

一种面向绿色云计算的任务调度算法

秦 军¹, 孙 蒙², 冯亮亮²

(1. 南京邮电大学 教育科学与技术学院, 江苏 南京 210003;
2. 南京邮电大学 计算机学院, 江苏 南京 210003)

摘 要:任务调度时的服务器能耗是云计算系统动态能耗的重要组成部分。目前云计算带来的巨大能耗已经成为制约云计算发展的技术瓶颈,因此节约能源和提高能源利用率是实现绿色云计算系统的重要基础。为实现减少能耗和缩短任务执行时间的绿色云计算目标,将遗传算法和蚁群算法相结合,提出了一种动态融合的任务调度算法。该算法利用遗传算法全局搜索查找能力强的优点寻找任务调度的较优解,并将该较优解转化为蚁群的初始信息素值,再通过蚁群算法的蚁群信息交流和正反馈机制寻找任务调度问题的最优解,以有效降低云计算数据中心和计算中心的能耗。仿真实验结果表明,所提出的任务调度算法显著降低了云计算系统计算的运行时间和总能耗。

关键词:绿色云计算;节能;任务调度;GCA

中图分类号:TP301.6

文献标识码:A

文章编号:1673-629X(2017)08-0092-05

doi:10.3969/j.issn.1673-629X.2017.08.019

A Task Scheduling Algorithm for Green Cloud Computing

QIN Jun¹, SUN Meng², FENG Liang-liang²

(1. College of Education Science & Technology, Nanjing University of Posts and
Telecommunications, Nanjing 210003, China;

2. College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: The energy generated by the server during the scheduling system is an important part of the dynamic energy consumption of the cloud computing system and the huge energy consumption of the cloud computing has become the technical bottleneck which restricts the development of cloud computing. Therefore, saving energy and improving energy efficiency is an important foundation to achieve green cloud computing system. To achieve the goal of reducing energy consumption and shortening the task execution time of the green cloud computing, an energy-efficient scheduling algorithm based on genetic algorithm and ant colony algorithm has been proposed, which takes advantage of the strong global search ability of genetic algorithm to find the optimal solution of the scheduling problem, then converts it to the initial pheromone of ant colony optimization algorithm. After information communication and positive feedback, the global optimal solution of the task scheduling problem has been found out to effectively reduce the energy consumption in cloud computing center and calculating center. Simulation results show that the proposed algorithm has significantly reduced the task execution time and the total energy consumption.

Key words: green cloud computing; energy saving; task scheduling; GCA

0 引 言

当今,人们的环保意识达到了空前高度,新能源、减排、绿色经济这些话题的热度越来越高。2007年,绿色网格组织(Green Grid)^[1]成立,目标是要降低数据中心和商业计算系统的能耗。此外,标准性能评估组织(Standard Performance Evaluation Corporation,

SPEC)、事务处理性能委员会(The Transaction Processing Performance Council, TPC)等国际性能评估标准化组织也都在致力于能耗标准化评价和优化问题。随着云计算规模越来越大,它对能源与环境的影响已越来越突出,其本身的能耗越来越不可忽视,能耗问题已成为云计算发展道路上必须要跨过的障碍^[2]。托尼·马

收稿日期:2016-07-20

修回日期:2016-10-27

网络出版时间:2017-06-05

基金项目:江苏省自然科学基金项目(BK20130882)

作者简介:秦 军(1955-),女,教授,研究方向为计算机网络技术、多媒体技术、数据库技术;孙 蒙(1990-),女,硕士研究生,研究方向为分布式计算机技术与应用。

网络出版地址: <http://cnki.net/kcms/detail/61.1450.TP.20170605.1505.002.html>

斯泰利奇在《云计算 节能之路》中提到,目前云计算的耗电量已经超过全球总耗电量的1%。云计算服务商建设数据中心,通常,一个占地500平方米的数据中心每天消耗的电量就高达38 000度。在2014年,只有8.5%的数据中心负责人预计在2015年后数据中心的容量仍然够用,到2020年,预计数据中心的建设规模几乎将是目前的两倍,使得云计算的能耗、对环境的影响等问题更为突出。

从上述数字可以看出,为云计算设计高能效的解决方案已经迫在眉睫。绿色云计算主要考虑影响生态环境的相关影响因素,近年来绿色云计算已得到研究者的众多关注。文献[3]提出了云计算技术中的绿色节能模式。文献[4]提出了如何利用虚拟化来降低能耗,通过虚拟化实现绿色云计算。文献[5]对STF-OS、LTF-OS和RT-OS三种绿色任务调度算法进行了相关的理论分析,并验证了其能有效地减少能源消耗。

在云环境下,任务调度属于NP完全问题,启发式智能算法普遍被认为是解决NP完全问题的较优算法,可以有效得到最优解,其搜索过程较复杂,但是有良好的寻优性能,相比传统算法有着不可比拟的优越性。启发式智能算法已被一些学者运用到解决云计算任务调度问题,目前应用比较成熟的有基于遗传算法(Genetic Algorithm,GA)、蚁群算法(Ant Colony Optimization,ACO)、粒子群算法(Particle Swarm Optimization,PSO)等资源调度策略。文献[6]提出了一种基于GA的云计算任务调度策略,以减少任务的最大完成时间。文献[7]提出了一种基于ACO的云计算任务调度算法,该算法综合考虑了计算节点的性能属性和该节点在负载压力下的任务响应能力。文献[8]针对云计算服务集群资源调度和负载平衡的优化问题,提出一种基于改进的粒子群优化算法的云计算资源调度策略。

然而每一种算法都有自身的局限性,GA在查找最优解的过程中,虽然具有较强的全局搜索能力,但对解空间的局部搜索能力较弱,容易陷入局部最优;且算法的变异操作基本是随机性的,同时在子代个体的进化过程中,缺乏对求解问题的启发信息的利用,因此算法的进化率较低,后期收敛速度较慢。而ACO作为分布式启发搜索算法,能够利用蚁群特有的信息素正反馈机制,通过蚁群的整体协作,找到问题的最优解;但ACO在寻优的初期由于缺乏信息素,蚁群中蚂蚁的路径搜索会有很大的随机性,导致算法的效率较低。粒子群算法搜索速度快,效率高,但容易陷入局部最优。近年来许多学者根据启发式智能算法加以改进,在解决云计算任务调度问题时,取得了较好的效果。

为此,结合GA的全局寻优能力和ACO的局部寻优能力,提出了一种基于GA和ACO的云计算任务调度算法(Genetic and ant Colony Algorithm,GCA),以加快收敛速度,缩短任务执行时间,有效提高任务调度效率,从而降低云计算平台中数据中心和计算中心的能源消耗,实现绿色云计算。

1 算法实现

该算法的基本思想是结合GA的全局寻优能力和ACO的局部寻优能力,利用GA较强的全局搜索能力,快速生成云任务调度问题的较优解,并作为ACO的初始化信息素分布,在算法后期主要是利用ACO的局部搜索能力,正反馈、分布式、求解效率高特性。

1.1 遗传算法的相关设定

(1) 染色体编码和初始种群的生成。

GA是从问题的解空间开始的,在解决实际问题时,种群中的每个个体都使用染色体进行编码,也就是先需要将所有可能存在的任务调度方案进行染色体编码,用来表示个体和问题的解之间的映射关系,即一条染色体就表示该算法的一个可行解,算法目标就是找到可行解中的最优解,来实现任务与资源的匹配。染色体的编码方式有多种,如直接编码、间接编码和混合编码^[9-10]。

在解决云计算调度问题时,根据任务调度的特点,可以直接对任务的执行状态编码,也可以采用间接编码。为此,结合云计算任务调度特点,采用节点任务的间接编码方式,即对每个任务分配到的计算节点进行编码。染色体的长度 n 为正在进行调度的任务数量。例如染色体 $\{f_1, f_2, \dots, f_n\}$,其中 f_i 表示第 i 个任务分配到的节点。

染色体编码的操作步骤如下:

Step1:创建一个数组 `int chromosome[n]`,存储编码后的染色体;

Step2:for($i = 0; i < n; i++$),循环体,判断任务数;

Step3:for($j = 0; j < n; j++$),循环体,判断计算节点数;

Step4:if($G[i][j] = 1$),判断矩阵 G 中的元素是否为1,如果为1,执行Step5;

Step5: `chromosome[i] = j`,将任务 i 分配给计算节点 j ;

Step6: return `chromosome[n]`,返回编码后的染色体。

如有任务数 $n = 7$,计算节点数 $m = 4$ 的云计算任务调度,现随机生成一个任务分配矩阵 G ,如下:

$$G = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad (1)$$

采用上述编码算法进行染色体编码,获得一条染色体:Chromosome[7] = (4,2,1,3,1,4,2),即 7 个任务对应的计算节点号分别为 4,2,1,3,1,4,2。

(2) 适应度函数。

GA 在进化搜索的过程中,不善于利用外部启发信息,仅仅以适应度函数为依据。GA 中的适应度函数是衡量 GA 寻找最优解的约束条件,适应度函数的设计决定着遗传算法的搜索性能。

融合算法的任务调度目标是降低能耗,即考虑任务的执行时间和执行费用。为了在缩短任务完成时间的同时降低计算节点的能源消耗,因此,将第 k 个染色体的适应度函数定义为:

$$f(k) = \sum_{j=1}^m F(k, v_j) \quad (2)$$

(3) 遗传操作。

遗传操作主要包括选择操作、交叉和变异操作。

选择操作:目的是为了使优良的染色体个体以较高的概率遗传到下一代,体现了自然世界中适者生存的进化理论。它是 GA 中一个很重要的步骤,通过对个体的适应性评价,计算种群中每个个体的选择概率,如式(3):

$$P(k) = \frac{f(k)}{\sum_{j=1}^q f(j)} \quad (3)$$

其中, $f(k)$ 为个体 k 的适应度值; q 为种群数量。

交叉和变异操作:在 GA 的迭代过程中,交叉操作负责染色体对之间的基因交叉或基因重组,具体操作是从两个父代染色体中选取部分基因片段,将这部分基因片段进行替换重组,生成一对新的子代染色体,模拟了自然界中有性繁殖的基因重组现象。将父代的优良基因遗传到下一代个体,交叉操作决定了全局搜索能力。变异操作仅对种群个体中少量染色体进行操作,拓展新的搜索空间,决定了局部搜索能力。变异操作保持了种群的个体多样性,防止算法陷入局部收敛。交叉和变异操作都是采用文献[6]中提出的自适应方式。

交叉概率函数为:

$$P_c = \begin{cases} k_1(f_{\max} - f') / (f_{\max} - f_{\text{average}}), & f' \geq f_{\text{average}} \\ k_2 f_{\text{average}}, & f' < f_{\text{average}} \end{cases} \quad (4)$$

变异概率函数为:

$$P_m = \begin{cases} k_3(f_{\max} - f) / (f_{\max} - f_{\text{average}}), & f \geq f_{\text{average}} \\ k_4, & f < f_{\text{average}} \end{cases} \quad (5)$$

其中,取 $k_1 = 0.4$, $k_2 = 0.06$, $k_3 = 1$, $k_4 = 0.2$; f 为变异个体的适应度值; f_{average} 为种群平均适应度值; f_{\max} 为种群中个体的最大适应值; f' 为交叉个体中较大的适应度值。

GA 在进化后期求解效率不高,并且容易陷入局部最优,因此选择合适的时机进行 GA 到 ACO 的过渡,可以解决此问题。

1.2 加入蚁群优化算法的 GA

执行上述 GA,最终可以获得问题的最优解或近似最优解。但是一般情况下,需要根据不同的问题迭代几百次甚至上千次。为了减少算法的执行时间,加快算法的收敛速度,采取 GA 与 ACO 相融合的方式。GA 与 ACO 的进化率伴随时间变化趋势见图 1。

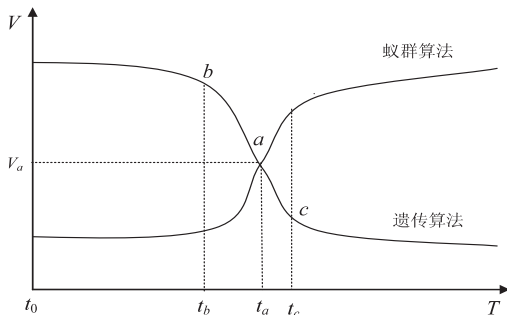


图 1 遗传算法与 ACO 的进化率-时间图

从图 1 可看出,在初始时间段 $t_0 \sim t_a$, GA 保持较高的进化率,在 t_b 之后,由于不能有效利用启发信息,个体进化率开始大幅下降;相反,ACO 在时间段 $t_0 \sim t_a$ 由于信息素的缺乏,保持较低的进化率,在 t_b 之后,随着信息素浓度的提高,进化率开始大幅提升。

因此,在 t_a 点将 GA 切换到 ACO 是最优方案,能够显著提高算法性能。具体操作方法:在 $t_0 \sim t_a$,利用 GA 对任务调度问题进行求解,选取种群中适应度值高的个体组成优化解作为 ACO 的初始信息素^[11],解决 ACO 对初始信息素的依赖问题,然后利用 ACO 的正反馈机制对较优解做进一步优化,最后搜索找到调度问题的最优解。

(1) 信息素初始化。

GA 迭代结束时,以一定概率选取任务调度问题较优解转化为 ACO 的初始信息素。这里将 GA 求得的最优解的前 10% 个体作为遗传优化算法的解合集 T ,各计算节点的初始信息素值为 τ_j :

$$\tau_j(0) = \tau_{c_j} - \tau_{T_j}(0) \quad (6)$$

其中, τ_{c_j} 表示计算节点 j 固有的最大处理能力; $\tau_{T_j}(0)$ 表示根据 GA 获得的最优解转换成的信息素

值,反映了节点 j 的能耗状态。

(2) 转移概率。

信息素初始化后,执行ACO,蚂蚁根据GA转化来的信息素分布开始搜索,以一定概率 p 进行下一跳路径选择(即在任务调度时将任务分配到下一个资源节点)。 t 时刻任务 i 调度到计算节点 j 的概率为:

$$P_{v_j}(t) = \frac{[\tau(t, v_j)]^\alpha \times [\eta(v_j)]^\beta}{\sum_{j=1}^q [\tau(t, v_j)]^\alpha \times [\eta(v_j)]^\beta}$$

(7)

其中, $\tau(t, v_j)$ 表示时刻 t 计算节点 v_j 上关于待执行任务 i 的信息素浓度值; $\eta(v_j)$ 表示计算节点 v_j 的处理能力; α 表示信息素的重要程度; β 表示计算节点处理能力的重要程度。

(3) 信息素更新。

为了防止陷入局部最优,需要对蚂蚁所选的线路进行局部信息素更新,以保证每轮搜索中的最优方案都在信息素的分布中有所体现,这种反馈方式可以加快ACO的收敛速度^[12-13]。随着任务的执行,将计算节点的信息素浓度根据任务的执行情况进行更新,见式(8):

$$\tau(t_1, v_j) = \rho \tau(t, v_j) + \Delta \tau(t, v_j)$$

(8)

其中, $\rho \in [0, 1]$ 表示信息素的残留系数;

$\Delta \tau(t, v_j) = \sum_{i=1}^m \Delta \tau_i(v_j)$ 表示第 i 只蚂蚁在时间 $[t, t_1]$ 在节点 v_j 上留下的信息素浓度,由式(9)进行计算:

$$\Delta \tau_i(v_j) = \frac{S}{L_j}$$

(9)

其中, S 为常数; L_j 表示计算节点 j 的实时负载。

ACO的终止条件是设置最大搜索步数或当进化进入停滞时算法结束。另外,在GCA算法中,设定了禁忌表,已搜索过的节点加入禁忌表,可以防止搜索陷入局部收敛。

1.3 遗传-ACO的任务调度

GCA的任务调度步骤如下:

Step1:根据用户提交的作业,进行初始值设置,根据作业的一些特征设置GA的相关参数:种群大小、交叉概率、变异概率、最小迭代次数和最大迭代次数;

Step2:根据待调度任务的规模进行染色体间接编码,定义适应度函数,随机生成初始种群;

Step3:对种群个体进行解码,然后准备进行遗传性操作;

Step4:计算个体的适应度值,根据个体适应度值进行选择操作,选择适应度值高的个体进入下一步遗传操作;

Step5:计算交叉概率 P_c 、变异概率 P_m ,对选出的较优个体进行交叉、变异操作,经过选择、交叉、变异操作后产生下一代群体;

Step6:比较新个体与父代个体,根据替换的原则进行优劣替换,选择优良的个体作为最终子代个体;

Step7:根据GA的终止条件进行判定,若满足则执行Step8,否则跳转至Step4;

Step8:将GA得到的拥有最大适应度染色体作为最优解集转化为ACO的初始信息素;

Step9:将GA得到的最优解的前10%转换成ACO的初始信息素,并将蚂蚁随机分布于待分配资源节点上,进入ACO;

Step11:蚁群中的蚂蚁根据式(7)选择下一个任务的可分配节点,同时将当前节点加入禁忌表;

Step12:更新被选择节点上的信息素;

Step13:判断ACO是否满足终止条件,若满足,输出任务调度方案结果,否则跳转步骤Step11。

2 仿真结果

采用云计算仿真软件CloudSim进行仿真实验,用于模拟云环境下的任务调度和资源分配。

通过将提出算法与GA、ACO进行比较,验证该算法的性能。初始参数的设置见表1。

表1 算法的主要参数

算法	参数	数值
GA	种群数量	100
	交叉概率	0.8
	变异概率	0.2
	蚂蚁数量	10
	迭代次数	200
ACO	传递到ACO的比例	10%
	α	3
	β	1
	ρ	0.6
	S	1 000

在模拟云计算实验中,为了检验GCA能耗优化的性能,从算法的任务调度时间和能耗方面与GA、ACO进行数据对比分析。采取执行任务数量200个,应用三种算法分别进行10次调度,然后将10次结果取平均值,采用任务数量为30、50、70、90、110、130、150、170、190的平均数据,然后进行数据对比分析。GCA的参数和表1保持一致。

三种算法不同任务数执行时间对比见图2。

从图2可以看出,在仿真初期,任务量较少,数目为30~90的范围内各算法的执行时间相差不大。随着任务数量的增加,任务数大于90的时候,GCA的调度优势明显体现出来。这是因为在算法后期转化为ACO,加快了算法的收敛速度,GCA的执行时间明显少于GA和ACO。仿真结果证明了GCA可以有效减少调度任务的执行时间。

不同任务数能源消耗对比见图 3。

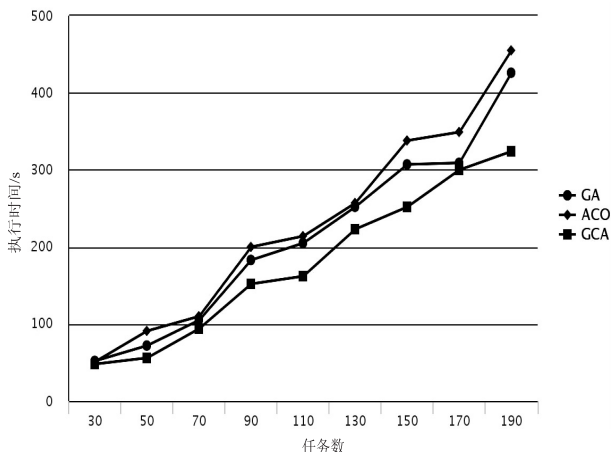


图 2 不同任务数执行时间对比

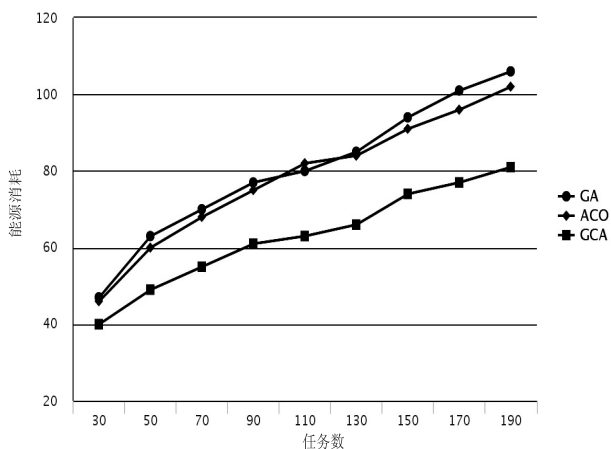


图 3 不同任务数的能源消耗对比

从图 3 可以看出,在相同任务数的情况下,GCA 的能源消耗远低于 GA 和 ACO,在任务数目少于 70 时,GCA 在能耗方面的优势表现不明显,随着任务数增加,可以明显看到能耗优化的效果。这是因为算法后期加入的 ACO 进行路径更新,避免了任务调度集中于不理想的资源上,增加了能耗。

3 结束语

为了降低云计算系统的能耗,实现能耗优化的绿色云计算,提出了一种基于 GA 和 ACO 动态融合的任务调度算法。该算法利用遗传算法全局搜索能力和蚁群算法的信息交流和正反馈机制寻找任务调度问题的

最优解。将云计算运行时间和总能耗作为评判标准,将该算法与 GA 和 ACO 进行了对比仿真分析。结果表明,该算法能够实现云计算总能耗的优化控制。

参考文献:

- [1] The Green Grid Consortium[EB/OL]. [2011-08-01]. <http://www.thegreengrid.org>.
- [2] Wang L, von Laszewski G, Dayal J, et al. Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with DVFS[C]//10th IEEE/ACM international conference on cluster, cloud and grid computing. [s. l.]:IEEE, 2010:368-377.
- [3] Kaur M, Singh P. Energy efficient green cloud; underlying structure[C]//2013 international conference on energy efficient technologies for sustainability. [s. l.]:[s. n.], 2013: 207-212.
- [4] 赵肄江,胡蓉.基于虚拟化的绿色云计算[J].湖南科技大学学报:自然科学版,2010,25(4):86-89.
- [5] 王永贵,张伟,韩瑞莲.云环境下绿色任务调度策略[J].计算机工程与应用,2012,48(34):81-87.
- [6] 李建峰,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.
- [7] 华夏渝,郑骏,胡文心.基于云计算环境的蚁群优化计算资源分配算法[J].华东师范大学学报:自然科学版,2010(1):127-134.
- [8] 刘万军,张孟华,郭文越.基于 MPSO 算法的云计算资源调度策略[J].计算机工程,2011,37(11):43-44.
- [9] Madivi R, Kamath S. An hybrid bio-inspired task scheduling algorithm in cloud environment[C]//International conference on computing, communication and networking technologies. [s. l.]:IEEE, 2014:1-7.
- [10] 王小平,曹立明.遗传算法:理论、应用及软件实现[M].西安:西安交通大学出版社,2002:34-70.
- [11] 何雪海,胡小兵,赵吉东,等.基于自适应转移概率的蚁群优化算法[J].计算机工程,2010,36(23):165-167.
- [12] 段海滨.蚁群算法原理及其应用[M].北京:科学出版社,2005.
- [13] 刘永,王新华,邢长明,等.云计算环境下基于蚁群优化算法的资源调度策略[J].计算机技术与发展,2011,21(9):19-23.