文章编号：2095-2163(2024)02-0028-07 中图分类号：TP391 文献标志码：A

基于改进灰狼算法的云资源调度策略研究

苏鸿斌
(浙江理工大学 计算机科学与技术学院，杭州 330018)

摘 要：针对云计算中任务调度考虑因素单一和大规模任务环境下任务存在调度效率低、分配不合理等问题，本文提出了一种基于改进的灰狼优化算法的云计算任务调度策略。通过建立基于多目标的评价模型，使其在单一适应度中处理多目标，其中包含任务总耗时、功耗以及系统负载度；提出一种将粒子群算法和灰狼优化算法相结合的搜索方法，以增强灰狼层次的全局最优搜索；引入自适应权重以增强灰狼优化算法的局部搜索能力；同时引入随机对立学习策略以避免陷入局部最优。将本文提出的改进算法与粒子群优化算法(PSO)、标准灰狼优化算法(GWO)及基于 MakeSpan 适应度的灰狼优化算法(MGWO)在 CloudSim 平台进行对比实验。仿真结果表明，该方法适用于大规模任务调度，且在任务完成总耗时、功耗以及系统的负载均衡度方面较 PSO、GWO 和 MGWO 均有明显提升，其中较 MGWO 算法综合提升 14%。

关键词：云计算；任务调度；灰狼优化；自适应；随机对立

Cloud source scheduling strategy based on improved Grey Wolf Optimization Algorithm

SU Hongbin
(School of Computer Science and Technology, Zhejiang Sci-Tech University, Hangzhou 330018, China)

Abstract: Aiming at the problems of single consideration of task scheduling in cloud computing and low scheduling efficiency and unreasonable allocation of tasks in large-scale task environment, this paper proposes a cloud computing task scheduling strategy based on improved Grey Wolf Optimization Algorithm. Firstly, an evaluation model based on multi-objectives is established to deal with multi-objectives in a single fitness, which includes the total time consumption, power consumption and system load. Secondly, a search method combining particle swarm optimization algorithm and Grey Wolf Optimization Algorithm is proposed to enhance the global optimal search at the Grey Wolf level. Thirdly, the adaptive weight is introduced to enhance the local search ability of the Grey Wolf Optimization Algorithm. At the same time, a stochastic opposition-based learning strategy is introduced to avoid falling into local optimum. Finally, the improved algorithm proposed in this paper is compared with Particle Swarm Optimization algorithm (PSO), standard Grey Wolf Optimization Algorithm (GWO) and Grey Wolf Optimization Algorithm based on MakeSpan fitness (MGWO) on CloudSim platform. The simulation results show that the proposed method is suitable for large-scale task scheduling, and the total time consumption of task completion, power consumption and load balancing degree of the system are significantly improved compared with PSO, GWO and MGWO.

Key words: cloud computing; task scheduling; Grey Wolf Optimization(GWO); self-adaption; opposition learning

0 引 言

云计算被认为是 IT 领域最突出的技术，其适用于各种领域，如：医疗保健、商业贸易、智能体系、移动端系统等等。由于竞争对手和世界的需要，云计算领域在最近一段时间得到了快速的发展。通过虚拟化的概念，云计算资源分布在客户端内部，将几个远程环境聚集在一起，并充分利用其能力[1]。云计算提供了 3 种服务：一是基础设施即服务(IaaS)，其为云用户提供各种用途的基础设施，如：存储系统和计算资源；二是平台即服务(PaaS)，其向客户端提供平台，以便用户可以在该平台上开发应用程序；第三种是软件即服务(SaaS)[2]，即向用户提供软件，用户不需要在自己的机器上安装软件，可以直接通

基金项目：浙江省重点研发计划项目(2020C03094)；浙江省教育厅一般科研项目(Y202147659)；浙江省教育厅项目(Y202250706)；浙江省基础公益研究计划项目(QY19E050003)。

作者简介：苏鸿斌(1997-)，男，硕士研究生，主要研究方向：计算机网络与分布式处理。Email：908377408@qq.com

收稿日期：2023-02-21

过云使用软件[3]。任务调度是云计算中最重要、最关键的问题之一，许多研究都试图找到云环境中对现有资源进行任务调度的最优解决方案。

云任务调度被称为 NP 完全问题[4]。更准确地说，检测解决方案所需的时间随问题大小[5]而变化。NP 类问题的解可以通过枚举技术、启发式或元启发式技术得到。枚举技术方法通常不被首选，因为其需要构建所有可能的任务调度程序，然后比较每个任务调度程序，以达到最佳解决方案[6]。启发式搜索技术以较高的操作成本生成可行的解决方案，在考虑成本的前提下可以完全地忽略该类技术[7]。因此，采用元启发式技术方法是唯一现实可行的解决方案。例如，遗传算法(GA)[8]、粒子群优化算法(PSO)[9]或灰狼优化算法(GWO)[10]，这些算法使用候选解决方案池遍历可用的解决方案空间。此外，适应性探索行为、小控制参数等因素也推动了 GWO 在本研究工作中的使用[11]。

灰狼优化算法展示了狼的实际生存本能，狼在捕获猎物时的合作本性，以及狼的领导等级和狩猎本性。此外，其还包含了狩猎的 3 个主要步骤：寻找猎物、包围和攻击猎物[12]。由此可见，灰狼优化算法在寻找优化问题通用解方面存在实现的可能性，值得进一步分析和发展。

综上所述，本文在云任务调度中采用改进的灰狼优化算法实施调度。首先，构建针对多目标的评价模型；其次，对灰狼优化算法进行层次结构改进、自适应权重改进和随机对立学习策略改进，增强算法全局最优搜索能力的同时优化局部搜索能力，并有效避免算法陷入局部最优解；最后，仿真实验表明，本文提出的算法在任务总耗时、任务执行完成后系统功耗以及任务执行时系统的负载均衡度上得到较好的提升。

1　云计算任务调度

在云计算领域中，考虑到不同任务对不同资源的需求存在差异[13]，因此有必要简要模拟任务成本的细节，并解释不同资源成本和用户预算成本之间的联系。针对该问题，本文提出一种资源开销模型，将资源开销模型分解为 CPU、内存、带宽 3 部分。根据资源定义，在资源成本模型的基础上，提出了多目标优化调度模型，以实现云计算环境下的多目标优化调度。

1.1　数学模型

本文中，假设有 n 个任务，这些任务应该在 m 个计算资源(也称虚拟机节点)上进行处理，最终目标是使执行时间最小化，并优化 CPU、内存及带宽的资源利用。当任务数小于资源节点时，按照先到先得算法分配任务；如果任务数大于资源节点，则根据调度算法进行分配。此外，不允许任务在资源之间迁移，即一个任务只能被分配到一个计算资源节点。为了表述问题，定义任务集 $T_i = \{t_1, t_2, t_3, \cdots, t_n\}$ 为 n 个独立任务，$R_i = \{r_1, r_2, r_3, \cdots, r_n\}$ 为资源节点。因此，计算资源与任务的关系表示如下：

$$P = \begin{pmatrix} p_{11} & \cdots & p_{1n} \\ \vdots & \ddots & \vdots \\ p_{m1} & \cdots & p_{mn} \end{pmatrix} \tag{1}$$

式中：元素 P_{ji} 代表一个计算资源节点 R_j 与一个任务 T_i 之间的对应关系。当任务 T_i 在计算资源 R_j 上运行时，P_{ji} 为 1，反之为 0，即 $P_{ji} \in \{0,1\}$。

1.2　资源成本模型

在云计算领域，任务和资源是相互矛盾的[14]。不同资源的成本是不同的，这也导致了任务成本的差异。针对这一问题，本文提出了一种资源成本模型，该模型结合 CPU 处理能力、内存及网络带宽，给出任务总耗时 $RT_{\max}$、功耗 E 以及系统负载度 B_d 3 个优化目标。

1.2.1　任务总耗时 $RT_{\max}$

每个任务的处理时间因各个计算资源节点的 CPU 运算力而异。第 i 个任务在计算资源节点 j 上的处理时间可以表示为

$$RT_{ji} = \frac{L_i}{R_{Cj}} \tag{2}$$

式中：L_i 为任务 i 的长度，R_{Cj} 表示计算资源节点 j 的 CPU 运算力，则计算资源节点 j 处理当前节点上所有任务的完成时间为

$$RT_j = \sum_{i=1}^{n} P_{ji} RT_{ji} \tag{3}$$

因系统总耗时 $RT_{\max}$ 为各计算资源完成各自任务所消耗时间的最大值，则计算公式如下

$$RT_{\max} = \mathrm{MAX}(RT_j),\ 1 \leqslant j \leqslant m \tag{4}$$

1.2.2　功耗 E

完成任务所需功耗与 CPU 的利用率相关[15]，系统功耗为各个计算资源的功耗相加。根据能耗计算公式 $E = P \times T$，则系统的总功耗 E 为

$$E = \sum_{j=1}^{m} R_{OCj} RT_j \tag{5}$$

式中：R_{OCj} 和 RT_j 分别表示计算资源 j 的 CPU 利用

率和任务完成时间。其中，R_{OCj} 计算公式可表示为

$$R_{OCj} = \frac{\sum_{i=1}^{n} P_{ji} T_{Ci}}{R_{Cj}} \times 100\% \tag{6}$$

1.2.3 系统负载度 B_d

任务被执行过程中，计算资源节点的负载度主要是由当前节点被分配的任务数量及节点的运算能力所决定的。取 R_{Cj}、R_{Mj}、R_{Bj} 分别表示计算节点 j 上的 CPU、内存和带宽的可利用情况；T_{Ci}、T_{Mi}、T_{Bi} 分别表示任务 i 相对 CPU、内存和带宽所需的资源数。则计算节点 j 的利用率为：

$$R_{Oj} = \frac{k_1 \sum_{i=1}^{n} P_{ji} T_{Ci} + k_2 \sum_{i=1}^{n} P_{ji} T_{Mi} + k_3 \sum_{i=1}^{n} P_{ji} T_{Bi}}{R_{Cj} + R_{Mj} + R_{Bj}} \times 100\% \tag{7}$$

$$k_1 + k_2 + k_3 = 1 \tag{8}$$

式中：k_1、k_2、k_3 分别表示 CPU、内存和带宽的权值。以各个计算资源之间利用率的标准差系数反映出系统的负载情况，则系统负载度 B_d 为：

$$R_{O\text{avg}} = \frac{\sum_{j=1}^{m} R_{Oj}}{m} \times 100\% \tag{9}$$

$$B_d = \sqrt{\frac{\sum_{j=1}^{m} (R_{Oj} - R_{O\text{avg}})^2}{m - 1}} \tag{10}$$

式(9)中，$R_{O\text{avg}}$ 为整个系统的资源平均利用率。

2 任务调度策略

2.1 灰狼优化算法

在自然界中，灰狼是群居的，并且具有严格的等级制度，如图 1 所示。

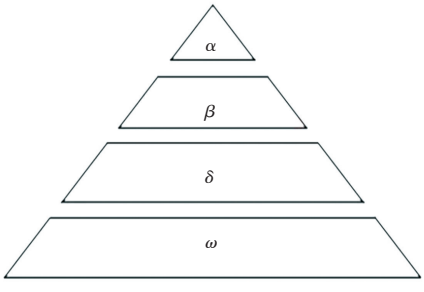

图 1 灰狼的社会等级制度

Fig. 1 Social hierarchy of Gray Wolf

根据狼群的社会等级，群体中的灰狼主要分为 *Alpha*(α)、*Beta*(β)、*Delta*(δ) 和 *Omega*(ω) 4 个等级，每个等级各司其职。灰狼的另一个主要社会特征是群体狩猎，包括跟踪猎物、包围猎物和攻击猎物[16] 3 个主要步骤。其中，最优解为 α，第二和第三优解分别为 β 和 δ，其余解为 ω。在 GWO 中，捕猎(优化)由 α、β 和 δ 引导，而 ω 只是跟随。包围过程的数学模型如下：

$$D = |\boldsymbol{C} \cdot \boldsymbol{X}_p(t) - \boldsymbol{X}(t)| \tag{11}$$

$$\boldsymbol{X}(t+1) = \boldsymbol{X}_p(t) - \boldsymbol{A} \cdot D \tag{12}$$

式(11)中：t 表示当前迭代，$\boldsymbol{X}_p$ 为猎物的位置向量，$\boldsymbol{X}$ 为灰狼的位置向量；式(12)中，$\boldsymbol{A}$、$\boldsymbol{C}$ 为系数向量，$\boldsymbol{A}$ 和 $\boldsymbol{C}$ 的计算方法如下：

$$\boldsymbol{A} = 2a \cdot r_1 - a \tag{13}$$

$$\boldsymbol{C} = 2 \cdot r_2 \tag{14}$$

其中，a 是收敛因子，并随着迭代次数从 2 线性减小到 0，r_1 和 r_2 的模取[0,1]之间的随机数。

攻击行为的数学模型如下：

$$\begin{aligned} D_\alpha &= |C_1 \cdot X_\alpha - \boldsymbol{X}| \\ D_\beta &= |C_2 \cdot X_\beta - \boldsymbol{X}| \\ D_\delta &= |C_3 \cdot X_\delta - \boldsymbol{X}| \end{aligned} \tag{15}$$

式中：X_α、X_β 和 X_δ 是当前迭代时，群体中排前三的最佳解，C_1、C_2 和 C_3 由公式(14)计算得到。

$$\begin{aligned} X_1 &= X_\alpha - A_1 \cdot (D_\alpha) \\ X_2 &= X_\beta - A_2 \cdot (D_\beta) \\ X_3 &= X_\delta - A_3 \cdot (D_\delta) \end{aligned} \tag{16}$$

$$X(t+1) = \frac{X_1 + X_2 + X_3}{3} \tag{17}$$

式(16)中：X_1、X_2、X_3 分别表示受 α、β 和 δ 影响，ω 个体调整后的位置；式(17)定义了 ω 的最终位置。

如上所述，ω 在 α、β 和 δ 的领导下更新其位置。在 GWO 中，两个向量 $\boldsymbol{A}$ 和 $\boldsymbol{C}$ 是随机向量，GWO 需要在搜索空间中进行探索和利用。当 $|\boldsymbol{A}| > 1$ 时，GWO 强调搜寻，并允许 GWO 勘探整个空间寻找猎物(最优解)；当 $|\boldsymbol{A}| < 1$ 时，GWO 专注于攻击。GWO 的另一个有利于探索的组件是 $\boldsymbol{C}$。从式(14)可以看出，$\boldsymbol{C}$ 是范围在[0,2]中的随机值。该分量为猎物提供随机权重，以便随机强调($|\boldsymbol{C}| > 1$) 或弱化($|\boldsymbol{C}| < 1$)猎物对式(11)中距离定义的影响。

2.2 改进灰狼优化算法

2.2.1 基于灰狼优化算法层次结构的改进

在 GWO 算法中，狼被分为 4 组：α、β、δ 和 ω，同时 α、β、δ 的位置更新方式相同，并强制其他狼 ω 根据 α、β、δ 的位置重新定位。然而，α 是最好的搜索智能体，对猎物的位置有最好的引导。β 和 δ 的位置都没有达到 α 的位置，因此如果 α 采用与 β 和 δ 相同的更新方式，就会出现 α 更新后的位置被 β 和

δ 误导的情况。同样的，δ 的位置不如 β，可能无法引导 β，此时群体会以较低的效率找到最优解，容易陷入局部最优。因此，本文受 PSO 中群体智能概念的启发，提出了一种基于增强狼群层次结构的 GWO 算法(PGWO)。PSO 算法是一种经典的 SIA 算法，其通过对自身最优位置和群体中最优个体进行学习，能够在复杂的空间中搜索到全局最优。在 GWO 算法中，每只狼有两种更新方式，一种是个体采用基于 PSO 算法的全局最优搜索方式，即增强狼群的层次结构；另一种是个体采用 GWO 的更新方式。每只灰狼根据自己的社会等级选择自己的更新方式。

在 PGWO 中，一组有 n 个解，狼也被分为 4 组：α、β、δ 和 ω。本文在此使用 $S_i(t)$ 记录每只狼的水平。如果狼属于 α，令 $S_i(t)=1$。同样，如果狼属于 β、δ 和 ω，则分别为 $S_i(t)=2$，$S_i(t)=3$，$S_i(t)=4$。然后引入参数 $DF_i(t)$ (Decision Factor)，使每只灰狼决定选择哪种更新方式来更新其位置。$DF_i(t)$ 的计算公式为

$$DF_i(t)=\frac{L-S_i(t)}{L-1} \tag{18}$$

式中，L 是狼群等级的总数，狼群有 4 级，因此 L 为 4。如果狼属于 α，$DF_i(t)=1$。类似地，对于 β、δ 和 ω，$DF_i(t)$ 的值分别为 0.67、0.33 和 0。

所以，灰狼根据 $DF_i(t)$ 的值决定是否猎取猎物。其中，r 是 0～1 之间的随机数，如果 $r \leqslant DF_i(t)$，灰狼会选择第一种更新模式来重新定位；否则，灰狼会选择跟随更高级别的狼来更新自己的位置。显然，由于 α 狼的 $DF_i(t)$ 值为 1，而 r 总是小于等于 1，α 狼只能选择第一种模式重新定位。对于 ω 只狼，$DF_i(t)$ 的值是 0，r 总是大于等于 0，所以 ω 只狼也只能选择第二种模式重新定位。这 4 组狼会以不同的方式更新自己的位置，具体描述如下：

(1) 如果 $DF_i(t)=1$，$X_i(t)$ 的位置更新公式为

$$X_i(t+1)=X_i(t)+a\times(X_{i\mathrm{Best}}-X_i(t))+a\times(X_{g\mathrm{Best}}-X_i(t)) \tag{19}$$

式中：$X_{i\mathrm{Best}}$ 表示该狼的历史最佳值，$X_{g\mathrm{Best}}$ 表示狼群的历史最佳值，a 为式(13)中的收敛因子。

(2) 如果 $DF_i(t)=2$，$X_i(t)$ 的位置更新公式为：

$$\begin{cases} X_i(t+1)=X_1, & r>0.67 \\ X_i(t+1)=\dfrac{X_1+X_2}{2}, & r\leqslant 0.67 \end{cases} \tag{20}$$

(3) 如果 $DF_i(t)=3$，$X_i(t)$ 的位置更新公式为：

$$\begin{cases} X_i(t+1)=\dfrac{X_1+X_2}{2}, & r>0.33 \\ X_i(t+1)=\dfrac{X_1+X_2+X_3}{3}, & r\leqslant 0.33 \end{cases} \tag{21}$$

(4) 如果 $DF_i(t)=4$，$X_i(t)$ 的位置更新公式为式(16)。式(19)～式(21)中，X_1、X_2 和 X_3 的值来自于式(16)。

因为 ω 狼只能选择第二种更新方式，所以其更新位置的方式和 GWO 算法相同。在传统 GWO 中，狼群均使用式(15)－(17)更新位置，α 狼受到 β 狼和 δ 狼的影响，如果 β 狼和 δ 狼是次一等的狼，这就导致 α 狼会向次狼的方向更新，影响求解的准确性。因此，第一种更新模式的动机是对较高层次的狼采用全局最优搜索方式，为较高级的狼提供更多倾向于全局最优的行为，以促进其探索能力和利用能力。这种更新模式增强了狼群的层次性，避免了等级较高的狼群搜索能力的退化，使种群能够以更高的效率和精度捕获猎物。

2.2.2　基于灰狼优化算法自适应权重的改进

原始 GWO 算法中，狼群的位置更新只受 α、β 和 δ 三只狼的位置影响[17]，并且其对狼群的影响效果是一致的，而不是充分发挥 α 狼作为最优狼的作用，进而导致算法的收敛速度下降。因此，本文提出一种基于适应度值的自适应权重，加强优秀狼体的领导作用。

$$\begin{aligned} w_1&=1-\frac{|f_\alpha|}{|f_\alpha|+|f_\beta|+|f_\delta|} \\ w_2&=1-\frac{|f_\beta|}{|f_\alpha|+|f_\beta|+|f_\delta|} \\ w_3&=1-w_1-w_2 \end{aligned} \tag{22}$$

式中：f_α、f_β 和 f_δ 分别是 α 狼、β 狼和 δ 狼的适应度值。根据式(17)可以改写为

$$X(t+1)=w_1\times X_1+w_2\times X_2+w_3\times X_3 \tag{23}$$

2.2.3　基于灰狼优化算法随机对立学习策略的改进

虽然采用了灰狼层级增强，但 PAGWO 的全局搜索能力仍然不足。为了进一步提高狼的探索能力，随机选择灰狼采用对立学习策略更新其位置，称为随机对立学习策略。2005 年，Tizhoosh[18] 提出了对立学习的概念，旨在防止算法陷入局部最优。之后，对立学习策略被广泛应用于许多优化算法的改进中。如：改进的带学习操作的全局最佳引导 PSO

(IGPSO)[19]。Dong[20]等人提出了一种自适应变异的对立粒子群算法(Opposition-Based PSO, PSO),并采用广义对立学习(Generalized Opposition-Based Learning, GOBL)来增强其全局能力。对立学习策略的主要思想,是在群体中寻找可行解的逆解,从而引导狼群寻找全局最优解。随机反向学习策略在数学上可用如下公式表示:

$$X_r = lb + (ub - X_\alpha) \tag{24}$$

式中:X_r 为随机从 ω 狼群中选择一只狼进行对立学习后的新位置,ub 和 lb 分别表示搜索空间的上下界。

本研究使用随机策略,从 ω 狼群中随机选择一只狼进行对立学习,既不影响种群质量,又能提高算法的探索能力。

3 基于改进灰狼的云任务调度设计

3.1 狼群与适应度函数设计

设 n 为任务的数量,m 为计算资源的数量。每种任务分配方案就是一只狼,设 $\boldsymbol{P} = \{p_1, p_2, p_3, \cdots, p_n\}$ 为解向量。通过采用 Z-score 标准化方法,对本研究提到的 3 个目标参数($RT_{\max}$、E、B_d)归一化,转化为单目标优化问题:

$$F = \frac{f(x) - f_\mu}{f_\sigma} \tag{25}$$

式中:F 代表归一化后的值,$f(x)$ 表示某一优化目标的参数值,f_μ 为该优化目标的均值,f_σ 为该优化目标的标准差。

通过对各个目标的适应度值添加权重,将多目标转化为单目标。适应度函数如下所示:

$$f_i = k_1 F_{RT} + k_2 F_E + k_3 F_B \tag{26}$$

式中:f_i 表示第 i 只狼的适应度值,k_1、k_2、k_3 分别是总耗时 $RT_{\max}$、功耗 E 和系统负载度 B_d 等 3 个目标所对应的权值,且 $k_1 + k_2 + k_3 = 1$。

3.2 改进的灰狼优化算法任务调度流程

本文提出的 PARGWO 算法的流程如图 2 所示:

实验步骤如下:

Step 1 初始化灰狼种群和参数 α、$\boldsymbol{A}$ 和 $\boldsymbol{C}$,将任务调度方案跟狼的位置一一对应。

Step 2 计算每只灰狼的适应度值,并记最好的 3 匹狼为 X_α、X_β 和 X_δ。

Step 3 根据狼群位置计算 α、$\boldsymbol{A}$ 和 $\boldsymbol{C}$。

Step 4 判断当前灰狼是否为随机狼,是则执行步骤 5,否则跳至步骤 6。

Step 5 执行 PARGWO 算法更新灰狼位置。

Step 6 执行不含对立学习的 PAGWO 算法,对灰狼位置进行跟新。

Step 7 判断是否达到最大迭代次数,是则转到步骤 8,否则跳回步骤 2。

Step 8 得到最优狼的位置。

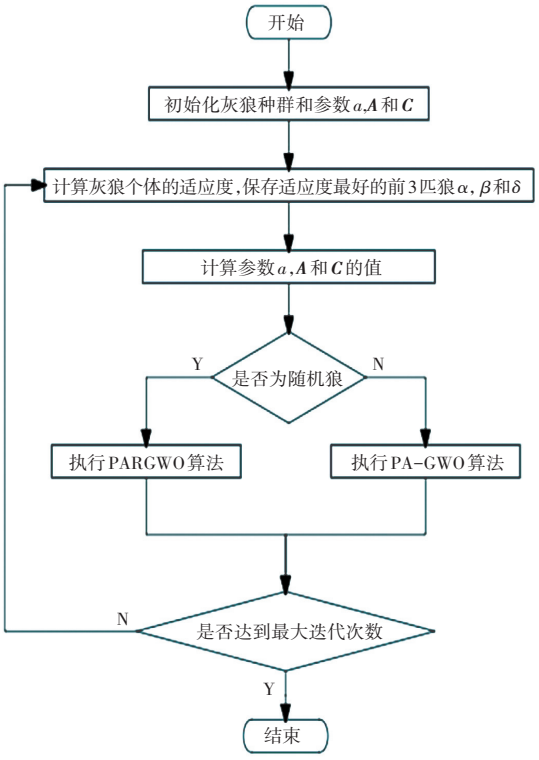

图 2 PARGWO 流程图

Fig. 2 Flow chart of PARGWO

4 实验结果分析

本文利用 CloudSim 工具对所提算法进行仿真,该工具包的基础平台依赖于 JAVA。实验在 AMD Ryzen 5 5600X、3.7 GHz、32 GB RAM 和 Windows10 操作系统配置的 PC 机上进行。从总耗时、功耗和负载均衡度三方面对算法进行了仿真,并与 PSO 算法、标准 GWO 算法以及文献[21]中提出的 MGWO 算法进行对比实验。相关的实验参数设置见表 1。

表 1 实验主要参数配置

Table 1 Main parameter configuration

参数名称	取值
计算资源节点数	20
节点 CPU 运算力	[500,2 000] mi/s
节点内存	[512,2 048] MB
节点带宽	[1 000,10 000] bit/s
任务长度	[100,1 000]
迭代次数	200

实验设定 T_1 和 T_2 两个任务集，其中 T_1 为小规模任务集，任务数量在 10~100 之间；T_2 为大规模任务集，任务数量在 1 000~5 000 之间。随机设置任务所需内存和带宽资源数，其中每个任务对内存需求在[50,500]MB 之间，每个任务对带宽的需求在[200,5 000] mi/s 之间。为了使本实验更加可靠有效，分别在大小两个规模任务集下，对任务总耗时、功耗和负载均衡 3 个方面做出对比。

4.1　任务总耗时对比

本小节将 PSO 算法、GWO 算法、MGWO 算法和本文提出的 PARGWO 算法，分别在大、小两种规模任务集下进行实验。根据图 3、图 4 可以看出，4 种调度策略在任务数量较少时差异并不大，但在大规模任务集的实验下，MGWO 和 PARGWO 算法的完成时间明显优于 PSO 和 GWO 算法，而本文的 PARGWO 算法的任务总耗时明显低于 MGWO 算法，说明 PARGWO 算法在减少任务总耗时方面有不错的效果。

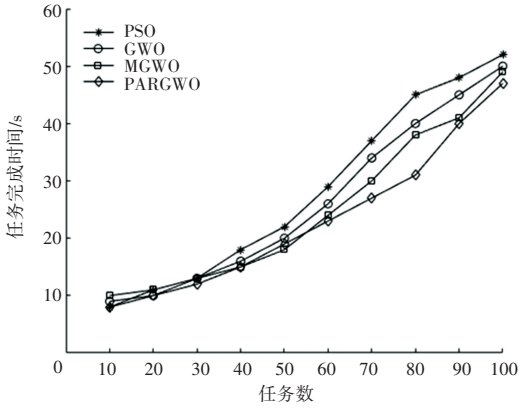

图 3　小规模任务下完成时间对比图

Fig. 3　Comparison of time under small-scale tasks

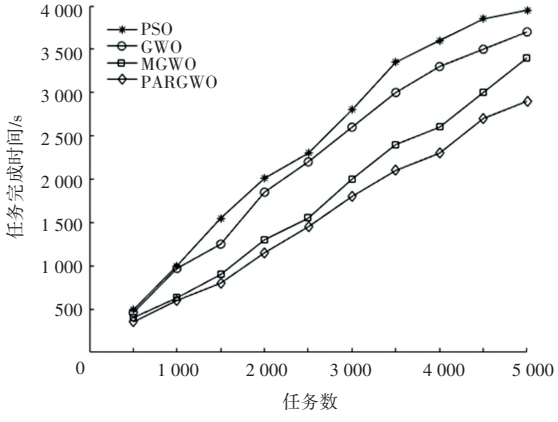

图 4　大规模任务下完成时间对比图

Fig. 4　Comparison of time under large-scale tasks

4.2　任务完成功耗比较

系统功耗的对比实验结果如图 5、图 6 所示。可以看出，在小规模任务量下，MGWO 和 PARGWO 两种策略虽然与 PSO 和 GWO 两种策略的差异不大，但随着任务量的增加，在大规模任务集的实验中，MGWO 和 PARGWO 两种策略的优越性逐渐得到体现，而 PARGWO 的功耗更是明显低于 MGWO 算法的功耗，说明本研究改进的算法对功耗的减少有所提升。

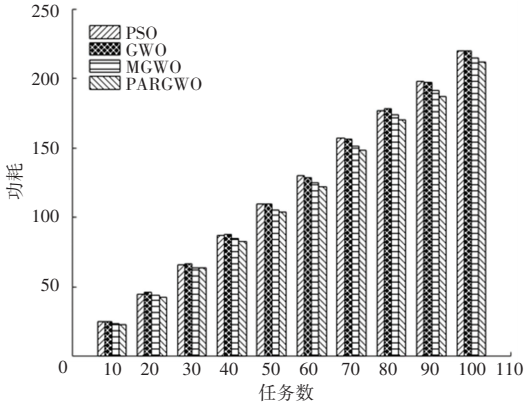

图 5　小规模任务下功耗对比图

Fig. 5　Comparison of power consumption under small-scale tasks

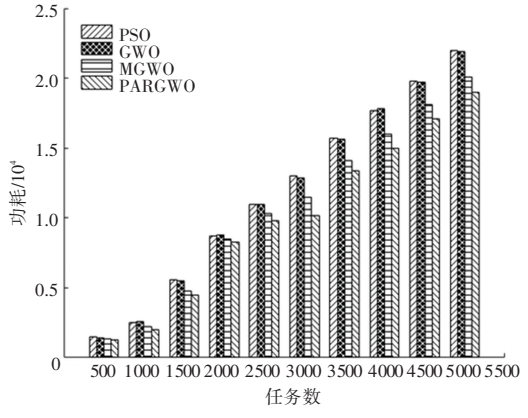

图 6　大规模任务下功耗对比图

Fig. 6　Comparison of power consumption under large-scale tasks

4.3　任务完成负载均衡度比较

系统负载度的对比实验结果如图 7、图 8 所示。由此可以看出，小规模任务集下，PSO 算法和 GWO 算法之间的系统负载度差异不大，MGWO 算法和 PARGWO 算法之间差异也不大，而 MGWO 和 PARGWO 算法略优于 PSO 和 GWO 算法。在大规模任务集下，随着任务量的增加，MGWO 和 PARGWO 算法逐渐与 PSO 和 GWO 算法拉开差距，而 PARGWO 的系统负载度一直优于 MGWO 算法。因此本文改进算法在系统负载度方面也有所提升。

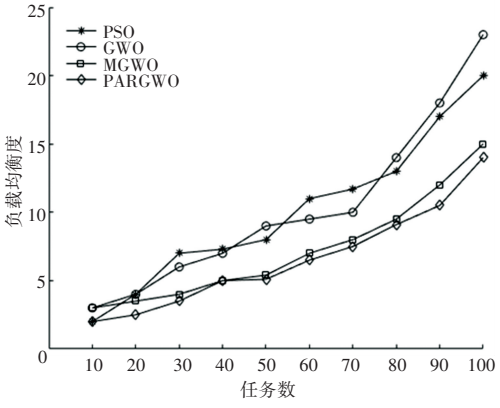

图 7 小规模任务下系统负载度对比图

Fig. 7 Comparison of system load under small-scale tasks

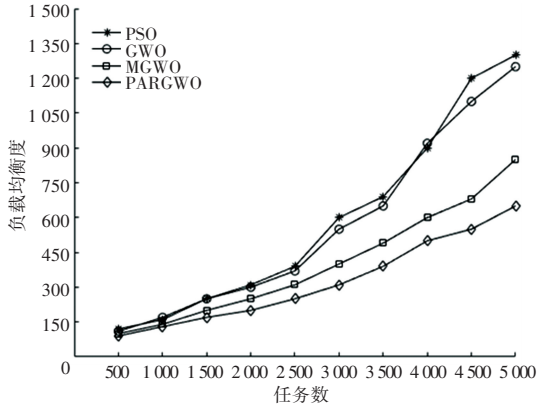

图 8 大规模任务下系统负载度对比图

Fig. 8 Comparison of system load under large-scale tasks

5 结束语

为提高 GWO 算法处理大规模问题的能力,提出一种基于增强狼群层次结构、自适应权重和随机对立学习的 GWO 算法(PARGWO)。该方法利用狼群加强层次结构的方式使种群在搜索过程中不发生退化,利用自适应权重提高头狼在种群中的带领作用,引入随机对立学习策略来避免算法陷入局部最优。仿真实验结果表明,PARGWO 算法在服务质量(任务总耗时、功耗、系统负载度)方面具有更好的性能。特别是在大规模任务量的情况下,PARGWO 较于 MGWO 在任务总耗时、功耗、负载均衡度 3 方面分别提升 5%、14%和 23%。在未来的工作中, PARGWO 算法可以很好地应用于大规模云计算环境下的任务调度问题。

参考文献

[1] 田倬璟,黄震春,张益农. 云计算环境任务调度方法研究综述[J]. 计算机工程与应用,2021,57(2):1-11.

[2] 武志学. 云计算虚拟化技术的发展与趋势[J]. 计算机应用,2017,37(4):915-923.

[3] 马小晋,饶国宾,许华虎. 云计算中任务调度研究的调查[J]. 计算机科学,2019,46(3):1-8.

[4] KUMAR M, SHARMA S C. PSO-based novel resource scheduling technique to improve QoS parameters in cloud computing[J]. Neural Computing and Applications, 2020, 32(16): 12103-12126.

[5] PENG Z, LIN J, CUI D, et al. A multi-objective trade-off framework for cloud resource scheduling based on the deep Q-network algorithm[J]. Cluster Computing, 2020, 23(4): 2753-2767.

[6] GOBALAKRISHNAN N, ARUN C. SIS: A scheme for dynamic independent task scheduling in a cloud environment [C]// Proceedings of the 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT). IEEE, 2016: 272-277.

[7] SHI L, ZHANG Z, ROBERTAZZI T. Energy-aware scheduling of embarrassingly parallel jobs and resource allocation in cloud[J]. IEEE Transactions on Parallel and Distributed Systems, 2016, 28(6): 1607-1620.

[8] GUTIERREZ-GARCIA J O, SIM K M. GA-based cloud resource estimation for agent-based execution of bag-of-tasks applications [J]. Information Systems Frontiers, 2012, 14(4): 925-951.

[9] 周丽娟,王春影. 基于粒子群优化算法的云计算资源调度策略研究[J]. 计算机科学,2015,42(6):279-281,292.

[10] 张晓凤,王秀英. 灰狼优化算法研究综述[J]. 计算机科学,2019,46(3):30-38.

[11] NATESAN G, CHOKKALINGAM A. An improved Grey Wolf Optimization Algorithm based task scheduling in cloud computing environment[J]. The International Arab Journal of Information Technology, 2020, 17(1): 73-81.

[12] 龙文,蔡绍洪,焦建军,等. 一种改进的灰狼优化算法[J]. 电子学报,2019,47(1):169-175.

[13] 卢莎莎,肖海力,王小宁. 容器技术在高性能计算环境中的应用[J]. 数据与计算发展前沿,2021,3(6):118-126.

[14] MIRJALILI S, SAREMI S, MIRJALILI S M, et al. Multi-objective Grey Wolf optimizer: A novel algorithm for multi-criterion optimization [J]. Expert Systems with Applications, 2016, 47: 106-119.

[15] DINESH REDDY V, GANGADHARAN G R, RAO G. Energy-aware virtual machine allocation and selection in cloud data centers [J]. Soft Computing, 2019, 23(6): 1917-1932.

[16] 郭振洲,刘然,拱长青,等. 基于灰狼算法的改进研究[J]. 计算机应用研究,2017,34(12):3603-3606,3610.

[17] JOSHI H, ARORA S. Enhanced Grey Wolf Optimization Algorithm for global optimization[J]. Fundamenta Informaticae, 2017, 153(3): 235-264.

[18] TIZHOOSH H R. Opposition-based learning: a new scheme for machine intelligence[C]//Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce. IEEE, 2005: 695-701.

[19] 王皓,欧阳海滨,高立群. 一种改进的全局粒子群优化算法[J]. 控制与决策,2016,31(7):1161-1168.

[20] DONG W, KANG L, ZHANG W. Opposition-based particle swarm optimization with adaptive mutation strategy [J]. Soft Computing, 2017, 21(17): 5081-5090.

[21] ALZAQEBAH A, AL-SAYYED R, MASADEH R. Task scheduling based on modified Grey Wolf optimizer in cloud computing environment[C]//Proceedings of the 2019 2nd International Conference on New Trends in Computing Sciences (ICTCS). IEEE, 2019: 1-6.