

基于变异算子的云计算任务调度算法

陈超^a, 蔡乐才^b, 高祥^c

(四川理工学院 a. 自动化与电子信息学院; b. 计算机学院; c. 机械工程学院, 四川 自贡 643000)

摘要:为了高效调度云计算中海量的任务,提出一种改进遗传算法(IGA),将变异操作分为两种:变异操作 a 和变异操作 b 。变异操作 a 为随机位置的基因值变异,而变异操作 b 则是先找出满足一定条件的基因位置,再将该位置的基因值变异成目标基因值,使得每次变异后的染色体都优于变异前的染色体。在算法的前期使用变异操作 a ,在算法后期即将收敛于最优解时,采用变异操作 b 以加快收敛的速度。为了避免改进变异操作使算法陷入局部解,在种群初始化时,采用染色体匹配率的方式选择初始化种群,使其均匀的分布在解空间上。实验仿真结果表明,改进算法不但使最终完成时间更短,收敛效率更高,而且可以在一定程度上均衡负载,能更有效地实现任务调度。

关键词:云计算;任务调度;遗传算法;匹配率;变异

中图分类号:TP393

文献标志码:A

引言

云计算作为一种全新的超级计算模式,其目的是通过网络将大量分散的资源集中于互联网的数据中心,并由数据中心提供计算、软件、数据访问及存储服务等。在云计算中,计算能力作为一种资源,通过互联网按需分配给用户^[1]。其中,用什么样的方法去分配计算资源就变得非常重要了,它决定着计算能力的好与坏。对于云计算服务提供商来说,其关键技术就是如何有效利用“云”中的资源,使大量任务进行合理高效的调度和分配,其分配效率直接影响到整个云计算环境的性能以及企业经济效益,因此找出更好的分配方法是非常有应用价值的。

目前,国内外涌现出了大量的云计算资源分配算法,但方法理论还不是十分成熟完善,缺乏一个统一的科学方法。一些简单的资源分配方法,如轮转法、哈希法、最小负载优先法等,其结果往往不够理想,并且很容

易造成物理服务器的服务性能不均衡等问题。这就使得越来越多的学者开始关注神经网络、禁忌搜索、蚁群算法^[2-4]、遗传算法等现代优化算法。而其中,将遗传算法作为云计算环境下的任务调度算法已渐渐成为研究的热点。一些改进的遗传算法被提出,如以任务总完成时间和平均时间为适应度的双适应度遗传算法(DF-GA)^[5]、动态调整任务分配的遗传算法^[6]、基于染色体编码方式和适应度函数的改进遗传算法^[7]、时间—成本约束的遗传算法(TCGA)^[8]等。现已提出的改进遗传算法中,收敛速度都还有待提高,同时,负载均衡也应该是考虑的重要因素。本文基于遗传变异算子思想,提出一种改进遗传算法(IGA),以能更好适应具有海量任务的云环境的任务调度。

1 遗传算法与任务调度

1.1 任务调度描述

根据调度层面的不同,云计算中的任务调度方法可

收稿日期:2013-12-06

基金项目:物联网技术与应用四川省青年科技创新团队项目(2011JTD0031);四川省教育厅重点科研项目(09Z087);四川理工学院研究生创新基金项目(B20306)

作者简介:陈超(1988-),女,四川什邡人,硕士生,主要从事物联网与云计算方面的研究,(E-mail)236537194@qq.com

以分为两大类:资源层面的调度和应用层面的调度。资源层面的调度问题实际上是云计算环境中应用的需求怎样得到满足的问题;应用层面的调度方法把底层看成许多计算节点,在应用层面对用户提交的作业进行分解,把分解后的子任务合理分配给不同的节点来实现。

本文中采用的遗传算法则是属于应用层面的调度方法。在 Map/Reduce 模型下,把任务分割成多个较小的子任务,然后分配给多个计算节点并行执行,如何给众多子任务合理分配资源是个复杂的问题。在云环境中,假设有 R 个资源: $\{VM_1, VM_2, \dots, VM_R\}$; 用户提交的任务为 M 个,表示为 $\{J_1, J_2, \dots, J_M\}$, MapReduce 模型^[9]将 M 个任务分割成 N 个子任务,表示为 $\{T_1, T_2, \dots, T_N\}$ 。则云计算环境下应用层面的调度问题^[10]即可描述为:在有限的 R 个资源情况下,高效合理的调度 N 个子任务,使任务总完成时间最小,并且在占用带宽、费用、可靠性等方面都能达到满意的效果。

1.2 遗传算法

遗传算法(Genetic Algorithm)是一种模拟生物进化论中自然选择过程的计算模型,是较经典的搜索最优解的方法。遗传算法中,首先会生成一组候选解,然后根据适应度函数计算每个候选解的适应度,并依据适应度的大小对群体进行选择操作,适应度大者生存,适应度小者淘汰。最后,对保留的个体进行交叉、变异操作,以产生出更优的个体。传统遗传算法具有全局解空间搜索和并行性两个显著优点,同时也存在早熟等现象。本文将结合云环境下任务调度的特点以及遗传算法自身的特点,提出一种改进遗传算法,作为云计算环境下的任务调度算法。改进的遗传算法在算法的前期与后期采用不同的变异操作,促进算法加速收敛。为了避免算法陷入局部解,在初始化阶段采用染色体匹配率^[8]来选择初始种群,使初始群体均匀遍布于整个解空间上。

2 采用改进遗传算法进行云任务调度

2.1 染色体编码与解码

遗传算法中常用的编码的方式有两种:实数制编码与二进制编码。为使算法在遗传操作中的计算更简单方便,本文中采用资源—任务直接编码方式,即染色体长度为任务总数,而其中每个基因的取值为该位置对应的任务分配到资源的资源编号。

2.2 产生初始群体

实际云计算中的任务数非常庞大,因此会存在许多

不同的解。当遗传算法中初始种群个体数远远小于解空间中解的数量时,即初始解不能均匀分布于解空间,算法很容易陷入局部解。因此本文在产生初始群体时将采用文献[8]提出的方法:计算染色体的匹配率,通过调节匹配率的大小来选择初始染色体,使得初始种群个体均匀地分布在解空间上。

若种群规模为 S ,子任务总数(染色体长度)为 L ,资源数为 W 。染色体 y_i 和染色体 y_j 的匹配率为

$$Matching(y_i, y_j) = Sumgene(y_i, y_j) / L \quad (1)$$

其中, $i \in (1, 2, \dots, S)$, $j \in (1, 2, \dots, S)$, 且 $i \neq j$; $Sumgene(y_i, y_j)$ 为 y_i 与 y_j 中相同等位基因的个数。

在选择初始化种群个体时,通过调节匹配率的大小选择,能够保证初始种群个体的多样性,使其均匀遍布在整个解空间上,促使算法收敛于全局解。该方法不仅克服了标准遗传算法易陷入局部解的缺点,并且可以避免本文中改进变异操作使算法陷入局部解,保证了改进变异操作的效率和正确性。

2.3 构造适应度函数

遗传算法遵循生物进化中优胜劣汰的原则,适应度大的个体最终被保留下来,适应度小的个体会因无法适应环境而被淘汰掉。遗传算法就是通过多次的优胜劣汰,最终找到最优解。因此适应度函数的选取非常重要,决定了算法的性能。

云计算环境下任务调度的最重要的目标是任务总完成时间。本文中任务总完成时间不仅是任务在计算资源上的执行时间,还包括了任务传输时间。任务执行时间主要取决于计算资源的计算能力,而传输时间主要取决于计算资源的带宽。因此,第 i 个任务被分配到第 j 个资源上的总完成时间可以用公式(2)来衡量:

$$T(i, j) = \frac{Inputfilesize_i + Outputsize_i}{bw_j} + \frac{Length_i}{E_j} \quad (2)$$

其中, $Inputfilesize_i$ 表示第任务 i 输入文件的大小, $Outputsize_i$ 表示任务 i 输出文件的大小。 $Length_i$ 表示任务 i 的长度。表示资源 j 的通讯带宽。表示资源 j 的计算能力,其计算公式为:

$$E_j = Num_j(Pe) \times Mips_j(Pe) \quad (3)$$

其中, $Num_j(Pe)$ 表示资源 j 处理器的数目, $Mips_j(Pe)$ 表示资源 j 处理器的平均速度。

假设分配到资源 j 上的任务数为 n , 则资源节点 j 完成所分配任务的时间为:

$$T_j = \sum_{i=1}^n T(i, j) \quad (4)$$

由于各个计算资源是并行处理任务序列的,所以任务的总完成时间为最后完成任务的资源节点所用的时间,即求的最大值。因此,基于任务总完成时间的适应度函数定义为:

$$f(x_k) = -\max_{j=1}^w(T_j) \quad (5)$$

其中,表示第 k 个染色体, $k \in (1, 2, \dots, S)$ 。

2.4 选择操作

选择的目的是为了保存优良基因,淘汰掉适应度低的个体。本文中通过计算适应度值比例来作为选择标准。对于给定规模为 S 的种群 (x_1, x_2, \dots, x_s) , 染色体 x_i 的适应度值为 $f(x_i)$, 则其入选概率为:

$$p(x_i) = \frac{f(x_i)}{\sum_{k=1}^s f(x_k)} \quad (6)$$

2.5 交叉操作

交叉操作是遗传算法中起核心作用的遗传算子,所谓交叉就是替换重组两个父代个体的部分结构,从而生成新个体的操作,即基因重组的过程。交叉操作是产生新个体最主要的方法,通过交叉可以将父代群体的优良基因遗传给下一代,并使新一代个体拥有更好的基因。交叉算子是决定遗传算法全局搜索能力的关键,通过交叉,遗传算法的搜索能力可以得到飞跃式的提高。交叉概率一般在 0.3~0.9 之间,本文将以一定概率 P_n ($0.3 < P_n < 0.9$) 进行交叉操作,从种群中选择两个个体,然后随机选择交叉点,交换交叉点后的基因。

2.6 变异操作

遗传算法中,变异算子的作用有两个。一是维持种群的多样性。交叉算子用以产生新个体,而变异算子则是产生新个体的辅助方法。通过变异可以拓宽解的搜索空间,促进算法在全局上搜索最优解,防止出现早熟现象或陷入局部解,这种情况下变异概率应该取较大值。二是增加算法的局部搜索能力。在算法后期,算法已接近最优解领域,此时通过交叉操作很难完成局部的细节搜索,而利用变异操作就可以从局部加速收敛于最优解。此时变异概率应取较小值,以防止最优解领域因变异而遭到破坏。

基于上述变异算子的特点,本文将变异操作归结为变异操作 a 和变异操作 b。

变异操作 a: 对要变异的个体随机选择一个变异点,随机选取一个值替代该位置上的基因值。

变异操作 b: 云计算环境下有着海量的数据,任务数非常巨大,远远大于资源数,因此,资源上的任务分部不

均。此变异操作就是在满足条件(7)下,将有最多任务的资源上的其中一个任务放到任务数最少的资源上去。

基于此,变异操作过程为:在算法的前期,以较大概率 P_{m1} 对群体执行变异操作 a,增加群体的多样性,使搜索遍布于整个解空间上。在算法的后期,以较小概率 P_{m2} 对群体执行变异操作 b。此时算法已接近最优解,但是由于个体的适应度值较为接近,以至于进化困难,收敛速度变慢,因此本文通过变异操作 b 来促进算法加速收敛于最优解。

例如染色体 $\{3, 1, 2, 3, 2, 4, 3, 1, 2, 2, 4, 2, 1, 3, 2\}$, 解码后如图 1 所示。

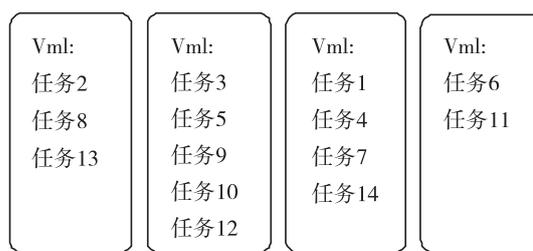


图 1 任务分配图

由图 1 可看出,任务数最多的是 $Vm2$, 任务数最少是 $Vm4$ 。 $Vm2$ 完成它所有任务的时间为 $T(Vm2)$, $Vm4$ 完成它所有任务的时间为 $T(Vm4)$, 对于 $Vm2$ 上的任务,如任务 3,若满足

$$T(Vm2) > T(Vm4) + T(3, 4) \quad (7)$$

其中 $T(3, 4)$ 是任务 3 在 $Vm4$ 上的完成时间,则将任务 3 移到 $Vm4$ 上,即染色体变异成 $\{3, 1, 4, 3, 2, 4, 3, 1, 2, 2, 4, 2, 1, 3, 2\}$ 。若 $Vm2$ 上的任务没有满足此条件的,就随机在 $Vm2$ 上选取一个任务移到 $Vm4$ 上。通过此变异操作可以使每次变异后的染色体比变异前的染色体更优,同时,还可以在一定程度上均衡负载。

3 实验仿真分析

采用云计算仿真平台 CloudSim^[11-12] 对算法进行仿真分析。CloudSim 是澳大利亚墨尔本大学网络实验室和 Gridbus 项目共同提出的一种云仿真软件,它是在离散事件模拟包 SimJava 上开发出来的函数库,其体系结构组件包括四个层次: SimJava、GridSim、CloudSim、User-Code。通过 CloudSim 仿真平台,在相同的环境条件下,对改进的遗传算法(IGA)与传统遗传算法(GA)进行对比实验。种群规模为 200,交叉概率为 0.25,变异操作 a 的变异概率取 0.1,变异操作 b 的变异概率取 0.05,染色体匹配率取 0.05。算法终止条件为:(1)如果连续 50 代适应度值不再变化;(2)达到最大迭代次数 $gnMax$ (这

里取 $gnMax = 200$)。

(1) 当资源数为 10 时,任务数分别取 20、40、80、150、300、600,对改进遗传算法和传统遗传算法进行对比仿真。记录每次任务总完成时间,统计结果见表 1。

表 1 任务总完成时间

算法	不同任务数 n 下的完成时间					
	20	40	80	150	300	600
GA	4.2	25.5	82.8	266	1126	6368
IGA	4.2	16.7	66.7	240	1070	6070

从表 1 可以看出,采用改进遗传算法作为任务调度算法时,任务的完成时间要短很多,优势较明显。

(2) 任务数为 2000,资源数为 20 时,改进遗传算法与传统算法的收敛情况如图 2 所示。

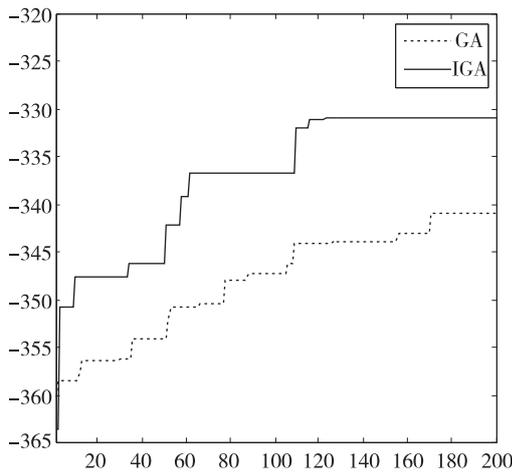


图 2 算法收敛结果比较

从图 2 可看出,在同样的条件下,改进遗传算法在迭代 120 次就已基本收敛,而标准遗传算法要迭代到 180 次左右才开始呈收敛趋势。这是因为改进遗传算法在迭代的后期采用了变异操作 b,加快了收敛速度,而标准遗传算法由于在迭代后期染色体适应度值较为接近,进化困难,导致收敛速度非常慢。同时还可以看出,改进遗传算法的最后任务完成时间比标准遗传算法更短,结果更为理想。

(3) 任务数为 2000,资源数为 3 时,并且调整资源的性能参数,使资源节点的处理能力有较大差异,这时资源节点 $Vm1$ 、 $Vm2$ 、 $Vm3$ 上的负载情况如图 3 所示。

从图 3 可以看到,当有大量任务而资源节点数有限并且资源节点运算能力差异较大时,改进遗传算法的负载较均衡,而原算法中资源节点分配到的任务数有较大差异,负载情况表现的不太理想。由此可看出,改进遗传算法的负载情况在一定程度上要优于标准遗传算法。

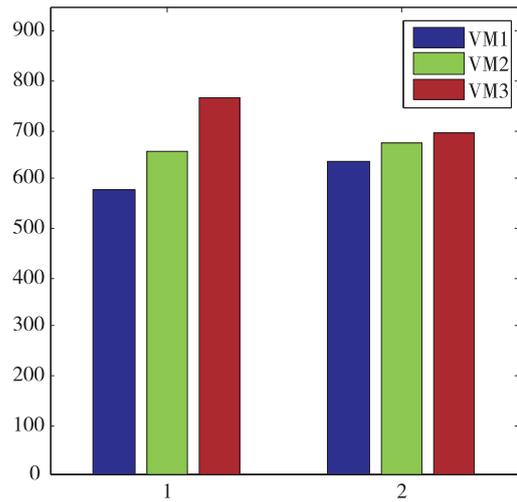


图 3 大量任务情况下资源的负载情况

4 结束语

提出了一种改进遗传算法作为云计算环境下的任务调度算法,通过两种变异操作促使算法加速收敛,同时,采用染色体匹配率来初始化种群,通过调节匹配率使初始群体均匀的分布在解空间上。最后,通过 Cloudsim 仿真分析表明,该改进算法不仅使得任务完成时间更短、收敛速度更快,而且在一定程度上改善了资源节点的负载情况,是一种云计算环境下有效的任务调度算法。

参考文献:

- [1] Foster I,Zhao Y,Raicu I,et al.Cloud computing and grid computing 360-degree compared[C]//Proceedings of the 2008 Grid Computing Environments Workshop. Washington,DC:IEEE Computer Society,2008:1-10.
- [2] 张春燕,刘清林,孟珂.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.
- [3] 刘永,王新华,邢长明,等.云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):19-27.
- [4] 范杰,彭舰,黎红友.基于蚁群算法的云计算需求弹性算法[J].计算机应用,2011,31(增刊 1):1-3.
- [5] 李剑锋,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] 王文枫,帅建梅.一种云计算环境下任务调度策略[J].电子技术,2012(7),35-38.
- [7] 刘愉,赵志文,李小兰,等.云计算环境中优化遗传算

- 法的资源调度策略[J].北京师范大学学报:自然科学版,2012,48(4):378-384.
- [8] 熊聪聪,冯龙,陈丽仙,等.云计算中基于遗传算法的任务调度算法研究[J].华中科技大学学报:自然科学版,2012,40(增刊1):1-4.
- [9] Dean J,Ghemawat S.MapReduce:simplified data processing on large clusters [C]//Proceedings of the 6th Symposium on Operating System Design and Implementation.New York:ACM,2004:137-150.
- [10] Pham H.Springer Handbook of Engineering Statistics [M].New York:Springer,2006:229-247.
- [11] Callheiro R N,Ranjan R,Rose C A F D,et al.Cloudsim: a novel framework for modeling and simulation of cloud computing infrastructures and services[J].Computing Research Repository,2009(1):1-9.
- [12] Calheiros N,Rajiv R,Belogazov A,et al.CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms[J].Journal of Software:Practice and Experience,2011,41:23-50.

Task Scheduling Algorithm Based on Mutation Operator in Cloud Computing

CHEN Chao^a, CAI Lecai^b, GAO Xiang^c

(a. Department of Automation & Electronic Information; b. Department of Computer;

c. College of mechanical engineering, Sichuan University of Science & Engineering, Zigong 643000, China)

Abstract: In order to dispatch a huge number of tasks efficiently, an improved genetic algorithm which includes two types of mutation operations: mutation operation *a* and mutation operation *b*, is put forward. Mutation operation *a* is the gene-variation on random position. And in mutation operation *b*, a gene position that meets certain conditions is found out first, then the value of this position is replaced by the target gene value. The chromosome after mutation operation *b* is always superior to that before mutation. Mutation operation *a* is used in earlier time of the algorithm. In later period, algorithm tends to converge to the optimal solution, so mutation operation *b* is used to improve the convergence speed. To avoid algorithm falling into local solution due to the improved mutation operations, the method that use the matching ratio of chromosomes to select initial population is adopted in the process of population initialization, for which the population can distribute in the whole solution space uniformly. The simulation results show that, the improved algorithm not only makes the final completion time shorter and convergence efficiency higher, but also balances the load to some extent. It can realize task scheduling more effectively.

Key words: cloud computing; task scheduling; genetic algorithm; matching ratio; mutation