基于粒子群优化的云计算低能耗资源调度算法

贾嘉, 慕德俊

(西北工业大学 自动化学院, 陕西 西安 710072)

摘 要:针对云计算环境中能耗过高问题,提出一种基于粒子群优化方法的云计算低能耗资源调度算法。首先建立了云环境中资源调度的能耗模型;在此模型基础上,指出能耗最优是多目标优化的帕累托(Pareto)最优问题。根据能耗模型,将粒子参数设为服务器分配状态和频率分配状态,从而寻找获得单粒子的局部最优帕累托解集;合并多个粒子最优解集,得到单个分配方案下帕累托全局最优解(Pareto optimality)集合;最后,在不同分配方案对应的最优解集合中寻找最优解。实验验证了所提算法的有效性。与广泛使用的轮询调度算法比较,所提算法的动态能耗为轮询算法的 45.5%。

关 键 词:云计算;代价函数;帕累托最优;资源调度;粒子群;调度算法
中图分类号:TP391 **文献标志码**:A **文章编号**:1000-2758(2018)02-0339-06

随着对云计算需求的不断扩大,几大云计算服务提供商,如 Amazon 等,建立了越来越多的数据中心,以满足云计算对基础设施资源的需要。为了维护大规模的数据中心运行,需要大量能耗。这不仅提高了服务成本,也给相应基础设施带来巨大压力。有研究表明,数据中心的使用率一般在 5% ~ 20%[1-2];而空闲服务器的电量消耗也超过满负荷情况下的 50%。能耗是云计算成本中重要一环。针对云环境中数据中心的能耗问题,本文研究了面向低能耗的基于粒子群优化算法的数据中心资源分配算法。采用帕累托最优解集合描述所有可行的最优解。为了得到帕累托最优解集合,采用粒子群优化算法,对不同的分配方案迭代寻优,并按存活周期随机变异分配方案。通过比较不同分配方案下能耗,得到最优的资源分配方案。

1 云计算与粒子群优化

1.1 云计算概述

云计算作为一种新兴的并行计算技术,是分布式处理、并行处理、网格计算的发展和延伸。它提供了更可靠、更安全的存储和计算数据能力、简化计算交付、降低成本、具有更高的扩展性和灵活性。云计算采用计时付费的形式,程序员只需关注应用程序本身,关于集群的处理问题,则交由平台处理。云计算重要特点之一是虚拟化。虚拟化技术范畴从简单的硬件抽象逐渐发展成为虚拟云操作系统,云主机能够大量根据用户定义的服务质量(quality of service,QoS)规范执行应用程序的 VMs 并行分享。

1.2 粒子群优化

粒子群优化(particle swarm optimization,PSO)算法[3]是一种模拟鸟群觅食行为的演化计算算法。因其结构简单、参数少、易实现,所以受到广泛重视并被应用到了许多自然科学和工程科学领域。标准的 PSO 的形式如下

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (p_i^t - y_i^t) + c_2 r_2 (p_g^t - y_g^t)$$
$$y_i^{t+1} = y_i^t + v_i^{t+1} \quad (1)$$

式中,ω 为惯性权重;y_i 是粒子 i 当前的位置;v_i 表示其速度;p_i^t 是 t 时刻粒子 i 的最优位置;p_g^t 是 t 时刻所有粒子的全局最优位置;c_1,c_2 是学习因子,控制粒子更倾向于全局最优解还是局部最优解;r_1, r_2 是在 [0,1] 内均匀分布随机数。

收稿日期:2017-06-12 **基金项目**:科技部重大专项基金(2016YFB0301203)资助
作者简介:贾嘉(1984—),女,西北工业大学博士研究生,主要从事云计算及物联网技术研究。

2 云计算能耗模型

云计算体系下,数据服务需要满足很多限制性条件,如相应时间、吞吐量等。对于资源调度而言,这些指标构成了调度算法的约束条件。本节首先研究了与调度相关的能耗模型,得到能耗的代价函数及约束条件;并指出,所求问题的解为帕累托最优解集合。然后采用粒子群优化算法对代价函数求取最优解集合。

2.1 能耗模型

在不考虑散热系统的情况下,数据中心最大的能量消耗是计算过程的能量消耗,即动态能量消耗。假设现有 m 台服务器;第 j 号服务器对应最低和最高工作频率分别为 $F_j^{\min}$ 和 $F_j^{\max}$;需要将 n 个虚拟主机调度到服务器上。根据云计算虚拟资源优化分配研究中的模型,能耗的代价函数为

$$\min \sum_{j=1}^{m} \left(\sum_{i=1}^{n} f_{ij} x_{ij} \right)^3 \tag{2}$$

$$\min \sum_{j=1}^{m} \left(\sum_{i=1}^{n} f_{ij} x_{ij} - F_j^{\max} \right)^3 \tag{3}$$

约束条件为

$$x_{ij} \in \{0,1\},\ i = 1,2,\cdots,n;\ j = 1,2,\cdots,m \tag{4}$$

$$\sum_{j=1}^{m} x_{ij} = 1,\ \forall i \tag{5}$$

$$F_j^{\min} \leqslant f_{ij} \leqslant F_j^{\max},\ \forall i,j \tag{6}$$

$$\sum_{i=1}^{n} f_{ij} x_{ij} \leqslant \max F_j,\ \forall j \tag{7}$$

式中,$x_{ij} = 1$ 表示第 i 个虚拟主机调度到第 j 个服务器,取 0 则否;f_{ij} 表示第 i 个虚拟主机在第 j 个服务器上占用的频率。

2 个代价函数分别对应了不同的能耗指标。代价函数(2) 代表了所有虚拟主机总动态功耗之和最小;代价函数(3) 表示尽可能提高某一台服务器的利用率。代价函数(2) 和(3) 存在一定的冲突关系,从而构成一个典型的多目标优化问题。与之相关,约束条件则反应了虚拟主机与服务器的硬性关系。约束条件(4) 中,$x_{ij} = 1$ 表示虚拟主机 i 分配到服务器 j 上,而约束(5) 表示虚拟主机只能被分配到 1 台服务器;约束(6) 指出每个虚拟主机占用频率的上下限;约束(7) 表示 1 台服务器上所有虚拟主机占用的频率和不能超过服务器负载。

决策变量 $x = \{x_{ij}\}$ 和 $f = \{f_{ij}\}$ 构成了最终的解。x 决定了虚拟主机在服务器上的分配;而 f 则指出该分配下虚拟主机使用的频率大小。如前所述,代价函数(2) 和(3) 计算决策变量 x 和 f 是一个多目标优化问题,传统的最优化算法难以求解。本节剩余部分讨论了求解 x 和 f 的方法。

2.2 多目标优化的帕累托最优

一般而言,代价函数(2) 和(3) 是相互冲突的,所以不存在使二者同时达到最小的惟一解,而是产生一组解的集合。数学上,把这种没有一个解比其他解更优的最优解集合称为帕累托最优解集或帕累托最优前沿(Pareto optimal front, POF)。此时,对于(2) 和(3) 对应的代价函数,任意 2 个解 f_1 和 f_2 存在 2 种可能的关系:一个解优于另一个解,或者两者都不比对方优。

本文采用帕累托占优条件[4] 来判断解之间的关系。针对(2) 和(3) 对应的代价函数的解 f_1 和 f_2,当 f_1 的一个代价函数严格优于 f_2,且另一个代价函数 f_1 不严格劣于 f_2,则称 f_1 帕累托占优 f_2,即当满足下式时,f_1 帕累托占优 f_2。

$$\begin{aligned} &\forall i:\ g_i(f_1) \leqslant g_i(f_2) \\ &\exists j:\ g_j(f_1) < g_j(f_2) \end{aligned} \tag{8}$$

式中,g_i,$i = 1,2$ 是代价函数(2) 或(3)。在给定了一个服务器分配方案 x 后,由(8) 可以评估任意 2 个频率分配方案 f_1 和 f_2。

3 基于粒子群优化的低能耗资源调度算法

由第 2 节可知,待求的决策变量为 x 和 f。由于需要指定一个服务器分配方案 x 后,才可能获取一个频率分配方案 f,因此,本文首先随机给出一个分配方案 x;在该方案下,使用经典的粒子群算法进行寻优,得到该方案对应的帕累托最优解集合;通过随机改变方案,得到不同方案的最优解集合,合并成为全局最优解集合,并在该集合中寻找最终解。

3.1 频率分配方案

首先随机产生一个服务器分配方案 x_t。在 x_t 下,按照约束条件随机生成关于频率分配 f_t 的粒子群 $\boldsymbol{y} = \{y_i\}$。对于每个粒子 y_i,按照式(1) 所示的迭代过程,进行寻优。

1) 粒子群寻优。对于每一个方案,按照经典的粒子群算法(1),需要使用粒子的局部最优解 p_i 和

全局最优解 p_g。多目标最优情况下,每个粒子 y_i 存在多个局部最优,进而合并所有粒子将产生多个全局最优。因此,本文算法中允许每个粒子最多产生的 N_l 个局部最优,即 $\boldsymbol{p}_i = \{p_{i,1}, p_{i,2}, \cdots, p_{i,N_l},\}$,而合并所有局部最优得到的全局最优集合,最多包含 N_g 个元素,即 $\boldsymbol{p}_g = \{p_{g,1}, p_{g,2}, \cdots, p_{g,N_g}\}$。对于多余的解,有不同的处理方法。与文献[5]不同,本文算法采用随机抛弃多余解的方法。显然,这有利于提高算法的实时性。

2)迭代终止条件。为了停止粒子群迭代寻优,本文设计依据服务器分配方案存活时间的服务器分配方案跳转概率 P_x。P_x 具体的构造方法在 3.2 节讨论。发生跳转时,算法存储当前 x_t 及其对应的全局最优解 $f_t = \boldsymbol{p}_g$。取遍所有可能的组合 x_t 及其对应的 f_t,合并所有解得到最终的解集合 $\boldsymbol{x} = \{x_t\}$ 和 $\boldsymbol{f} = \{f_t\}$。

3)3 种帕累托最优解集合。本文涉及到 3 种不同的帕累托最优解集合,为了避免混淆,在此指明其相互关系。给定服务器分配方案 x_t 下,最底层的集合是一个粒子 y_i 迭代寻优过程中产生的局部最优解 p_i;合并局部最优解,得到所有粒子 $\boldsymbol{y} = \{y_i\}$ 的全局最优解集合 $f_t = \boldsymbol{p}_g$;对于不同的方案 $\boldsymbol{x} = \{x_t\}$,所有 f_t 将共同组成最终的解集合 $\boldsymbol{f} = \{f_t\}$。

3.2　服务器分配方案

本文采用服务器分配方案跳转概率 P_x 来终止 3.1 节所述的频率分配寻优。

本文使用分配方案的存活周期为自变量,对跳转概率 P_x 赋值。直观而言,如果对一个服务器分配方案 x_t 已经进行了多次粒子群迭代寻优,则可以认为此时的结果已经近似达到该方案下的最优解,从而停止计算当前的 x_t,转而生成新的分配方案 x_{t+1},重新进行寻优。出于以上的考虑,令

$$P_x = I/N_t \tag{9}$$

式中,I 为粒子群迭代次数;N_t 为指定的参数。在 3.1 节所述粒子群寻优时,每次迭代初始都产生一个[0,1]内均匀分布的随机数,与 P_x 进行比较,如果小于 P_x,则发生跳转。

实际中,服务器的个数一般在 $10^2 \sim 10^3$ 间,本文算法随机抽取 N_x 个分配方案进行寻优。遍历所有 N_x 个服务器方案后,得到 1 组帕累托最优解集合 $\boldsymbol{f} = \{f_t\}$。

根据决策者不同的需要,选择最终的惟一解的方法也有所不同。本文采用类似文献[5]的方法,计算每个解到理想最优点的距离,以距离最小原则选取最终解,如图 1 所示。图 1 中,最终帕累托最优解集合存在 5 个解,$f_1, f_2, \cdots, f_5$。解 f_1 的动态能耗最小,为 $g_{1\min}$,解 f_5 的服务器空闲能量最小,为 $g_{2\min}$。于是,理想最优点为 P 点,对应坐标($g_{1\min}, g_{2\min}$)。计算每个最优解到 P 点的距离,按照距离最小选取最终的解。图 1 中,f_2 为选取的最优解。

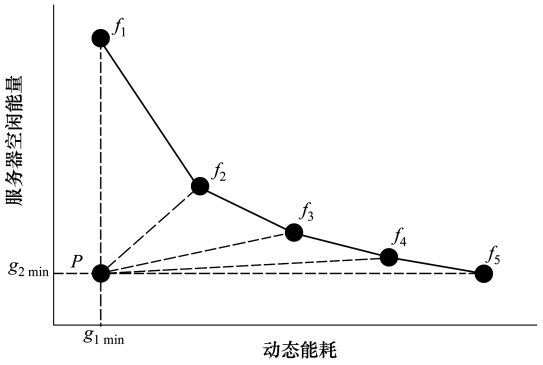


图 1　距离评价方法

4　实验分析

本文采用 Matlab 模拟了云计算下服务器和虚拟主机。程序运行于 1 台 PC 机,内存 1.75 GB,CPU 为 3.0 GHz。

4.1　参数设置

实验中,本文所提算法使用了如下的参数。粒子群中,惯性权重 $\omega = 1$,学习因子 $c_1 = c_2 = 1.5$;采用 20 个粒子进行迭代寻优。每个粒子最多保留 $N_l = 5$ 个局部帕累托最优,而所有粒子共保留 $N_g = 20$ 个全局帕累托最优。用于生成跳转概率的 N_t 是关键参数,较小的 N_t 将使算法更快结束,而较大的 N_t 则增加了每个方案的迭代寻优次数,从而提高得到该方案下最优解的概率。文中取 $N_t = 20$。显然,当迭代次数 $n = 20$ 时,按照(9)式所示,算法必然发生跳转,因此,N_t 还起到了传统粒子群算法中迭代次数最大值的作用。

服务器上每个虚拟主机允许占用的最低频率统一为 $F_j^{\min} = 1$ GHz,最高频率为 $F_j^{\max} = 2$ GHz。每个服务器最多可以提供 $\max F_j = 8$ GHz。因此,每个服务器可供 4 ~ 8 个虚拟主机使用。

4.2　不同服务器资源下对比实验

通过调整服务器数量 m 和虚拟主机数量 n,在不同的任务类型下进行了实验。对比算法采用工业

界普遍使用的轮询调度方法。

首先验证了在 $m = 5$ 个服务器和 $n = 10$ 个虚拟主机环境下算法的性能。图 2 显示了所提算法的调度结果。其中,图 2a)显示了某个帕累托局部最优解集合,而图 2b)展示了最终的所有帕累托全局最优解。按照 3.2 节所述距离评价方法,算法最终选择图 2b)中左下点对应的决策变量作为最终解,其动态功耗为 18.6 W,服务器空闲能量为 1.157 W。在本实验中,每个粒子群平均存活了 5.4 次迭代。算法的实时性基本满足实际要求。采用轮询算法,动态功耗为 80 W,服务器空闲能量为 20 W。

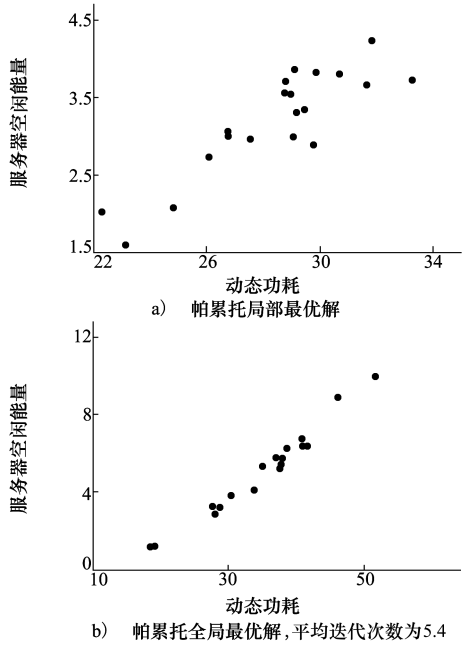


图 2 少量服务器/虚拟主机资源调度结果

在中量负载情况下,取 $m = 100$ 个服务器,$n = 300$ 个虚拟主机,所提算法得到如图 3 所示结果。此时,动态功耗为 1 029 W,服务器空闲能量为 566.3 W。采用轮询算法,动态功耗为 2 400 W,服务器空闲能量为 1 600 W。在大量负载情况下,取 $m = 1\,000$ 个服务器,$n = 2\,500$ 个虚拟主机,所提算法得到如图 4 所示结果。此时,动态功耗为 9 038 W,服务器空闲能量为 3 657 W。采用轮询算法,动态功耗为 20 000 W,服务器空闲能量为 10 000 W。

了验证所提算法对变化的服务其数量及虚拟主机数量的鲁棒性,取 $m = \{2^1, 2^2, \cdots, 2^{10}\}$ 个服务器,对应虚拟主机数量 n 为 m 的 $2^{1+(t-1)/10}$ 倍,$t = 1, 2, \cdots, 10$ 随序号依次增多。例如,当 $m = 2^{10}$ 时,$n = 3\,821$。由于在 4.1 节参数设置中已经指明,1 台服务器最多可以容纳 4 台虚拟主机,此时 $n/m = 3.73$ 已经接近服务器饱和状态。图 5 显示了对比结果。图 5a)为动态能耗对比图,图 5b)为服务器空闲能量对比图。出于可视化考虑,采用了 log 坐标系。

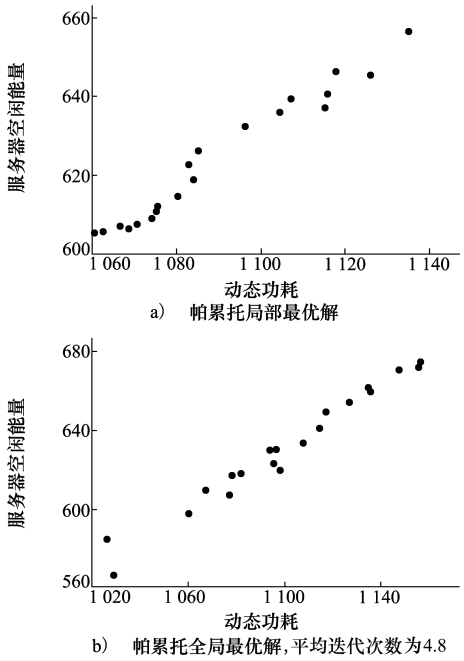


图 3 中量服务器/虚拟主机资源调度结果

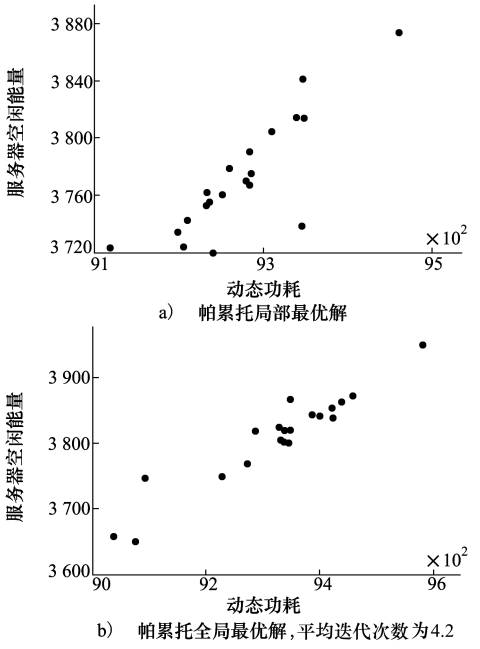
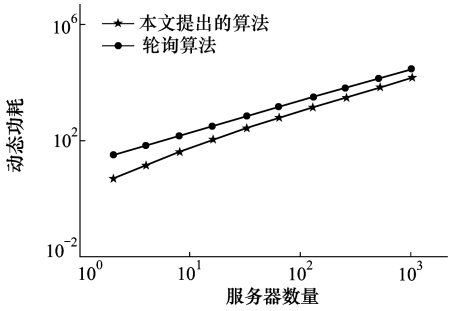
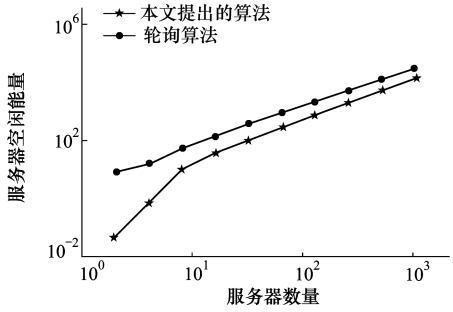


图 4 大量服务器/虚拟主机资源调度结果



a)　不同数量服务器下动态功耗对比图



b)　不同数量服务器下服务器空闲能量对比图

图 5　不同数量服务器下性能对比图

由图 5 可见,所提算法比轮询调度算法在动态能耗和服务器空闲能量 2 个指标上,均取得更优的结果。在全部 10 组实验上,所提算法的动态能耗均小于轮询调度的 50%。

4.3　能耗优先对比实验

对比算法中,除了 4.2 节使用的轮询调度外,还采用了文献[6]中优先受限的(precedence-constrained)并行应用调度算法,以及文献[7]采用的状态转移算法。

设定服务器数量 $m = \{80, 100, 120\}$,对应虚拟主机数量 $n = \{1,2,3\} \times m$。为了直观对比各算法的能耗,将轮询调度的能耗指定为 x,而其余算法的能耗相应换算为轮询调度的占比[8-9]。

表 1 展示了不同算法在 $n/m = 1$ 时的实验结果。表 2 对应 $n/m = 2$ 的结果;表 3 为 $n/m = 3$ 的结果。各表中,黑体标出了能耗最低的算法。

表 1　不同算法能耗对比表,$n/m=1$　　(%)

服务器数量	轮询调度	precedence-constrained method	状态转移	所提算法
$m=80$	100	51.88	55.46	**50.56**
$m=100$	100	**48.15**	53.58	48.88
$m=120$	100	44.54	51.22	**40.38**

表 2　不同算法能耗对比表,$n/m=2$　　(%)

服务器数量	轮询调度	precedence-constrained method	状态转移	所提算法
$m=80$	100	45.76	48.66	**43.78**
$m=100$	100	42.67	47.87	**41.78**
$m=120$	100	40.12	45.33	**40.02**

表 3　不同算法能耗对比表,$n/m=3$　　(%)

服务器数量	轮询调度	precedence-constrained method	状态转移	所提算法
$m=80$	100	**42.26**	47.24	42.80
$m=100$	100	40.38	45.66	**39.45**
$m=120$	100	38.28	43.45	**37.98**

从表 1~表 3 可见,轮询调度的耗能最多,且普遍为智能算法能耗的 2 倍以上。在 3 种能耗相关的智能算法中,状态转移算法的表现一般,在不同服务器/虚拟机比例下,均没有获得最少的能耗。这是由于状态转移受限于其固有的模型假设,而实际中,转移阵实时、动态的变化,往往给该算法带来极强的扰动因素,造成算法性能下降。所提算法在 7 组实验中取得最低的能耗。

5　结　论

针对云计算中能耗过高的问题,提出一种基于粒子群优化的资源调度算法。首先给出了能耗模型,以虚拟主机在服务器上的分配方案和占用的频率为决策指标,进行优化。指出了该模型为多目标优化问题,从而存在多个最优解,即帕累托最优解集合。采用粒子群优化算法对于每个分配方案进行迭代寻优,按照从属关系逐次构造了单粒子帕累托局部最优解集合、粒子群帕累托全局最优解集合和最终的跨分配方案的帕累托最优解集合共 3 个集合。在最终的集合上,采用距离评价标准,自主选择惟一的解。在不同数量的服务器和虚拟主机环境下进行了仿真实验。实验结果表明所提算法所需能耗低于轮询算法的 50%。

参考文献:

[1] Ludwig Siegele. Let it Rise: A Special Report on Corporate IT[M]. Economist Newspaper, 2008

[2] Rangan Kash, Cooke A, Post J, et al. The Cloud Wars: $ 100+billion at stake[R]. Tech Rep, Merrill Lynch, 2008

[3] James Kennedy. Particle Swarm Optimization[M]. Encyclopedia of Machine Learning, Springer, 2010

[4] Abido M A. Multiobjective Particle Swarm Optimization for Environmental/Economic Dispatch Problem[J]. Electric Power Systems Research, 2009, 79(7): 1105-1113

[5] 刘静,罗先觉. 采用多目标随机黑洞粒子群优化算法的环境经济发电调度[J]. 中国电机工程学报, 2010, 30(34): 105-111

Liu Jing, Luo Xianjue. Environmental Economic Dispatching Adopting Multi-Objective Random Black-Hole Particle Swarm Optimization Algorithm[J]. Proceedings of the CSEE, 2010, 30(34): 105-111 (in Chinese)

[6] Mezmaz M, Melab N, Kessaci Y, et al. A Parallel Bi-Objective Hybrid Metaheuristic for Energy-Aware Scheduling for Cloud Computing Systems[J]. Journal of Parallel and Distributed Computing, 2011, 71(11):1497-1508

[7] Lee Young Choon, Zomaya A Y. A Novel State Transition Method for Metaheuristic-Based Scheduling in Heterogeneous Computing Systems[J]. IEEE Trans on Parallel and Distributed Systems, 2008, 19(9): 1215-1223

[8] Li Y, Chen M, Dai W, Qiu M. Energy Optimization with Dynamic Task Scheduling Mobile Cloud Computing[J]. IEEE Systems Journal, 2017, 11(1): 96-105

[9] Rimal B P, Maier M. Workflow Scheduling in Multi-Tenant Cloud Computing Environments[J]. IEEE Trans on Parallel and Distributed Systems, 2017, 28(1): 290-304

Low-Energy-Orientated Resource Scheduling in Cloud Computing by Particle Swarm Optimization

Jia Jia, Mu Dejun

(School of Automation, Northwestern Polytechnical University, Xi'an 710072, China)

Abstract: In order to reduce the energy cost in cloud computing, this paper represents a novel energy-orientated resource scheduling method based on particle swarm optimization. The energy cost model in cloud computing environment is studied first. The optimization of energy cost is then considered as a multiobjective optimization problem, which generates the Pareto optimization set. To solve this multiobjective optimization problem, the particle swarm optimization is involved. The states of one particle consist of both the allocation plan for servers and the frequency plans on servers. Each particle in this algorithm obtains its Pareto local optimization. After the assembly of local optimizations, the algorithm generates the Pareto global optimization for one server plan. The final solution to our problem is the optimal one among all server plans. Experimental results show the good performance of the proposed method. Comparing with the widely-used Round robin scheduling method, the proposed method requires only 45.5% dynamic energy cost.

Keywords: cloud computing; cost function; Pareto optimality; resource scheduling; particle swarm; scheduling alogorithms