Journal of Computer Applications
计算机应用,2018,38(12):3490-3495
ISSN 1001-9081
CODEN JYIIDU
2018-12-10
http://www.joca.cn

文章编号:1001-9081(2018)12-3490-06
DOI:10.11772/j.issn.1001-9081.2018040898

云计算资源的动态随机扰动的粒子群优化策略

喻德旷,杨　谊*,钱　俊
(南方医科大学 生物医学工程学院,广州 510515)
(*通信作者电子邮箱 yiyang20110130@163.com)

摘　要:云计算环境中的资源具有动态性和异构性,大规模任务资源分配的目标是最小化完成时间和资源占用,同时具有尽可能好的负载均衡,这是一个非确定性多项式(NP)问题。借鉴智能群体算法的优点,提出基于改进的粒子群优化(PSO)算法构建混合式群体智能调度策略——动态随机扰动的PSO策略(DRDPSO)。首先,将PSO的惯性权重常数修改为变量,实现对求解过程收敛速度的合理控制;其次,缩小每次迭代的搜索范围,在保留候选最优集合的前提下减少无效搜索;然后,引入选择操作,筛选出优质个体并传递到下一代;最后,设计随机扰动,提高候选解的多样性,在一定程度上避免了局部最优陷阱。在CloudSim平台上进行了两类仿真测试,结果表明,处理同构任务时,在大部分情况下DRDPSO的指标都优于模拟退火遗传算法(SAGA)和遗传算法(GA)+PSO算法,总执行时间比SAGA减少13.7%~37.0%,比GA+PSO减少13.6%~31.6%;其资源耗费比SAGA减少9.8%~17.1%,比GA+PSO减少0.6%~31.1%;其迭代次数比SAGA减少15.7%~60.2%,比GA+PSO减少1.4%~54.7%;其负载均衡度比SAGA减小8.1%~18.5%,比GA+PSO减少2.7%~15.3%,且波动幅度最小。处理异构任务时,三种算法表现出相似的规律:CPU型任务的总执行时间最多,混合型任务次之,IO型任务最少,DRDPSO的综合指标最好,较为适合处理多种类型的异构任务,而GA+PSO算法适合快速求解混合型任务,SAGA则适合快速求解IO型任务。所提DRDPSO在处理较大规模的同构和异构任务时,能够较为明显地缩短总的任务执行时间,不同程度地提高资源利用率,并适当兼顾计算节点的负载均衡。

关键词:云计算资源;动态调度;群体智能算法;混合式调度策略;随机扰动
中图分类号:TP391　　**文献标志码**:A

Dynamic random distribution particle swarm optimization strategy for cloud computing resources

YU Dekuang, YANG Yi*, QIAN Jun
(*School of Biomedical Engineering, Southern Medical University, Guangzhou Guangdong* 510515, *China*)

Abstract: Resources in cloud computing environment are dynamic and heterogeneous. The goal of resource allocation in large-scale tasks is to minimize the completion time and resource occupation while having the best load balancing, which is a Non-deterministic Polynomial (NP) problem. Drawing on the advantages of intelligent swarm optimization, a hybrid swarm intelligence scheduling strategy named Dynamic Random Distribution PSO (DRDPSO) was proposed based on an improved PSO algorithm. Firstly, the inertia weight constant of PSO was modified to be a variable to control the convergence speed of solution process reasonably. Secondly, the search scope of each iteration was shrinked so as to reduce invalid search on the premise of retaining candidate optimal set. Then, selection operation was introduced to select high-quality individuals and pass them on to the next generation. Finally, random disturbance was designed to improve the diversity of candidate solutions and avoid the local optimal trap to some extent. Two kinds of simulation tests were carried out on the CloudSim platform. The experimental results show that, the proposed DRDPSO is better than Simulated Annealing Genetic Algorithm (SAGA) and Genetic Algorithm (GA) + PSO in most cases when dealing with isomorphic tasks. The total execution time of the proposed algorithm is less than SAGA by 13.7% - 37.0% and less than GA + PSO by 13.6% - 31.6%, the resource consumption of the proposed algorithm is less than SAGA by 9.8% - 17.1% and less than GA + PSO by 0.6% - 31.1%, the number of iterations of the proposed algorithm is less than SAGA by 15.7% - 60.2% and less than GA + PSO by 1.4% - 54.7%, the load balance degree of the proposed algorithm is less than SAGA by 8.1% - 18.5% and less than GA + PSO by 2.7% - 15.3% with the smallest fluctuation amplitude. When dealing with heterogeneous tasks, three algorithms has the similar properties: in aspect of the total execution time consumption, CPU tasks are the most, the mixed tasks take the second place, and IO tasks are the least. The comprehensive performance of DRDPSO is the best, which is the most suitable for dealing with

收稿日期:2018-05-02;**修回日期**:2018-07-10;**录用日期**:2018-07-19。　**基金项目**:广东省科技计划项目(2013B060500046,2014A020212545,2013B051000054,2017A030304009);广东省教育厅重点平台及科研项目(2016GXJK021);广东省高等教育教学研究和改革项目(C1032165);广州市高校创新创业教育项目课程与教学研究项目(201709k50,201709T40)。

作者简介:喻德旷(1972—),男,江西南昌人,副教授,博士,主要研究方向:医学信号仿真、医学信息处理;　杨谊(1973—),女,广东河源人,副教授,博士,主要研究方向:信息系统设计;　钱俊(1975—),女,安徽黄山人,副教授,博士研究生,主要研究方向:统计模型与应用、云计算。

multiple types of heterogeneous tasks. GA + PSO algorithm is suitable for solving hybrid tasks and SAGA algorithm is suitable for solving IO tasks quickly. When dealing with large-scale isomorphic and heterogeneous tasks, the proposed DRDPSO can significantly shorten the total task execution time and improve the utilization of resources in varying degrees with proper load balancing of computing nodes.

Key words: cloud computing resource; dynamic scheduling; swarm intelligence algorithm; hybrid scheduling strategy; random disturbance

0　引言

云计算通过虚拟化技术将网络计算资源整合在一起,组成一个庞大的计算节点池,用户通过浏览器按需获得资源,完成数据处理任务[1]。网络计算资源庞大且分散,要根据用户请求将资源动态地分配给各个任务,就需要进行合理的资源调度。任务调度策略对用户任务的执行效率、系统资源的使用效率、任务执行成本、负载均衡、系统稳定性等均有直接的影响[2]。云计算环境中的资源具有动态性和异构性,对大规模任务进行资源分配和调度时,不仅需要最小化完成时间和提高系统使用率,而且要考虑资源负载均衡、服务质量,是一个非确定性多项式(Non-deterministic Polynomial, NP)问题[3]。

针对云计算资源调度问题,国内外学者进行了大量的研究,提出了许多调度策略,根据算法原理可分为三大类。第一类是静态资源调度策略[4],如数学规划、专家系统、神经网络和模糊逻辑策略,这类方法的优点是规则严密、逻辑清晰,但是规则较为复杂,只适合小规模资源的静态调度。第二类是传统动态资源调度策略,包括基于贪心思想的策略(Min-min 算法、Max-min 算法、Sufferage 算法等)[5]和基于操作系统调度的策略,如 FIFO(First In First Out)算法、公平调度算法、计算能力调度算法等[6]。这些策略可以实现资源动态调度,但由于参与操作的因子很多,用于大规模任务调度计算复杂度太高。第三类是启发式策略,主要包括遗传算法和粒子群算法[7]。例如,文献[8]提出了一种基于模拟退火思想的改进遗传算法,在退火过程中以一定概率接受劣质解从而避免调度早熟现象;文献[9]在遗传算法基础上,提出了一种考虑多维约束的任务调度算法 MCGA(Multiple Constraints based on Genetic Algorithm),在算法的编码与解码、适应度函数、交叉变异等操作环节上进行了改进;文献[10]建立了多目标优化模型,并结合最新的径向基函数(Radial Basis Function,RBF)神经网络和粒子群优化(Particle Swarm Optimization, PSO)算法对其求解;文献[11]改进了 PSO 的搜索方向、适应度函数等方面;文献[12]将粒子群算法和差分遗传算法结合求解;文献[13]设计了双向搜索的改进粒子群算法等。

1　动态随机扰动 PSO 策略的基本思路

从以上分析可知,相对于传统任务调度策略,启发式任务调度策略在求解 NP 类问题时具有更高的自适应性和灵活性:实验表明,模拟退火算法体现了贪心(Greedy)策略,在搜索到局部最优解后,会进行扰动,以一定的概率接受一个比当前解要差的解,进而有可能跳出局部最优解,找到一个比当前解更好的解。但贪心策略往往不能达到全局最优解,扰动幅度和时机不容易控制,并且在并行性方面较弱。PSO 算法具有参数设置少、全局搜索能力强等优点,其并行性和分布式的特点能够处理海量数据,较为适合作为云计算资源调度策略。但 PSO 在迭代后期随着种群多样性的降低,往往陷入局部最优而错失全局最优解。遗传算法(Genetic Algorithm, GA)从问题解集开始搜索,而不是从单个解开始,通过对不同解的组成进行交叉或修改,能够有效地提高解的组成的多样性,利于全局择优,且具有并行优势,覆盖面大。GA 采用概率规则来指导搜索方向,比确定性规则具有更好的自适应和自学习性,但是概率筛选规则存在机会风险,获得的解是不稳定的。从这些分析可以知道,采用单一算法处理大规模计算资源分配,得到的结果往往不是最优或稳定的。本文利用 PSO 与遗传算法本身的优点和对大量资源调度的适应性,将二者结合起来,设计了基于随机扰动的优化技术的群体智能混合策略,来提高云计算任务调度的综合效率,减少任务执行时间、降低计算成本,并改善负载平衡。

本文提出动态随机扰动的 PSO(Dynamic Random Distribution PSO, DRDPSO)策略如下:首先建立云计算资源调度模型;改进 PSO 算法,将其惯性权重常数修改为变量,使得求解过程的搜索不是匀速而是可控变速的;在每次迭代中以局部范围代替全局范围,减少盲目搜索操作;在 PSO 算法内引入遗传算法的选择操作,筛选出更优质的个体并传递到下一代,并引入变异操作实现随机扰动,试图改善粒子组成,提高粒子多样性;结束条件体现了多重约束的合理运用。

1.1　建立云计算资源模型

云计算的系统资源是有限的,如何为用户任务合理地分配资源,使各个计算节点达到负载均衡,是整个服务过程中需要考虑的问题。为了突出调度问题而减少其他因素干扰,本文约定:所有任务都具有原子性(不再可分);多个任务可以同时在数据中心的计算节点(虚拟机)上执行;每个计算节点都可分配给任何一个任务(任务没有计算特殊性);根据计算能力一个计算节点可以只接受一个任务或同时接受多个任务;不考虑任务切换、传输等时间耗费。对模型参数进行如下定义:

1) 任务集合 $\boldsymbol{T} = \{t_1, t_2, \cdots, t_m\}$,由 m 个相互独立的原子任务组成。

2) 计算节点集合 $\boldsymbol{CN} = \{\boldsymbol{cn}_1, \boldsymbol{cn}_2, \cdots, \boldsymbol{cn}_n\}$,由 n 个计算节点组成,每个计算节点采用如下属性线性表形式描述:$\boldsymbol{cn}_i = \{CPU, MEM, DISK, \cdots\}$,$CPU$ 表示内核数,MEM 表示内存大小,$DISK$ 表示磁盘空间大小,可根据需要增加或删除属性元素。

3) 决策矩阵 $\boldsymbol{D} = \{d_{ij}\}, i = 1,2,\cdots,m, j = 1,2,\cdots,n$, $d_{ij} \in \{0,1\}$,$d_{ij} = 1$ 表示第 i 个任务在第 j 个计算资源上执行;$d_{ij} = 0$ 表示第 i 个任务不在第 j 个计算资源上执行。

4) 执行时间矩阵 $\boldsymbol{ExeTime} = \{et_{ij}\}, i = 1,2,\cdots,m, j = 1,2,\cdots,n$, et_{ij} 表示第 i 个任务在第 j 个计算资源上完成所需要的时间。

5) 计算成本矩阵 $\boldsymbol{CalCost} = \{calc_{ij}\}, i = 1,2,\cdots,m, j = 12,\cdots,n, calc_{ij}$ 表示第 i 个任务在第 j 个计算资源上完成所消

耗的资源成本(CPU、内存、外存)。

6) 总时间成本(Execute Time Cost, ETC)和总计算资源成本(Calculate Resource Cost, CRC)。

$$ETC = \sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} \times et_{ij} \tag{1}$$

$$CRC = \sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} \times calc_{ij} \tag{2}$$

云计算资源调度目标包括:所有任务完成时总的执行时间最小,总的资源占用最少,即使得式(1)和/或式(2)取最小值。实际应用中,式(1)、(2)同时获得最小值的概率较小,可以根据实际需要赋予总时间成本和总计算成本以不同的权重如式(3):

$$TotalCost = \mu_1 ETC + \mu_2 CRC \tag{3}$$

如侧重时间成本,则 μ_1 取较大值;若侧重计算成本,则 μ_2 取较大值,目标是综合代价 *TotalCost* 最小。

在本文实验中,云计算资源模拟环境下 *ETC* 和 *CRC* 为同一数量级,如果在其他应用场景下这两个变量不是同一数量级,则应当进行归一化处理后再加权累加。

1.2 动态随机扰动 PSO 策略

PSO 算法初始化为一群随机粒子,代表随机解。所有的粒子都具有速度属性,在每一次迭代中,每个粒子通过跟踪两个值来更新自己的速度和位置:一个是粒子本身所找到的最优解 *lBest*,另一个是整个种群目前的最优解 *gBest*。每个粒子根据适应度函数来判断自己当前是否到达最优解的位置。粒子之间利用信息共享进行从无序到有序的演化,从而获得全局最优解。

1.2.1 简化的编码设计

通常粒子群算法和遗传算法可以采用两种编码方法:实数编码法和二进制编码法。前者表示形式容易理解,解码简单;后者能够表示多样化的种群,但占用存储空间较大,解码过程难以理解。为了提高策略可读性和一致性,本文采用实数编码法对粒子进行编码。每个粒子表示任务与资源的对应关系,如粒子{(1,3),(2,2),(3,4),(4,3),(5,1)}表示任务 1 分配到资源(计算节点)3 上、任务 2 分配到资源 2 上、任务 3 分配到资源 4 上等。在同一时间窗口内的任务的数量决定了编码的长度,上例中的粒子的长度为 5。

1.2.2 改进的粒子更新策略为“前快后慢”

设群体粒子集合(即解的集合)为 $\boldsymbol{X} = \{x_1, x_2, \cdots, x_s\}$,$s$ 为解空间规模,粒子 $x_i (i \in [1,s])$ 在 t 时刻的位置为 $p_i(t)$,此时单个粒子的最佳位置为 $lbest\_p(t)$,当前群体的最佳位置为 $gbest\_p(t)$,粒子 x_i 运动的速度公式为:

$$\boldsymbol{v}_i(t+1) = w \times \boldsymbol{v}_i(t) + c_1 \times (lbest\_p(t) - p_i(t)) + c_2 \times (gbest\_p(t) - p_i(t)) \tag{4}$$

位置公式为:

$$p_i(t+1) = p_i(t) + \boldsymbol{v}_i(t+1) \tag{5}$$

其中:w 为惯性权重常数;c_1 和 c_2 为常系数。

在实验中发现,由于速度更新率即式(4)中的惯性权重 c_1 和 c_2 设置为常数,使得求解过程中收敛速度保持不变。而实际的解搜索往往需要在早期进行快速的大致定位,在后期放慢速度在确定的候选区域内进行比较仔细的搜索,因此本文将惯性权重修改为变量,根据经验设计如下:

$$w(t+1) = w(t) + k_1 \times \frac{IterNum - IterNow}{IterNow} + k_2 \times \frac{fit(t)}{fit(t-1)} \tag{6}$$

其中:*IterNum* 为算法的总迭代次数;*IterNow* 为当前迭代次数;k_1 和 k_2 为常数,一般取值在 1 ~ 2,其比例关系为1∶1;$fit(t)$ 为 t 时刻群体的适应度函数值。惯性权重的初始值默认为 1。

式(6)体现了对粒子运动变化速率的合理控制,惯性权重变化受到两个因素的影响:

一是迭代次数。迭代过程中粒子的更新速度应当是前快后慢,这是由于起始点是随机解,应当尽快收敛到全局最优解可能处在的小区域中,所以收敛速度要快;而后期则应当在确定的小区域内进行较为细致的搜索,所以收敛速度要慢。按照经典算法,收敛速度一直是常量,将不利于提高收敛速度,也不利于在最可能产生最优解的区域细致搜索。

二是适应度(综合代价 *TotalCost*)。根据前面分析的云计算资源调度目标和所涉及的衡量指标,本文策略的适应度函数主要由式(3)决定,即同时考虑总的执行时间和总的资源开销。此外,各台服务器节点的负载平衡度不应差异过大,否则会造成某些节点压力重,容易出现瓶颈,而某些节点的资源得不到有效利用,因此本文定义了负载均衡指标。云平台中某个计算节点 j 的负载均衡因子 LB_j 定义为:

$$LB_j = \mu_1 \times \left(\sum_{i=1}^{m} d_{ij} \times et_{ij}\right)/ETC + \mu_2 \times \left(\sum_{i=1}^{m} d_{ij} \times calc_{ij}\right)/CRC \tag{7}$$

其中 μ_1 和 μ_2 为权重参数,其他参数含义同前面介绍。

负载均衡度反映了调度策略对计算资源利用的公平性和均衡性。定义负载均衡度 τ 为:

$$\tau = \sqrt{\sum_{j=1}^{n} (LB_j - \overline{LB_j})^2/(n-1)} \tag{8}$$

其中:$\overline{LB_j}$ 为所有 LB_j 的均值;τ 值越小,则说明负载均衡性越好,解的适应度越高。

由于 τ 和适应度函数值 *TotalCost* 在数值上往往不是同一数量级,在不同类型的任务中也难以归一化,所以解的优良性的判断规则为两个:优先依据 *TotalCost* 选择局部最优解;如果存在多个局部最优解,则根据负载均衡度 τ 判断,取 τ 最小的解。也就是说在综合代价最小的前提下兼顾负载均衡。

1.2.3 缩小搜索范围

由于粒子在上一时刻的最佳位置直接影响到下一时刻的最佳位置的选择,不必每次迭代都重新搜索整个种群,所以本文用 $t-1$ 时刻 $gbest\_p(t-1)$ 粒子的若干个邻居中的最佳位置(邻居数量 *NeighNum* 根据经验设置,通常为总粒子数量的 1/3),取代 t 时刻整个种群的最佳粒子位置作为 $gbest\_p(t)$ 的值,从而减少大量无效的搜索和比较,降低计算代价,提高计算效率。如果最佳位置确实需要发生大的偏移,则可以通过适应度函数控制,函数值的改变能够调整粒子运动的范围(发散或收敛),越靠近当前最佳位置的粒子下一步的搜索范围应当收敛,反之应当发散。

1.2.4 引入遗传算法的选择和变异操作

遗传算法也属于搜索启发式算法,从完全随机个体的种群开始,选择多个适应度较高的个体,通过选择、交叉和突变产生新的生命个体,构成下一代种群。选择运算的作用是把优质的个体基因直接遗传到下一代;交叉运算则交换两个个

体的若干基因,产生新的个体;变异运算是对个体的某些基因位点作修改,产生新的个体。迭代结束时,以进化过程中所得到的具有最大适应度个体作为最优解输出。

资源调度求解的搜索过程中也会出现某些时刻粒子的相似度高,而不利于发现更优解的现象。因此本文在改进的PSO过程中引入遗传算法的选择和变异操作。在每一代中选择一定比例的优质粒子,即 N_{top} 个适应度最高的粒子即优质粒子(N_{top} 根据经验设置为粒子总数的1/10),直接进入下一代。对于其他的粒子则执行随机变异操作,具体做法是对 t 时刻随机选取的非优质粒子 x_i 的第 j 个分量 $x_{ij}(t)$ 按照随机概率 $\varphi(i,j,t)$ 进行随机变异,变异率定义如下:

$$\varphi(i,j,t) = \begin{cases} 1, & rand(seed) > \sigma \\ 0, & \text{其他} \end{cases} \tag{9}$$

其中,σ 取值范围是(0,1),一般取[0.6,0.8]。例如,σ 取值为0.7,对于粒子{(1,3),(2,4),(3,4),(4,3),(5,1)},某个时刻的 $j=3$ 的分量的变异概率 $rand$ 为0.8,变异后的新的对应资源编号为资源集合中的随机值(假设计算得到1),如果编号1的资源能够满足任务3的需求,则实施变异为{(1,3),(2,4),(3,1),(4,3),(5,1)},否则不执行变异操作。通过变异操作,改变粒子的基因组成,从而有可能将适应度较低的粒子进行优化,但是也不排除变异操作导致粒子的适应度降低的情况。以上的变异操作实质上也是一种随机扰动技术,可以防止算法过早地陷入局部最优解。

遗传算法中的交叉操作也能够提高基因的多样性,减少陷入局部最优的可能,但粒子之间基因的交叉可能导致多个任务需求与资源的连锁不匹配,而判断更新后的任务和资源是否匹配,会大大增加算法的复杂性,所以本文没有盲目采用遗传算法的交叉操作,而是用随机变异的扰动技术来达到目的。

1.2.5　算法结束条件综合考虑多个指标

本文把分配成功的指标($TotalCost$,运行时间少,计算资源占用少)和计算公平指标(τ,节点的负载均衡度)结合起来,这使得本文策略的综合性能更好。实际上这些指标不可能同时达到最优,因为它们之间存在相互冲突性。在解决实际问题时,往往达到需要的综合最优就可以了。结束条件表示为:当 t 时刻粒子集的适应度函数与上一时刻相比不再增加(近似),或者迭代次数到达了一个上限,就令算法结束。形式化设置为:(presumption1)t 时刻的适应度(综合代价)$TotalCost$ 与 $t-1$ 时刻相比,增加率小于阈值 ε_c,且 t 时刻的负载均衡度 τ 与 $t-1$ 时刻相比,降低率小于阈值 ε_τ;(presumption2)达到最大迭代次数 $IterNum$。

1.2.6　DRDPSO 策略流程

Step1　初始化种群,随机生成符合任务-资源约束条件的粒子编码,设置最大迭代次数、解规模等参数的初值。

Step2　初始化粒子局部最优和全局最优值。

Step3　根据式(3)、(6),更新式(4)、(5),即更新所有粒子的速度和位置。

Step4　根据适应度函数式(3)计算各个粒子的适应度和群体适应度(群体搜索范围控制在 $NeighNum$ 而不是所有粒子),根据式(8)计算负载均衡权值。

Step5　若满足结束条件(presumption1),则输出当前解,并结束;否则,若满足条件(presumption2),则输出当前解,并结束;若不满足条件(presumption1),也不满足条件(presumption2),转到Step6。

Step6　对当前群体 N_{top} 个适应度最高的优质粒子直接进入下一代。按照式(9)概率对非优质粒子进行选择和随机变异操作,生成下一代群体;转到Step3。

1.2.7　算法复杂度分析

经典PSO算法每一次迭代中的粒子数量不变,记为 N_c。设第 i 次迭代中粒子的数量为 $N_i(i=1,2,\cdots,m)$,m 为最大迭代次数,则有 $N_i = N_c$。设每个粒子每一次迭代需要的运算时间为 T_i,则经典PSO算法总的运行时间为 $O(N_c * T_i * m)$。本文DRDPSO策略将群体搜索范围控制在 $NeighNum$ 而不是所有粒子($NeighNum$ 根据经验设置,通常为总粒子数量的1/3),即搜索范围缩小至原来的1/3,且随着迭代的进行,后期符合条件的粒子的数量还会逐渐减少(非单调)。设每个粒子每一次迭代需要的运算时间为 T_i,则本文DRDPSO策略总的运行时间为 $O(N_i * T_i * m)$,其中 $N_i \leqslant 1/3N_c$。比经典PSO算法增加的计算式(6)、(8)、(9)均为常数级计算,对时间复杂度带来的增量远远小于搜索范围带来的减量,可以忽略。本文DRDPSO策略改进的效果在运行时间方面主要体现在:每一次迭代中的搜索范围大大减少,参与下一步运算的粒子数大大减少,使得运行时间减少。而增加的计算部分使得对资源调度的评估标准更合理,解的多样性更好。

在空间存储方面,本文DRDPSO策略比经典PSO算法在每次迭代中增加了常量数量级的中间变量如 $fit(t)$、τ、$\varphi(i,j,t)$ 等,以及与之相关的临时变量的存储,但每次迭代搜索范围相比经典算法缩小至原来的1/3,使得临时粒子的保存量也大为减少,所以空间复杂度有所降低。

需要明确的是,本文策略属于启发式搜索算法,这一类算法所获得的解都并不是理论上的最优解。对于诸多NP-hard问题,使用确定性算法时间复杂度的代价极大,到不能接受的程度(运行时间很长),有时候需要保存的中间变量数量也很大,导致空间复杂度也较高。而求解实际问题往往并不需要理论上的最优解,只需要一个满足一定条件、符合工程需求的次最优解就能解决问题。

2　实验结果与分析

本文采用云计算仿真工具CloudSim进行实验仿真,首先创建数据中心、用户代理和计算节点(虚拟资源),其次将用户代理与计算节点进行映射,生成云任务集合,在此基础上使用不同的调度策略将任务分配给计算节点。文献[8]的模拟退火遗传算法(Simulated Annealing Genetic Algorithm, SAGA)和文献[12]的GA+PSO算法分别对云计算的资源调度问题对标准遗传算法和标准PSO算法作了有特色的改进,并提供了较详细的实现过程,因此作为本文策略的对照算法。在相同环境和条件下将本文DRDPSO策略与SAGA、GA+PSO进行对比实验。计算机仿真实验环境的配置为:Windows 7操作系统,CPU 4核,内存16 GB,硬盘2 TB。

2.1　任务规模单变量实验结果

在任务类型相同的条件下,以任务规模为单变量,分别生成 $m=100,200,\cdots,1000$ 个模拟的用户任务,计算资源 n 取50。本文DRDPSO策略中的参数初始化为:种群规模 $Size$(即粒子的个数,或候选解的个数)初始化为100,$N_{top} = Size/10 = 10$,$NeighNum = Size/3 = 33$,最大迭代次数 $IterNum = 2000$。$\mu_1 = 4$,$\mu_2 = 1$,表示本文更重视总执行时间

的优化,而把资源成本放在较次要的位置。$c_1 = c_2 = 1, k_1 = k_2 = 1, \sigma = 0.7$。为了挖掘最优值,体现每次迭代与上一次的细微改变,设置为较小的门限值:$\varepsilon_c = 0.01, \varepsilon_\tau = 0.01$(实际运行时,根据具体问题域的变化幅度不同,或者为了避免迭代次数过多,总是超过上限,可以根据经验调整为小于等于0.05的值)。SAGA 和 GA + PSO 的公共参数与本文算法相同,其各自的局部变量大部分按原文献设置参数值,但为了在本实验中获取最好的结果,对部分参数值作了适当调整。三种算法在处理不同任务规模下的 CPU 型任务的性能如图 1 所示。由于本文算法中多处使用了随机技术,所以同样参数的实验多次运行结果在数值上不完全相同,但是多次实验结果反映出的规律是基本一致的。

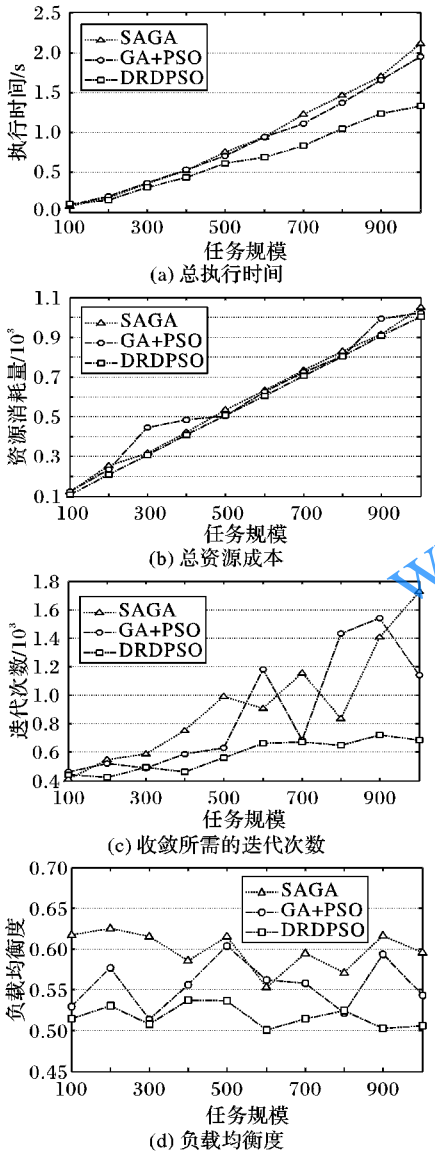

图 1 不同算法在不同任务规模下的性能比较

Fig. 1 Performance comparison of different algorithms under different task scales

从图 1(a) 可以看出,在任务规模比较小的情况下(规模为 100),本文算法 DRDPSO 比 SAGA、GA + PSO 的最优调度方案所需的总执行时间略大。而随着任务规模的增加,三种调度策略的总执行时间都有增加,本文 DRDPSO 策略的增加率明显小于 GA + PSO,而 GA + PSO 则略小于 SAGA。DRDPSO 的总执行时间比 SAGA 少 13.7% ~37.0%,比 GA + PSO 少 13.6% ~31.6%。上述结果表明本文调度方案能够适应大规模任务调度,在较短的时间内完成用户任务。

由图 1(b)可以看出,本文 DRDPSO 策略与 SAGA 在任务规模增加时,资源耗费近似于线性增长,表明这两种策略在任务-计算节点匹配的适应性和稳定性均较好,而 GA + PSO 的资源耗费波动较大。在大多数情况下 DRDPSO 的资源耗费比其他两种算法少,随任务规模不同,本文 DRDPSO 策略的资源耗费比 SAGA 少 9.8% ~17.1%,比 GA + PSO 少0.6% ~31.1%。

从图 1(c)可以看出,在达到收敛所需的迭代次数方面,文本算法 DRDPSO 明显少于其他两个算法,且不易受任务规模变化的影响,而另外两种算法在任务规模增大时迭代次数也显著增加,直到任务规模增大到一定程度才停止大幅增加。DRDPSO 的迭代次数比 SAGA 少 15.7% ~60.2%,比 GA + PSO 少 1.4% ~54.7%。文本算法 DRDPSO 迭代次数出现波动的原因主要来自于 $\varphi(i,j,t)$ 的值和每次实验随机产生的任务序列的情况(子任务大小、子任务出现顺序等),由于适应度函数和负载均衡度等多种指标的共同制约,使得迭代次数较少。同时也看到,在大部分情况下,本文 DRDPSO 策略的时间耗费没有与迭代次数的减少幅度成线性关系,是每次迭代中增加了用于改进的多个计算导致的。

从图 1(d)可以看出,在不同的任务规模条件下,本文算法 DRDPSO 的负载均衡度总体来看最小,比 SAGA 减小 8.1% ~18.5%,大部分情况下比 GA + PSO 减少 2.7% ~15.3%,且变化最小。GA + PSO 虽然有时获得比 DRDPSO 更好的负载均衡度,但它的整体性能不够稳定,而 SAGA 的负载均衡度偏大,表明其对计算节点的利用不够均衡。

2.2 任务类型单变量实验结果

为了比较对不同类型任务的处理效果,将任务划分为三类:1)CPU 型任务,占用 CPU 的比例远大于 IO 比例(比值大于 3∶1);2)混合型任务,占用 CPU 和 IO 比例大致相当(比值介于 3∶1 和 1∶3 之间);3)IO 型任务,占用 CPU 的比例远小于 IO 的比例(比值小于 1∶3)。实验结果表明,三种算法各自在相同任务规模下的性能基本保持一定的规律。以任务规模 500 为例,三种算法对三种类型任务的平均总执行时间、平均总的资源成本、平均迭代次数和平均资源负载均衡度如表 1 所示。

从表 1 可以看出,在总执行时间方面,三种调度策略表现相似,都是对于 CPU 型任务的总执行时间最多,混合型任务次之,IO 型任务最少,显然,这是由于本文定义的适应度函数为综合代价 *TotalCost*,而 *TotalCost* 只考虑执行时间和资源占用情况,未考虑任务传输时间(而实际应用也往往不计入任务传输时间,或者认为任务可以预传输,传输时间为常量)。在总执行时间方面,本文策略 DRDPSO 表现最好。类似的,在占用资源数量方面,三种调度策略对于三种类型任务的总资源成本也呈现与总执行时间相似的规律。在收敛迭代次数方面,本文算法 DRDPSO 在各类型任务都是最少的,GA + PSO 算法适合快速求解混合型任务,SAGA 则适合快速求解 IO 型任务。在负载均衡度方面,本文算法 DRDPSO 和 GA + PSO 算法的负载均衡度对于三种类型的任务的表现相似,即处理 CPU 型任务的均衡性最弱,处理 IO 型任务的均衡性最好,而 SAGA 则适合于混合型任务的均衡调度。

多次实验结果表明,本文 DRDPSO 策略在大部分情况下在多个指标上比其他两种算法均有更好的表现。但由于算法

的随机性本质,不能保证在每次运行时在各个方面(执行时间、资源成本、迭代次数)都优于对比算法。

表 1　在任务规模 500 条件下,三种算法对三种类型任务的性能比较

Tab. 1　Performance comparison of three algorithms for three types of task under task scale of 500

算法	执行时间/ms			占用资源数量			收敛所需迭代次数			负载均衡度		
	CPU 型	混合型	IO 型	CPU 型	混合型	IO 型	CPU 型	混合型	IO 型	CPU 型	混合型	IO 型
SAGA	781.11	485.26	282.42	526	495	310	687	649	554	0.608 6	0.570 0	0.485 3
GA + PSO	740.67	432.00	281.42	518	468	274	713	772	607	0.593 1	0.505 5	0.549 4
DRDPSO	558.77	364.43	286.48	508	342	248	465	476	328	0.522 1	0.478 3	0.429 3

以上实验结果表明,本文 DRDPSO 策略有较为显著的效果:将惯性系数由常量修改为变量,使得过程前期搜索速度快而后期搜索精度高,实现对迭代过程的合理控制;以局部最优位置代替全局最优位置,减少了大量的无效搜索;引入遗传算法的选择操作,筛选出优质个体并传递到下一代;引入变异操作实现随机扰动,改善非优质粒子的组成,提高了全局最优解出现的概率;综合约束条件使得结束点更为合理。这些改进操作都促进了总的执行时间、资源成本、迭代次数的降低,资源-任务匹配度升高,以及资源负载均衡度减小。

3　结语

基于云计算的资源调度目标和要求,本文通过对现有调度策略的总体分析得出群体智能算法具有较好的处理能力。本文结合两种表现较为突出的群体智能算法 PSO 与遗传算法优势,并加入了多种改进方式,形成了混合式群体智能调度策略 DRDPSO。在 CloudSim 平台上进行了两类仿真测试,大量实验统计结果表明,与对比算法 SAGA 和 GA + PSO 相比,DRDPSO 能够较为明显地缩短总的任务执行时间,提高了不同类型任务下的资源利用率,并适当兼顾计算节点的负载均衡,能够较好地解决云计算资源的动态调度问题。

参考文献(References)

[1] MARTINEZ G, ZEADALLY S, CHAO H C. Editorial: cloud computing service and architecture models [J]. Information Sciences, 2014, 258 (3): 353 - 354.

[2] FU X, CANG Y L. Task scheduling and virtual machine allocation policy in cloud computing environment [J]. Journal of Systems Engineering and Electronics, 2015, 26 (4): 847 - 856.

[3] LIANG Q, ZHANG J, ZHANG Y H. The placement method of resources and applications based on request prediction in cloud data center [J]. Information Sciences, 2014, 279: 735 - 745.

[4] ALAHMADI A, ALNOWISER A, ZHU M M, et al. Enhanced first - fit decreasing algorithm for energy aware job scheduling in cloud [C]// Proceedings of the 2014 IEEE International Conference on Computational Science and Computational Intelligence. Piscataway, NJ: IEEE Press, 2014: 69 - 74

[5] ZOU D Q, XIANG Y, MIN G Y. Privacy preserving in cloud computing environment [J]. Security and Communication Networks, 2016, 9(15): 2752 - 2753

[6] LI Y S, TANG X Y, CAI W T. Dynamic bin packing for on-demand cloud resource allocation [J]. IEEE Transactions on Parallel & Distributed Systems, 2016, 27(1): 157 - 170.

[7] 谢丽霞,严焱心. 云计算环境下的服务调度和资源调度研究[J]. 计算机应用研究,2015,32(2):528 - 531.(XIE L X, YAN Y X. Analysis of service scheduling and resource allocation based on cloud computing [J]. Application Research of Computers, 2015, 32(2): 528 - 531.)

[8] 刘峰,毕利,杨军. 一种用于云计算资源调度的改进遗传算法[J]. 计算机测量与控制,2016,24(5):202 - 206.(LIU F, BI L, YANG J. An improved genetic algorithm for cloud computing resource scheduling [J]. Computer Measurement & Control, 2016, 24 (5): 202 - 206.)

[9] 李超,戴炳荣,旷志光,等. 云计算环境下基于改进遗传算法的多维约束任务调度研究[J]. 小型微型计算机系统,2017,38(9): 1945 - 1949.(LI C, DAI B R, KUANG Z G, et al. Research on task scheduling with multiple constraints based on genetic algorithm in cloud computing environment [J]. Journal of Chinese Computer Systems, 2017, 38(9): 1945 - 1949.)

[10] 赵宏伟,李圣普. 基于粒子群算法和 RBF 神经网络的云计算资源调度方法研究[J]. 计算机科学,2016,43(3): 113 - 117.(ZHAO H W, LI S P. Research on resources scheduling method in cloud computing based on PSO and RBF neural network [J]. Computer Science, 2016, 43(3): 113 - 117.)

[11] 温聪源,徐守萍,曾致远. 云计算环境下基于 ABC-QPSO 算法的资源调度模型[J]. 计算机应用与软件,2015,32(5): 30 - 32, 64.(WEN C Y, XU S P, ZENG Z Y. ABC and QPSO algorithm-based resource scheduling model in cloud computing environment [J]. Computer Applications and Software, 2015, 32(5): 30 - 32, 64.)

[12] 陈海涛. 云计算中的基于粒子群算法和差分遗传算法的资源调度[J]. 计算机系统应用,2015,24 (10):136 - 141.(CHEN H T. Resource scheduling based on particle swarm and differential genetic algorithm in cloud computing [J]. Computer Systems & Applications, 2015, 24 (10): 136 - 141.)

[13] WANG X L, WANG Y P, CUI Y. An energy-aware bi-level optimization model for multi-job scheduling problems under cloud computing [J]. Soft Computing, 2016, 20(1): 303 - 317.)

This work is partially supported by the Guangdong Science and Technology Project (2013B060500046, 2014A020212545, 2013B051000054, 2017A030304009), the Key Platforms and Research Projects by Foundation of Guangdong Educational Committee (2016GXJK021), the Research and Reform Project of Higher Education of Guangdong Province (C1032165), the Guangzhou University Innovation and Entrepreneurship Education Project Curriculum and Teaching Research Project (201709k50, 201709T40).

YU Dekuang, born in 1972, Ph. D., associate professor. His research interests include medical signal simulation, medical information processing.

YANG Yi, born in 1973, Ph. D., associate professor. Her research interests include information system design.

QIAN Jun, born in 1975, Ph. D. candidate, associate professor. Her research interests include statistical model and application, cloud computing.