doi:10.3969/j.issn.1001-4616.2024.01.011

一种基于改进人工鱼群的云计算任务调度算法

孙　鉴[1,2],吴佳伟[1],刘陈伟[1],武　涛[1]

(1.北方民族大学计算机科学与工程学院,宁夏 银川 750021)
(2.北方民族大学图像图形智能处理国家民委重点实验室,宁夏 银川 750021)

[**摘要**]　为提高云计算任务调度的效率,减少系统执行任务的最大完工时间以及成本,本文提出一种改进的人工鱼群任务调度算法(improved artificial fish swarm algorithm,IAFSA). 首先,将反向学习策略应用于种群初始化和鱼群的行为选择中,以提高改进人工鱼群算法在迭代中的收敛速度和种群多样性. 其次,将自适应全局-局部记忆机制引入到标准 AFSA 算法的觅食行为中,以进一步提高勘探能力. 最后,增加了基于平均适应度的行为选择机制,以提供更合理的行为选择,减少算法的复杂性. 通过使用 CloudSim 平台进行实验验证,分别测试在不同任务规模下 IAFSA 的算法效能. 实验结果表明,改进人工鱼群算法在降低系统任务最大完工时间和成本上均表现出了显著的优势.

[**关键词**]　云计算,任务调度,人工鱼群,CloudSim,最大完工时间,成本
[**中图分类号**]TP391　[**文献标志码**]A　[**文章编号**]1001-4616(2024)01-0091-12

A Cloud Computing Task Scheduling Algorithm Based on Improved Artificial Fish Warm

Sun Jian[1,2],Wu Zhuiwei[1],Liu Chenwei[1],Wu Tao[1]

(1.School of Computer Science and Engineering,North Minzu University,Yinchuan 750021,China)
(2.The Key Laboratory of Images & Graphics Intelligent Processing of State Ethnic Affairs Commission,
North Minzu University,Yinchuan 750021,China)

Abstract:In order to improve the efficiency of cloud computing task scheduling and reduce the makespan and cost of tasks,this paper proposes an improved artificial fish swarm task scheduling algorithm(IAFSA). Firstly,the opposition-based learning strategy was applied to the population initialization and the behavior selection of the fish swarm to improve the convergence speed and population diversity of the improved artificial fish swarm algorithm in iterations. Secondly,the adaptive global-local memory mechanism was introduced into the foraging behavior of the standard AFSA algorithm to further improve the exploration ability. Finally,an action selection mechanism based on average fitness was added to provide more reasonable action selection and reduce the complexity of the algorithm. By using CloudSim platform for experimental verification,the algorithm efficiency of IAFSA under different task scales was tested respectively. The experimental results show that the improved artificial fish swarm algorithm has significant advantages in reducing the maximum completion time and cost of the system task.

Key words:cloud computing,task scheduling,AFSA,CloudSim,makespan,cost

随着互联网产业的不断发展,对计算和大规模存储资源的需求也随之快速增长. 因此,云计算因其作为软件即服务(SaaS)、基础设施即服务(IaaS)和平台即服务(PaaS)提供给用户的高性能计算服务和设施而受到关注[1]. 云计算中有许多热门问题需要研究,如数据安全、可扩展存储管理、移动云、任务调度等,其中如何实现对任务进行一个合理的调度,以此来减少任务计算时延和成本的降低成为云计算研究中的重点[2].

收稿日期:2023-05-21.
基金项目:国家自然科学基金项目(62062002)、宁夏科学自然基金项目(2022AAC03289)、北方民族大学中央高校基本科研业务费专项资金项目(FWNX09)、北方民族大学校级一般项目(2021XYZJK01).
通讯作者:孙鉴,博士,副教授,研究方向:云计算、任务调度. E-mail:550177201@qq.com

目前,很多学者将传统的元启发式算法用于解决云计算任务调度问题,并在各自的研究角度上取得了一定的效果[3]. 如粒子群算法[4]、遗传算法[5]、蚁群算法[6]等. 但单一的元启发式算法往往容易出现局部搜索与全局搜索失衡,陷入局部最优解、求解时间长等问题. 面对实际任务调度中用户需求越来越动态多变的现状,单一元启发式算法难以广泛应用,实用性较差,故产生了将不同算法融合的混合式算法. 混合式算法是通过保留传统式启发算法部署简单的特点与启发式算法的高性能的同时,将不同的元启发式算法之间进行一个取长补短的结合,以达到改善算法性能、提高迭代产生解质量的目的. 例如文献[7]提出一种布谷鸟粒子群算法(cuckoo particle swarm optimization,CPSO),该算法将 CSA 和 PSO 两种算法混合在一起,既利用 CSA 算法的搜索随机性来扩大算法搜索范围,又可以利用 PSO 的学习因子提高算法的收敛性,从而有效地优化任务和资源的调度. 文献[8]提出一种混合粒子群遗传算法(hybrid particle swarm algorithm and genetic algorithm,PSO_PGA)[8],该算法通过遗传算法的交叉、变异机制扩大搜索解空间,以及利用粒子群的记忆机制将自身和全局最优粒子的飞行经验反馈给下一代粒子群来确保粒子群始终朝着优秀解的方向移动,实验表明,所提出的算法能提高解的质量和收敛性. 文献[9]提出一种鲸鱼遗传算法(whale genetic optimization algorithm,WGOA),该算法将 GA 算法的交叉和变异机制引入 WOA 算法,提升全局寻优的能力,避免 WOA 算法陷入局部最优解. 以上算法在仿真实验中往往比单一元启发式算法有更好的优化效果. 因此将混合式算法应用在云计算任务调度中具有一定研究意义与前景.

人工鱼群算法(artificial fish swarm algorithm,AFSA)是李晓磊等于 2002 年提出的一种仿生群智能优化算法[10],具有算法稳定性高、寻优速度快且易和其他算法相结合的优点[11]. 但 AFSA 算法中人工鱼的觅食行为会受限于视野和步长的大小[12],存在很大的随机性和盲目性,且每条人工鱼在迭代过程中都需要先执行觅食、聚群、追尾 3 种行为,再选取其中最优行为进行执行[10],从而导致算法的复杂性较高,运行速度较慢. 针对以上现象,文献[13]提出采用非线性递减函数动态地调整人工鱼的视野范围和步长,同时结合禁忌搜索算法来提升算法性能. 文献[14]提出一种人工鱼群与粒子群相结合的算法,该算法在人工鱼的行为上做了改进,引入了速度和惯性权重机制,提升了收敛速度和寻优结果. 文献[15]提出了双自适应人工鱼群优化算法,将高斯变异引入到人工鱼群算法中,同时进行自适应调整视野和步长大小,该算法有效地提高了寻优质量并避免了早熟现象. 但是以上改进都仍然受到人工鱼的视野、步长等因素的约束,使得寻优效率仍然不高、求解时间长.

本文将针对利用元启发式算法在云环境下得到任务调度的最优解展开讨论,为了寻找使得任务完成时间短、成本低的云任务调度方案,本文借鉴花朵授粉算法(flower pollination algorithm,FPA)[16]中的交叉授粉和自花授粉机制,在人工鱼群算法中引入了自适应全局-局部记忆策略,提出一种改进的人工鱼群算法,并在 CloudSim 上完成任务调度过程的模拟实验,验证了本文算法的高效性. 算法主要改进如下:

(1)将反向学习策略引入到人工鱼群种群初始化和行为选择中,提升初始种群质量,扩大人工鱼的搜索解空间.

(2)改进了鱼群的觅食行为,通过借鉴花粉算法中的交叉授粉和自花授粉行为,加强了人工鱼之间信息交流,使得人工鱼具有向全局最优位置和局部最优位置移动的能力,减少了搜索的盲目性,提高解的精度.

(3)为了减少算法的复杂性,提出了基于种群平均适应度的行为选择机制,对不同个体的人工鱼进行合理的行为分配,减少重复计算,提升鱼群搜索效率.

1 云任务调度模型

1.1 任务调度问题描述

云计算平台可以对不同的计算资源和服务进行统一管理,并根据用户需求的变化,自动对计算资源进行分配和调度. 在云计算任务调度过程中,首先会在逻辑上将用户所提交的任务进行细分,具体分成若干个独立的子任务,各个子任务根据特定的策略分配到合适的计算资源上. 其核心问题就是为用户的任务合理分配云资源,并实现计算时间和成本的最小化等目标的过程. 具体任务调度模型如图 1 所示.

首先在该模型中,假设有 n 个任务的集合 Task 为:

$$\mathrm{Task}=\{\mathrm{task}_1,\mathrm{task}_2,\cdots,\mathrm{task}_n\}, \tag{1}$$

云资源中 m 个虚拟机 Vm 集合为:

$$Vm=\{vm_1,vm_2,\cdots,vm_m\},\tag{2}$$

则任务与虚拟机的映射关系用矩阵表示为:

$$\boldsymbol{S}=\begin{bmatrix}s_{11}&s_{12}&\cdots&s_{1m}\\s_{21}&s_{22}&\cdots&s_{2m}\\\vdots&\vdots&\ddots&\vdots\\s_{n1}&s_{n2}&\cdots&s_{nm}\end{bmatrix}.\tag{3}$$

其中,s_{ij}表示第 i 个任务是否分配给第 j 个虚拟机. 当 $s_{ij}=0$ 时,任务 i 没有分配给虚拟机 j,当 $s_{ij}=1$ 时,任务 i 分配给虚拟机 j.

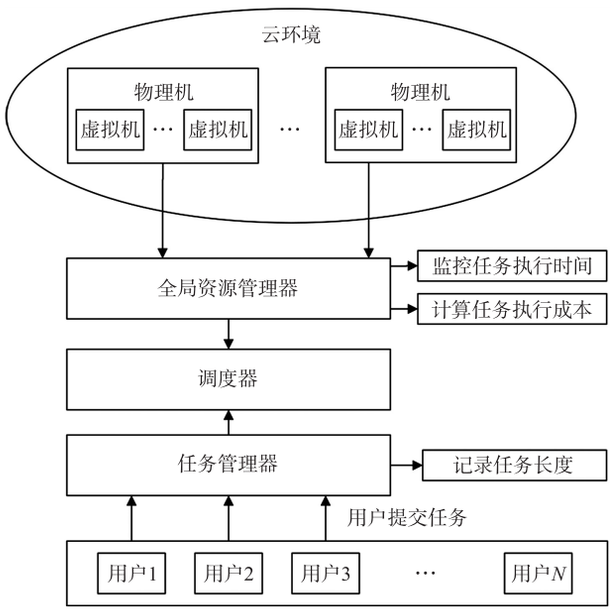


图 1　任务调度模型

Fig. 1　Task scheduling model

1.2　适应度函数设计

适应度函数用途非常广泛,可以将分布在种群中的解决方案的复杂性降低到学习算法可以处理的程度. 为降低云计算中任务执行时间和花费的成本,本文主要在设计适应度函数时考虑任务最大完工时间和总成本,具体设计如下所述:

(1)任务最大完工时间(MakeSpan)

该指标为用户提交的任务在数据中心计算集群中执行所花费的总时间,即最后一个执行完毕任务的完成时间,最大完工时间越小,说明该算法的任务调度的执行效率越高.

首先任务的执行时间如公式(4)所示:

$$t_{ij}=\frac{\text{length}(i)}{\text{mips}(j)},\tag{4}$$

则每个任务分配到各个虚拟机的执行时间可以用矩阵 $\boldsymbol{ET}$ 表示:

$$\boldsymbol{ET}=\begin{bmatrix}t_{11}&t_{12}&\cdots&t_{1m}\\t_{21}&t_{22}&\cdots&t_{2m}\\\vdots&\vdots&\ddots&\vdots\\t_{n1}&t_{n2}&\cdots&t_{nm}\end{bmatrix}.\tag{5}$$

定义 $vm\text{Time}_j$ 为虚拟机 j 上执行所分配任务的时间开销总和,如公式(6)所示:

$$vm\text{Time}_j=\sum_{i=1}^{n}s_{ij}t_{ij}.\tag{6}$$

最后将系统中任务的最大完工时间 MakeSpan 计算如公式(7)所示:

$$\text{MakeSpan}=Max(vm\text{Time}_1,vm\text{Time}_2,\cdots,vm\text{Time}_m).\tag{7}$$

其中,t_{ij}表示第 i 个任务在第 j 个虚拟机上的执行时间,length(i)表示第 i 个任务的长度,mips(j)表示第 j 个虚拟机的计算能力.

(2)执行任务总成本(Cost)

该指标为在云环境中完成计算用户提交任务所需的成本开销,是用户实际使用的资源和服务的费用. 执行任务总成本越低,代表所用的任务调度算法提供了更高质量的解决方案.

定义 C_j 为第 j 个虚拟机执行所分配任务的总成本,计算公式如(8)所示:

$$C_j=vm\text{Time}_j\text{Base}_j,\tag{8}$$

则系统执行任务总成本 Cost 用公式(9)计算:

$$\text{Cost}=\sum_{j=1}^{n}c_j,\tag{9}$$

其中,Base_j 为虚拟机 j 在单位时间内执行任务所耗费的成本.

(3)多目标适应度函数

本文以最小化任务最大完工时间和成本为优化目标. 在迭代过程中,适应度作为人工鱼位置更新的条件,适应度小的个体更优良,则多目标适应度函数如公式(12)所示:

$$F_1=\log_e^{\text{MakeSpan}}, \tag{10}$$

$$F_2=\log_e^{\text{Cost}}, \tag{11}$$

其中,F_1、F_2 分别是某个人工鱼的执行时间和成本的适应度,为将两个相差较大的函数整合到一起,把每一个函数值取对数,最后相加得到多目标适应度函数为:

$$\text{Fitness}=F_1+F_2. \tag{12}$$

2 基本算法介绍

2.1 基本人工鱼群算法

基本人工鱼群算法[10]是通过观察鱼类的活动发现,鱼群总是聚集在营养物质最丰富的地方,根据鱼类这一特点模拟其基本行为,由此实现全局优化. 在本文中设人工鱼个体的位置为向量 $\boldsymbol{X}=(x_1,x_2,\cdots,x_n)$,人工鱼其所处位置的食物浓度(适应度)表示为 $Y=f(X)$,下文介绍其主要行为,具体描述如下:

(1)觅食行为

设人工鱼的当前位置为 X_i,首先人工鱼在其视野范围 Visual 内根据公式(13)随机选择一个位置 X_j,若 $Y_j<Y_i$(在极大值问题中,$Y_j>Y_i$),则根据公式(14)向位置 X_j 移动;否则再次重新随机选择一个状态 X_j,若反复尝试 Try_number 次仍不满足前进条件,则根据公式(15)执行随机行为.

$$X_j=X_i+\text{Rand}()\,\text{Visual}, \tag{13}$$

$$X_{i|\text{prey}}=X_i+\text{Rand}()\,\text{Step}\,\frac{X_j-X_i}{X_j-X_i}, \tag{14}$$

$$X_{i|\text{prey}}=X_i+\text{Rand}()\,\text{Step}, \tag{15}$$

其中,人工鱼的视野范围为 Visual,每次移动步长为 Step,人工鱼个体之间的距离为 X_j-X_i,Rand()是[0,1]之间的随机数.

(2)聚群行为

设人工鱼的当前位置为 X_i,搜索其视野内的伙伴数量 n_f 及中心位置 X_c,若$\frac{Y_c n_f}{Y_i}<\delta$,则说明伙伴中心有较多食物且不太拥挤,则根据公式(16)朝中心位置移动一步;否则执行觅食行为.

$$X_{i|\text{swarm}}=X_i+\text{Rand}()\,\text{Step}\,\frac{X_c-X_i}{X_c-X_i}, \tag{16}$$

其中,δ 表示拥挤度因子.

(3)追尾行为

设人工鱼的当前位置为 X_i,搜索其视野内的食物浓度 Y_j 最大的伙伴 X_j,若$\frac{Y_j n_f}{Y_i}<\delta$,则说明伙伴 X_j 的状态具有较高的食物浓度且周围不太拥挤,则根据公式(17)朝伙伴的方向前进一步;否则执行觅食行为.

$$X_{i|\text{follow}}=X_i+\text{Rand}()\,\text{Step}\,\frac{X_j-X_i}{X_j-X_i}. \tag{17}$$

2.2 花粉算法

花粉算法是 2012 年学者 Yang[16]提出的一种新型群智能优化算法,该算法的思想源于自然界中植物花朵授粉过程,具有参数少、算法简单易实现,能在全局搜索与局部搜索之间转换的的优点. 其原理为,首先在搜索空间中进行种群初始化,再根据设定的概率,对每一个花粉配子进行全局授粉行为或者局部授粉行为,以此反复迭代,更新种群,直到寻找到最优解.

本文中花粉配子的位置设为向量 $\boldsymbol{X}_i=(x_{i1},x_{i2},\cdots,x_{in})$,种群全局最佳位置为 $X_g=(x_{g1},x_{g2},\cdots,x_{gn})$,故花粉配子的更新策略如下:

(1)首先确定一个概率 p,p 在原算法中取固定 0.8. 再取一个介于[0,1]之间的随机数 Rand,通过

Rand 决定执行(2)或者(3).

(2)当 p>Rand 时,通过公式(18)进行全局授粉行为:

$$X_i^{t+1}=X_i^t+L(X_i^t-X_g), \tag{18}$$

$$L\sim\frac{\lambda\Gamma(\lambda)\sin\left(\frac{\pi\lambda}{2}\right)}{\pi s^{1+\lambda}}. \tag{19}$$

(3)当 p<Rand 时,通过公式(20)进行局部授粉行为:

$$X_i^{t+1}=X_i^t+\varepsilon(X_j^t-X_k^t). \tag{20}$$

其中,参数 L 步长,服从 *Lévy* 分布. $\lambda=1.5$,$\Gamma(\lambda)$ 为标准伽马函数,ε 为[0,1]之间的随机数,X_i^t 表示第 t 次迭代的解,X_g 表示全局最优解,X_j^t 和 X_k^t 表示第 t 次迭代中随机选取的其他花粉配子.

3　改进的人工鱼群算法

3.1　反向学习

反向学习(opposition-based learning,OBL)[17]是由学者 Tizhoosh 于 2005 年首次提出,并在短时间内被证明是提高各种优化算法的性能的一个有效方法[18-19]. 其主要思想是在评估给定问题的某些解时,运用反向学习策略计算其相反的解可以提供找到另一个更接近全局最优解的候选解的可能. 应用公式如下:

设在 N 维空间中有一点 $S=(s_1,s_2,\cdots s_n)$,则必然存在其反向点 $S^*=(s_1^*,s_1^*,\cdots,s_n^*)$. s_1^* 由公式(21)求得.

$$s_i^*=a+b-s_i, \tag{21}$$

其中,a 和 b 分别是 s_i 取值的上下界.

3.2　基于反向学习的种群初始化策略

AFSA 是一种群体随机搜索算法,其计算时间与其初始种群中个体与最优个体的距离有关,如果初始种群个体在最优值附近,那么在计算过程中,种群的所有个体都会朝向最优值进行快速地收敛[20]. 为加速种群收敛,提高最终解的精度,本文将反向学习策略应用于种群初始化阶段. 基于反向学习的种群初始化流程如算法 1 所示.

算法 1　利用反向学习初始化种群.

输入:种群大小 N,位置向量维度 dim;

输出:种群位置向量集 $\boldsymbol{pos}$;

(1)随机产生 N 个 dim 维位置向量,命名为 $\boldsymbol{pos}_x$.

for i in $|\boldsymbol{pos}_x|$

(2)对向量集 $\boldsymbol{pos}_x$ 中的第 i 个向量 $\boldsymbol{pos}_{xi}$ 根据公式(21)生成反向学习向量 $\boldsymbol{pos}_{yi}$,将其加入到反向学习向量集 $\boldsymbol{pos}_y$.

end for

(3)计算 $\boldsymbol{pos}_x$ 和 $\boldsymbol{pos}_y$ 的适应度.

(4)从 $\boldsymbol{pos}_x$ 和 $\boldsymbol{pos}_y$ 中挑选出 N 个适应度较好的向量,并将其加入到 $\boldsymbol{pos}$.

return $\boldsymbol{pos}$

其中,$|\boldsymbol{pos}_x|$ 为鱼群中人工鱼的总数量.

3.3　改进的行为选择

人工鱼的每个行为都有其不同的作用. 人工鱼的觅食行为奠定算法收敛的基础,聚群行为增强收敛的稳定性与全局性,追尾行为增强了算法收敛的快速性与全局性[21]. 而在原算法中并没有对人工鱼的行为进行合理规划,采取对每条人工鱼分别执行 3 种行为后,再择优执行的策略. 这样虽然使得算法具有很强的局部搜索能力,但是同时也造成了大量的无用搜索,使得搜索效率下降,计算时间长. 因此本文提出了一种基于种群平均适应度的行为选择.

首先根据公式(22)计算出每次迭代后的种群平均适应度,再根据平均适应度将种群进行动态分组,对优秀个体和一般个体使用不同移动策略. 采用对每次迭代效果较好的方向作为引导的思想,对小于平

均适应度的优秀个体选择觅食行为进行全局寻优,对大于平均适应度的一般个体选择聚群和追尾行为向优秀个体处聚拢.

同时,为解决随着迭代次数的增加,种群多样性逐渐减少的问题,采取对最优行为进行反向学习,并从二者中选取最佳点进行执行的策略,以此来扩大解空间.

通过以上方法对人工鱼群进行了动态分类,不同的个体采取不同的移动策略,使得全局最优值更快地突现出来,搜索效率也得到提升.

$$Y_{\text{avg}}^{t}=\frac{\sum_{i=1}^{N}f(X_i^t)}{N}, \tag{22}$$

其中,X_i^t 为第 t 次迭代中鱼群的第 i 个人工鱼,$Y(X_i^t)$ 为第 i 个人工鱼的适应度,N 为鱼群中人工鱼个数,Y_{avg}^{t} 为第 t 次迭代的人工鱼群平均适应度.

3.4 概率 p 的改进

概率 p 主要是控制两种搜索方式的比例,但是在算法迭代过程中,原算法的概率 p 取一个固定值有一定的不合理性,并没有考虑到不同人工鱼之间适应度的差异. 因此,本文将概率设置为与迭代次数和适应度相关的函数,如公式(23)所示. 该改进使得一般人工鱼在算法前期,更倾向于往全局最优位置靠拢,增强了全局搜索能力和收敛速度,而优秀人工鱼在算法后期,更倾向于局部搜索,提高搜索解的精度,避免陷入局部最优.

$$p_i^t=\alpha\left(1-\frac{e^{\frac{t}{\text{maxite}}-1}}{e-1}\right)+\beta\left(\frac{Y_i^t-Y_{\text{best}}^t}{Y_{\max}^t-Y_{\text{best}}^t}\right), \tag{23}$$

其中,t 是当前迭代次数,maxite 是最大迭代次数,p_i^t 是第 t 次迭代中第 i 个人工鱼的选择概率. α 和 β 各取 0.5.

3.5 改进的觅食行为

在基本人工鱼群算法中,觅食行为是最为核心的行为,其直接影响了迭代过程中的搜索精度和收敛速度. 但在基本人工鱼群算法中,觅食行为的移动主要依靠人工鱼视野进行搜索寻优,与全局最佳位置交流学习能力较弱,导致容易陷入局部最优解. 因此,人工鱼仅依靠视野来寻优具有一定的局限性. 故本文提出一种改进的觅食行为,将局部授粉和全局授粉行为融合到觅食行为中,有效地提升了全局搜索与局部寻优能力,提高了算法的收敛速度. 具体改进如下所述:

设人工鱼的当前位置为 X_i,授粉行为选择概率为 p. 首先人工鱼在其视野范围内根据公式(13)随机选择一个位置 X_j^t,若 $Y_j^t<Y_i^t$,则进行授粉行为的选择. 先得到一个随机数 Rand,若 Rand<p,则根据公式(24)进行全局搜索,扩展搜索范围,若 Rand>p,则根据公式(25)进行局部搜索,提升解的精度. 若反复尝试多次仍不满足前进条件,则根据公式(26)随机移动一步.

$$X_{i|\text{prey}}=X_i^t+L(X_j^t-X_g^t), \tag{24}$$

$$X_{i|\text{prey}}=X_i^t+\varepsilon(X_j^t-X_i^t), \tag{25}$$

$$X_{i|\text{prey}}=X_i^t+L(X_i^t-X_g^t), \tag{26}$$

其中,$X_{i|\text{prey}}$ 为第 i 条人工鱼的当前行为的预备解,X_j^t 为视野范围内较优解,X_g^t 为第 t 次迭代的全局最优解,X_i^t 为第 t 次迭代的第 i 条人工鱼的解. 执行流程如算法 2 所示.

算法 2 改进的觅食行为.

输入:X_i^t;

输出:$X_{i|\text{prey}}$;

for i in Try__number

①$X_j=X_i^t+\text{Rand}()\,\text{Visual}$

②if($Y_j<Y_i^t$)

③根据公式(23)更新概率 p_i^t.

④if(Rand<p_i^t)

⑤　$X_{i|\text{prey}}=X_i^t+L(X_j^t-X_g^t)$

⑥　return $X_{i|\text{prey}}$

⑦　else

⑧　$X_{i|\text{prey}}=X_i^t+\varepsilon(X_j^t-X_i^t)$

⑨　return $X_{i|\text{prey}}$

⑩　end if

⑪end if

end for

$X_{i|\text{prey}}=X_i^t+L(X_i^t-X_g^t)$

⑫Return $X_{i|\text{prey}}$.

3.6　基于 IAFSA 算法的云计算任务调度

算法流程文字描述如算法 3 所示.

算法 3　改进的人工鱼群算法.

输入:定义输入参数视野 Visual、拥挤度因子 δ、步长 Step、最大迭代次数 maxite、人工鱼群种群大小 N、位置向量维度 dim、最大尝试次数 Try_number.

输出:最优调度解 X_g.

①进行云计算任务和虚拟机编码.

②根据算法 1 进行人工鱼群位置向量集初始化.

③while 不满足停止条件.

④　根据公式(22)计算 Y_{avg}^t.

⑤　for i in N

⑥　if($Y_i^t \leqslant Y_{\text{avg}}^t$)

⑦　根据算法 2 执行改进的觅食行为.

⑧　else

⑨　执行聚群和追尾行为.

⑩　end if

⑪　利用从预备解中的最佳行为 X_{next}
根据公式(21)产生反向解 X_{obl}.

⑫　if($Y_{\text{obl}}<Y_{\text{next}}$ && $Y_{\text{obl}}<Y_i$)

⑬　$X_i^{t+1}=X_{\text{obl}}$

⑭　else if($Y_{\text{next}}<Y_i$)

⑮　$X_i^{t+1}=X_{\text{next}}$

⑯　end if

⑰end for

⑱更新公告板,选出全局最优的人工鱼.

⑲判断是否满足终止条件,若满足则返回最优解 X_g,否则继续从步骤④进行下一次迭代.

基于 IAFSA 的云计算任务调度过程如图 2 所示.

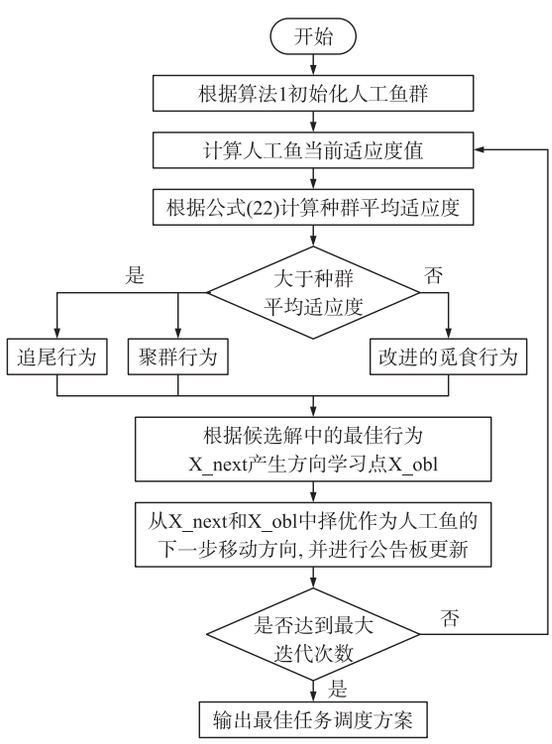


图 2　算法 IAFSA 流程图

Fig. 2　Algorithm IAFSA flowchart

3.7　IAFSA 时间复杂度分析

时间复杂度是一种用于分析算法效率的方法,它表示算法运行所需的时间与输入规模之间的关系. 通常用大 O 符号表示,例如 $O(N)$ 表示算法的时间复杂度与输入规模 N 成正比. 时间复杂度越低,算法的效率越高. 如下,表 1 详细列出了 IAFSA 各个寻优步骤的时间复杂度的值.

表 1　IAFSA 时间复杂度分析

Table 1　Complexity estimation of the IAFSA

IAFSA 寻优步骤	时间复杂度	IAFSA 寻优步骤	时间复杂度
1.初始化 N 条人工鱼	$O(N)$	5.觅食行为	$O(Try_{number}*N)$
2.初始化公告版	$O(N)$	6.随机行为	$O(N^2)$
3.聚群行为	$O(N^2+2*N)$	7.终止条件判断	$O(1)$
4.追尾行为	$O(N^2+2*N)$	8.公告板信息输出	$O(1)$

在表 1 中,以种群大小规模为 N,可以求得原始人工鱼群算法时间复杂度为

$$O(\text{Maxite}*(3*N^2+\text{Try}_{\text{number}}*N+6*N),$$

同样地,可以求得花粉算法的时间复杂度为

$$O(\text{Maxite}*(N^2+N),$$

在基于种群平均适应度的行为选择下,IAFSA 算法时间复杂度为

$$O(\text{Maxite}*(3*N^2+6*N),$$

由以上分析可以得出,时间复杂度主要与种群规模有关. 3 种算法的时间复杂度均为 $O(N^2)$ 级别,但花粉算法和 IAFSA 的时间复杂度要优于 AFSA.

4　仿真实验与结果分析

4.1　实验环境设置

本文实验是通过使用澳大利亚墨尔本大学网格实验室开发的 CloudSim[22] 平台进行仿真模拟. 实验是在 Intel i5 处理器、16GB 内存、Windows 11 操作系统下进行. 在实验中,通过 CloudSim 模拟一个 IaaS 云提供商,其拥有 1 个数据中心、5 个不同配置的虚拟机. 云系统中规定使用 5 台虚拟机进行实验,虚拟机配置信息详情如表 1 所示,每个对比算法都将在相同虚拟机和任务配置下运行.

表 2　虚拟机信息

Table 2　VM's information

VM 编号	VM 性能/MIPS	内存/GB	带宽/(GB/s)	费用/(元/s)
1	100	512	1	0.19
2	200	512	1	0.27
3	300	512	1	0.43
4	400	512	1	0.51
5	500	512	1	0.62

同时,整个实验的任务数量规模分为三类:即 50 个任务的小规模,200 个任务的中规模,500 个任务的大规模,其中每个任务大小都在[100,1 000 000]中随机产生. 因为 IAFSA 算法和对比算法均是随机搜索算法,每次搜索时得到的解可能不一致,为了保证实验的准确性,以及减少任务大小的不确定性给实验结果带来的影响,本文实验结果均为在相同运行环境下执行 20 次,再取其平均值进行分析得出. 表 3 所示为 7 种算法的性能对比结果,结果均使用平均值表示.

表 3　IAFSA 算法与其他算法性能比较

Table 3　Performance comparison between IAFSA algorithm and other algorithms

	小规模任务			中规模任务			大规模任务		
	成本/元	最大完工时间/s	适应度函数值	成本/元	最大完工时间/s	适应度函数值	成本/元	最大完工时间/s	适应度函数值
IAFSA	8.900 2	16 000.488	11.860 8	37.185	70 808.031	14.787 1	95.345	234 382.7	16.920 9
AFSA	9.447 9	17 503.031	12.013 7	38.375	85 601.648	15.002 6	97.579	286 916.4	17.146 7
OBL-TP-PSO	9.340 0	16 851.825	11.960 1	37.871	81 565.462	14.940 2	97.551	316 179.9	17.242 6
SAPSO	9.369 9	16 872.108	11.964 4	38.342	81 994.421	14.947 6	98.844	278 835.3	17.129 6
P-ADAFSA	9.083 3	16 286.497	11.899 2	37.527	72 509.088	14.812 9	96.784	242 699.7	16.973 8
CAFSA	9.284 1	17 260.811	11.979 9	37.531	84 785.371	14.970 1	96.084	283 999.3	17.120 8
PAFSA	9.050 3	16 299.766	11.894 8	37.283	74 130.444	14.812 9	96.214	273 846.6	17.084 7

4.2　算法参数设置

为验证改进的算法在云计算任务调度中的良好性能,本文将提出的 IAFSA 算法与 AFSA、SAPSO、

OBL-TP-PSO、P-ADAFSA[23],以及在 AFSA 基础上仅根据文中 3.3 小节做出改进的 CAFSA 和仅根据文中 3.5 小节做出改进的 PAFSA 这 6 种云计算任务调度算法进行对比. 以上算法的迭代次数均设为 200,种群大小为 100. IAFSA、AFSA、CAFSA、PAFSA 以及 P-ADAFSA 算法参数设置如表 2 所示,OBL-TP-PSO、SAPSO 分别参照文献[4]、文献[24]设置.

4.3　小规模任务

图 3 显示了 7 种算法在小规模任务环境下适应度的迭代情况对比. 从图 3 中的对比情况来看,IAFSA 算法的适应度明显低于其他 6 种对比算法,这说明 IAFSA 算法能很好地应用在云计算的任务调度中. 改进的人工鱼群算法在迭代一开始就明显优于其他算法,体现出了优化后的种群初始化策略与觅食行为的效果,使得 IAFSA 算法能够在迭代前期更快、更精准地找到全局最优解,而仅改进行为选择的 CAFSA 则比 AFSA 略优.

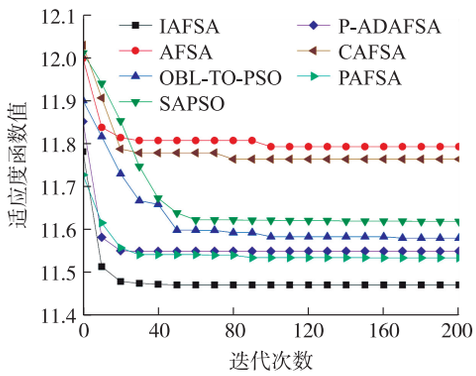


图 3　任务规模为 50 的收敛情况

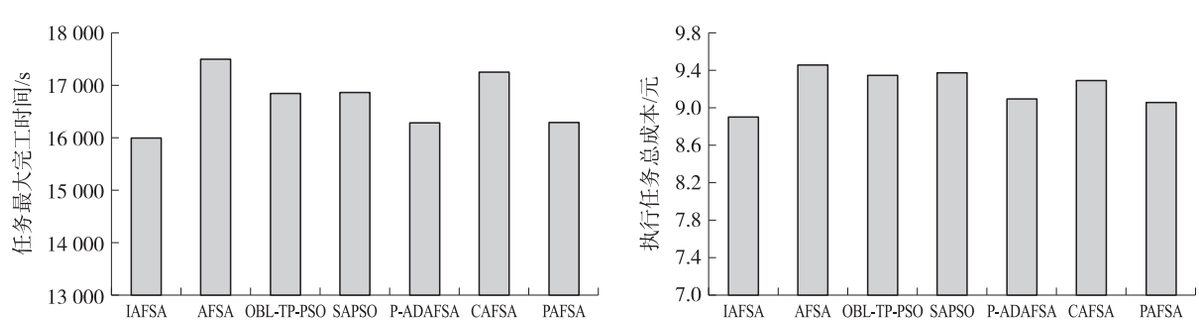
Fig. 3　Convergence of task size 50

图 4 为任务规模在 50 的情况下对系统最大完工时间和成本指标的优化效果. 从结果来看,IAFSA 算法对指标优化情况有明显优势,最大完工时间与成本都是最少的. 虽然 OBL-TP-PSO 与 SAPSO 算法都优于原始 AFSA 算法,但与 IAFSA 算法相比,仍有较大差距.

表 4　算法参数设置

Table 4　Algorithm parameter setting

参数	取值	参数	取值
视野 Visual	15	种群大小 N	100
拥挤度因子 δ	10	位置向量维度 Dim	与任务数量一致
步长 Step	10	Try_number	10
最大迭代次数 Maxite	200		



(a) 最大完工时间　　(b) 执行任务总成本

图 4　任务规模为 50 的指标优化情况

Fig. 4　Optimization of indicators with task size of 50

4.4　中规模任务

图 5 为任务规模 200 时的算法收敛过程. 从图 5 中可以得出,IAFSA、AFSA、PAFSA、P-ADAFSA、OBL-TP-PSO 算法都能取得一个较好的初值,而 SAPSO 算法适应度初值相对较高. 在算法迭代过程中,IAFSA 算法、PAFSA 算法、P-ADAFSA 算法都能保持较快的收敛速度,其中 IAFSA 算法因为其改进的觅食行为,增强了鱼群间个体交流的能力,同时能够合理分配人工鱼进行全局搜索或局部搜索,使得在迭代过程中,不仅能快速找到全局最优解,还能够凭借局部搜索策略继续提升解的精度,避免了在迭代后期陷入局部最优状态. 同时,融入粒子群记忆因子的 P-ADAFSA 算法与仅改进觅食行为的 PAFSA 算法能够基本保持一致. 从图 5 中可以看出,IAFSA 算法迭代效果仍优于其他 6 种对比算法.

图 6 为任务规模为 200 时的各算法的总成本和最大完工时间的实验结果图. 从图中可得知,CAFSA 算法、AFSA 算法、OBL-TP-PSO 算法、SAPSO 算法的最大完工时间都在 80 000 s 以上,而 IAFSA 算法、P-ADAFSA 算法、PAFSA 算法都在 70 000 s 附近,但 IAFSA 仍保持着最少的最大完工时间,同时执行任务总成本也是低于其他 6 种对比算法. 由此可见,IAFSA 算法在任务调度的多目标优化中有着非常显著的优势.

4.5　大规模任务

图 7 为在大规模任务时的算法收敛过程. 可以从中看出,IAFSA 在随机性更强的大规模任务情景下,能够在算法迭代初期就具有优秀的初值,并且在算法迭代后期体现出了突破局部最优解的能力. 相比于 6 种对比算法,IAFSA 具有更快的寻优速度和更高的寻优精度,其算法迭代结果表现更加优异.

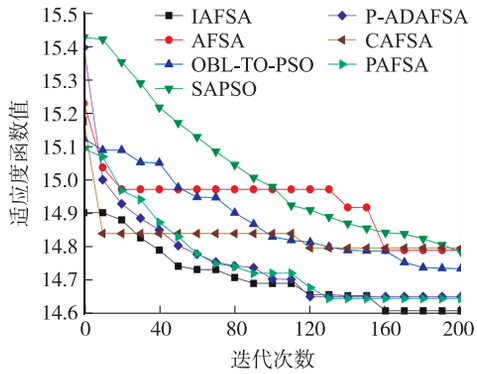
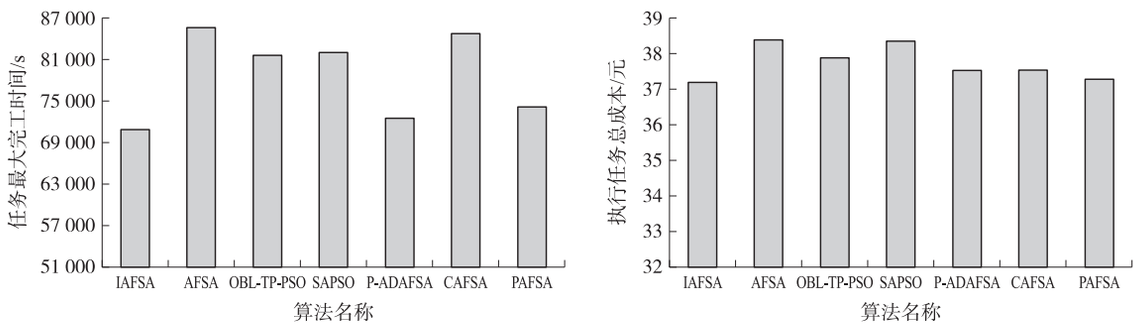


图 5　任务规模为 200 的收敛情况

Fig. 5　Convergence of task size 200



(a) 最大完工时间　　(b) 执行任务总成本

图 6　任务规模为 200 的指标优化情况

Fig. 6　Optimization of indicators with task size of 200

由图 8 可知,在大规模任务中,执行任务总成本差异比较明显,IAFSA 算法仍然保持着最小值,其中 AFSA、OBL-TP-PSO 算法基本保持一致,CAFSA 算法略优于 AFSA 算法. 从最大完工时间指标中来看,IAFSA 算法执行任务所耗时间为 234 382.755 2 s,低于其他 6 种对比算法. 这表明,即使在大任务规模的云环境中,IAFSA 算法仍然能取得较优的任务调度方案,使得任务的最大完成时间和所消耗的成本都低于其他对比算法.

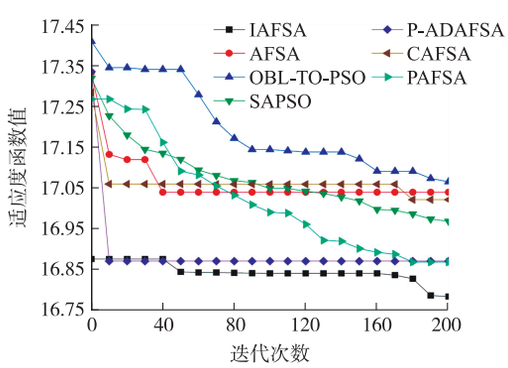
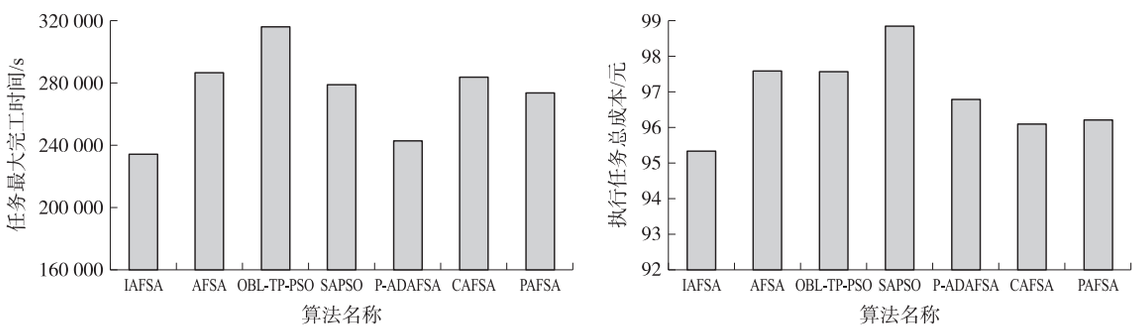


图 7　任务规模为 500 的收敛情况

Fig. 7　Convergence of task size 500



(a) 最大完工时间　　(b) 执行任务总成本

图 8　任务规模为 500 的指标优化情况

Fig. 8　Optimization of indicators with task size of 500

4.6 IAFSA 算法优势分析

在基本人工鱼群算法中,能并列执行聚群、追尾、觅食 3 种行为,并将随机行为作为缺省嵌入以上 3 种行为中,这使得该算法在快速寻优、跳出局部最优解等方面具有优良的表现,但并列执行同样会使得该算法的时间复杂度较高. 随着人工鱼数目的增多,将会需求更多的存储空间,也会造成计算量的增长,并且由于 3 种行为的职能都有所不同,并列执行多种行为将会导致整个鱼群移动的无序性和混沌性. 同时由于视野的寻优限制和步长的随机性,使得寻优的精度难以很高. 而在花朵授粉算法(FPA)中,花粉配子所具有的全局授粉或者局部授粉行为具有优秀的群体信息交流能力,将其融入至人工鱼的行为中能够进行更加有效的移动,从而提升鱼群的寻优精度,避免盲目地落入局部最优解当中. 同时在引入基于种群平均适应度的行为选择机制后,不同适应度的人工鱼选择相应的行为,使得鱼群的移动更加有序和合理,并使得人工鱼在迭代过程中从每次并列执行 3 种行为减少至 1 种或 2 种行为,减少了不必要的并行运算,符合智能算法追求至简化的原则. 以上通过第四章中 CAFSA 算法和 PAFSA 算法的实验结果可以证实上述改进的有效性.

5 结论

本文通过基于反向学习的种群初始化策略、改进的行为选择、融入花粉算子的觅食行为、概率 p 的改进提出了一种改进的人工鱼群计算任务调度算法. 通过 CloudSim 云计算仿真平台对 IAFSA 算法与其他 6 种算法进行对比,本文提出的 IAFSA 算法在不同的任务规模下,其任务完成时间、任务执行成本、收敛速度等指标均体现出了显著的优化效果. 同时,在下一步研究中,将会在优化目标上考虑加入虚拟机负载、能耗、资源利用率等指标,以求能更好地解决云计算任务调度问题.

[参考文献]

[1] ASHRAF H, ALJAMMAL, NEDAL M, et al. A new architecture of cloud computing to enhance the load balancing[J]. International journal of business information systems, 2017, 25(3): 393-405.

[2] 徐保民,倪旭光.云计算发展态势与关键技术进展[J]. 中国科学院院刊,2015,30(2):170-180.

[3] 郑爽,吕遐东,陈杰.面向多目标优化的云计算调度研究综述[J]. 舰船电子工程,2022,42(9):13-19.

[4] ZHOU Z, LI F, ABAWAJY J H, et al. Improved PSO algorithm integrated with opposition-based learning and tentative perception in networked data centres[J]. IEEE access, 2020, 8: 55872-55880.

[5] 孙敏,叶侨楠,陈中雄.云环境下方差定向变异遗传算法的任务调度[J]. 计算机应用,2019,39(11):3328-3332.

[6] JIA Z, WANG Y, WU C, et al. Multi-objective energy-aware batch scheduling using ant colony optimization algorithm[J]. Computers & industrial engineering, 2019, 131: 41-56.

[7] PREM J T, PRADEEP K. A multi-objective optimal task scheduling in cloud environment using cuckoo particle swarm optimization[J]. Wireless personal communications, 2019, 109: 315-331.

[8] FU X L, SUN Y, WANG H F, et al. Task scheduling of cloud computing based on hybrid particle swarm algorithm and genetic algorithm[J]. Cluster computing, 2023, 26(5): 2479-2488.

[9] NATESAN G, CHOKKALINGAM A. Multi-objective task scheduling using hybrid whale genetic optimization algorithm in heterogeneous computing environment[J]. Wireless personal communications, 2020, 110: 1887-1913.

[10] 李晓磊. 一种新型的智能优化方法—人工鱼群算法[D]. 杭州:浙江大学,2003.

[11] 李君,梁昔明. 人工鱼群算法收敛速度改进优化仿真[J]. 计算机仿真,2018,35(1):232-238.

[12] 王联国,施秋红. 人工鱼群算法的参数分析[J]. 计算机工程,2010,36(24):169-171.

[13] 杨文杰,巨涛,杨阳,等. 面向边缘计算的人工鱼群搜索任务调度[J]. 电子测量与仪器学报,2022,36(11):149-159.

[14] 陆俊明,张向锋. 一种改进的粒子群人工鱼群算法[J]. 上海电机学院学报,2019,22(1):50-55.

[15] 徐建波,戴月明,严大虎. 双自适应人工鱼群优化算法[J]. 微电子学与计算机,2018,35(4):53-57.

[16] YANG X S. Flower pollination algorithm for global optimization[C]//Unconventional Computation and Natural Computation: 11th International Conference. Berlin Heidelberg: Springer, 2012.

[17] TIZHOOSH H R. Opposition-based learning: a new scheme for machine intelligence[C]//International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web

Technologies and Internet Commerce(CIMCA-IAWTIC'06). Vienna,Austria:IEEE,2005.
[18] MALISIA A R,TIZHOOSH H R. Applying opposition-based ideas to the ant colony system[C]//2007 IEEE Swarm Intelligence Symposium. Honolulu,USA:IEEE,2007.
[19] WANG H,LI H,LIU Y,et al. Opposition-based particle swarm algorithm with Cauchy mutation[C]//2007 IEEE Congress on Evolutionary Computation. Singapore:IEEE,2007.
[20] RAHNAMAYAN S,TIZHOOSH H R,SALAMA M M A. Quasi-oppositional differential evolution[C]//2007 IEEE Congress on Evolutionary Computation. Singapore:IEEE,2007.
[21] 黄光球,刘嘉飞,姚玉霞. 人工鱼群算法的全局收敛性证明[J]. 计算机工程,2012,38(2):204-206.
[22] CALHEIROS R N,RANJAN R,BELOGLAZOV A,et al. CloudSim:a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms[J]. Software:Practice and experience,2011,41(1):23-50.
[23] 刘志锋,舒志浩,胥越峰,等. 基于 PSO 自适应双策略的人工鱼群算法[J]. 计算机与现代化,2022,321(5):46-53.
[24] WANG X H,LI J J. Hybrid particle swarm optimization with simulated annealing[C]//Proceedings of 2004 International Conference on Machine Learning and Cybernetics. Shanghai:IEEE,2004.

[责任编辑:黄　敏]