DOI: 10.13382/j.jemi.B1902852

云计算多目标任务调度的优化粒子群算法研究*

马学森[1,2,3] 谈 杰[1,3] 陈树友[1,3] 储昭坤[1,3] 石 雷[1,3]

(1. 合肥工业大学 计算机与信息学院 合肥 230009; 2. 广东三水合肥工业大学研究院 佛山 528000;
3. 安全关键工业测控技术教育部工程研究中心 合肥 230009)

摘 要:针对传统粒子群算法求解云计算多目标任务调度的收敛速度慢、精度低的缺陷,提出一种优化多目标任务调度粒子群算法(MOTS-PSO)。首先,引入非线性自适应惯性权重,改变粒子的寻优能力,避免算法陷入局部最优;其次引入花朵授粉算法概率更新机制,平衡粒子的全局搜索和局部寻优,并对粒子的全局搜索位置更新公式进行改进;最后引入萤火虫算法,产生“精英解”对局部搜索位置更新公式进行改进;同时利用“精英解”对粒子的位置进行扰动,跳出局部最优状态。实验表明,MOTS-PSO 算法在收敛速度和收敛精度上,比 PSO 算法提高了 27.1%、19.9%,比 FA 算法提高了 22.09%、5.2%。进一步实验表明,MOTS-PSO 算法在解决不同规模数量的任务调度时,比 PSO、FA 算法效果更优。

关键词: 多目标任务调度;粒子群优化;自适应惯性权重;全局搜索;局部寻优;位置扰动

中图分类号: TP393 **文献标识码**: A **国家标准学科分类代码**: 520

Research on optimal particle swarm optimization for multi-objective task scheduling in cloud computing

Ma Xuesen[1,2,3] Tan Jie[1,3] Chen Shuyou[1,2,3] Chu Zhaokun[1,2,3] Shi Lei[1,3]

(1. School of Computer and Information, Hefei University of Technology, Hefei 230009, China; 2. Research Institute of Sanshui & Hefei University of Technology in Guangdong, Foshan 528000, China; 3. Engineering Research Center of Safety Critical Industiral Measurement and Control Technology, Ministy of Education, Hefei 230009, China)

Abstract: To overcome the slow convergence and low accuracy of traditional particle swarm optimization (PSO) for slow convergence and low accuracy of multi-objective task scheduling in cloud computing, an optimized multi-objective task scheduling particle swarm optimization algorithm (MOTS-PSO) is proposed. Firstly, the nonlinear adaptive inertial weight is introduced to change the particle's optimization ability to avoid the algorithm from running into local optimum. Secondly, the flower pollination algorithm probability update mechanism is introduced to balance the global search and local optimization of the particles. In addition, we improve the global search position update formula. Finally, the firefly algorithm (FA) is introduced to generate the elite solution to improve the local search position update formula. At the same time, we utilize the elite solution to perturb the particle position and to jump out of the local optimal state. Experiments show that the MOTS-PSO algorithm has 27.1% and 19.9% higher convergence speed and precision than the PSO algorithm, and 22.09% and 5.2% higher than the FA algorithm. Further experiments show that the MOTS-PSO algorithm is more effective than the PSO and FA algorithms in solving tasks of different sizes and numbers.

Keywords: multi-objective task scheduling; particle swarm optimization; adaptive inertial weighting; global search; local optimization; Elite solution

收稿日期:2019-12-22 Received Date: 2019-12-22

*基金项目:广东省科技发展专项基金(2017A010101001)、中央高校基本科研业务费专项基金(PA2019GDKPK0079)、国家留学基金、安徽省教育厅高等学校省级质量工程项目(2017JYXM0055,2019MOOC020)资助

0 引　言

云计算作为一种新型的商业服务模式[1],云计算的计算中心是由一系列提供超强网络服务和超强计算资源的异构主机所组成。从用户体验的角度来看,云计算可以分为基础设施即服务(IaaS)、平台即服务(PaaS)和软件即服务(SaaS)三层,并且每一层都拥有自己的虚拟化资源。虚拟化技术的使用使得每个服务平台可以对其基础资源进行抽象。

在云计算中,任务调度问题是一个组合优化问题,任务调度的结果关系到整个云计算设施的效率,并且在提高用户服务质量中起到十分关键的作用。任务调度就是在适应性时间上将任务调度到适应性资源上,建立适当的任务执行顺序,从而可以在一定的约束条件下执行任务,分散处理器上的负载,最大程度地减少总任务执行时间。在云计算中,工作流中的任务数量以及可用资源数量会随着虚拟资源的分配而迅速增长,计算所有可能的任务调度策略并选择一个最佳调度策略是不可行的,因为这种方法的复杂性会随着任务数量和资源数量的增加而呈指数增长。近些年,利用启发式优化算法,解决任务调度问题的研究众多。启发式算法的优点在于,在处理任务调度问题的过程中,一方面算法本身的运行时间是可以被接受的,另一方面启发式优化算法可以显著降低搜索空间的复杂性。在协调调度策略运行时间和任务最优分配上启发式算法提供了一种折中的解决方案。为此,粒子群算法作为启发式优化算法的一种,本文将引入粒子群算法作为任务调度策略。

尽管在该领域已进行了大量研究,但在大多数研究者的目标优化模型中,目标函数彼此不冲突且具有相同的趋势,在求解过程中实质上是将其转化为单目标机制求解或单目标优化问题。本文在研究云计算中任务调度问题时,建立了一个综合的多目标任务调度优化模型,优化总任务执行时间和数据传输时间,以及单节点最大任务的执行时间和数据传输时间最优,这些优化的目标函数之间存在一定的冲突。考虑到粒子群优化算法(PSO)及其改进算法在解决云计算任务调度优化方面的优势[2-5],本文首先引入非线性自适应惯性权重策略,改变粒子在前期迭代过程中全局寻优能力以及后期迭代过程中局部寻优能力,提高粒子种群多样性,避免陷入局部最优;其次,引入花朵授粉算法概率更新机制,平衡全局搜索过程和局部寻优过程,并对粒子的位置更新机制进行改进;最后,算法寻优的初期,引入萤火虫算法,先产生一个“精英解”进行“仓储”,利用该“精英解”对局部搜索的位置更新公式进行进一步改进;在监控到粒子群算法在迭代的过程中,最优解连续恒定不变的迭代次数超过设定值时,利用“仓储”的“精英解”对粒子的位置进行扰动,避免算法陷入局部最优的状态。将该多目标任务调度粒子群算法(MOTS-PSO)作为任务调度策略进行部署。通过在 CloudSim 仿真平台上的实验显示,在多目标任务调度问题中,本文提出的 MOTS-PSO,相较于 PSO 算法、萤火虫算法(FA)具有更快的收敛速度,更高的收敛精度。MOTS-PSO 算法无论是在小规模还是在大规模多目标任务调度过程中都能比 PSO 算法和 FA 算法取得更好的调度效果。

1 相关工作

在现有的研究工作中将启发式优化算法运用于云计算任务调度优化问题的研究工作有很多。文献[6]提出在经典的蚁群算法上引入局部、全局信息深度更新方式,并将改进后的蚁群算法作为云计算任务调度策略。通过模拟云计算的任务调度实验,证明了与经典的蚁群优化算法(ACO)相比,改进后蚁群算法的搜索速度加快,同时在任务调度过程中,大幅度减少了云计算任务的执行时间,资源负载问题也得到了很好的解决。文献[7]通过遗传算法(GA)开发了一种负载均衡策略,从而在并行计算系统运行期间,优化任务分配,并取得良好的调度效果。文献[8]使用简单的遗传算法优化任务调度和计算资源的分配,来最小化制造周期和运营成本。此外,使用非支配的排序遗传算法-Ⅱ(最成功的多目标优化技术)获得双重目标的非支配解。实验表明使用简单遗传算法可以显著降低制造周期和运营成本,并且在多目标优化问题中可以获得广泛的非支配解。为了提高任务调度过程中 GA 调度结果的准确性,文献[9] 对经典的 GA 进行了相应的优化,研究者在经典遗传算法的基础之上,引入双精英保留策略,并根据云计算中虚拟节点资源的性能引入虚拟机相对适应的概念。与传统的遗传算法将染色体随机交叉变异相比,改进后的变异操作为一种有目标的变异操作。实验研究表明,改进后的遗传算法收敛于最优解的速度更快,同时算法也降低了调度任务的完成时间,是一种能让虚拟机资源负载更加均衡的调度策略。除将蚁群算法和遗传算法及其改进算法作为任务调度策略运用于云计算任务调度,有研究将收敛效果更好的粒子群算法应用于云计算任务调度研究中。文献[10]研究证实,无论是在网格计算,还是在云计算环境中,从调度效果上看,遗传算法与粒子群算法相比,后者能取得更好的任务调度效果。文献[11]提出了一种基于小位置值规则的 PSO 算法,制定用于任务调度的模型,该模型将最大程度地减少执行和传输的总时间。他们使用基于粒子群的交叉,变异和局部搜索算法对 PSO 进行了比较和分析。实验结果证明, PSO 算法的解决方案不仅比

GA 更好,而且比 GA 更快。文献[12]在标准的粒子群算法的基础上,引入混沌优化搜索技术,并提出一种基于 Tent 映射的自适应混沌粒子群任务调度策略。改进后的算法克服了传统 PSO 算法容易陷入局部最优解的缺点,将改进后的算法作为云计算中任务调度策略进行部署,实验表明在 CloudSim 平台上,改进后的 PSO 算法与改进的遗传算法、改进后的蚁群算法相比,改进后的 PSO 算法能更有效缩短总任务的完成时间,具有更好的寻优能力和实时性。

以上研究表明,PSO 更适合云计算中的任务调度。但是上述研究依然存在一些不足之处,尽管基于 PSO 的单目标算法效率很高,但它们对于解决在云计算中多目标任务调度问题并不实用,因为这两个目标相互冲突并且不存在一个可以同时优化每个解决方案的解决方案。另外在任务规模增加后,原始的粒子群算法容易出现收敛过早,收敛精度过低等问题。为了克服这些缺点,在后续的研究工作中,文献[13]在解决异构环境下与工作流中任务代价问题时,提出了一个任务的执行代价和依赖任务之间的数学模型,并基于一种改进的粒子群算法 WSA-IPSO 算法去求解一个最优化调度方案,实验证明改进后的粒子群算法与 MCT 算法和标准的粒子群算法相比,算法可以快速收敛,并得到一个最优的调度方案。文献[14]提出了一种基于分数阶达尔文粒子群优化算法,在研究任务调度的过程中,综合考虑任务最短等待时间、资源负载均衡以及任务完成所耗费用 3 个目标,在此基础之上,实验验证表明改进后的粒子群算法能在任务调度过程中不仅有最低的任务最大完成时间,同时还较好的完成了虚拟机节点资源的负载均衡,改进后的粒子群算法在处理多目标任务调度问题时表现良好。

2　多目标任务调度模型

云计算中,任务调度问题就是将数量庞大的任务通过调度策略分配到不同的虚拟机节点上执行的过程,合理的调度策略可降低总任务完成代价,提高用户服务质量。本节首先将对任务调度问题进行相应的描述,并在此基础上建立一个多目标任务调度的数学模型。

2.1　问题描述

假设在云环境中,部署有数量为 m 的物理主机,虚拟机节点总数量为 n, 待处理任务序列数量为 k。 多目标任务调度问题的描述如下。

1)云计算数据中心部署的物理主机集合为 P, 其中 $P=\{p_1,p_2,\cdots,p_i\}(i=1,2,\cdots,m)$。 物理主机之间相互独立,每台物理主机拥有 CPU、内存、存储、GPU、带宽等类型的资源。

2)数据中心物理主机上需要部署的虚拟机 VM, $VM=\{vm_1,vm_2,\cdots,vm_n\}(j=1,2,\cdots,n)$, 表示数据中心需要部署的所有虚拟机节点。各虚拟机节点的部署和运行相互独立,虚拟机节点拥有 CPU、内存、存储、GPU、带宽等类型的资源; $VMm_j(j=1,2,\cdots,n)$ 表示虚拟机 j 的处理速度; $VMc_j(j=1,2,\cdots,n)$ 表示虚拟机 j 的 CPU 核心数目。

3) 任务序列集合 $T=(t_1,t_2,\cdots,t_l)(l=1,2,\cdots,k)$。其中 DE_{ik} 表示虚拟机 k 上,任务 i 的长度; DT_{ik} 表示任务 i 需要上传到虚拟机 k 上的数据量。

4) 资源分配矩阵 $\boldsymbol{X}_{ij}=(x_{ij})$, 其中 $(i=1,2,\cdots,k,j=1,2,\cdots,n)$, x_{ij} 的取值为:

$$x_{ij}=\begin{cases}0, \text{任务 } i \text{ 不分配到虚拟机 } j \text{ 上}\\ 1, \text{任务 } i \text{ 分配到虚拟机 } j \text{ 上}\end{cases} \tag{1}$$

调度算法每次执行时都会生成一个资源矩阵 $\boldsymbol{X}_{ij}$, $\boldsymbol{X}_{ij}$ 中的所有 x_{ij} 的取值就是一次调度策略,调度算法将在这些取值中,找到一个最佳的资源矩阵,作为最佳任务调度方案。

问题描述的符号如表 1 所示。

表 1　任务调度问题符号表

Table 1　Symbol table for task scheduling

符号	说明
m	物理主机数量
n	虚拟机数量
k	调度任务数量
P	云计算中心物理主机,$P=\{p_1,p_2,\cdots,p_i\}(i=1,2,\cdots,m)$
VM	虚拟节点资源,$\{vm_1,vm_2,\cdots,vm_n\}(j=1,2,\cdots,n)$
VMm_j	表示虚拟机 j 的处理速度
VMc_j	表示虚拟机 j 的 CPU 核心数目
T	调度的任务序列,$T=(t_1,t_2,\cdots,t_l)(l=1,2,\cdots,k)$
DE_{ik}	分配在虚拟机 k 上执行的任务 i 的长度
DT_{ik}	任务 i 需要上传到分配的虚拟机 k 上的数据量的大小
X_{ij}	资源分配举证,当 $x_{ij}=1$,将任务 i 分配到虚拟节点 j 上

2.2　多目标优化调度数学模型

在云计算任务调度的过程中,每个虚拟机节点的计算能力和网络传输带宽是不同的,依据不同的调度策略将产生不同的执行结果。在本模型中我们考虑的并非传统的单目标问题而是一个多目标的数学模型。本文不仅仅考虑总任务的执行时间代价,数据传输时间代价总和最优,同时也将调度后单节点最大任务代价也考虑在任务调度模型当中,确保最大单节点任务的执行时间和数据传输时间最优,该目标与总任务的执行时间代价,总的数据传输时间代价存在一定的冲突。在本节中,对多目标任务调度问题进行如下的数学模型描述:

定义 1　任务总执行时间

总任务执行时间是按照调度策略将任务分配到虚拟节点上执行时所花费的时间代价总和,第 k 个任务执行时间为:

$$Texe_k = \sum_{i=1}^{n}\left(x_{ik} \times \frac{DE_{ik}}{VMm_k \times VMc_k}\right) \tag{2}$$

因此,总的任务执行时间定义为:

$$Texe = \sum_{k=1}^{m} Texe_k \tag{3}$$

定义 2　任务数据的总传输时间

总的传输时间是调度过程中,数据的传输时间。数据传输时间与虚拟机的带宽存大小有关,第 k 个任务传输时间为:

$$Trans_k = \sum_{i=1}^{m}\left(x_{ik} \times \frac{DT_{kz}}{Band_{kz}}\right) \tag{4}$$

因此,任务调度过程中,总数据传输时间定义为:

$$Trans = \sum_{k=1}^{n} Trans_k \tag{5}$$

定义 3　单节点任务执行代

在任务调度中,单虚拟机节点的执行代价为:

$$T_{cost} = T_{exe} + T_{trans} \tag{6}$$

定义 4　多目标优化问题

1) 总任务执行时间最优,则其数学描述为:

$$f_1(x) = Min(Texe) \tag{7}$$

2)　任务的数据传送总时间最优,则其数学描述为:

$$f_2(x) = Min(Trans) \tag{8}$$

3) 单虚拟机节点最大任务执行代价最优,则其数学描述为:

$$f_3(x) = Min\{Max(T_{cost})\} \tag{9}$$

4) 最优问题的数学描述:

多目标问题的数学模型可以描述为:

$$MinF(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \cdots, f_n(\vec{x})] \tag{10}$$

因此本文的多目标优化问题可以定义为:

$$MinF(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), f_3(\vec{x})] \tag{11}$$

其中, $\sum_{k=1}^{m} x_{ik} = 1, \forall i = 1,2,\cdots,n; x_{ik} \in \{0,1\}, \forall i,k$。

3　改进粒子群算法的任务调度策略

3.1　基本粒子群算法

Kennedy 和 Eberhart 在对鸟群觅食行为研究的基础上,于 1995 年首次提出 PSO 算法[15]。PSO 算法是众多启发式迭代优化算法中的一种,算法在实现较为简单,迭代过程中收敛速度较快。在处理优化问题的过程中,被优化的数学函数可以不具备连续、微分、可导等特性。

算法的基本原理可以总结为假设存在空间的维度为 n 维,粒子种群规模为 N 的搜索空间。其中,每个粒子在搜索空间中都看作是不具备体积的粒子,对于粒子 i, 其速度的 n 维向量可以表示为 $V_i = \{v_{i1}, v_{i2}, \cdots, v_{in}\}$, 位置的 n 维向量可以表示为 $X_i = \{x_{i1}, x_{i2}, \cdots, x_{in}\}$。在搜索过程中,第 i 代粒子搜索到的最优位置信息有两种,一个是粒子在种群内搜索到的局部最优位置 $pbest$, 另一个最优位置信息是整个粒子群迄今为止搜索到的全局最优位置 $gbest$, 局部最优解的向量表示为 $P_i = \{P_{i1}, P_{i2}, \cdots, P_{in}\}$, 最优解的向量表示为 $Gbest_i = \{Gbest_{i1}, Gbest_{i2}, \cdots, Gbest_{in}\}$。全局粒子群算法模型中粒子 i 在第 $t+1$ 次迭代时,速度更新公式为:

$$v_{in}^{(t+1)} = \omega \times v_{in}^{(t)} + c_1 \times r_1 \times (P_{in} - x_{in}^{(t)}) + c_2 \times r_2 \times (Gbest_{in} - x_{in}^{(t)}) \tag{12}$$

式中:ω 代表迭代过程中的惯性权重,调节粒子的搜索能力; r_1、r_2 是两个随机数,取值在 [0,1]; c_1 和 c_2 表示粒子飞行中加速因子,也称为学习因子。

粒子 i 在第 $t+1$ 次迭代时,位置更新公式为:

$$x_{in}^{(k+1)} = x_{in}^{(k)} + v_{in}^{(k+1)} \tag{13}$$

3.2　算法改进策略

1) 自适应惯性权重

粒子群算法的惯性权重 ω 是由 Shi 等提出。Shi 等指出,不同的 ω 值,代表着粒子在寻找最优解过程中拥有不同的搜索能力,尤其是对粒子群的全局搜索和局部探索过程的影响。文献的研究表明,当 $\omega > 1$ 时,粒子在多维空间中的飞行速度随着时间会逐渐增大至最大值,此时群体粒子处于发散的状态。相反,当 $0 < \omega \leqslant 1$ 时,粒子的速度不断的减小。飞行中较大的 ω 对搜索空间的探索是十分有利的,大大增加种群的多样性,而较小的 ω 对粒子局部开发能力的提升是十分有利的。前期粒子在寻优迭代的过程中需要一个较大的搜索空间进行探索,并且增加种群的多样性,这个时候就需要一个较大的 ω 值,在算法执行的后期,需要一个较小的惯性权重值 ω 来提升局部开发的能力。Shi 等通过线性递减函数对惯性权重 ω 进行自适应的改进,取得了较好的迭代效果。文献[16-18]提出了采用非线性递减函数对惯性权重 ω 进行自适应的改进,实验结果表明,采用非线性递减函数的 PSO 算法要优于采用线性递减函数改进的 PSO 算法。因此,本文采用的是一种非线性递减函数,对惯性权重 ω 进行调优。优化后的惯性权重值在粒子迭代过程中自适应改变,改进后的惯性权重定义为:

$$\omega = \omega_{Max} - (\omega_{Max} - \omega_{Min}) \times (e^{\left(\frac{t}{Tmax}-1\right)}) \times rand(0,1) \tag{14}$$

式中: $\omega_{Max} = 0.9, \omega_{Min} = 0.4$ [19], t 是算法的迭代次数, Tmax 是算法最大的迭代次数。

2) 引入花朵算法的改进机制

英国学者 Yang 于 2012 年提出的花朵授粉算法(flower pollination algorithm, FPA),是一种新的启发式群智能算法[20]。花朵授粉算法模拟自然界花朵自花授粉

和异花授粉两个过程。异花授粉的过程被看做是算法的全局搜索过程,而自花授粉的过程则被看做是局部搜索的过程。局部搜索和全局搜索之间可以通过切换概率 $p \in [0,1]$ 来控制。Yang 在设计花朵授粉算法的时候,通过转换概率 p 选择授粉方法,来平衡局部搜索和全局搜索之间的关系。在粒子群算法中,平衡算法的全局搜索过程和局部寻优过程,有利于提高算法的收敛精度和收敛速度。

因此,在粒子群算法中引入花朵授粉算法依概率切换搜索过程的机制,算法在优化时全局搜索次数和局部搜索次数得到了更好的平衡,能更好帮助粒子提高收敛速度,同时也有利于粒子跳出局部最优解。对 p 值做一个自适应的调整[21]:

$$p = 0.75 + 0.2 \times rand(0,1) \tag{15}$$

改进后的全局搜索的位置更新公式为:

$$x_i^{(t+1)} = x_i^{(t)} + L(\lambda) \times (V_{pbest} - v_{id}^{(t+1)}) + \varepsilon \times (V_{gbest} - v_{id}^{(t+1)}) \tag{16}$$

式中:$L(\lambda)$ 是花朵授粉算法中的 levy 飞行函数;ε 是一个在 $[0,1]$ 的随机数;V_{pbest} 表示局部最优速度值;V_{gbest} 表示粒子寻优过程中全局最优速度值。

3)混合萤火虫算法的改进机制

粒子群算法在迭代的后期容易陷入局部最优状态而无法跳出,此时对粒子位置进行一次扰动将有助于粒子跳出局部最优解,重新进行寻优。初期利用萤火虫[22]算法,产生一个"精英解"存储,用向量 u 表示。因此,利用萤火虫算法产生的"精英解",对粒子群算法局部搜索位置更新公式进行改进。新的局部搜索位置更新公式改进为:

$$x_i^{(t+1)} = u + rand \times (V_{gbest} - v_{id}^{(t+1)}) \tag{17}$$

式中:$rand$ 表示的取值满足标准正态分布的随机数;V_{gbest} 表示粒子寻优过程中全局最优速度值。

粒子群算法在后期的寻优过程中往往需要跳出局部最优解范围。在迭代过程中,设置监测值 $Maxstep$,在监测到粒子群算法的最优解连续不发生变化的迭代次数超过 $Maxstep$ 时,此时就判定粒子在迭代过程中陷入了局部最优的"陷阱"中,这时算法需要采用一定的变异策略帮助粒子跳出当前的局部最优值,利用萤火虫算法产生的仓储"最优解"重新初始化一个位置向量,帮助算法跳出局部最优解的设计方案为:

$$if(CurrentStep > MaxStep)\{x_i^{(t+1)} = u + trnd(t) \times (V_{gbest} - v_{id}^{(t+1)})\} \tag{18}$$

式中:V_{gbest} 表示粒子寻优过程中全局最优速度值;$trnd(t)$ 表示的是 t 分布。

3.3　适应度函数

求解多目标问题的数学方法众多,可采用约束法、目标规划法、分层序列法以及函数评价法等常用的方法求解。为提高云计算的资源利用率以及用户服务质量,根据 2.2 节数学模型的描述,定义的 3 个目标函数为:

$$f_1(x) = \sum_{k=1}^{m}\sum_{i=1}^{n}\left(x_{ik} \times \frac{DE_{ik}}{VMm_k \times VMc_k}\right) \tag{19}$$

式中:$f_1(x)$ 是任务总执行时间。

$$f_2(x) = \sum_{k=1}^{m}\sum_{i=1}^{n}\left(x_{ik} \times \frac{DT_{ik}}{Band_{ik}}\right) \tag{20}$$

式中:$f_2(x)$ 表示任务与虚拟机之间数据传输过程中总传输时间。

$$f_3(x) = \sum_{i=1}^{n}\left(x_{ik} \times \frac{DE_{ik}}{VMm_k \times VMc_k}\right) + \sum_{i=1}^{m}\left(x_{ik} \times \frac{DT_{kz}}{Band_{kz}}\right) \tag{21}$$

式中:目标函数 $f_3(x)$ 表示单虚拟机节点任务的调度代价。

在求解多目标问题时,为相应的优化指标分配相应的权重系数 w,以显示调度过程中优化目标的重要性级别。适应度函数的定义为:

$$F = Min\{w \times Min(f_1(x) + f_2(x))\} + (1 - w) \times Min(Max(f_3(x))) \tag{22}$$

其中,$(0 \leqslant w \leqslant 1)$。当 $w = 0$ 或者 $w = 1$ 时,多目标任务调度问题将转化为单目标任务调度问题。在 w 逐渐增大时,优化的目标则更侧重于最优化总任务的执行时间和数据传输时间;反之,任务更偏向于最优化最大单节点任务的调度代价。

3.4　编码与任务分配策略

针对标准的粒子群算法在处理多目标问题时容易出现收敛速度慢,收敛精度低的缺点,本文引入了自适应惯性权重策略、结合花朵授粉算法概率更新机制以及混合萤火虫算法对标准的粒子群算法进行改进,并提出了一种 MOTS-PSO 算法。

当 MOTS-PSO 算法作为多目标任务调度策略时,需要对粒子进行编码处理。编码就是将任务调度问题的可行解空间与粒子的位置进行映射的过程。假设在任务调度模型中,存在 n 个数量的任务 $\{t_1,t_2,\cdots,t_l\}(l = 1,2,\cdots,k)$,,这些任务需要分配到 m 个虚拟机 $\{vm_1,vm_2,\cdots,vm_n\}$ 上执行。

MOTS-PSO 算法被用来在最小化执行时间、数据传输时间以及最小化单节点最大任务执行时间这 3 个目标中,找到一个最佳的任务调度方案。所有可行调度方案都是由 MOTS-PSO 算法产生。粒子的位置向量 $X_i = \{x_{i1},x_{i2},\cdots,x_{in}\}$ 的值是连续值的,在任务调度中将任务部署到相应的虚拟机上是一类离散问题,关键问题在于将粒子的位置信息转化为具有实际意义的虚拟机编号。因此,在将连续的位置信息转化为离散的位置信息的过程中,采用文献[23]所提到的最小位置(SPV)原则,对连

续的位置信息进行离散化。

具体处理过程为若第 i 代粒子所对应的位置信息为 $X_i = \{x_{i1}, x_{i2}, \cdots, x_{in}\}$，对应的任务的部署方案向量为 $d(\vec{x_i}) = \{|floor(x_1)|, |floor(x_2)|, \cdots, |floor(x_i)|\}$，即将连续的位置值转换为离散值。假设需要将 7 个调度任务分配到 5 个数量的虚拟机上，其种虚拟机的标号为 {0,1,2,3,4}，某时刻粒子 i 在迭代过程中的位置信息为 {1.33,0.45,4.56, -1.98,2.72,3.41,2.65}，编码后可得 {1,0,4,1,2,3,2}，表 2 是对应的任务调度方案。

表 2　任务分配策略

Table 2　Task allocation strategy

任务	t_1	t_2	t_3	t_4	t_5	t_6	t_7
VMID	vm_1	vm_0	vm_4	vm_1	vm_2	vm_3	vm_2

3.5　MOTS-PSO 任务调度策略部署

1）初始化云计算数据中心物理主机列表和每个物理主机上虚拟机的列表；

2）初始化粒子群算法参数；

3）利用萤火虫算法，得出初始最优解；

4）初始化粒子群算法的速度向量信息以及位置向量信息，使用 SPV 规则将的连续位置值向量 $\boldsymbol{X}_i$ 转换为离散向量 $\vec{d}$，以确定为每个到达任务分配的。根据评价函数评估任务调度代价值，并记录，从历史最优值中选出全局最优值，并记录相应的编码策略；

5）粒子迭代过程开始，依据改进式(14)计算迭代的权重参数 w；

6）依据式(12)更新粒子速度向量信息，依据式(16)、(17)的变异策略更新粒子位置向量信息；

7）使用 SPV 规则将的连续位置值向量 $\boldsymbol{X}_i$ 转换为离散向量 $\vec{d}$，以确定为每个到达任务分配的 VM；

8）根据评价函数评估任务调度代价值，并记录，从历史最优值中选出全局最优值，并记录相应的编码策略；

9）监测最优解，若迭代中最优解保持不变的次数超过阈值，则利用“仓储”的“精英解”对粒子的位置进行扰动；

10）判断迭代条件是否满足，不满足则跳出迭代，否则转到步骤 5)；

11）得出 MOTS-PSO 算法的最优解，并与萤火虫算法输出的最优解进行比较，取两者中最优值，作为任务调度的部署方案；

12）调度任务结束。

4　仿真实验结果与分析

为进一步说明本文提出的 MOTS-PSO 调度策略是一种效率更高的多目标任务调度算法，选取了另外两种启发式优化算法：PSO 算法和 FA 算法与本文提出的 MOTS-PSO 算法进行实验对比。本文采用 CloudSim 云计算仿真实验平台进行仿真实验，表 3 是实验的硬件环境配置。

表 3　硬件环境配置表

Table 3　Hardware environment configuration table

配置信息	参数
CPU	Intel 酷睿 i7-8750H
内存	16G DDR4
硬盘	希捷 1TB
操作系统	微软 Windows10

4.1　云计算任务调度环境

本文采用 CloudSim 云计算仿真实验平台，在进行云环境任务调度前，需要对云环境中相关设备进行配置。包括：云环境中物理主机的类型以及主机的相关参数信息、物理主机中虚拟机的类型以及相关参数信息、调度任务的任务长度以及传输的数据量的大小，同时还包括相关调度算法中的各个参数配置信息。

在 CloudSim 云计算仿真实验平台中，表 4 是实验平台中物理主机的类型以及相关参数设置。

表 4　物理主机参数

Table 4　Physical host parameter

物理机类别	CPU 核心数	内存/GB	硬盘/TB	带宽/Mbps
1	16	30	3	1 000
2	16	45	4	2 500
3	24	45	5	1 500
4	32	60	6	2 000

物理主机中将配置一定数量的虚拟机，任务通过调度策略分配到虚拟机上执行，虚拟机的类型以及各类型虚拟机的相关参数设置如表 5 所示。

表 5　虚拟机参数

Table 5　Virtual machine parameter

虚拟机类别	CPU 核心	内存/GB	处理速度/MIPS	带宽/Mbps
1	1	3.75	256	1 000
2	1	3.75	300	2 000
3	1	7.50	300	2 500
4	1	1.70	400	1 500
5	1	3.75	256	2 000

除了表 4 的物理主机参数设定、表 5 的虚拟机参数的设定，对于调度任务自身参数，为了便于与其他算法横向比较，本文的任务调度设计采用与文献[24]相同的方式，其调度任务参数的设定如表 6 所示。

表 6　调度任务参数

Table 6　Schedule task parameter

参数	取值
任务长度	25 000~250 000
传输数据量	100~600

云环境中,调度算法将会把任务以最优的方式分配到虚拟机上进行执行,本文在实验中采用 3 种调度算法,包括原始 PSO 算法、FA 算法以及本文提出的 MOTS-PSO 算法。表 7 是这 3 种算法的相关参数设置。

表 7　各调度算法参数

Table 7　Parameters of each scheduling algorithm

调度算法	参数名称	参数值
PSO	种群大小	25
	迭代次数	500
	惯性因子	0.9
	学习因子	$c_1 = 1.494\,45, c_2 = 1.494\,45$
FA	种群大小	25
	迭代次数	500
MOTS-PSO	种群大小	25
	迭代次数	500
	惯性因子	$\omega_{Max} = 0.9, \omega_{Min} = 0.4$
	学习因子	$c_1 = 1.494\,45, c_2 = 1.494\,45$

4.2　结果与分析

1)算法性能比较与分析

算法的收敛速度和收敛精度是评价启发式算法性能的两个重要指标。在所述硬件环境以及相关属性的基础之上,对改进后的算法进行评测。在进行算法性能对比时,取标准 PSO 算法、FA 算法与本文所提出的 MOTS-PSO 算法作比较,参数设置如表 7 所示,3 种算法多次重复运行。图 1 所示为 3 种算法多次重复运行的平均结果迭代效果对比。

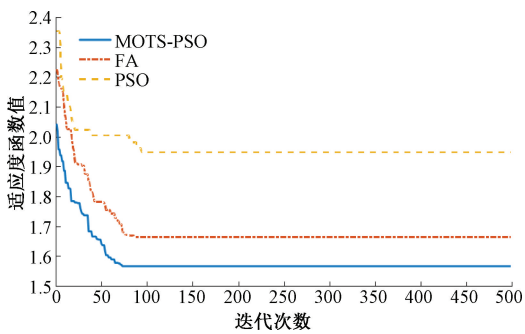

图 1　迭代对比

Fig. 1　Iterative comparison chart

依据图 1 的分析,改进后的 MOTS-PSO 算法在都保持着优于 PSO 算法和 FA 算法的趋势。3 种算法在迭代过程中相关比较数据如表 8 所示。

表 8　收敛性对比

Table 8　Convergence comparison

任务调度问题		数值
PSO 算法	迭代最小值	1.800 9
	迭代最大值	2.170 4
	迭代平均值	1.941 1
	收敛代数平均值	92
FA 算法	迭代最小值	1.601 7
	迭代最大值	1.776 2
	迭代平均值	1.652 9
	收敛代数平均值	86
MOTS-PSO 算法	迭代最小值	1.516 9
	迭代最大值	1.681 1
	迭代平均值	1.56 627
	收敛代数平均值	67

由表 8 可知,本文提出的 MOTS-PSO 算法,迭代的最大值是优于 PSO 算法和 FA 算法,迭代的最小值也是优于 PSO 算法和 FA 算法; MOTS-PSO 算法的收敛平均值为 1.566 27,收敛的平均迭代次数为 67,迭代平均值相比经典的 PSO 算法提高了 27.17%,比 FA 算法提高了 22.09%;收敛精度上,相较于 PSO 算法提高了 19.49%,相较于 FA 算法提高了 5.2%。

结合图 1 和表 8 的分析结果,可以得出 MOTS-PSO 无论是在收敛速度上,还是在收敛精度上相较于经典 PSO 算法和 FA 算法都要更优。因此,通过引入自适应惯性权重,增加了粒子种群多样性;引入花朵授粉算法依概率切换机制,平衡粒子群算法的全局搜索与局部搜索,以及萤火虫算法“精英解”扰动机制,改进后的算法性能提高较为明显。

2)多目标任务调度实验对比

针对第 2 节提出的 3 个优化目标,本文设计了不同实验场景的对所提出的 MOTS-PSO 任务调度策略进行验证。实验场景 1,小规模实验条件的部署,在数据中心中部署 5 台物理主机,每台主机上部署有相应的虚拟机,调度的任务数量为 10~100,实验中每次增加 10 个任务序列,物理主机的数量固定不变。实验场景 2,与小规模实验场景相比,设计一个大规模任务调度场景,任务数量为 1 000~5 000,实验过程中,每次增加 1 000 个任务量。

(1)小规模任务数量部署的实验对比

图 2~4 所示为实验场景 1 下算法任务调度总代价、任务总执行时间和数据传输时间、单节点最大任务执行代价的效果对比。

从实验结果可以看出,在小规模任务数量场景下,本文提出的 MOTS-PSO 在各个优化目标的结果上都要优于 PSO 算法和 FA 算法。在求最优执行代

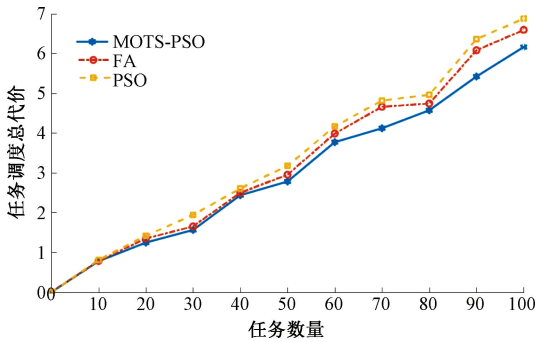

图 2　小规模任务调度总代价对比

Fig. 2　Comparison chart of total cost of small-scale task scheduling

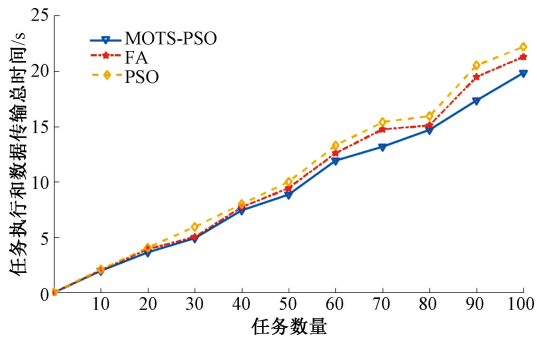

图 3　小规模总执行时间和数据传输时间对比

Fig. 3　Comparison of small-scale total execution time and data transfer time

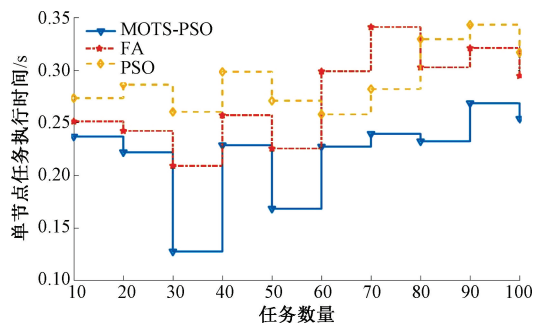

图 4　小规模任务单节点最大执行代价对比

Fig. 4　Comparison of single node maximum execution costs for small-scale tasks

价，任务执行时间和数据传送的总时间，初始阶段三种算法的差距并不明显，随着任务数量的不断增加，本文提出的 MOTS-PSO 调度策略在优化这些目标函数时，效果更好；不仅如此，MOTS-PSO 算法在 3 种的调度方案中，单节点的最大执行代价也是最优的，因此，引入的优化改进策略，确保了算法寻优能力，增加粒子群算法种群的多样性，能更好的帮助算法找到最优值。因此，在实验场景 1 中，MOTS-PSO 算法更优。

(2)大规模任务数量部署的实验对比

图 5~7 所示为在实验场景 2 下，算法任务调度总代价、任务总执行时间和数据传输时间、单节点任务最大执行代价的效果对比。

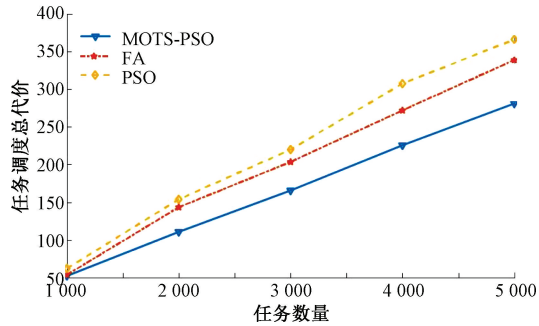

图 5　大规模任务调度总代价对比

Fig. 5　Comparison chart of total cost of large-scale task scheduling

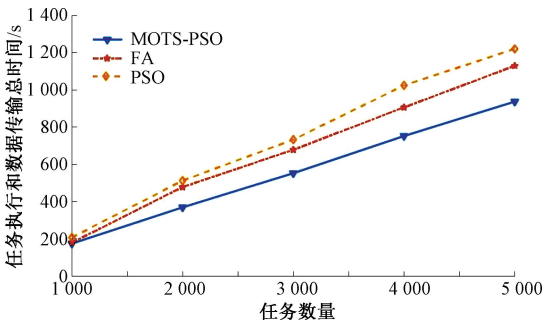

图 6　大规模总执行时间和数据传送时间对比

Fig. 6　Comparison of large-scale total execution time and data transfer time

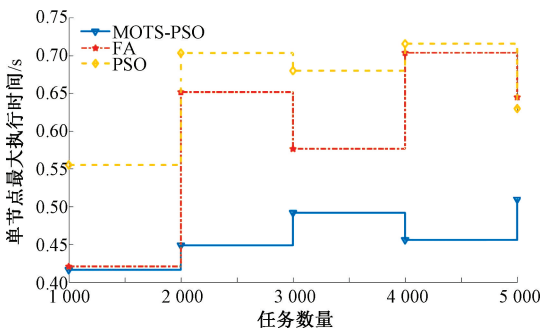

图 7　大规模任务单节点最大执行代价对比

Fig. 7　Comparison of the maximum execution costs of a single node for a large-scale task

从实验结果可以看出，在大规模任务场景下，通过 MOTS-PSO 调度策略求出的任务总执行代价，总任

务执行时间和数据传输时间都要优于 PSO 算法和 FA 算法。通过 MOTS-PSO 算法调度产生的最大单节点总执行代价也是最优的。与小规模试验场景下的实验结果相比较,在大规模任务场景下 MOTS-PSO 调度策略更加的稳定,性能优势也更加明显。由此,在任务规模较大时,引入非线性自适应惯性权重策略,改变了粒子在前期迭代过程中全局寻优能力以及后期迭代过程中局部寻优能力,提高粒子种群多样性,避免陷入局部最优;引入花朵授粉算法依概率更新机制,平衡了粒子群算法的全局搜索和局部寻优过程;最后,引入萤火虫算法,产生“精英解”对局部搜索的位置更新公式进行进一步改进,在提高粒子寻优的收敛度、收敛精度有较好的效果。在迭代过程中“监控到”最优解恒定不变超过阈值时,利用“仓储”的“精英解”对粒子的位置进行扰动,能有效帮助粒子跳出局部最优解,从而能够在大规模任务调度时获得一个更好的任务调度策略。因此,本文所提出的 MOTS-PSO 算法在大规模任务应用场景下,能更好的解决多目标任务调度问题。

从以上两个实验场景的结果中,可以得出改进后的粒子群算法,能够作为更好的任务调度策略应用于多目标任务调度过程中,大大降低任务调度的总代价,相比于 PSO、FA 任务调度策略,是一个更好的任务调度策略。

5　结　论

本文针对标准的 PSO 算法作为多目标调度策略,存在收敛速度慢,收敛精度低的缺陷,提出改进后的 MOTS-PSO 算法。通过引入非线性自适应惯性权重策略;花朵授粉算法的依概率平衡全局搜索与局部搜索机制,并对全局搜索的位置更新公式进行改进;混合萤火虫算法产生“精英”对粒子局部搜索位置进行改进以及位置扰动机制;提高了原始粒子群算法的收敛速度和收敛精度。仿真实验结果对比来看,本文提出的 MOTS-PSO 算法比标准 PSO 算法收敛精度更高,收敛 速度更快,调度策略上要优于 PSO 算法和 FA 算法。因此,本文所提出的多目标任务调度策略能够较好的解决云计算中,任务调度分配问题。

然而,本算法依然存在待改进的地方,首先粒子群算法的初始化种群分散度不够,初始化状态种群的多样性较低,其次,在解决任务调度问题的时候忽略了虚拟机部署的负载均衡问题,以及数据中心物理主机能量的消耗问题,这些问题将在以后的工作中进行进一步研究。

参考文献

[1] 黄启成,陈羽中,江伟,等. 一种面向云环境虚拟机部署的粒子群优化策略[J]. 小型微型计算机 2018, 39(7):1554-1559.
HUANG Q CH, CHEN Y ZH, JIANG W, et al. Particle swarm optimization strategy for VM placement in cloud [J]. Journal of Chinese Computer Systems, 2018, 39(7):1554-1559.

[2] MAPETU J. P. B, CHEN Z, KONG L F. Low-time complexity and low-cost binary particle swarm optimization algorithm for task scheduling and load balancing in cloud computing [J] Artificial Intelligence, 2019, 49(9):3308-3330.

[3] 刘春燕,杨巍巍. 云计算中基于遗传粒子群算法的多目标任务调度[J]. 计算机技术与发展,2017,27(2):56-59.
LIU CH Y, YANG W W. A multi-objective task scheduling based on genetic and particle swarm optimization algorithm for cloud computing [J]. Commuter Technology And Development, 2017, 27(2):56-59.

[4] 蔡晓丽,钱诚. 基于改进的粒子群算法的云资源任调度策略[J]. 微电子学与计算机,2018,35(6):28-35.
CAI X L, QIAN CH. Cloud resource scheduling strategy based on improved particle swarm optimization [J]. Micro-electronics and Computer, 2018,35(6):28-35.

[5] 杨晓光,张奇松,张益民,等. 基于改进混合粒子群算法的云计算任务调度研究[C],内蒙古大学学报(自然科学版),2019, 50(6):674-678.
YANG X G, ZHANG Q S, ZHANG Y M, et al. Research on cloud computing task scheduling problem based on improved hybrid particle swarm optimization[C]. Journal of Inner Mongolia University (Natural Science Edition), 2019, 50(6):674-678.

[6] 张海玉. 基于改进蚁群算法的云计算任务调度研究[J]. 微电子学与计算机, 2016,33(9):110-113.
ZHANG H Y. Task scheduling in cloud computing by using ant colony algorithm [J]. Micro-electronics & Computer, 2016, 33(9):110-113.

[7] 肖志娇,常会友, 衣杨. 启发式规则与 GA 结合的优化方法求解工作流动态调度优化问题[J]. 计算机科学, 2007, 34(2):157-160.
XIAO ZH J, CHANG H Y, YI Y. An optimization method of workflow dynamic scheduling based on heuristic GA [J]. Computer Science, 2007, 34(2):157-160.

[8] SOFIA A S, NICHOLAS P E. MCAMC: Minimizing the cost and of cloud service using non-dominated sorting

genetic algorithm-II [J]. Sadhana-Academy Proceedings sings in Engineering Sciences, 2019, 44(10):407-415.

[9] 钟小康. 云环境下任务调度算法的研究[C]. 赣州:江西理工大学,2018.
ZHONG X K. Research on the task scheduling algorithmin cloud computing environment[C]. Ganzhou: Jiangxi University of Science and Technology, 2018.

[10] ZHANG L, CHEN Y H, SUN R Y, et al. A task scheduling algorithm based on PSO for grid computing[J]. International Journal of Computational Intelligence Research, 4(1):37-43 (2008).

[11] GUO L, ZHAO S, SHEN S, et al. Task scheduling optimization in cloud computing based on heuristic algorithm[J]. Journal of Networks, 2012, 7(3): 547-553.

[12] 张晓丽. 自适应CPSO算法在云计算任务调度中的应用[J]. 计算机技术与发展,2016,26(8):161-165.
ZHANG X L. Application of self-adaptive chaos particle swarm optimization in task scheduling for cloud computing[J]. Computer Technology and Development, 2016,26(8):161-165.

[13] 郭文涛,卢少武. 基于优化代价的云工作流调度改进PSO算法[J]. 计算测量与控制, 2017, 25(6): 162-166.
GUO W T, LU SH W. Cloud workflow scheduling with improved particle swarm optimization algorithm based on cost optimization[J]. Computer Measurement & Control, 2017,25(6):162-166.

[14] 侯欢欢. 自适应动态调整粒子群的云计算任务调度[J]. 计算机应用与软件,2019,36(9):46-51.
HOU H H. Adaptive dynamic adjustment of particle swarm optimization for cloud computing task scheduling[J]. Computer Applications and Software, 2019, 36(9): 46-51.

[15] 张国荣,陈夏冉, 颜丽花. 基于隔离小生境粒子群算法的APF优化配置[J]. 电子测量与仪器学报,2017, 31(2):208-215.
ZHANG G R, CHEN X R, YAN L H. APF optimization allocation based on isolation niche particle swarm algorithm [J]. Journal of Electronic Measurement and Instrumentation, 2017, 31(2):208-215.

[16] 胡堂清,张旭秀,曹晓月. 一种动态调整惯性权重的混合粒子群算法[J]. 电光与控制,2020, 27(6): 16-21.
HU T Q, ZHANG X X, CAO X Y. A hybrid particle swarm optimization with dynamic adjustment of inertial weight[J]. Electronics Optics & Control, 2020, 27(6): 16-21.

[17] 王丽,王晓凯. 一种非线性改变惯性权重的粒子群算法[J]. 计算机工程与应用,2007,43(4):47-48,92.
WANG L, WANG X K. Modified particle swarm optimizer using non-liner inertia weight[J]. Computer Engineering and Application, 2007,43(4):47-48,92.

[18] 张晓莉,王秦飞, 冀汶莉. 一种改进的自适应惯性权重的粒子群算法[J]. 微电子学与计算机,2019,36(3): 66-70.
ZHANG X L, WANG Q F, JI W L. An improved particle swarm optimization algorithm for adaptive inertial weights [J]. Microelectronics Computer, 2019,36(3): 66-70.

[19] 顾清华,孟倩倩. 优化复杂函数的粒子群-鸽群混合优化算法[J]. 计算机工程与应用,2019,55(22):46-52.
GU Q H, MENG Q Q. Hybrid particle swarm optimization and pigeon-inspired optimization algorithm for solving complex functions[J]. Computer Engineering and Applications, 2019, 55(22):46-52.

[20] ZHOU Y Q, WANG R, LUO Q F. Elite opposition-based flower pollination algorithm [J]. Nerocomputing, 2016,188(S1):294-310.

[21] 卞京红, 贺兴时,杨新社. 基于萤火虫算法的自适应花粉优化算法[J]计算机工程与应用,2016,52(21): 162-167.
BIAN J H, HE X SH, YANG X SH. Hybrid algorithm of firefly algorithm and self-adaptive flower pollination algorithm[J]. Computer Engineering and Applications, 2016,52(21):162-167.

[22] YANG X S. Nature-Inspired Metaheuristic Algorithms [M]. Forme: Luniver Press, 2008.

[23] ALGULIYEV R M, IMAMVERDIYEV Y N, ABDULLAYEVA F J. PSO-based load balancing method in cloud computing [J]. Automatic Control and Computer Sciences, 2019, 53(1):44-55.

[24] 林涛,王昊, 李鹏. 基于改进差分进化算法的云计算任务调度策略[J]. 传感器与微系统,2019,38(9): 24-27.
LI T, WANG H, LI P. Cloud computing task scheduling strategy based on improved differential evolution algorithm[J]. Transducer and Microsystem Technologies, 2019, 38(9):24-27.

作者简介

马学森,2016年于合肥工业大学获得博士学位,现为合肥工业大学副教授,硕士生导师,主要研究方向为无线传感器网络、移动边缘计算。
E-mail:mxs@hfut.edu.cn

Ma Xuesen received Ph. D. degree from Hefei University of Technology in 2016. Now he is an associate

professor and a M. Sc. supervisor at Hefei University of Technology. His main research interests include wireless sensor network, multi-access edge computing.

谈杰,2017 年于北华大学获得学士学位,现为合肥工业大学硕士研究生,主要研究方向为高可靠性嵌入式系统、移动边缘计算、云计算。
E-mail:1406002732@ qq. com

Tan Jie received B. Sc. degree from Beihua University in 2017. Now he is a M. Sc. candidate at Hefei University of Technology. His main research interests include high-reliability embedded system, multi-access edge computing, cloud computing.

陈树友,2018 年于合肥师范学院获得学士学位,现为合肥工业大学硕士研究生,主要研究方向为分布式实时系统、移动边缘计算。
E-mail:1156796454@ qq. com

Chen Shuyou received B. Sc. degree from Hefei Normal University in 2018. Now he is a M. Sc. candidate at Hefei University of Technology. His main research interests include distributed real-time system, mobile edge computing.

储昭坤,2019 年于安徽工程大学获得学士学位,现为合肥工业大学硕士研究生,主要研究方向为分布式实时系统、移动边缘计算。
E-mail:11503011643@ qq. com

Chu Zhaokun received B. Sc. degree from Anhui Polytechnic University in 2019. Now he is a M. Sc. candidate at Hefei University of Technology. His main research interests include distributed real-time systems, mobile edge computing.

石雷,2012 年于合肥工业大学获得博士学位,现为合肥工业大学副教授,硕士生导师,主要研究方向为无线传感器网络、无线通信。
E-mail:shilei@ ialab. hfut. edu. cn

Shi Lei received Ph. D. degree from Hefei University of Technology in 2012. Now he is an associate professor and a M. Sc. supervisor at Hefei University of Technology. His main research interests include wireless sensor network and wireless communications.