mutation Particle Swarm Optimization (AMPSO)

0　引言

负载均衡是提高资源利用率和系统稳定性的重要手段,是云计算相关研究的重点。负载均衡就是将系统流量按照节点的实际情况均衡分配,使集群中资源的利用率得到提高[1]。

目前针对负载均衡应用最多的两类策略是基于RR(Round Robin)算法和遗传算法(Genetic Algorithm, GA)实现的[2]。其中RR算法通过将任务按照顺序依次分配给不同的节点执行,具有简单快速的优点,但却因为无法随着系统负载的动态变化而自我调整,所以只适合于规模不大的任务调度[3]。使用智能算法解决大规模系统的负载均衡问题是近年来研究的热点,其中包括遗传算法、蚁群算法和粒子群算法[4-5]。文献[6-7]提出了用遗传算法来解决异构分布式系统中的调度问题。然而,遗传算法对新空间的探索能力是有限的,很容易收敛到局部最优解,出现"早熟"现象[8]。当涉及到大量个体的计算时,因为牵扯到交叉变异等复杂操作,使得算法无法在短时间内结束。与遗传算法相比,粒子群算法具有更快的收敛速度,但也同样存在"早熟"现象,尤其是处理多峰优化问题时[9]。文献[10]提出了使用关系矩阵编码的粒子群算法解决负载均衡问题。在此基础上,自适应变异粒子群算法(Adaptive Mutation Particle Swarm Optimization, AMPSO)通过对全局最优位置 *gBest* 进行自适应变异操作,一定程度上增加了算法跳出局部最优的能力,然而由于变异的随机性,使得很大程度上变异后的解不如变异前的解更优,因此其跳出局部最优的能力也是有限的。

"二八"定律是经济学经典定律,由意大利经济学家帕累托提出,它认为在任何一组东西中,最重要的只占其中一小部分,约20%,其余的80%尽管是多数,却是次要的。与此定律吻合,云服务提供商大多数的利润来源于一小部分长期使用的客户,这部分客户可以称为大客户。在云计算中,"二八"定律强调了大客户的重要性,认为通过提高大客户的满意度,能更好地增加提供商的收益。因此,大客户满意度在任务调度中是不容忽视的。此外,任务的执行时间反映了任务的执

收稿日期:2012-02-04;**修回日期:**2013-03-28。
基金项目:国家自然科学基金资助项目(61203135);国家科技支撑计划项目(2012BAH19F01)。
作者简介:刘卫宁(1965-),女,重庆人,教授,博士生导师,主要研究方向:智能计算与服务、物流与供应链管理、网络与分布式计算;　高龙(1988-),男,山东邹平人,硕士研究生,主要研究方向:云计算、智能计算。

行效率,所以缩短任务的执行时间也是任务调度追求的目标之一。

针对上述问题及相关启示,本文在自适应变异粒子群算法的基础之上对其进行了改进,提出了一种异构集群环境中面向集群负载均衡的任务调度模型和策略,通过在合适的时机对合适的粒子进行变异,从而更大程度上增加了算法跳出局部最优的能力。

1　相关模型

异构环境下的面向负载均衡的任务调度可以简单描述为:以由 n 个性能不同的节点所构成的异构集群处理由 m 个不同用户所提交的 m 类独立不相关任务[11],使得在确保负载尽量平衡的前提下,最小化任务总执行时间的同时,最大限度地提高大客户满意度。基于文献[12]描述的架构,本文将调度过程分为任务到虚拟机配置的映射和虚拟机到物理机节点的映射两步。将任务构成的集合记为 $S_{\text{job}}=\{J_1,J_2,\cdots,J_m\}$,将虚拟机配置构成的集合记为 $S_{\text{vm}}=\{V_1,V_2,\cdots,V_v\}$。

1.1　节点

为了描述异构集群中各个节点的能力差异,选取集群中一个节点 i 作为参考,并令其节点性能为 1,将节点 j 的固有能力定义为节点 j 的性能与节点 i 的性能之比:

$$C_j=Cap_j/Cap_i \tag{1}$$

其中:Cap_i 和 Cap_j 表示节点 i 和节点 j 的性能,可以参考文献[13]提及的性能表示方法。定义节点 j 的资源数量为 RP_j,t 时刻节点 j 的剩余资源数量为 RPL_j,故异构集群中物理节点 j 可以定义为

$$N_j=(C_j,RP_j,NS_j,NR_j) \tag{2}$$

其中:NS_j 代表节点 j 的安全性级别,NR_j 代表节点 j 的可靠性级别。

1.2　负载

负载的定义需要考虑资源利用率和机器固有能力两个方面。比如,对于资源利用率同为 80% 的 A、B 两个节点而言,固有能力较高的 A 节点比固有能力较低的 B 节点具有更小的运行压力。为了描述 t 时刻整个集群的负载信息,参考文献[14]中所介绍的负载计算方法,将 t 时刻前 T 时间段作为负载观察期,通过定时检测虚拟机运行情况,计算出观察期内各个虚拟机的平均资源利用率作为 t 时刻虚拟机 i 的负载,并记作 L_{vi}。将节点 j 在 t 时刻的负载定义为

$$L_{pj}=\frac{1}{T\times C_j}\sum_{i=1}^{n}(L_{vi}\times t_{vi}) \tag{3}$$

其中:n 表示观察周期 T 内节点 j 运行的虚拟机个数(包括中途结束运行的虚拟机);t_{vi} 表示虚拟机 i 在 T 时间段中运行的时间长度;C_j 表示节点 j 的固有能力。

在将虚拟机分配到物理机节点执行的过程中,为了实现分配过程保证负载均衡,需要预估这 m 个虚拟机在运行时的负载,根据式(3)对于虚拟机负载的计算方法,将对虚拟机负载的估算转换为对虚拟机资源利用率的估算。计算方法如下:

将任务根据规模大小分成 K 类,分类结果记作 $S_c=\{S_{c1},S_{c2},S_{c3},\cdots,S_{ck}\}$。每个任务对应于集合 S_c 中的一个元素,用于指明任务的规模大小;对于每个任务,也对应于集合 S_v 中的一个元素,用于指明系统为此任务分配的虚拟机。系统动态维护一个矩阵 $\boldsymbol{A}_{k\times m}$,其中 a_{ij} 表示在一定的观察周期内,V_i 配置的虚拟机运行 S_{cj} 规模任务的资源利用率。假设预估的虚拟机 i 的负载是 L_{vi},t 时刻将虚拟机 i 部署到节点 j 上之后的负载就是 $L_{pj}'=L_{pj}+L_{vi}$,L_{pj} 为 t 时刻任务分配到节点 j 之前的负载大小。对于集群负载稳定状态的描述使用负载方差来表示:

$$\sigma^2=\frac{1}{N}\sum_{j=1}^{N}(L_{pj}-\bar{L}_p)^2 \tag{4}$$

其中:$\bar{L}_p=\frac{1}{N}\sum_{j=1}^{N}L_{pj}$,$N$ 表示集群中节点的个数。σ^2 的值越大,表示系统的负载均衡性越差。

1.3　任务描述

任务的执行情况受集群节点的安全性和可靠性的影响。集群节点越安全可靠,任务完成得就越高效,客户的满意度就越高,因此客户的满意程度与节点安全性和可靠性呈正相关关系。基于此思想,将任务的数学模型定义为

$$J_i=(RJ_i,RS_i,RR_i,U_i,V_i,C_i) \tag{5}$$

其中:RJ_i 表示任务请求的资源数;RS_i 表示用户对执行此任务的节点的安全性等级的最低期望值,该值指定了用户希望执行此任务的节点具有的最低安全性等级;RR_i 表示用户对执行此任务的节点的可靠性等级的最低期望值,该值指定了用户希望执行此任务的节点具有的最低可靠性等级;U_i 表示提交任务的用户等级;V_i 表示任务 i 对应的虚拟机配置,对应于 S_{vm} 中某个元素对应的序号;C_i 表示任务的规模,对应于 S_c 中某个元素对应的序号。

1.4　目标函数

为了保证本文所提及的在保证负载尽量均衡的前提下,最小化任务总执行时间的同时提高大客户满意度的目标,指定一个代表大客户的等级值 l 和一个负载方差上限 σ_c,并建立如下目标函数:

$$\max(F)=\left(k_1\times 100\times\frac{|S_s|}{|S_{\text{all}}|}\right)\Big/\left(k_2\times\sum_{i=1}^{m}T_i\right) \tag{6}$$

其中:$S_{\text{all}}=\{J_i\mid J_i\in S_{\text{job}}\ \text{and}\ U_i\geqslant l\}$,$S_s=\{J_i\in S_{\text{all}}\ \text{and}\ RS_i\leqslant NS_j\ \text{and}\ RR_i\leqslant NR_j\}$,且必须满足如下两个约束条件:

$$\sigma^2\leqslant\sigma_c \tag{7}$$

$$\sum RJ_i<RPL_j \tag{8}$$

其中:约束(7)表示,在每一次调度方案下,计算所得的系统负载方差必须小于预先设定的方差上限;约束(8)表示分配到节点 j 上的所有任务请求的资源总和要小于当前时刻节点 j 剩余的资源。本文的目的就是在约束的前提下使目标函数(6)的值最大。本文将基于改进的自适应变异粒子群算法对问题进行求解。

2　调度策略

2.1　改进的自适应变异粒子群算法

自适应变异的粒子群算法在基本粒子群算法的基础之上,为了克服基本粒子群算法极易出现“早熟”的缺点,通过加入群体适应度方差和随机变异算子,对全局极值 $gBest$ 进行随机变异,以此来提高算法跳出局部最优解的能力,算法的相关公式如下。

1)粒子速度和位置的更新公式[15]:

$$v_{id}^{k+1}=w*v_{id}^{k}+c_1*rand_1^k*(pBest_{id}^{k}-x_{id}^{k})+c_2*rand_2^k*(gBest_{id}^{k}-x_{id}^{k}) \tag{9}$$

$$x_{id}^{k+1}=x_{id}^{k}+v_{id}^{k+1} \tag{10}$$

为了保证算法前期具有较好的全局搜索能力,在后期具有较好的局部搜索能力,公式中的惯性权重 w 采用线性减小的方式设定。

2)粒子群群体适应度方差计算公式:

$$\sigma^2 = \sum_{i=1}^{n}\left(\frac{f_i - f_{avg}}{f}\right)^2 \tag{11}$$

其中：n 表示粒子群的粒子数目；f_i 表示粒子 i 的适应度；f_{avg} 表示 n 个粒子的平均适应度；f 是一个随着算法进化而不断变化的任意值，目的是归一化群体适应度方差，使得 $|f_i - f_{avg}|$ 的值不大于 1。一般 f 的取值采用式(12)：

$$f = \begin{cases} \max\{|f_i - f_{avg}|\}, & |f_i - f_{avg}| > 1 \\ 1, & \text{其他} \end{cases} \tag{12}$$

3)随机算子 pm 的计算公式：

$$pm = \begin{cases} k, & \sigma^2 < \sigma_d^2 \text{ 且 } f(pBest) < f_d \\ 0, & \text{其他} \end{cases} \tag{13}$$

其中：$k \in [0.1, 0.3]$；σ_d^2 表示期望群体适应度方差，此值与实际问题相关；f_d 表示期望最优值，可以设置为理论最优解。

4)为了克服原有自适应变异粒子群算法跳出局部最优能力有限的不足，对原有算法进行改进，将原来只针对全局最优点执行变异的操作扩展到更多的粒子，变异公式如下：

$$P_k = P_k \times (1 + 0.5 \times \eta) \tag{14}$$

其中：η 是符合标准正态分布的随机变量，P_k 指粒子 p 的第 k 维数值。将 n 个粒子按照适应度从低到高进行排序，选择前 $n/2$ 的粒子和 $gBest$ 粒子作为待变异粒子执行式(14)的变异操作。将变异后的粒子适应度值与变异前 $gBest$ 适应度值进行比较，取两者中适应度最大的粒子作为新的 $gBest$，将另一个粒子作为当前的 $pBest$。

2.2 调度策略

2.2.1 粒子编码

本文对粒子的编码采用多维向量的方式，每一维表示的是接收任务的节点编号。如使用(1,3,2)来表示一个粒子，表示任务 1 分配到节点 1 上，任务 2 分配到节点 3 上，任务 3 分配到节点 2 上。当在算法中通过式(9)、(10)来更新粒子的位置和速度后，会出现浮点数的情况，为了将浮点数变得有意义，采用四舍五入的方式对数据进行规约。如更新后的粒子编码是(2.1,3.6,2.5)，则规约成(2,4,3)，将数据变得具有实际意义。

2.2.2 算法步骤

将目标函数(6)作为粒子的适应度函数，对于不满足约束的粒子，采用保留 $pBest$ 值并重新初始化的方式对粒子进行修复。基于此前的模型和描述，任务调度的步骤描述为：

1)为 S_{job} 集合中的每个任务分配资源最适配的虚拟机，指定代表大客户的用户等级 l 和可接受的负载方差上限 σ_c。

2)在约束(7)和(8)允许的范围内，初始化各个粒子的位置和速度，将粒子的 $pBest$ 设置为当前位置，$gBest$ 设置为初始粒子群中最佳粒子的位置。

3)判断算法终止条件是否满足，如果满足则跳到 10)，否则执行 4)。

4)对每个粒子根据式(9)和(10)更新速度和位置，通过式(4)计算当前调度方案下的集群负载方差。

5)判断更新后的粒子是否满足约束(7)和(8)，如果满足，则跳到 7)，否则执行 6)。

6)保留该粒子的 $pBest$ 值，对其循环进行重新初始化，直到满足约束(7)和(8)时停止。

7)通过式(6)计算粒子的适应度，如果粒子适应度优于 $pBest$ 的适应度，则将 $pBest$ 设置为新位置；如果粒子群中最高适应度优于 $gBest$ 的适应度，则将 $gBest$ 设置为新位置。

8)根据式(11)和(12)计算粒子群适应度方差 σ^2，根据式(13)计算变异概率 pm。

9)随机产生数 $r \in [0,1]$，如果 $r < pm$，则根据式(14)提及的方法，选择需要变异的粒子，对其进行变异操作，并更新 $pBest$ 和 $gBest$，否则执行 3)。

10)输出结果，算法结束。

3 实验分析

为了验证本文所提及的策略解决任务调度问题的能力并分析算法的性能，采用仿真实验的形式，对本文所采用的改进自适应变异粒子群算法与基本粒子群算法和未改进的自适应变异粒子群算法进行了比较。实验模拟了 4 台异构物理机节点，3 个任务规模和 2 个虚拟机配置，通过模拟 30 个任务和 60 个任务两种情况绘制了图 1 和图 2。

图 1 通过统计的方式，给出了不同迭代次数下三种算法得到最优解的能力。可以看出，改进的 AMPSO 算法和未改进的 AMPSO 算法都明显比基本 PSO 算法具有更快的收敛速度，即使用更少的迭代次数便可以逼近最优值。迭代超过 500 代以后，两种 AMPSO 算法的 20 次平均最佳适应度值有了明显的差距，这说明在使用同样的迭代次数时，本文提出的改进 AMPSO 算法比未改进的 AMPSO 算法更容易逼近最优值。

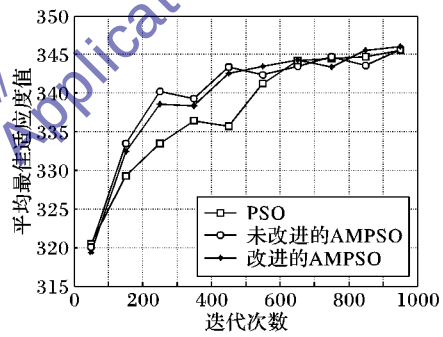

图 1 30 个任务时 20 次实验的平均最佳适应度曲线

图 2 比较了三种算法的目标函数值，可以看出，基本 PSO 算法在 80 代左右就陷入了局部最优解，而通过自适应变异的方式，未改进的 AMPSO 算法和改进的 AMPSO 算法都顺利地跳出了局部最优。同时，在第 250 次迭代之后，两种 AMPSO 算法的目标函数值出现了分叉。本文改进的 AMPSO 算法的目标函数值明显高于未改进的 AMPSO 算法的目标函数值，这是因为改进过的 AMPSO 算法具有更大的变异可能性，能更好地增加群体的多样性，进而具有更强的跳出局部最优的能力。

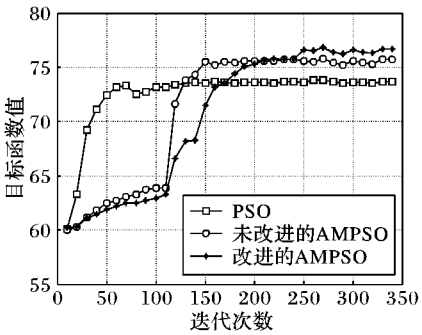

图 2 60 个任务时各算法的目标函数值对比

通过图 1 和图 2 可以看到，使用未改进 AMPSO 算法和本文提出的改进 AMPSO 算法都可以用来解决本文提出的模型，但是改进的 AMPSO 算法在收敛速度和跳出局部最优能力方面表现更加出众，故对解决本文提出的问题更加有效。

(下转第 2166 页)

一件困难的事情,特别是一旦当前绑定的 Web 服务失效,需要重新进行服务的查找,因此效率较低。为了缩减服务查找空间,提高查找效率,本文提出服务簇的概念以及面向服务簇的服务请求/响应模式。文中给出了服务簇的形式化定义、面向服务簇的服务请求/响应模式以及服务体系结构,设计了服务查找算法,并进行了仿真实验,结果表明面向服务簇的服务请求/响应模式在服务发现方面具有较高效率,能够大幅度地降低服务再查找时间。下一步的工作是对服务簇的粒度划分进行研究,发现粒度划分方案以及粒度与服务发现效率之间的关系。

参考文献:

[1] MAGUIRE T, SNELLING D. Web services service group 1.2[M]. London: OASIS, 2006:1 - 42.

[2] LIU X Z, HUANG G, MEI H. Discovering homogeneous Web service community in the user-centric Web environment [J]. IEEE Transactions on Service Computing, 2009, 2(2):167 - 181.

[3] 刘譞哲, 黄罡, 梅宏. 用户驱动的服务聚合方法及其支撑框架[J]. 软件学报, 2007, 18(8): 1883 - 1895.

[4] SHENG Q Z, BENATALLAH B, MAAMAR Z, *et al.* Configurable composition and adaptive provisioning of Web services [J]. IEEE Transactions on Services Computing, 2009, 2(1):34 - 49.

[5] BENATALLAH B, SHENG Q Z, DUMAS M. The Self-Serv environment for Web services composition [J]. IEEE Internet Computing, 2003, 7(1):40 - 48.

[6] LIU J, WANG H Y, CUI L Z, *et al.* Method and supporting framework for business domain-oriented Web service discovery [J]. Journal of Southeast University: English Edition, 2008, 24(3): 369 - 371.

[7] 陈科, 成毅, 谢明霞, 等. 基于服务簇的空间信息服务自动发现[J]. 计算机工程, 2012, 38(24):182 - 190.

[8] 栾文静, 杜玉越. 一种基于服务簇网元模型的 Web 服务发现方法[J]. 计算机科学, 2012, 39(8): 147 - 152.

[9] 李庆诚, 左珊珊, 董振华, 等. 中文 RSS 信息自动检索与分类研究[J]. 计算机工程, 2011, 37(6):79 - 81.

[10] ALEXANDER B, HIRST G. Semantic distance in WordNet: an experimental, application-oriented evaluation of five measures [C]// Workshop on WordNet and Other Lexical Resources: Second Meeting of the North American Chapter of the Association for Computational Linguistics. Pittsburgh: [s.n.], 2001:29 - 34.

[11] LI Y, BANDAR Z A, MCLEAN D. An approach for measuring semantic similarity between words using multiple information sources [J]. IEEE Transactions on Knowledge and Data Engineering, 2003, 15(4): 871 - 882.

[12] GANJISAFFAR Y, ABOLHASSANI H, NESHATI M, *et al.* A similarity measure for OWL-S annotated Web services [C]// Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence. Washington, DC: IEEE Computer Society, 2006: 621 - 624.

[13] ZHONG J W, ZHU H P, LI J M, *et al.* Conceptual graph matching for semantic search [J]. Conceptual Structures: Integration and Interfaces, 2002 23(9): 92 - 106.

[14] 孙萍, 蒋昌俊. 利用服务聚类优化面向过程模型的语义 Web 服务发现[J]. 计算机学报, 2008, 31(8): 1340 - 1353.

[15] YALCIN N, BAYRAKDAROGLU A, KAHRAMAN C. Application of fuzzy multi-criteria decision making methods for financial performance evaluation of Turkish manufacturing industries [J]. Expert Systems with Applications, 2012, 39(1): 350 - 364.

(上接第 2142 页)

4 结语

本文研究了异构云中面向集群负载的任务调度问题,通过融入“二八”定律,对任务调度过程加入了大客户满意度概念,提出了一种更利于提升云服务提供商利润的新模型。模型充分考虑了异构云环境下各节点固有能力之间的差异,把用户对集群节点可靠性和安全性的偏好程度加入到策略中来,以求达到在保证集群负载均衡的情况下,最小化任务执行时间的同时,尽可能提高大客户满意度的目标。并提出了基于改进自适应变异粒子群算法的实现,新算法对原 AMPSO 算法做了改进,通过将全局最优位置和适应值较低的粒子进行自适应变异,使得算法具有更强的跳出局部最优的能力。通过仿真实验分析表明,改进的自适应变异粒子群算法在解决本文提出的问题时,比基本 PSO 算法和未改进的 AMPSO 算法具有更高的效率。但本文的重心放在了新模型和策略的阐述上,并没有对可以实现此策略的其他算法进行深入的研究和比较,故无法得出一个最适合本模型和策略的实现算法,这将是以后重点研究的方向。

参考文献:

[1] 刘晔, 沈潇军, 刘摩西, 等. 基于云模式的资源调度与负载均衡研究[J]. 电脑知识与技术, 2011, 7(33):8208 - 8210.

[2] 杨锦, 李肯立, 吴帆, 等. 异构分布式系统的负载均衡调度算法[J]. 计算机工程, 2012, 38(2):166 - 168.

[3] 凌云, 周华锋. 面向异构集群系统的动态负载均衡技术研究[J]. 计算机工程与设计, 2008, 29(12):3068 - 3070.

[4] 张春艳, 刘清林, 孟珂, 等. 基于蚁群优化算法的云计算任务分配[J]. 计算机应用, 2012, 32(5):1418 - 1420.

[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):184 - 186.

[6] MARTINO D, MILILOTTI M. Sub optimal scheduling in a grid using genetic algorithms[J]. Parallel Computing, 2004, 30(5):553 - 565.

[7] ANDREW P, THOMAS N. Framework for task scheduling in heterogeneous distributed computing using genetic algorithms [J]. Artificial Intelligence Review, 2005, 24(3):415 - 429.

[8] 张铃, 张钹. 遗传算法机理的研究[J]. 软件学报, 2000, 11(7): 945 - 952.

[9] 吕振肃, 侯志荣. 自适应变异的粒子群优化算法[J]. 电子学报, 2004, 32(3):416 - 420.

[10] 秦勇, 宋继光, 蔡昭权, 等. 基于关系矩阵编码的粒子群负载均衡算法研究[J]. 计算机应用与软件, 2011, 28(5):126 - 128.

[11] QIN X, XIE T. An availability-aware task scheduling strategy for heterogeneous systems [J]. IEEE Transactions on Computers, 2008, 57(2): 188 - 199.

[12] FANG Y Q, WANG F, GE J W. A task scheduling algorithm based on load balancing in cloud computing [C]// Web Information Systems and Mining, LNCS 6318. Berlin: Springer, 2010:271 - 277.

[13] TIAN W H, ZHAO Y, ZHONG Y L, *et al.* A dynamic and integrated load-balancing scheduling algorithm for Cloud datacenters [C]// CCIS 2011: Proceedings of the 2011 IEEE International Conference on Cloud Computing and Intelligence Systems. Piscataway: IEEE, 2011:311 - 315.

[14] HU J H, GU J H, SUN G F, *et al.* A scheduling strategy on load balancing of virtual machine resources in cloud computing environment [C]// PAAP 2010: Proceedings of the 2010 Third International Symposium on Parallel Architectures, Algorithms and Programming. Piscataway: IEEE, 2010: 89 - 96.

[15] 杨维, 李歧强. 粒子群优化算法综述[J]. 中国工程科学, 2004, 6(5):87 - 94.