第 37 卷 第 5 期
2022 年 10 月

天津科技大学学报
Journal of Tianjin University of Science & Technology

Vol. 37 No. 5
Oct. 2022

DOI:10.13364/j.issn.1672-6510.20210195
数字出版日期：2022–03–31；数字出版网址：http://kns.cnki.net/kcms/detail/12.1355.N.20220329.1146.002.html

基于天牛须遗传混合算法的大规模云任务调度

张 锐[1]，王随园[1]，张春霞[1]，王建喜[2]，郭洪飞[3]
（1. 天津科技大学电子信息与自动化学院，天津 300222；2. 航天银山电气有限公司，珠海 519090；
3. 暨南大学物联网与物流工程研究院，珠海 519070）

摘 要：针对目前云计算环境下大规模任务调度效率低下的问题，提出一种基于天牛须搜索算法与遗传算法混合的优化算法. 为改善遗传算法优化结果不稳定和局部搜索能力差的问题，采用新型交叉方式，动态改变变异概率，拓展天牛搜索方向，并采用精英解保留策略. 最后基于 CloudSim 仿真平台，在不同任务规模调度情况下，将混合算法和其他 3 种算法在相同环境下进行仿真调度. 实验结果表明：该混合算法比其他算法得到更优质解，相比于遗传算法，小规模任务调度优化结果提高 7.87%，大规模任务调度提高 30.23%，是一种高效的云任务调度优化算法.

关键词：云计算；任务调度；遗传算法；天牛须搜索算法
中图分类号：TP18 **文献标志码**：A **文章编号**：1672-6510（2022）05-0044-06

Large-Scale Cloud Task Scheduling Based on Beetle Antennae Search Algorithm Hybrid Genetic Algorithm

ZHANG Rui[1]，WANG Suiyuan[1]，ZHANG Chunxia[1]，WANG Jianxi[2]，GUO Hongfei[3]
（1. College of Electronic Information and Automation，Tianjin University of Science & Technology，
Tianjin 300222，China；
2. Aerospace Yinshan Electric Co.，Ltd.，Zhuhai 519090，China；
3. Institute of Physical Internet，Jinan University，Zhuhai 519070，China）

Abstract：Aiming at the low efficiency of large-scale cloud task scheduling in the current cloud computing environment，an optimization algorithm based on beetle antennae search and genetic algorithm is proposed in this article. In order to improve the unstable optimization results and poor local search ability of genetic algorithm，a new crossover method is adopted，and the mutation probability is changed dynamically. Then the search direction of beetle is expanded，and the elite solution retention strategy is adopted. Finally，based on CloudSim，the hybrid algorithm and the other three algorithms are simulated and scheduled in the same environment in the case of different task scale scheduling. The experimental results show that the hybrid algorithm obtains better solutions than other algorithms. Compared with genetic algorithm，the optimization result of small-scale task scheduling is improved by 7.87% and that of large-scale task scheduling is improved by 30.23%，it proves an efficient cloud task scheduling optimization algorithm.

Key words：cloud computing；task scheduling；genetic algorithm；beetle antennae search algorithm

云计算自 2006 年首次被 Google 公司提出以来，众多大型互联网公司相继进入云计算市场，其中 Amazon 公司和 Microsoft 公司相关云计算业务营收已达百亿美元规模[1]. 云计算之所以发展如此迅速，是由于云计算拥有传统计算不具备的超大规模、虚拟化、高可靠性等优点[2].

目前，国内外使用不同优化策略求解云计算任务调度的算法研究有很多. 文献[3]提出改进猫群优化算法，算法对云计算任务调度的时间、能耗和成本目标的优化效果显著. 文献[4]针对雾计算中任务调度

收稿日期：2021–10–09；修回日期：2021–11–30
基金项目：广州市科技计划资助项目（202002030321）；广东省研究生教育创新计划资助项目（82620516）
作者简介：张 锐（1979—），男，辽宁人，副教授；通信作者：郭洪飞，副教授，ghf-2005@163.com

问题，通过将遗传算法与蚁群算法相结合，在减少调度时间的同时提高了内存分配效率. 文献[5]将罗尔斯正义分配伯格模型和博弈算法运用到海量云计算资源调度上，提出的算法显著提高了整体服务质量. 文献[6]将微生物遗传算法和改进粒子群算法结合，加入动态惯性权重策略，增强算法搜索能力，有效减少任务完成时间和执行成本. 文献[7]建立新的资源分配模型，并采用珊瑚礁优化算法，有效改善负载均衡问题的同时降低了执行成本. 文献[8]针对工作流云任务调度优化问题，提出匹配和多轮匹配算法，优化结果可有效改善任务完成时间以及资源利用率. 文献[9]提出一种基于模糊自防御算法的云计算多目标任务调度优化算法，在最大完成时间、截止期违规率和虚拟机资源利用率方面目标优化效果明显.

现有文献大多没有考虑大规模云任务调度算法的性能表现，随着云计算服务应用场景不断扩展以及云计算用户激增，如何快速合理分配海量云任务至计算资源成为目前研究的重要问题. 云计算环境下任务调度研究是 NP-hard 问题[10]，运用传统精确求解方法，计算量巨大且难以在可接受时间内得到最优解.

文献[11]通过测试函数验证了全局搜索的遗传算法和个体寻优天牛须搜索算法结合的优越性. 本文将该思想应用于云任务调度领域，根据迭代进程动态改变交叉方式，既加快前期寻优能力又避免算法早熟；采用精英解保留策略，保护最优解不被交叉变异破坏；调整混合策略，增强算法局部寻优能力. 4 种算法的对比实验结果表明：本文提出的算法在任何规模云任务调度中，尤其是大规模云任务调度中，性能表现优越.

1　云计算模型

在云计算中，独立云任务调度的情况下：当用户将云任务提交至服务器时，云任务会被 Map/Reduce 模型分割成若干独立、不存在依赖关系的子任务，然后这些子任务根据调度算法被分配到不同虚拟机中执行. 云计算调度模型如图 1 所示，所需研究问题主要分为一级调度问题和二级调度问题. 如何将不同任务分配到不同虚拟机，被称为一级调度问题；如何在不同容量物理机上分配不同性能的虚拟机，被称为二级调度问题. 在分配后的虚拟机中执行一级调度分配的任务，则整个云计算调度过程完成，本文研究一级调度问题.

根据本文所研究的问题，结合实际调度环境，作出如下假设：各云任务长度不同，且相互独立；每台虚拟机性能不同；调度任务个数远大于虚拟机个数；分配到一台物理机的所有虚拟机内存大小、带宽和计算速度等性能不能超过该物理机实际性能.

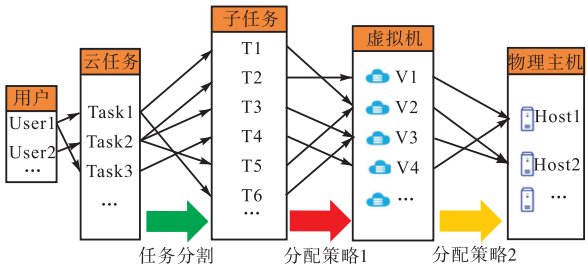

图 1　云计算调度模型

Fig. 1　Cloud computing scheduling model

任务执行时间主要与任务长度以及虚拟机运算速度相关，ETC 表示各任务分配到各虚拟机的执行时间，计算公式为

$$\mathrm{ETC}(i,j)=\frac{\mathrm{Task}_i}{\mathrm{Capacity}_j} \tag{1}$$

式中：$\mathrm{ETC}(i,j)$ 表示任务 i 分配到虚拟机 j 上的执行时间，Task_i 表示任务 i 的大小，$\mathrm{Capacity}_j$ 表示虚拟机 j 的运算速度.

$$\mathrm{Time}(j)=\sum_{i=1}^{n}\mathrm{ETC}(i,j) \tag{2}$$

$$\mathrm{cTime}=\max(\mathrm{Time}(j)) \tag{3}$$

$$\mathrm{Fitness}=\frac{1}{\mathrm{cTime}} \tag{4}$$

式(2)—式(4)中：$\mathrm{Time}(j)$ 表示虚拟机 j 执行完所有分配任务所消耗的时间；cTime 表示所有虚拟机中完成任务花费时间最长虚拟机的运行时间，即任务完成所需时间；Fitness 表示个体适应度.

适应度越高，则任务完成所需时间越短，适应度决定种群进化方向和个体被选择的概率.

2　算法设计

2.1　遗传算法

遗传算法(GA)最初是由 Holland 提出的一种优化算法，主要针对非线性优化问题. 遗传算法是目前最常见且有效的优化算法之一，通过仿照生物进化过程中优胜劣汰的进化规律，算法随机选择两个父辈个体，并随机挑选各自基因片段进行交换，同时，种群中少数个体由于某个或多个基因变异，同样产生新的染色体；然后，根据每个个体对环境的适应程度，决定个体繁殖概率，这样拥有优质基因片段的高适应度个体比例不断增加，而适应度低的个体随着迭代的进

行则被淘汰；最后，经过若干次交叉变异后，留下最优个体，即为全局最优解或者近似全局最优解.

但传统遗传算法存在一些不确定性，例如：容易过早收敛，导致后续迭代不能持续优化；局部搜索能力差，遗传算法擅长全局搜索最优解[11]，传统遗传算法应用在云计算任务调度中会导致任务分配结果十分不稳定. 一方面，云任务分配到虚拟机存在海量可行方案，尤其是面对大规模云任务调度时，极易发生“维数爆炸”问题；另一方面，遗传算法自身的局部搜索能力不强，当遇到大规模任务调度时，较难得到最优解.

2.2　天牛须搜索算法

天牛须搜索算法(BAS)是通过模仿天牛的觅食过程开发出的一种算法. 通过计算左右两端适应度，持续更新天牛个体解的位置，寻找最优解. 假设天牛个体解表示为 $X=(X_1, X_2, X_3, \cdots, X_n)$，则天牛左右两须解 X_l 和 X_r 分别为

$$\begin{cases} X_l = X + d\boldsymbol{A} \\ X_r = X - d\boldsymbol{A} \end{cases} \tag{5}$$

式中：d 表示天牛两须中间位置与天牛须的距离；$\boldsymbol{A}$ 表示随机生成的方向不确定的单位向量，含义为天牛在解空间的方向.

天牛个体当前位置 X^t 通过不断迭代，持续判断左右两须的适应度并向适应度高的方向移动，下一个位置 X^{t+1} 为

$$X^{t+1} = X^t + \delta \boldsymbol{A} \times \mathrm{sign}\left[f(X_l) - f(X_r)\right] \tag{6}$$

式中：δ 表示天牛移动的步长，sign 决定天牛移动的方向.

2.3　遗传天牛须混合算法

针对传统遗传算法和天牛须搜索算法的优缺点，将两个算法进行改进，并将改进天牛须搜索算法作为改进遗传算法种群适应度优化的算子，将两个算法混合，提出 GA-BAS(genetic algorithm-beetle antennae search algorithm)算法，算法流程如图 2 所示.

2.3.1　算法编码

本研究属于独立云任务调度问题，不存在优先关系. 因此，本文采用实数直接编码方式. 一条染色体表示一种分配策略，染色体中基因个数代表任务调度过程中待分配子任务的个数，每个基因位置上的数字表示分配给该任务虚拟机的序号. 染色体编码示例如图 3 所示，该示例中 T1—T8 表示共计 8 个云任务；V1—V5 表示共计 5 台虚拟机，即染色体中每个基因可取的数值为 1 ~ 5. 因此，该示例表示编号 1—8 的云任务分别分配给 2、3、1、4、5、2、3、5 号虚拟机.

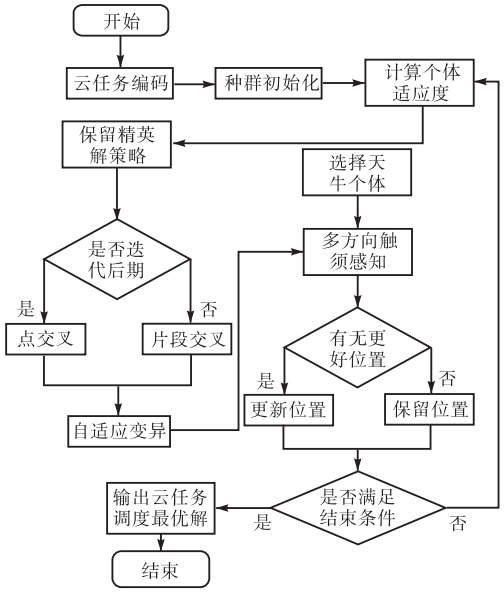

图 2　GA-BAS 云任务调度算法流程图

Fig. 2　Flow chart of cloud task scheduling algorithm

V2	V3	V1	V4	V5	V2	V3	V5
T1	T2	T3	T4	T5	T6	T7	T8

图 3　编码示例

Fig. 3　Coding example

2.3.2　选择算子

本文的选择算子采用保留精英解策略，即：种群每次迭代前，适应度最高的个体不进行任何交叉变异操作，直接替换适应度最低的个体，保留到下一代中. 这样，既可以保护种群中最优解不被交叉变异破坏，又可以提高种群平均适应度.

2.3.3　交叉算子

交叉操作是传统遗传算法中产生新个体最重要的方法. 交叉操作使不同个体中各自优秀的基因相互结合，充分挖掘和组合优质解，随着迭代进行，根据适者生存原则，不断产生更优质的解，直至迭代结束，产生当前最优解. 因此，交叉操作深刻影响算法性能和收敛速度. 本文交叉操作选用片段交叉和点交叉相结合的方法，如图 4 所示.

相比于片段交叉，单点交叉对解的改变更小，改变幅度更小，有利于找到最优解. 当算法需要分配大量云任务至虚拟机时，在算法迭代的初始阶段，选用片段交叉的方法，可以有效加快算法迭代速度，在算法迭代后半段，采用点交叉的方法，既可以改善算法局部搜索能力，也可以有效保护优质解不被破坏.

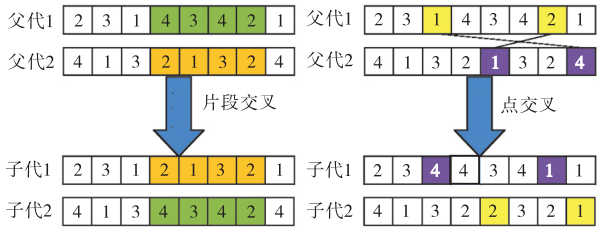

图 4　交叉操作示意图

Fig. 4　Schematic diagram of cross operation

2.3.4　变异算子

变异对遗传算法进化和产生新个体有辅助作用，在保持种群多样性和避免早熟方面有一定作用. 在一般情况下，个体适应度越低，则该个体变异概率越大；相反，个体适应度越高，则该个体变异概率越小. 本文参考文献[12]中自适应变异概率方式：首先，随机选择一个基因位置；然后，产生一个随机数，该随机数最大值不超过虚拟机个数，用产生的随机数代替该基因原有编号，产生新的个体. 自适应变异公式为

$$p=\begin{cases} p_{\max}-\dfrac{(p_{\max}-p_{\min})(f-f_{\text{avg}})}{f_{\max}-f_{\text{avg}}} & f\geqslant f_{\text{avg}} \\ p_{\max} & f<f_{\text{avg}} \end{cases} \tag{7}$$

式中：p 表示当前个体变异概率，$p_{\max}$ 表示种群进化过程中每个个体最大的变异概率，$p_{\min}$ 表示种群进化过程中每个个体最小的变异概率，f 表示当前被选择个体的适应度，$f_{\max}$ 表示种群中最优个体的适应度，f_{avg} 表示种群中所有个体的平均适应度.

2.3.5　混合策略

在算法执行完选择、交叉和变异后，混合 BAS 算法：随机选择种群中一部分个体经过 BAS 算法寻优，然后将产生的新种群替代旧种群.

传统 BAS 算法只有左右两个方向选择，很大程度限制了搜索范围，本文拓展天牛搜索方向，即：随机产生 n 个右方向的单位向量，则对应产生 n 个左方向的单位向量，如图 5 所示，改进后天牛须搜索算法可以选择多个方向，更有利于找到浓度高的信息点.

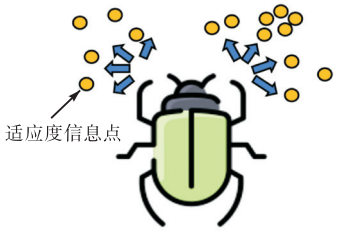

图 5　BAS算法模型

Fig. 5　BAS algorithm model

因此，本文通过改变天牛探索方向的个数，调节天牛须算法搜索范围和寻优能力，建立多方向感知模型，计算各个方向的适应度，反馈给天牛搜索模型，决定天牛下次移动方向，单位向量 $\boldsymbol{A}$ 计算公式为

$$\boldsymbol{A}=\frac{\text{rand}(n,D)}{\left\|\text{rand}(n,D)\right\|} \tag{8}$$

式中：rand 表示随机数函数，n 为天牛右须探索方向的个数，D 为个体解的维度.

为了避免 BAS 算法寻优搜索使得种群多样性降低，导致最优解过早出现，本文设定每次迭代参与天牛须寻优搜索的个体数量占种群规模的 50%. 若 BAS 算法搜索到更优位置，则更新位置，反之，保留当前位置. 最后判断是否达到迭代次数，若达到迭代次数，则输出种群中最优个体，结束算法，反之，返回种群适应度计算，继续循环.

3　实验仿真

为了测试 GA-BAS 算法的性能，本文选取传统遗传算法与 GA-BAS 算法进行对比，使用 CloudSim 3.0 云计算仿真平台对云任务调度过程进行仿真. 具体仿真实验的配置条件为：Windows 10 操作系统，Intel i5-8750 处理器，8 GB 内存，1 TB 硬盘.

3.1　仿真环境设置

为了保证各算法能够进行合理比较，且仿真过程能顺利进行，要在 CloudSim 运行仿真调度前设置仿真环境（物理机和虚拟机参数、云任务长度和传输数据大小）以及调度算法各参数. 本文云计算各运行条件设置与文献[13]相同. 作为云计算中最底层的计算资源，物理机性能参数直接影响云计算中调度任务的完成时间，本文设置两种类别物理机，物理机参数见表 1. 物理机性能参数设置完成后，需要在物理机上分配若干个性能不同的虚拟机，虚拟机参数见表 2.

表 1　物理机参数

Tab. 1　Physical machine parameters

物理机类别	核心数	内存/GB	存储/TB	带宽/(Mbit·s^{-1})
1	16	2	1	1 000
2	32	4	1	2 000

表 2　虚拟机参数

Tab. 2　Virtual machine parameters

虚拟机类别	核心数	内存/GB	处理速度/MIPS	带宽/(Mbit·s^{-1})
1	1	1	1 000	1 000
2	1	2	2 000	1 500
3	1	1	3 000	2 000
4	1	2	4 000	2 500

注：MIPS 表示每秒执行百万条指令

云任务的长度直接决定了云计算运行过程中计

算量的大小，本文云任务采用随机生成方式，云任务长度为 5 000 ~ 10 000.

3.2　结果与分析

本文设置不同规模云任务量，并经过多次仿真实验，分析在不同规模任务量下算法的性能表现. 在相同仿真实验条件下，将 GA-BAS 算法和其他 3 种对比算法通过各自分配策略将云任务分配到虚拟机上，算法参数见表 3（SA 为模拟退火算法）.

表 3　算法参数

Tab. 3　Algorithm parameters

算法名称	种群规模	交叉概率	变异概率	迭代次数/次	
				小规模任务	大规模任务
SA	60	无	无	100	200
BAS	60	无	无	100	200
GA	60	0.8	0.1	100	200
GA-BAS	60	0.8	自适应	100	200

3.2.1　小规模任务调度场景

设置 200 个云任务分配给 40 台虚拟机，迭代 100 次. 实验结果如图 6 所示.

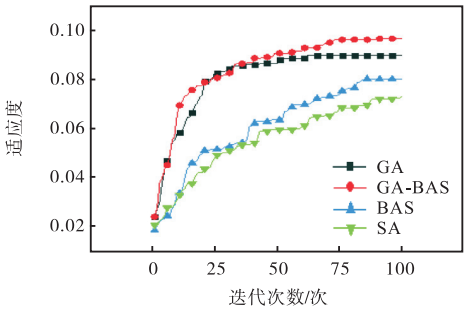

图 6　小规模任务调度

Fig. 6　Small scale task scheduling

在小规模任务调度情况下，相比于 SA 算法，BAS 算法有更多局部搜索方向，也就有更强局部寻优能力，因此 BAS 算法优化结果比 SA 算法更好. 但是，由于 BAS 算法和 SA 算法全局搜索能力差，优化结果明显低于其他两种全局优化算法.

在迭代初始阶段，GA 算法和 GA-BAS 算法差别不大，随着迭代进行，GA 算法在第 64 代得到最优解；而 GA-BAS 算法得益于交叉策略动态改变和混合局部搜索能力强的天牛须搜索算法，在第 93 代得到最优解.

根据表 4 可知：通过多次小规模任务调度仿真实验，相比于 GA 算法，GA-BAS 算法优化结果（迭代 100 次）的最小值、最大值和平均值分别提高了 10.47%、4.30% 和 7.87%.

表 4　小规模任务调度迭代 100 次结果对比

Tab. 4　Comparison of results of 100 iterations of small scale task scheduling

算法名称	适应度		
	最小值	最大值	平均值
SA	0.071	0.075	0.073
BAS	0.077	0.081	0.079
GA	0.086	0.093	0.089
GA-BAS	0.095	0.097	0.096

3.2.2　大规模任务调度场景

设置大规模任务调度，即：将 20 000 个云任务分配到 400 台虚拟机上，由于任务调度规模巨大，因此将所有算法迭代次数提升至 200 次. 仿真实验结果如图 7 所示.

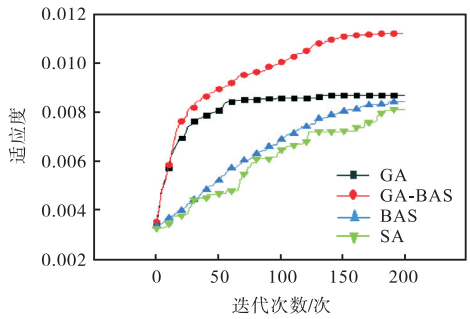

图 7　大规模任务调度

Fig. 7　Large scale task scheduling

由图 7 可知：在大规模任务调度情况下，BAS 算法、SA 算法和 GA 算法优化结果相近. 其中，GA 算法迭代曲线和小规模任务调度情况相似，收敛速度过快，在迭代第 57 次后，优化结果提升不大，并且在第 136 代得到最优解，陷入了局部最优解，导致后续迭代的计算资源被浪费. 而 GA-BAS 算法迭代曲线呈缓慢收敛态势，并在第 197 代得到最优解.

在迭代过程中，GA-BAS 算法每一代最优解都要优于 GA 算法最优解，这是由于大规模任务调度的调度方案数量巨大，这也就导致种群中个体需要局部寻优方向增多，而传统 GA 算法的交叉变异操作并不擅长局部搜索. 一方面，GA-BAS 算法加入改进 BAS 算法后，扩大了算法搜索方向，能更好适应大规模任务调度；另一方面，GA-BAS 算法在迭代后期采用点交叉方式，使得个体基因变化幅度变小，相比于片段交叉，不易错过更优质局部解.

由表 5 可知：经过多次大规模任务调度仿真实验，相比于 GA 算法，GA-BAS 算法优化结果（迭代 200 次）的最小值、最大值和平均值分别提高了 38.46%、30.77% 和 30.23%，优化效果提升明显.

表 5　大规模任务调度迭代 200 次结果对比

Tab. 5　Comparison of results of 200 iterations of large scale task scheduling

算法名称	适应度		
	最小值	最大值	平均值
SA	0.007 2	0.008 5	0.008 1
BAS	0.008 2	0.008 6	0.008 4
GA	0.007 8	0.009 1	0.008 6
GA-BAS	0.010 8	0.011 9	0.011 2

从以上两种不同任务规模仿真实验可知，本文提出的 GA-BAS 混合算法优化结果的平均值、最小值和最大值均优于其他 3 种对比算法，尤其在大规模云任务调度场景下，GA-BAS 算法优化结果显著优于其他对比算法. 得益于天牛须搜索算法强大的局部寻优能力，使 GA-BAS 算法拥有更强局部搜索能力和更稳定的优化结果，弥补了传统遗传算法局部搜索能力不强和容易过早收敛的问题；保留精英解策略可以有效保护优质解不被破坏，提升种群整体适应度；提出新型交叉策略，在迭代前期采用片段交叉，加速算法收敛速度，在迭代后期采用点交叉方式，不仅增强算法在迭代早期的寻优能力，而且有效避免算法在迭代后期过早收敛，同时保护优质解不被大范围破坏，增加了算法局部寻优能力；引入自适应变异方式，根据迭代过程，动态调整变异概率，提升种群多样性，避免算法早熟.

4　结　语

在云计算环境下，本文针对大规模任务调度效率低下的问题，同时考虑传统遗传算法应用于云任务调度中的优缺点，将传统遗传算法进行多方面改进，并加入改进天牛须搜索算法，提出 GA-BAS 算法，对云计算环境下任务调度问题进行优化. 实验结果表明：该算法能够有效减少大规模任务调度的完成时间，是一种高效云任务调度算法. 未来研究方向可以完整考虑整个云计算所有调度过程，加入 Map/Reduce 模型分割任务过程，同时增加二级调度，即：物理机分配虚拟机过程.

参考文献：

[1] 周悦芝，张迪. 近端云计算：后云计算时代的机遇与挑战[J]. 计算机学报，2019，42(4)：677–700.

[2] 文婷婷. 云服务平台中计算资源管理的研究与应用[D]. 成都：西南交通大学，2019.

[3] MANGALAMPALLI S，SWAIN S K，MANGALAMPALLI V K. Multi objective task scheduling in cloud computing using cat swarm optimization algorithm[J]. Arabian journal for science and engineering，2021，10：1–10.

[4] 王志刚，赵传信. 基于遗传蚁群算法的雾计算任务调度研究[J]. 计算机应用与软件，2021，38(8)：291–297.

[5] 孙红，赵娜. 基于改进伯格博弈模型的云计算任务调度[J]. 控制工程，2020，27(3)：500–506.

[6] 孙长亚，王向文. 基于 MGA-PSO 的云计算多目标任务调度[J]. 计算机应用与软件，2021，38(6)：212–218.

[7] WAN S Z，QI L X. An improved coral reef optimization-based scheduling algorithm for cloud computing[J]. Journal of mathematics，2021，2021：5532288.

[8] ZHU Q H，TANG H，HUANG J J，et al. Task scheduling for multi-cloud computing subject to security and reliability constraints[J]. IEEE/CAA Journal of automatica sinica，2021，8(4)：848–865.

[9] GUO X Y. Multi-objective task scheduling optimization in cloud computing based on fuzzy self-defense algorithm[J]. Alexandria engineering journal，2021，60(6)：5603–5609.

[10] 熊聪聪，高萌，赵青，等. 一种基于改进二进制粒子群的工作流云调度算法[J]. 天津科技大学学报，2021，36(4)：61–66.

[11] 赵玉强，钱谦，周田江，等. 天牛须搜索与遗传的混合算法[J]. 小型微型计算机系统，2020，41(7)：1438–1445.

[12] 曹阳，刘亚军，俞琰. 基于遗传–蚁群算法的云计算任务调度优化[J]. 吉林大学学报(理学版)，2016，54(5)：1077–1081.

[13] 马学森，谈杰，陈树友，等. 云计算多目标任务调度的优化粒子群算法研究[J]. 电子测量与仪器学报，2020，34(8)：133–143.

责任编辑：周建军