



云计算中基于生物共生机制 改进粒子群优化的任务调度方案

王琳杰

(铜仁学院数学科学学院, 贵州 铜仁 554300)

摘要:针对传统的基于智能算法的云计算任务调度方案获取最优解用时较多的问题,受生物界共生现象的启发,提出一种基于生物共生机制(SM)改进粒子群优化(PSO)的任务调度方案。首先,将 PSO 中的粒子分成 2 个种群,各自执行寻优。然后,每执行 k 次 PSO 迭代后,将两个种群中的个体进行互利共生和寄生操作。通过互利共生操作使搜索过程穿过最佳解区域,从而增强搜索能力;通过寄生操作排除较差解并引入较优解来防止过早收敛。最终获得任务调度的最优解。仿真结果表明,提出的优化算法可快速收敛,相比其他几种较新的调度方案,提出的方案能够获得最小的任务完成时间和响应时间。

关键词:云计算;任务调度;生物共生机制;粒子群优化;全局搜索能力

中图分类号:TP311

文献标识码:A

doi: 10.11959/j.issn.1000-0801.2016241

Task scheduling scheme based on improved particle swarm optimization with biological symbiosis mechanism in cloud computing

WANG Linjie

School of Mathematical Sciences, Tongren University, Tongren 554300, China

Abstract: For the issues that the existing task scheduling scheme based on intelligent algorithms can't obtain the optimal solution in cloud computing and inspired by nature symbiosis, a new task scheduling scheme based on improved particle swarm optimization (PSO) with biological symbiosis mechanism (SM) was proposed. Firstly, the particles in PSO were divided into two populations, and the optimization process were performed alone. Then, after each execution of the k iteration of PSO, the individual in the two populations performed the mutualism and parasitism operation. The search process was optimized by mutualism operation to through the optimal solution region, which could enhance the search ability. The parasitism operation was used to avoid premature convergence by eliminating the poor and introducing the optimal solution. Finally, the optimal solution of the task scheduling was obtained. Simulation results show that the optimal scheduling scheme can obtain the minimum task completion time and response time.

Key words: cloud computing, task scheduling, biological symbiosis mechanism, particle swarm optimization, global searching capability

收稿日期:2016-02-29;修回日期:2016-09-11

基金项目:国家自然科学基金资助项目(No.61364003, No.61562703);贵州省联合基金资助项目(黔科合 J 字 LKT[2012]16 号)

Foundation Items: The National Natural Science Foundation of China(No.61364003, No.61562703), Guizhou Province Mutual Foundation (No.LKT[2012]16)



1 引言

云计算^[1]是一种新的计算技术,用户可以利用云计算租借软件、硬件、基础设施和计算资源,作为每个用户的基础资源,并将他们的工作提交给云计算处理或者存储。云数据存在于不同服务器中,这些服务器通过网络连通并以虚拟机(virtual machine, VM)的形式实现^[2]。云计算中的用户任务通常以工作流的方式进行,工作流任务调度方案作为云计算工作流技术的重要组成部分,已经成为该领域的研究热点。无效任务调度会导致收益损失、性能退化和违反服务水平协议(service level agreement, SLA)^[3]。工作流任务调度的主要目标是减少任务执行的总时间和资源的空闲时间,提高资源利用率^[4]。

任务调度是一个组合 NP-完全问题,启发式智能算法是解决这种问题的一种有效手段,常用的智能算法有遗传算法(genetic algorithm, GA)、粒子群优化(particle swarm optimization, PSO)算法、蚁群算法(ant colony algorithm, ACO)等。其中,PSO 的搜索最优解决方案的能力比 GA 和 ACO 强,且收敛速度快^[5]。然而,这些方法都存在容易陷入局部最优的缺陷。为了解决这个问题,学者提出了多种基于改进型 PSO 算法的任务调度方案。例如,参考文献[6]提出了蚁群—粒子群优化(AC-PSO)方案,首先利用 PSO 算法迅速求得初始解,然后根据调度结果生成蚁群的初始信息素分布,最后利用 ACO 获得最优解。然而,蚁群算法在搜索开始时往往信息素不足,使得其早期搜索效率较低,影响调度方案的整体执行效率。参考文献[7]提出一种结合模拟退火(simulated annealing, SA)算法和 PSO(SA-PSO)的任务调度方案,在各温度下依次进行 PSO 和 SA 搜索,形成两层串行结构。其利用 SA 的概率跳性保证种群的多样性,避免陷入局部最优。然而,该方案执行时间较长,且初始参数的设置对 SA 算法性能的影响很大。

PSO 算法源于对社会型群居动物的行为模拟,因而将大自然中的一些生物行为融入 PSO 算法是一条可行的改进途径。国内外学者在这方面也开展了一些研究。例如,参考文献[8]受生物中捕食猎物的协同进化启发,将 PSO 算法中的粒子分成两类,相互排斥,使粒子在搜索过程中迫使陷入局部最优点的粒子逃离。然而,该方法没有对两个粒子群进行更新和淘汰,影响了解的质量。参考文献[9]根据细菌的趋化行为,使 PSO 算法中粒子不仅被当前的最优位置吸引,同时也会被自己历史最差位置和群体最差位置

所排斥。然而,该方法中的细菌在食物方向趋化过程中,若食物耗尽,其没有进行翻转操作以改变行进方向,影响了搜索速度。

目前,利用生物行为对 PSO 进行改进的研究还很少。本文受到生物共生行为的启发,提出一种基于共生机制(symbiosis mechanism, SM)的 PSO 改进算法(SM-PSO),用于求解云计算中的任务调度方案。将 PSO 中的粒子分成两个种群,各自进行寻优。每迭代 k 次之后,将两个种群中的个体进行互利共生和寄生操作,提高粒子的多样性,以此避免算法陷入局部最优。仿真结果表明,本文方案能够快速寻找到任务调度的最优方案,获得了最短任务执行时间和响应时间。

2 问题描述

2.1 任务调度简述

云系统是由多个分布在各地的数据中心(data center, DC)组成,可通过网络互相访问。每个数据中心由多个计算机、存储单元和其他资源组成。数据中心的 VM 通过一个高带宽的通信网络进行连接。因此,通信时延可以忽略不计。任务调度的目的是将用户任务有效地分配到不同的可用 VM 上,优化能源消耗和任务完成时间。

当云服务中心(CB)接收到需要调度的任务时,首先需要查询执行这些任务所需的服务,然后在所发现的可用服务上调度这些任务。例如,在给定时间间隔内将任务 $\{T_1, T_2, T_3, \dots, T_n\}$ 提交给 CB。各数据中心的 VM 是异构的,有着不同的处理速度和内存,即在不同 VM 上执行同一任务将会产生不同的执行成本。当 CB 接收到任务时,假设虚拟机 $\{V_1, V_2, V_3, \dots, V_m\}$ 可用。那么可在这些 VM 上调度任务,并基于截止时间优先执行任务。

2.2 任务调度目标函数

本文任务调度的目标是合理调度任务到 VM 上执行,从而获得较高的 VM 利用率和最小完成时间。因此,本文根据每个调度到 VM 上的任务的预期执行时间(expected time to compute, ETC)来制定调度决策。ETC 值根据任务长度和 VM 的计算能力的比值进行估计,计算能力单位为每秒百万条指令(million instructions per second, MIPS)。ETC 值通常用矩阵形式表示,矩阵的行表示需要调度的任务数,列表示可用 VM 数。ETC 矩阵的每一行代表了每个 VM 执行一个给定任务所需的时间,而每一列表示每个任务在一个给定的 VM 上执行所需的时间。

本文目标是通过寻找 VM 和任务的最佳组合来最小化完成时间。令 $C_{ij}(i \in \{1, 2, 3, \dots, m\}, j \in \{1, 2, 3, \dots, n\})$ 表示第 i 个 VM 上执行第 j 个任务的执行时间, 其中, m 是 VM 的数量, n 是任务的数量。所以, PSO 中每个粒子的适应度值 f 可以表示为:

$$f = \max\{C_{ij}, \forall \text{ all tasks } j \text{ mapped to } VM_i\}, i \in \{1, 2, 3, \dots, m\} \quad (1)$$

用户任务流必须在截止期限 d_i 前完成, 每个用户任务流可被分配到同一个数据中心。

2.3 粒子编码

在本文提出的基于改进 PSO 算法求解任务调度的过程中, 将每个粒子编码为一个 $1 \times n$ 维向量的任务调度方案 (n 为任务数), 表 1 为由一个 1×4 维向量表示的任务调度方案。其中, 向量的元素为 $[1, m]$ 内的自然数, m 为 VM 数。最优粒子 x^{best} 即获得的最优任务调度方案。

表 1 编码粒子为一个解向量的例子

任务	T_1	T_2	T_3	T_4
VM	1	1	3	2

3 利用共生机制改进 PSO 算法

生物之间的“共生”是指生物体生活在一起时的交互作用。共生的生物在生理上相互分工, 互换生命活动的产物, 在组织上形成新的结构。从生物体利弊关系的角度, 共生关系可分成互利共生、共栖和寄生^[10]。互利共生是指共生的生物体成员彼此都得到好处; 共栖是指仅对其中一方生物体有益, 对另一方没有影响; 寄生是指一种生物寄生在另一种生物的体内或体表, 吸取营养物质来维持生活。

在标准 PSO 算法的进化过程中, 每一代均更新信息, 即更新个体的历史最优位置和群体的历史最优位置。这样就没有充分利用当前所搜索到的最优位置, 一旦粒子陷入局部最优值, 算法将无法继续执行, 影响 PSO 的全局收敛性^[11]。针对 PSO 算法易发生过早收敛的问题, 受自然界生物共生现象的启发, 本文将粒子群分为多个种群, 改变传统的用一个种群在解空间中搜索最优解。各个种群各自进化, 定期使用互利共生和寄生机制来更新搜索空间中解向量的位置, 使整个系统协同进化, 直至获得最优解。

3.1 传统 PSO 算法

传统 PSO 中, 每个粒子在每次迭代中都会向吸引子所引导的方向移动, 从一个解跳跃至另一个解。如果结果是

一个较优解, 则其会吸引其他粒子向这个解移动, 最终获得最优解。PSO 算法的粒子位置更新计算式为^[12]:

$$v_i^j = w \cdot v_i^j + c_1 \cdot r_1 \cdot (p_i^j - x_i^j) + c_2 \cdot r_2 \cdot (p_g^j - x_i^j) \quad (2)$$

$$x_i^{j+1} = x_i^j + v_i^{j+1} \quad (3)$$

其中, x_i 为第 i 个粒子的位置; v_i 为第 i 个粒子的移动速度; p_i 为第 i 个粒子的个体最优解; p_g 为全局最优解; c_1 和 c_2 为学习因子, 表示粒子向个体最优解和全局最优解靠近的程度, 通常都取 2; r_1 和 r_2 为 $[0, 1]$ 内的随机数。 w 为惯性权重, 决定当前粒子速度继承的程度。较大的惯性权重 w 有利于跳出局部最优, 进行全局寻优; 较小的 w 值有利于局部寻优, 加速算法收敛。为了平衡算法的全局和局部搜索能力, 惯性权重的值通常会在一定范围内, 随着迭代次数的增加而线性递减, 即动态惯性权重^[13], 表达式如下:

$$w = w_{\min} + \frac{w_{\max} - w_{\min}}{N} \times (N - ite) \quad (4)$$

其中, $w_{\max}$ 和 $w_{\min}$ 为 w 的最大值和最小值, 通常取 0.9 和 0.4; N 为最大迭代次数, ite 为当前迭代次数。

3.2 改进 PSO 算法

本文将传统 PSO 算法中的粒子分成 2 个种群, 将个体之间的互利共生和寄生关系嵌入 PSO 算法中。即两个种群各自进化寻优, 每间隔一定迭代次数 k 后, 两个种群发生一次互利共生或寄生行为。之后, 将两个种群中的粒子按照适应度函数值进行排序, 当一个种群的最优粒子高于另一个种群时, 粒子向种群最优和全局最优方向飞行。

互利共生操作使 2 个种群中的粒子相互共享并进行融合, 如果融合结果的适应度高于原来粒子, 则将其进行替换。寄生操作可使一个种群从另一个种群中吸收优秀粒子替换较差粒子。通过这 2 个操作能够对种群中较差的粒子进行淘汰, 提高算法的寻优能力。

需要注意的是, 本文只将 PSO 粒子分成了 2 个种群, 而没有分成 3 个或更多。其主要原因为: 分群的数量直接影响算法的计算复杂度, 考虑到任务调度的实时性要求, 应尽量缩短计算时间; 实验表明, 当将 PSO 粒子分成 2 个种群时, 算法的寻优性能有明显提高, 而分成 3 个或更多种群时, 性能只比 2 个种群时略微提升, 效果不明显, 但计算量成倍增加。综上所述, 本文选择分为 2 个种群。

本文利用共生机制改进 PSO 算法的步骤如下所述。

步骤 1 生态系统初始化。初始化两个粒子种群的控制变量, 如种群大小、最大迭代次数。

步骤 2 选定当前最优个体。提取出 PSO 迭代至此所



获得的当前全局最优个体 x^{best} 。

步骤 3 互利共生。设置 PSO 第 k 次迭代时,进入共生阶段。此时,分别从两个粒子种群中随机选择个体 x_j 与个体 x_i ,根据式(5)和式(6)进行互利共生。

$$x_i' = x_i + r' \cdot (x^{best} - ((x_i + x_j)/2)b_1) \quad (5)$$

$$x_j' = x_j + r'' \cdot (x^{best} - ((x_i + x_j)/2)b_2) \quad (6)$$

其中, r' 和 r'' 是 0 到 1 之间的随机数。 b_1 和 b_2 为从共生个体之间产生的互利因子,取值为 1 或 2,其中,1 表示部分受益而 2 表示全面受益,即这两个个体从共生行为中获得的好处相等或一方可能比另一方受益多。 $((x_i + x_j)/2)$ 定义了两种个体对生态系统的适应度。根据适应度函数 $f()$ 计算适应度值 $f(x_i')$ 和 $f(x_j')$,仅当 $f(x_i') > f(x_i)$ 时,利用式(5)将 x_i 更新为 x_i' ,仅当 $f(x_j') > f(x_j)$ 时,利用式(6)将 x_j 更新为 x_j' 。 x^{best} 表示到目前为止具有最大适应度值的个体。

步骤 4 寄生阶段。设置 PSO 第 $2k$ 次迭代时,进入寄生阶段。此时,在一个粒子种群中随机选择一个个体 x_i 作为一个寄生体,并且从另一个种群中随机选择一个个体 x_j 作为 x_i 的宿主,寄生体根据式(7)从宿主中吸收营养。

$$x_i' = x_i + r' \cdot (x^{best} - x_j) \quad (7)$$

其中, r' 在 $-1 \sim 1$ 内。寄生行为结束后,如果适应度值 $f(x_i') > f(x_i)$,那么 x_i' 将代替 x_i 。

重复步骤 2~步骤 4 直至达到停止条件。互利共生和寄生优化过程如算法 1 所示。

算法 1 互利共生/寄生优化过程

创建并初始化生态系统中个体的种群 $X = \{x_1, x_2, x_3, \dots, x_N\}$;

设定停止条件: 达到迭代次数 $N=500$ 次或连续 50 代获得的解相同;

设定 x^{best} 为当前 PSO 获得的最优解。

while 停止条件

//互利共生阶段

从 2 个粒子种群中随机选择 x_i 和 x_j ;

$f_1 \leftarrow 1$ or $2, f_2 \leftarrow 1$ or 2 ;

$r' \leftarrow \text{rand}(0,1), r'' \leftarrow \text{rand}(0,1)$;

$x_i' = x_i + r' \cdot (x^{best} - ((x_i + x_j)/2)b_1)$;

$x_j' = x_j + r'' \cdot (x^{best} - ((x_i + x_j)/2)b_2)$;

if $f(x_i') > f(x_i)$ then $x_i \leftarrow x_i'$;

end if

if $f(x_j') > f(x_j)$ then $x_j \leftarrow x_j'$;

end if

//寄生阶段

从一个种群中随机选择 x_j 作为宿主,从另一种种群中随机选择 x_i 作为寄生体;

$r' \leftarrow \text{rand}(-1,1)$;

$x_i' \leftarrow x_i + r' \cdot (x^{best} - x_j)$;

if $f(x_i') > f(x_i)$ then $x_i \leftarrow x_i'$;

end if

//更新当前全局最优

if $f(x_i) > f(x^{best})$ then $x^{best} \leftarrow x_i$;

end if

end while

4 仿真及分析

为了测试本文方案的性能,利用 CloudSim 仿真器^[14]构建一个异构云计算场景。建立两个数据中心,每个中心分别包含两个主机。每个主机配备 20 GB 内存、1 TB 硬盘、10 Gbit/s 带宽和一个酷睿 i5 处理器,创建 20 个 VM,每个 VM 都具有 0.5 GB 内存和 1 Gbit/s 带宽。为了近似真实应用场景,设定 VM 的处理能力各不相同,在 1 000~10 000 MIPS 内以 1 000 为间隔随机取值。

仿真中,构建计算量为 50 亿~200 亿条指令不等的多个任务,分别组成具有 100、200、300、400、500、600、700、800、900、1 000 个任务集实例。另外,将各种任务集中不同大小的任务构建成两种分布,即均匀分布和正态分布。均匀分布中,任务集中的不同计算量大小的任务的数量相同。正态分布中,中等计算量的任务数量较多,小计算量和大计算量的任务数量较少。

将本文 SM-PSO 方案与传统 PSO 调度方案、参考文献[6]提出基于 AC-PSO 调度方案的和参考文献 [7] 提出的 SA-PSO 调度方案,在任务完成时间和响应时间方面进行比较。为了公平比较,各种方案中的 PSO 算法的参数设定见表 2。其中,通过算法的收敛性分析,本文算法在 150 次迭代后稳定,基本在 400 次迭代前就能获得最优解。所以,为了保证算法收敛,本文设定迭代次数 $N=500$ 。另外,对于互利共生阶段中的迭代间隔 k ,在考虑各自进化进程和共生操作次数下,设置 $k=50$ 。

图 1 和图 2 分别显示了各种方案,在不同数量的均匀分布和正态分布任务集下的平均完成时间,其中,数据为 10 次实验的平均值。可以看出,在多数场景下,本文 SM-PSO 方案的完成时间最小,特别是具有 300 个以上任务的实例中。这是因为本文方案寻优过程中获得了全局最

表 2 环境参数及 PSO 参数设定

参数	取值
环境参数	
数据中心数量	2
每个数据中心的 VM 数量	10
VM 的处理速度/MIPS	1 000~10 000
任务总数/个	100~1 000
每个任务的长度	50 亿~200 亿条指令
PSO 参数	
粒子种群大小	100
自我学习因子 c_1	2
社会学习因子 c_2	2
动态惯性权重 w	0.4~0.9
迭代次数 N	500
迭代间隔 k	50

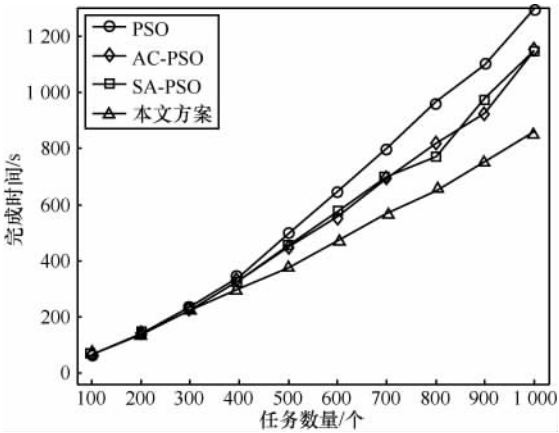


图 1 完成时间(均匀分布)

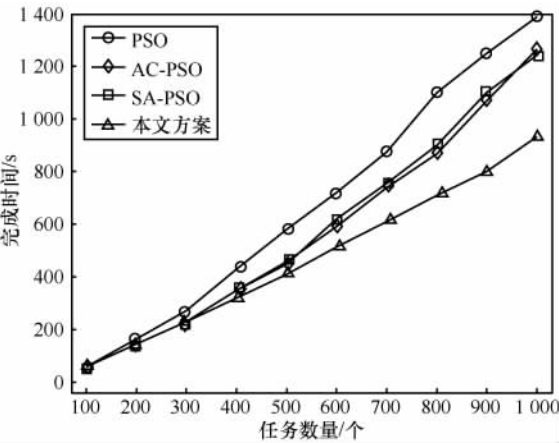


图 2 完成时间(正态分布)

优解,合理分配了云计算任务,充分利用了 VM 处理器,使总体完成时间最短。

图 3 和图 4 分别显示了各种方案在不同数量的均匀

分布和正态分布任务集下的平均任务响应时间。任务响应时间是任务调度系统实时性的判断标准之一。可以看出,本文 SM-PSO 方案具有较少的响应时间,这是因为本文方案能够快速地收敛到最优解。

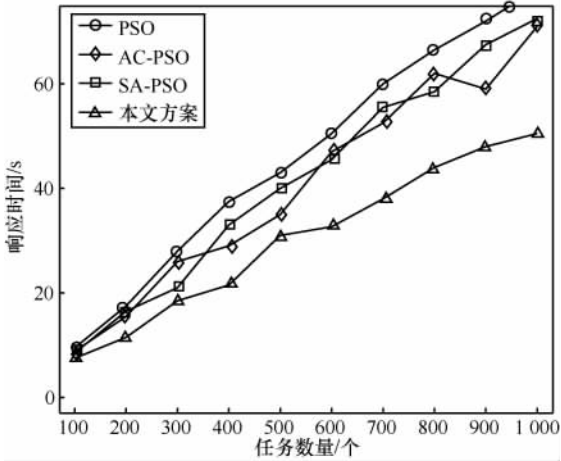


图 3 响应时间(均匀分布)

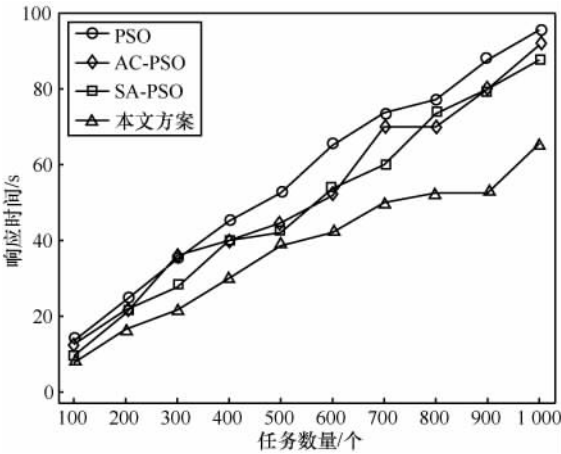


图 4 响应时间(正态分布)

图 5 描述了本文 SM-PSO 方案和 SA-PSO 方案在具有 500 个均匀分布的任务集场景下,求解最优调度方案的收敛性。可以看出,两种算法都能够在搜索开始时迅速收敛到较优解。但在后续寻找最优解过程中,本文算法能够快速收敛。在第 150 次迭代后,寻找到解的适应度值几乎稳定,而 SA-PSO 方案要在 350 次迭代后才稳定。另外,可以看出,本文 SM-PSO 方案获得的最优解的适应度函数值低于 SA-PSO 算法,这表明本文方案获得的最优解的质量较高。这是因为本文方案中通过互利共生和寄生机制提高了粒子的多样性,使其能够快速搜索到全局最优解。

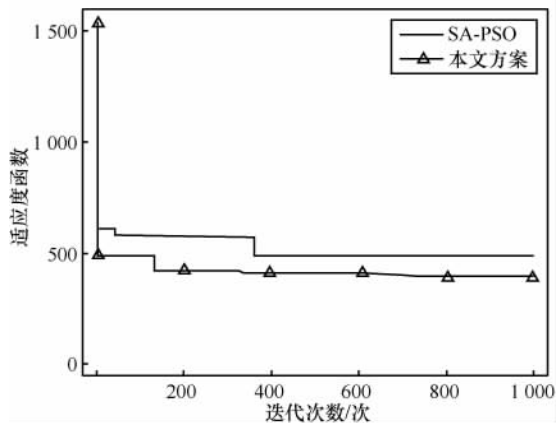


图5 算法收敛性(500个均匀分布任务)

表3列出了本文方案和SA-PSO方案在均匀分布的不同任务集数量上实验的数据统计,并采用 t 检验^[15]来表征算法性能。 t 检验是用 t 分布理论来推论差异发生的概率,从而比较两个平均数的差异是否显著。从统计表中获得的对应临界值为2.554,即当 t 值大于这个值时,两种算法存在显著性差异。

表3的统计分析表明,对于100~300个任务集的实际例,计算的 t 值要小于2.554(临界 t 值),这表明两种方案之间的性能差异并不显著。对于400~1 000条数据实例,计算得到的 t 值高于2.554,这意味着本文方案的性能优于SA-PSO方案。另外,样本数越大, t 值越大,说明本文方案更适合处理较大任务集的分配。

本文方案的优越性能归功于互利共生和寄生机制,互利共生阶段中的互利因素机制使搜索过程穿过最佳解区域来增强搜索能力。寄生阶段通过排除较差解并引入较优解来防止过早收敛。共生机制增强了算法的全局探索能力,使其不仅仅关注局部最佳解区域。

5 结束语

本文基于生物共生机制提出一种改进型PSO算法(SM-PSO),用于求解云计算最优任务调度方案。通过模拟生物的互利共生和寄生关系来提高传统PSO算法的粒子多样性,避免陷入局部最优,并提高了收敛速度。利用CloudSim构建云环境进行实验,以任务流完成时间和响应时间为性能指标,将本文算法和现有的几种改进型PSO算法进行比较。结果表明,本文算法能够快速获得任务调度的最优解。在不同任务场景下,获得了最短的完成时间,且随着任务量的增加,本文方案的性能优势越来越明显。

在今后工作中,将进一步改善所采用的共生机制,考虑加入寄生行为中寄生生物的免疫逃避机制,提高算法搜索能力。另外,对于云任务调度,将在目标函数中添加其他因素,如功耗等。

参考文献:

- [1] 张鹏,王桂玲,徐学辉. 云计算环境下适于工作流的数据布局方法[J]. 计算机研究与发展, 2013, 50(3): 636-647.
ZHANG P, WANG G L, XU X H. A data placement approach for workflow in cloud [J]. Journal of Computer Research and Development, 2013, 50(3): 636-647.
- [2] 刘丹琦,于炯,英昌甜. 云计算环境下多有向无环图工作流的节能调度算法[J]. 计算机应用, 2013, 33(9): 2410-2415.
LIU D Q, YU J, YING C T. Energy efficient scheduling for multiple directed acyclic graph in cloud computing [J]. Journal of Computer Applications, 2013, 33(9): 2410-2415.
- [3] 刘少伟,孔令梅,任开军,等. 云环境下优化科学工作流执行性能的两阶段数据放置与任务调度策略 [J]. 计算机学报, 2011, 34(11): 2121-2130.

表3 完成任务集(均匀分布)的时间/s

任务集大小	SA-PSO			本文方案			t 检验
	最佳值	最差值	平均值	最佳值	最差值	平均值	t 值
100	66.6	72.1	64.4	59.8	73.0	64.8	0.15
200	111.5	145.9	129.9	124.6	150.3	138.5	0.52
300	196.9	243.1	217.7	185.7	229.6	212.4	0.83
400	289.4	353.0	326.0	292.1	307.1	301.2	3.46
500	426.0	507.5	451.0	353.4	408.1	378.5	8.01
600	502.7	617.3	556.5	440.8	509.8	468.4	6.16
700	623.9	782.2	693.5	510.2	628.7	570.4	5.70
800	648.5	791.5	778.2	606.7	702.2	654.0	3.98
900	893.1	1 044.9	988.4	686.3	801.2	760.9	11.64
1 000	1 062.0	1 259.7	1 156.2	819.4	907.5	864.7	13.33

- LIU S W, SUN L M, REN K J, et al. A two-step data placement and task scheduling strategy for optimizing scientific workflow performance on cloud computing platform [J]. Chinese Journal of Computers, 2011, 34(11): 2121-2130.
- [4] LI W, WU J, ZHANG Q, et al. Trust-driven and QoS demand clustering analysis based cloud workflow scheduling strategies[J]. Cluster Computing, 2014, 17(3): 1-18.
- [5] 武善玉, 张平, 甓李方. 云制造系统中基于粒子群优化的多任务调度 [J]. 华南理工大学学报: 自然科学版, 2015, 43(1): 105-110.
- WU S Y, ZHANG P, YING L F. Multi-task scheduling based on particle swarm optimization in cloud manufacturing systems[J]. Journal of South China University of Technology(Natural Science Edition), 2015, 43(1): 105-110.
- [6] THANUSHKODI K. Hybrid intelligent algorithm [improved particle swarm optimization (PSO) with ant colony optimization (ACO)] for multiprocessor job scheduling[J]. Scientific Research & Essays, 2012, 7(20): 1935-1953.
- [7] KIM Y G, LEE M J. Scheduling multi-channel and multi-timeslot in time constrained wireless sensor networks via simulated annealing and particle swarm optimization[J]. IEEE Communications Magazine, 2014, 52(1): 122-129.
- [8] 刘冬华, 甘若迅, 樊锁海, 等. 基于捕食策略的粒子群算法求解投资组合问题[J]. 计算机工程与应用, 2013, 49(6): 253-256.
- LIU D H, GAN R X, FAN S H, et al. Particle swarm optimization based on predatory search for portfolio investment[J]. Computer Engineering and Applications, 2013, 49(6): 253-256.
- [9] XIA H. An improved particle swarm optimization based on biological chemotaxis [J]. Advanced Materials Research, 2014, 15(10): 737-740.
- [10] 石星宏, 张仲荣, 王博, 等. 基于生物寄生行为的双种群粒子群算法在 VRP 中的应用[J]. 兰州交通大学学报, 2012, 31(3): 65-68.
- SHI X H, ZHANG Z R, WANG B, et al. Particle swarm optimization based on parasitic behavior for the VRP in the application[J]. Journal of Lanzhou Jiaotong University, 2012, 31(3): 65-68.
- [11] JIA Q, SEO Y. An improved particle swarm optimization for the resource-constrained project scheduling problem[J]. International Journal of Advanced Manufacturing Technology, 2013, 67(12): 2627-2638.
- [12] KOULINAS G, KOTSIKAS L, ANAGNOSTOPOULOS K. A particle swarm optimization based hyper-heuristic algorithm for the classic resource constrained project scheduling problem[J]. Information Sciences, 2014, 277(2): 680-693.
- [13] 温涛, 盛国军, 郭权, 等. 基于改进粒子群算法的 Web 服务组合[J]. 计算机学报, 2013, 36(5): 1031-1046.
- WEN T, SHENG G J, GUO Q, et al. Web services composition based on modified particle swarm optimization [J]. Chinese Journal of Computers, 2013, 36(5): 1031-1046.
- [14] BUX M, LESER U. DynamicCloudSim: simulating heterogeneity in computational clouds [J]. Future Generation Computer Systems, 2014, 46(4): 85-99.
- [15] 史小露, 孙辉, 李俊, 等. 具有快速收敛和自适应逃逸功能的粒子群优化算法[J]. 计算机应用, 2013, 33(5): 1308-1312.
- SHI X L, SUN H, LI J, et al. Particle swarm optimization algorithm with fast convergence and adaptive escape [J]. Journal of Computer Applications, 2013, 33(5): 1308-1312.

[作者简介]



王琳杰(1981-),男,铜仁学院副教授,主要研究方向为计算机网络、密码学。