



基于格分布方差的多目标云 workflow 调度算法

包晓安¹, 曹云棣¹, 张娜¹, 钱俊彦², 曹建文³

(1. 浙江理工大学, 浙江 杭州 310018; 2. 桂林电子科技大学, 广西 桂林 541004;
3. 中国科学院软件研究所, 北京 100190)

摘要: 提出了基于格分布方差的多目标云 workflow 调度算法和差粒子自学习策略。首先, 考虑任务调度的特性, 进行粒子编码离散化。其次, 利用 Pareto 最优 workflow 调度解集映射到自适应网格坐标系的策略, 计算网格坐标系中每个 Pareto 最优解的格分布量。再次, 采用格分布方差评估当前 Pareto 前端的多样性程度, 并动态调整进化策略。最后, 设计了差粒子自学习策略。仿真实验表明, 通过该算法得到的 workflow 调度解集, 在 IGD 和 S 性能指标上均优于 MOPSO 算法, 在最优值方面优于 ϵ -FDPSO 和 NSGA-II 算法。

关键词: 云 workflow 调度; 多目标优化; 粒子群算法; 网格坐标系; 格分布方差; 动态调整

中图分类号: TP393

文献标识码: A

doi: 10.11959/j.issn.1000-0801.2019035

Multi-objective cloud workflow scheduling algorithm based on grid variance

BAO Xiaolan¹, CAO Yundi¹, ZHANG Na¹, QIAN Junyan², CAO Jianwen³

1. Zhejiang Sci-Tech University, Hangzhou 310018, China

2. Guilin University of Electronic Technology, Guilin 541004, China

3. Chinese Academy of Sciences, Beijing 100190, China

Abstract: Multi-objective cloud workflow scheduling algorithm based on grid variance and the strategy of bad particles self-learning were presented. Firstly, the characteristics of task scheduling was taken into consideration, and particle encoding was discredited. Secondly, the strategy of mapping Pareto optimal workflow scheduling set to self-adaptive grid coordinate system, and calculating the grid distribution value of each Pareto optimal solution was used. Thirdly, grid variance was adopted to evaluate the diversity of current Pareto front and dynamically adjust evolution strategies. Finally, the concept of being dominated times was introduced into bad particles self-learning strategy for filtering out bad particles in population. The simulation experiment shows that workflow scheduling solution set by this algorithm is better than the MOPSO algorithm on both IGD and S performance indexes, and the optimal value is superior to the ϵ -FDPSO and NSGA-II algorithm.

Key words: cloud workflow scheduling, multi-objective optimization, particle swarm algorithm, grid coordinate system, grid variance, dynamic adjustment

收稿日期: 2018-05-16; 修回日期: 2019-01-01

基金项目: 国家自然科学基金资助项目 (No.61502430, No.61562015); 广西省自然科学基金资助项目 (No.2015GXNSFDA139038); 浙江理工大学 521 人才培养计划资助项目

Foundation Items: The National Natural Science Foundation of China (No.61502430, No.61562015), Guangxi Natural Science Foundation of China (No.2015GXNSFDA139038), 521 Talent Project of Zhejiang Sci-Tech University



1 引言

云计算是一种高效、可伸缩、高可靠性的网络资源模型,为用户快速分配计算、存储等各种资源^[1]。在全球各地,由于云计算的迅速发展,数据中心被广泛建立并投入使用。数据中心的能源消耗约为 26 GW (吉瓦, $1 \text{ GW}=10^9 \text{ W}$), 约占全球总能耗的 1.4%, 并以每年 12% 的比例持续增长^[2]。高能耗、高污染问题日益明显, 甚至制约数据中心的发展^[3]。云工作流在需要高效计算性能和跨组织业务协作、科学计算领域的应用前景广泛^[4]。具有优先约束的云工作流任务调度到可用资源以最大限度地减少执行时间, 已成为 NP-完全问题^[5-6]。因此, 在保证工作流有效调度的同时考虑能源消耗, 已成为云工作流调度研究的热点。

目前云工作流调度问题已获得大量学者的关注并取得一定的成果。异构最早完成时间 (heterogeneous earliest finish time, HEFT) 算法是经典的列表启发式算法^[7], 利用任务的平均执行时间和任务间的通信开销计算每个任务的调度优先级, 选取高优先级的任务到能最快完成该任务的虚拟机上, 实现总调度时间最小化。参考文献[8]提出带期限约束的关键路径算法, 优先计算关键路径上任务的截止时间, 再采取递归方法依次求得其他路径上任务的截止时间, 在截止时间约束下将非关键路径上的任务分配至最便宜的虚拟机上。参考文献[9]基于 HEFT 算法, 根据整个工作流的剩余预算和当前任务的可用预算进行相应的调度, 提出预算约束的异构最早完成时间 (budget-constrained heterogeneous earliest finish time, BHEFT) 算法。应用 GA^[10-11]和 ACO (ant colony optimization)^[12]可以在时间约束下最大限度地降低执行费用。参考文献[13]使用粒子群优化 (particle swarm optimization, PSO) 算法求最小化工作流执行成本的调度解, 同时满足用户定义的完工时间约束, 该算法设置静态惩罚函数使得违反约束条件的调度解

劣于可行解。以上任务调度算法着重以优化时间和费用为目标, 均未考虑实际能耗问题。

参考文献[14]结合动态电压频率调整 (dynamic voltage and frequency scaling, DVFS) 技术与粒子群优化算法提出一种新的面向多目标的能耗感知工作流调度算法, DVFS 通过动态调整 CMOS 电路中的电压和频率以平衡资源性能和能源消耗^[15], 其优点是可减少执行能源消耗。参考文献[16]提出一种基于非占优排序的混合多目标粒子群优化的工作流调度算法, 通过改进的粒子群进化机制得到冲突目标间的工作流 Pareto 最优解集。少数考虑能耗的多目标粒子群优化 (multi-objective particle swarm optimization, MOPSO) 工作流调度算法难以得出真实的最优工作流调度解集, 主要是采用单一的搜索策略来更新粒子的速度, 缺少考虑多目标优化问题的复杂性。参考文献[21]使用基于离散粒子群的 ϵ -fuzzy 占优排序 (ϵ -fuzzy dominance sort based discrete particle swarm optimization, ϵ -FDPSO) 算法解决工作流调度问题, 使用模糊占优生成全局最优解集, 但算法容易陷入局部最优。参考文献[17]采用带精英策略的非支配排序遗传 (non-dominated sorting genetic algorithm: version II, NSGA-II) 算法达到优化调度时间、能源消耗和截止期超时惩罚的目标。但该算法使用固定的交叉率和变异率作为遗传算子, 造成进化后期种群多样性降低的问题, 不利于全局寻优。

本文在云计算环境下研究工作流的调度时间、费用和能耗这个多目标问题, 提出了基于格分布方差的多目标云工作流调度算法。该算法以 MOPSO 为基本思路, 结合工作流的调度过程, 建立调度模型与目标优化方程。考虑工作流任务的调度位置和电压缩放等级 (voltage scaling level, VSL), 进行粒子编码离散化。为了平衡种群进化过程中的开发与开采, 采用多维调度解空间转换到自适应网格坐标系的策略, 利用格分布方差评

估当前调度解空间的多样性程度,并以此动态调整外部文档自优化策略和差粒子自学习策略。同时,引入 Pareto 被占优次数的概念筛选出种群中的差粒子,引导差粒子向精英粒子进化以提高解空间的多样性。最终使得算法产生的最优工作流调度解集兼顾收敛性和多样性。

2 模型定义

假设云服务提供商提供 m 种拥有不同配置和价格的虚拟机资源,表示为 $P=\{p_1, p_2, \dots, p_m\}$ 。每台虚拟机 $p_j \in P$ 以每秒执行的百万指令数 (million instructions per second, MIPS) 表示其处理能力,同时均支持 DVFS,表明虚拟机可以运行在不同的 VSL^[18]上。

2.1 工作流模型

工作流采用有向无环图 (directed acyclic graph, DAG) 描述并表示为 $G=(T, E)$, 如图 1 所示。其中 $T=\{t_1, t_2, \dots, t_n\}$ 为任务集合, n 表示任务的个数。节点权重集合 $W=\{w_1, w_2, \dots, w_n\}$ 代表任务所包含的百万指令数。 $E=\{(t_i, t_j, e_{ij})\}$ 为图 1 中有向边的集合,即 t_i 是 t_j 的前驱任务,当 t_i 和 t_j 调度在同一个虚拟机上时,通信开销 e_{ij} 可忽略不计。任务 t_i 的直接前驱任务集合记为 $\text{pre}(t_i)$, 直接后继任务集合记为 $\text{succ}(t_i)$ 。只有当 $\text{pre}(t_i)$ 全部执行完成并且数据传输到任务 t_i 所在的虚拟机时, t_i 才能开始执行。

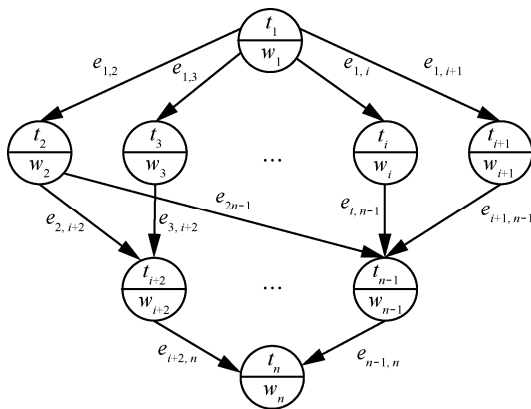


图1 工作流模型

2.2 多目标模型定义

关于多目标优化问题 (multi-objective optimization problem, MOP)^[19], 其解通常为多个彼此冲突目标的最优潜在解集合。 m 个决策变量和 n 个优化目标的 MOP 通常被定义为:

$$\min (F(x)=[f_1(x), f_2(x), \dots, f_n(x)]) \quad (1)$$

其中, $x=(x_1, x_2, \dots, x_m) \in S$ 代表 m 维的决策向量, S 为决策空间; $f_1, f_2, \dots, f_n \in \Omega$ 代表 n 维的目标向量, Ω 为目标空间。一组目标函数将决策空间映射到目标空间, $S_m \rightarrow \Omega_n$ 。以下定义了 MOP 中相关的 Pareto 概念。

定义 1 (Pareto 占优) 对于任意两个解 u^* , $v^* \in \Omega$, u^* 占优 v^* 定义为:

$$\begin{aligned} u^* > v^* &\Leftrightarrow \forall i=1, 2, \dots, n, u_i^* \leq v_i^* \\ v_i^* &\wedge \exists j=1, 2, \dots, n, u_j^* < v_j^* \end{aligned} \quad (2)$$

定义 2 (Pareto 被占优次数) 解 $x^* \in \Omega$, Ωx^* 表示在 Ω 中去除 x^* 后所形成的集合, 并对其中每个元素进行占优关系比较, x^* 的 Pareto 被占优次数定义为:

$$P_d = \text{num}\{(\text{each } \Omega x^*) \succ x^*\} \quad (3)$$

定义 3 (Pareto 最优解集合) 解 $x^* \in \Omega$ 是 Pareto 最优解或非占优解, 当且仅当:

$$\exists x \in \Omega: x \succ x^* \quad (4)$$

所有 Pareto 最优解的集合称为 Pareto 最优解集。所有 Pareto 最优解形成的区域称为 Pareto 前端。

定义 4 (完成时间) 从工作流的入口任务 t_{entry} 开始执行到出口任务 t_{exit} 结束的时间段称为整个工作流的完成时间 makespan。任务的最早开始时间 $\text{EST}(t_i, p_{t_i})$ 和最早结束时间 $\text{EFT}(t_i, p_{t_i})$ 为:

$$\text{EST}(t_i, p_{t_i}) = \begin{cases} p_{t_i}^{\text{avail}}, t_i = t_{\text{entry}} \\ \max \left\{ \begin{aligned} &p_{t_i}^{\text{avail}} \\ &\max_{t_j \in \text{pre}(t_i)} \left\{ \text{AFT}(t_j) + \frac{e_{ij}}{B(p_{t_i}, p_{t_j})} \right\} \end{aligned} \right\} \end{cases} \quad (5)$$

$$\text{EST}(t_i, p_{t_i}) = \text{ET}(t_i, p_{t_i}) + \text{EST}(t_i, p_{t_i}) \quad (6)$$



其中, $p_{t_i}^{\text{avail}}$ 为虚拟机 p_{t_i} 准备执行任务的就绪时间, $\text{AST}(t_i, p_{t_i})$ 和 $\text{AFT}(t_i, p_{t_i})$ 代表任务 t_i 在虚拟机 p_{t_i} 上的实际开始时间和实际完成时间, $B(p_{t_i}, p_{t_j})$ 表示虚拟机 p_{t_i} 和 p_{t_j} 之间的带宽, $\text{ET}(t_i, p_{t_i})$ 为任务 t_i 在虚拟机 p_{t_i} 上的执行时间。整个工作流的完成时间为:

$$\text{makespan} = \max \{ \text{AFT}(t_{\text{exit}}) \} \quad (7)$$

定义 5 (调度费用) 调度费用指所有任务在虚拟机上执行所产生的费用, 暂不考虑数据存储和传输费用。按照单位时间收取费用的方式, 则虚拟机 p_{t_i} 执行任务 t_i 所需的费用为:

$$\text{cost}_{t_i} = [\text{AFT}(t_i, p_{t_i}) - \text{AST}(t_i, p_{t_i})] \cdot C_{p_{t_i}} \quad (8)$$

其中, cost_{t_i} 中的时间为任务 t_i 的实际完成时间 AFT 和实际开始时间 AST 之差, 由于虚拟机执行任务时会进行动态电压频率调整, 因此不同于任务的最早结束时间 EFT 与最早开始时间 EST 之差。整个工作流的调度费用为:

$$\text{cost} = \sum_{i=1}^n \text{cost}_{t_i} \quad (9)$$

定义 6 (能耗模型) 工作流中任务执行的能源消耗模型参考 CMOS 模型^[20], 定义为:

$$P = ACV^2 f \quad (10)$$

其中, A 代表每个时钟周期的开关次数, C 代表总容量负载, V 代表供电电压, f 代表时钟频率。因此在虚拟机上执行任务的能耗表示为:

$$\begin{aligned} \text{energy} &= \sum_{i=1}^n ACV^2 f \cdot \text{ET}(t_i, p_{t_i}) = \\ &= \sum_{i=1}^n \beta V_{t_i}^2 \text{ET}(t_i, p_{t_i}) \end{aligned} \quad (11)$$

本文在考虑工作流调度时间和费用的基础上, 充分考虑能耗问题, 从而形成多目标工作流调度。调度目标是将工作流中所有的任务调度至虚拟机上, 最小化工作流的调度时间、费用和能耗。计算式如下:

$$\min(\text{makespan}, \text{cost}, \text{energy}) \quad (12)$$

3 多目标粒子群云 workflow 调度算法

本文提出一种基于格分布方差的多目标云 workflow 调度 (grid variance dynamic adjustment MOPSO, GVDA-MOPSO) 算法, 解决 workflow 任务调度的多目标优化问题。考虑到云 workflow 调度问题的离散空间性, 对任务所分配的虚拟机和执行任务时所处的电压缩放等级进行编码, 第 t 次迭代中第 i 个粒子定义为 $x_i(t) = (x_{i1}, x_{i2}, \dots, x_{iN})$, 其中 N 表示粒子维数即任务个数, $x_{ij} = (p_m, L_k)$, m 为虚拟机编号, k 为电压缩放等级。种群进化过程中根据式 (13) 更新粒子的速度和位置:

$$\begin{cases} v_i(t+1) = \omega v_i(t) + c_1 r_1 (\text{pBest}_i - x_i(t)) + \\ c_2 r_2 (\text{gBest} - x_i(t)) \\ x_i(t+1) = \lceil x_i(t) + v_i(t+1) \rceil \end{cases} \quad (13)$$

其中, ω 为惯性权重系数, c_1 和 c_2 为加速系数, r_1 和 r_2 为在 $[0, 1]$ 上均匀分布的随机数, pBest_i 和 gBest 分别表示第 i 个粒子的个体最优解和种群的全局最优解, $\lceil x \rceil$ 表示不小于 x 的最小整数。

workflow 调度流程如图 2 所示, 基本步骤如下。

步骤 1 用户提交 workflow, 通过向上排序 (upward rank) 算法^[7] 计算各个任务节点的排序值, 然后按照降序规则生成调度序列。

步骤 2 将调度序列和虚拟机列表作为输入, 调用 GVDA-MOPSO 算法生成调度方案。其中通过计算格分布方差来动态调整进化策略, 以平衡进化过程中的开发和开采。

步骤 3 根据算法得到的调度方案, 将任务调度至虚拟机。

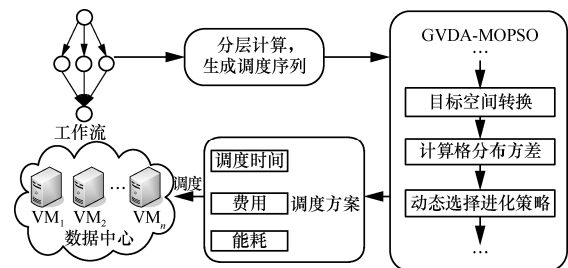


图 2 workflow 调度流程

3.1 任务排序

由于只有当 $\text{pre}(t_i)$ 全部执行完成并且数据传输到任务 t_i 所在的虚拟机时, t_i 才能开始执行, 故采用向上排序算法确定任务的优先级。任务 t_i 的排序 rank 值定义如下:

$$\text{rank}(t_i) = \bar{w}_i + \max_{t_j \in \text{succ}(t_i)} (\bar{c}_{i,j} + \text{rank}(t_j)) \quad (14)$$

其中, $\text{rank}(t_i)$ 为任务 t_i 的排序值, $\bar{w}_i$ 为 t_i 在虚拟机上的平均执行时间, $\bar{c}_{i,j}$ 为 t_i 到 t_j 的平均通信时间。排序值由 t_{exit} 开始计算, 依次向上递归得到每个任务的 rank 值。再对所有的 rank 值降序排序, 其对应的节点顺序即云工作流任务的调度顺序。

3.2 自适应网格坐标系

外部文档主要用于存储迭代过程中的 Pareto 最优工作流调度解集, 是第二代进化多目标优化算法的主要特征^[21]。平行坐标广泛用于多维数据分析和可视化方法, 多目标优化问题的目标向量正是一种多维数据^[22]。为此, 本文对外部文档中的 M 维 Pareto 最优工作流调度解集进行目标空间转换, 映射到具有 $M \times \max(G_{k,m})$ 个格子的二维自适应网格坐标系中, 以反映 Pareto 前端的分布情况。 $G_{k,m}$ 为第 k 个 Pareto 最优解的第 m 维的格坐标分量, 计算式如式 (15) 所示。第 m 维上 k 个格坐标分量组成集合, 该集合中值的不同程度越大, 反映 k 个 Pareto 最优解在第 m 维上越分散。

$$G_{k,m} = \left\lceil K \frac{|f_{k,m} - \bar{f}_m|}{f_m^{\max} - \bar{f}_m} \right\rceil \quad (15)$$

其中, $k=1,2,\dots,K$, K 为当前迭代周期中外部文档的成员个数, $m=1,2,\dots,M$ 为优化问题的目标维数; $f_{k,m}$ 表示第 k 个 Pareto 最优解在第 m 维上的取值, $f_m^{\max}$ 和 $\bar{f}_m$ 分别表示当前 Pareto 前端上第 m 维的最大值和平均值。为了使格坐标分量都落在自适应网格坐标系中, 当 $f_{k,m} = \bar{f}_m$ 时, 令 $G_{k,m}=1$ 。

图 3 (a) 中的 n 个点表示多目标优化问题测试函数 ZDT (包含两个目标) 的 Pareto 前端的 n 个

目标向量。图 3 (b) 由 $\max(G_{k,m})$ 行 2 列构成的网格坐标, 其中列数表示图 3 (a) 中的目标维数。例如, 图 3 (a) 中 P_5 的网格坐标是 (2,2), 在图 3 (b) 中分别用 $G_{5,1}=2$ 和 $G_{5,2}=2$ 表示。图 3 (b) 中用虚线连接同一个 Pareto 最优工作流调度解的格坐标分量。

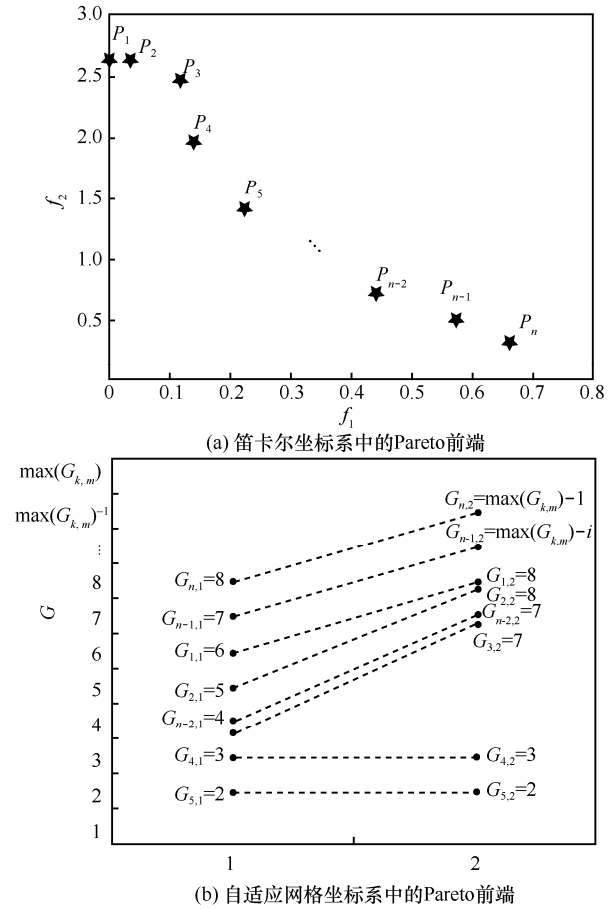


图 3 目标空间转换图

3.3 格分布方差

定义 7 (格分布量) 格分布量表示第 k 个 Pareto 最优解的 M 维上的格坐标分量之和。计算式如下:

$$D_k = \sum_{m=1}^M G_{k,m} (k=1,2,\dots,K) \quad (16)$$

图 4 表示图 3 (a) 所示的 Pareto 最优解集中的每个 Pareto 最优解对应的格分布量。例如, 图 3 (b) 中 P_3 的网格坐标是 (4,7), 由式 (16)



计算得 $D_3=G_{3,1}+G_{3,2}=11$ 。结合图 3(a) 和图 3(b), 发现图 3(a) 中 Pareto 最优解与原点的欧氏距离越小, 其映射到自适应网格坐标系中的格分布量也越小。

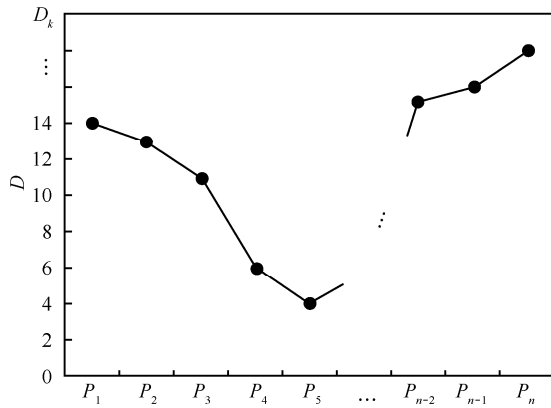


图 4 每个 Pareto 最优解的格分布量

定义 8 (格分布方差) 各维度上格坐标分量方差之和的开方称为格分布方差, 计算式如下:

$$\delta = \sqrt{\sum_{m=1}^M \frac{\sum_{k=1}^K (G_{k,m})^2}{K}}, m=1, 2, \dots, M \quad (17)$$

设中心点为最靠近原点的点 (格分布量最小的点)。 D_k 越大, 表示其对应的 Pareto 最优解离中心点越远, 则格分布方差反映了当前 Pareto 前端的多样性程度。因此利用格分布方差动态调整平衡开发和开采策略, 可进一步提高算法的优化效果。如果格分布方差 $\delta >$ 阈值 c , 说明 Pareto 前端分布较为多样, 则采取外部文档自优化策略以提高 Pareto 最优解的收敛性。反之, 说明 Pareto 前端分布失衡, 则采取差粒子自学习策略以提高 Pareto 最优解的多样性。动态调整算法如算法 1 所示。

算法 1 动态调整算法

输入: 外部文档 rep, 目标维度 M

输出: 当前 Pareto 前端的状态和采用的进化策略

for $i=1$ to num (rep) do

根据式 (15) 计算第 i 个粒子 M 维的格坐

标分量;

根据式 (16) 计算第 i 个粒子的格分布量;

end for

for $j=1$ to M do

计算第 j 维上格坐标分量方差;

end for

根据式 (17) 计算外部文档的格分布方差 δ ;

if $\delta > c$

return 多样状态并采取外部文档自优化策略;

else

return 失衡状态并采取差粒子自学习策略;

end if

3.4 外部文档自优化策略

外部文档粒子周围存在大量相似的前端粒子, 通过对外部文档粒子的变异能快速寻找到这些前沿粒子, 从而提高算法的收敛性。算法 2 通过优化变异粒子的一个维度, 使得 Pareto 前端逼近真实 Pareto 前端。首先, 取外部文档中的一个粒子, 随机取该粒子的一个目标维度, 改变该维度上的虚拟机编号及电压缩放等级; 其次, 判断变异后粒子的合法性, 如果虚拟机编号或电压缩放等级超出取值最小或最大边界, 则虚拟机编号或电压缩放等级对应取最小或最大边界; 再次, 计算新粒子的调度时间、费用和能耗, 判断新粒子是否占优旧粒子, 如果是, 则更新旧粒子; 最后, 迭代结束后, 返回优化后的外部文档。其中, 算法中的 pm 表示变异率, 通过实验和概率分析发现, pm 取 0.4 较为合适, 此时粒子变异后占优原粒子的概率较高。外部文档自优化算法如算法 2 所示。

算法 2 外部文档自优化算法

输入: 外部文档 rep, 目标维度 M , 虚拟机最小编号 $p_{\min}$ 和最大编号 $p_{\max}$, 最低电压缩放等级 $L_{\min}$ 和最高电压缩放等级 $L_{\max}$

输出: 优化后的外部文档

```

for i=1 to num(rep)do
    NewSolution = rep(i);
    j=random(M); //随机选取粒子的一个维度
    dx1=pm*(pmax-pmin); dx2=pm*(Lmax-Lmin);
//产生随机变量
    改变 NewSolution 的第j维的虚拟机编号
    及电压缩放等级;
    判断变异后粒子的合法性, 若不合法,
    则进行取值调整;
    计算 rep(i)的时间、费用和能耗;
    if NewSolution > rep(i) // 占优比较
        rep(i)=NewSolution; //更新 rep(i)
    end if
end for
return rep; //返回自优化后的外部文档

```

3.5 差粒子自学习策略

当 Pareto 前端处于失衡状态时, 算法可能陷入局部最优。本文采用差粒子自学习策略来辅助算法跳出局部极值并提高解的多样性。通过记录第 t 次迭代中种群每个粒子的 Pareto 被占优次数 P_d , 筛选出 P_d 大于 α 次的粒子, 记为差粒子集合 (由实验可得, 当种群规模为 n 时, $\alpha=n/5$ 时算法的性能较好)。

本文把外部文档的格分布量矩阵记为 $S=[D_1, D_2, \dots, D_M]$; 去掉矩阵 S 中重复的元素, 得到由不同元素构成且升序排列的矩阵 $R=ASC(unique(S))$; 遍历 S , 计算 R 中每个元素对应的个数, 存入矩阵 RN ; 考虑到 Pareto 前端分布范围较大, 将 RN 每隔 5 个元素分为一个区间, 计算每个区间的和, 得到矩阵 $Z=[z_1, z_2, \dots, z_s]$, 其中 $s=[num(RN)/5]$ 。 z_s 的大小代表该区间内粒子的个数。从粒子个数最少的区间 (排除个数为零的区间) 中选取精英粒子, 利用式 (18) 引导差粒子向精英粒子学习, 使得 Pareto 前端分布呈现多样性。其中, $bp(t)$ 代表第 t 次迭代的一个差粒子, $pattern$ 代表从外部文档中选择的精英粒子, $rand()$

代表 $[0,1]$ 上的随机数。差粒子引导图如图 5 所示, p_k 代表一个差粒子映射到目标空间上的点, 假设通过计算得出两侧圈中的粒子的格分布量属于区间 1, 中间的粒子的格分布量属于区间 2, 区间 2 的粒子个数明显少于区间 1, 故引导差粒子向区间 2 中的精英粒子进化。

$$xNew(t)=bp(t)+rand() \cdot (pattern - bp(t)) \quad (18)$$

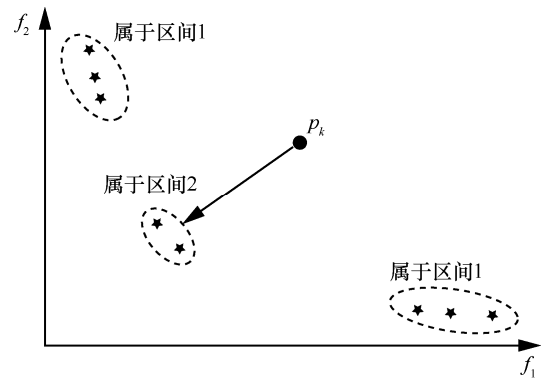


图5 差粒子引导图

差粒子自学习策略如算法 3 所示。首先, 第 t 次迭代中种群的所有粒子, 筛选出差粒子; 其次, 选取差粒子集合中的一个粒子, 根据格分布量从粒子个数最少的小集合中选择精英粒子, 引导差粒子根据式 (18) 向精英粒子学习; 再次, 计算新粒子的调度时间、费用和能耗。判断新粒子是否占优旧粒子, 如果是, 则更新旧粒子。最后, 所有差粒子遍历完成, 返回更新后的种群。

算法 3 差粒子自学习策略

输入: 第 t 次迭代中种群的所有粒子 pop , 每个粒子的被占优次数 P_d , 外部文档 rep

输出: 自学习后的种群

```

for j=1 to num(pop)do
    bp=(pop(j).Pd>α); //选择差粒子
end for

```

根据格分布量将 Pareto 最优工作流调度解集分为若干个小集合;

```

for i=1 to num (bp) do

```

从粒子个数最少的小集合中选择精英粒



子, 记为 pattern ;

粒子 $\text{bp}(i)$ 根据式 (18) 进行自学习, 得到新粒子 $x_{\text{New}}(i)$;

计算 $x_{\text{New}}(i)$ 的时间、费用和能耗;

if $\text{NewSolution} > \text{rep}(i)$ // 占优比较

$\text{bp}(i) = x_{\text{New}}(i)$;

end if

end for

$\text{pop} = (\text{pop}, P_d=0) \cap (\text{bp})$; // 更新后的种群

return pop

3.6 算法流程

根据 MOPSO 算法框架, 结合云 workflow 调度的特性, 本文采用目标空间转换策略, 利用格分布方差评估当前调度解空间的多样性程度, 并以此自适应调整进化策略。算法 4 给出了基于格分布方差的多目标云 workflow 调度算法的伪代码实现方案。

算法 4 基于格分布方差的多目标云 workflow 调度 (GVDA-MOSPO) 算法

输入: 云 workflow $T = \{t_1, t_2, \dots, t_n\}$, 虚拟机 $P = \{p_1, p_2, \dots, p_m\}$, 虚拟机的电压等级 $L = \{L_1, L_2, \dots, L_k\}$, 迭代次数 Iteration , 外部文档容量 cap

输出: ScheduleSet // 最优 workflow 调度解集

根据式 (14) 得到 T 中每个任务 rank 值, 生成任务调度序列;

for $i=1$ to n do

初始化粒子的调度方案 $\text{pop}(i)$ 和速度 $v(i)$, 令 $\text{pBest} = \text{pop}(i)$;

计算调度方案 $\text{pop}(i)$ 的时间 (式 (7))、费用 (式 (9)) 和能耗 (式 (11));

end for

对 n 个调度方案占优进行比较, 选取未被占优的粒子存入 rep ;

对 rep 采用目标空间转换策略, 计算粒子的格分布量;

for $t=1$ to iteration do

for $i=1$ to n do

利用格分布量, 采用轮盘赌转法从 rep 中选择全局最优解 $\text{gBest}(i)$;

按式 (13) 更新 $\text{pop}(i)$ 的速度和位置;

计算调度方案 $\text{pop}(i)$ 的时间、费用和能耗, 更新 $\text{pbest}(i)$;

end for

对更新后的 n 个调度方案占优比较, 选取未被占优的粒子存入 rep ;

调用算法 1;

if Pareto 前端为多样性

调用算法 2;

else

调用算法 3;

end if

if $\text{num}(\text{rep}) > \text{cap}$

利用格分布量, 采用轮盘赌转法从 rep 中删除 $(\text{num}(\text{rep}) - \text{cap})$ 个多余的粒子;

end if

end for

return Pareto 最优 workflow 调度解集, 即 ScheduleSet

4 实验仿真及结果分析

为了分析和评估本文提出的 GVDA-MOPSO 算法的性能, 选取 3 种求解多目标优化问题的同类型群智能算法作为对比, 包括 ϵ -FDPSO 算法^[21]、MOPSO 算法^[23]和 NSGA-II^[24]。实验采用 WorkflowSim^[25]仿真工具, 计算环境为: 虚拟机 Intel Core i5-3320M CPU@2.60 GHz, 内存 8 GB, Windows7 64 位操作系统。

4.1 实验环境配置

实验的数据中心提供 10 种拥有不同处理速率和租赁价格的虚拟机, 其定价参照 Amazon 云。每个虚拟机均可运行于不同的 VSL, 虚拟机忙时运行于最大电压缩放等级, 空闲时均运行于最小电

压缩等级, 见表1。实验使用 WorkflowGenerator 生成激光干涉引力波天文台 (laser interferometer gravitational wave observatory, LIGO) 工作流, LIGO 任务数为 30、50、100、150。

表1 虚拟机参数设置

编号	计算速率/ MIPS	带宽/ (Mbit·s ⁻¹)	单价/ (美元·h ⁻¹)	电压/V
1	967	870	3.13	1.5
	389			0.9
2	1 300	1 100	4.19	2.2
	650			1.0
3	1 250	854	3.67	2.2
	625			1.0
4	845	987	2.71	1.75
	338			0.9
5	1 077	730	3.26	1.5
	431			0.9
6	510	828	1.55	1.75
	204			0.9
7	1 290	921	3.73	2.2
	645			1.0
8	659	830	1.94	1.75
	267			0.9
9	1 160	568	3.45	1.5
	464			0.9
10	734	966	2.46	1.75
	294			0.9

4.2 性能指标分析

通常来说, 算法的评估主要从解的收敛性和多样性两方面考虑。本文采用逆世代距离 (inverted generational distance, IGD) 与间距 (spacing, S) [26] 对算法进行评价。

(1) 逆世代距离 IGD

表示为当前搜索到的非占优解与 Pareto 前端的距离:

$$IGD(P, PF^*) = \frac{\sum_{i=1}^{|PF^*|} \text{dist}(PF_i^*, P)}{|PF^*|} \quad (19)$$

其中, PF^* 为真实 Pareto 解集中的一部分, P 为算法得到的最优解集, $|PF^*|$ 为解集的个体数目, $\text{dist}(PF_i^*, P)$ 为 PF_i^* ($PF_i^* \in PF^*$) 到 P 中距离最近个体的欧氏距离。从式 (19) 可看出, 对于 PF^* , 采样越多, 越能接近覆盖算法整个真实边界。IGD 的值越小, 说明搜索到的解的收敛性越好。

(2) 间隔 S

评估解间的多样性, 表示为:

$$S = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (\bar{d} - d_i)^2} \quad (20)$$

其中, $d_i = \min(|f_1^i(x) - f_1^j(x)| + |f_2^i(x) - f_2^j(x)|), i, j = 1, 2, \dots, n$, $\bar{d}$ 表示距离度量 d_i 的平均值, n 为 Pareto 解的个数, S 值越小对算法越有利。

4.3 格分布方差阈值分析

在 GVDA-MOPSO 算法中存在一个格分布方差阈值 c , c 的选取对算法有一定的影响。当 c 过大时, 算法侧重 Pareto 最优解集的多样性, 由于扩大了其搜索空间, 解集的收敛性变得较好, 但存在解集收敛速度较慢的问题。当 c 过小时, 在最优解集多样性不佳的情况下优化其收敛性, 得到的最优解集效果不佳, 且搜索策略偏重于外部文档自优化时, 算法耗时相对较长。实验选取工作流为 LIGO-30, 外部文档容量为 30。收敛性指标 IGD 和多样性指标 S 的关系如图 6 所示, 其中曲线上的点为 c 的取值情况。根据实验结果表明, 当 $c > 110$ 时, IGD 指标和 S 指标均较优, 为了兼顾算法效率, 本文格分布方差阈值 c 取 110。

4.4 对比结果

4.4.1 与 MOPSO 对比

GVDA-MOPSO 和 MOPSO 参数取值: 种群个体数 150, 外部文档容量为 30, $c_1=1$, $c_2=1$, 惯性权重 ω 由 0.5 开始每次迭代的下降速率为 0.99。通过实验分析 GVDA-MOPSO 的格分布方差阈值 $c=110$, $\alpha=30$ 。为了得到满足 Pareto 最优的工作流调度解集, 算法均迭代 200 次。

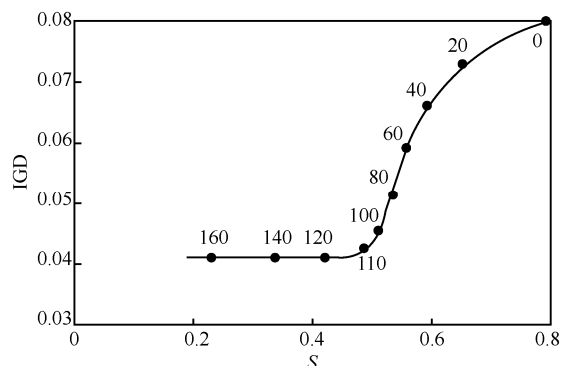


图6 IGD与S关系

当任务数为30、50、100、150时，两种算法得到的最优 workflow 调度解集如图7所示。通过对比发现，GVDA-MOPSO 算法在不同 LIGO 任务数下得出的最优 workflow 调度解集均比 MOPSO 算法更邻近于真实 Pareto 前端。尤其当任务数为100、150时，解之间的空间分布较为均匀。

图8展示了两个算法的IGD和S两项性能指标，考虑算法的随机性影响，所有结果都取200次仿真实验结果的均值和方差。GVDA-MOPSO 算法在不同 LIGO 任务数下均值和方差均比 MOPSO 算法小，这是因为 GVDA-MOPSO 算法自适应调整进化策略，使搜索到的调度解兼顾收敛性和多样性。同时从图8(b)看出，GVDA-MOPSO 算法的S性能指标明显优于 MOPSO 算法，这是因为算法优先根据格分布方差评估解的多样性，在多样性的基础之上再兼顾解的收敛性。

4.4.2 与 ϵ -FDPSO、NSGA-II 对比

现将 GVDA-MOPSO 与 ϵ -FDPSO 算法、NSGA-II 算法进行比较。

(1) ϵ -FDPSO 方法使用模糊占优机制确定 Pareto 最优解集，从而求解云 workflow 调度问题。

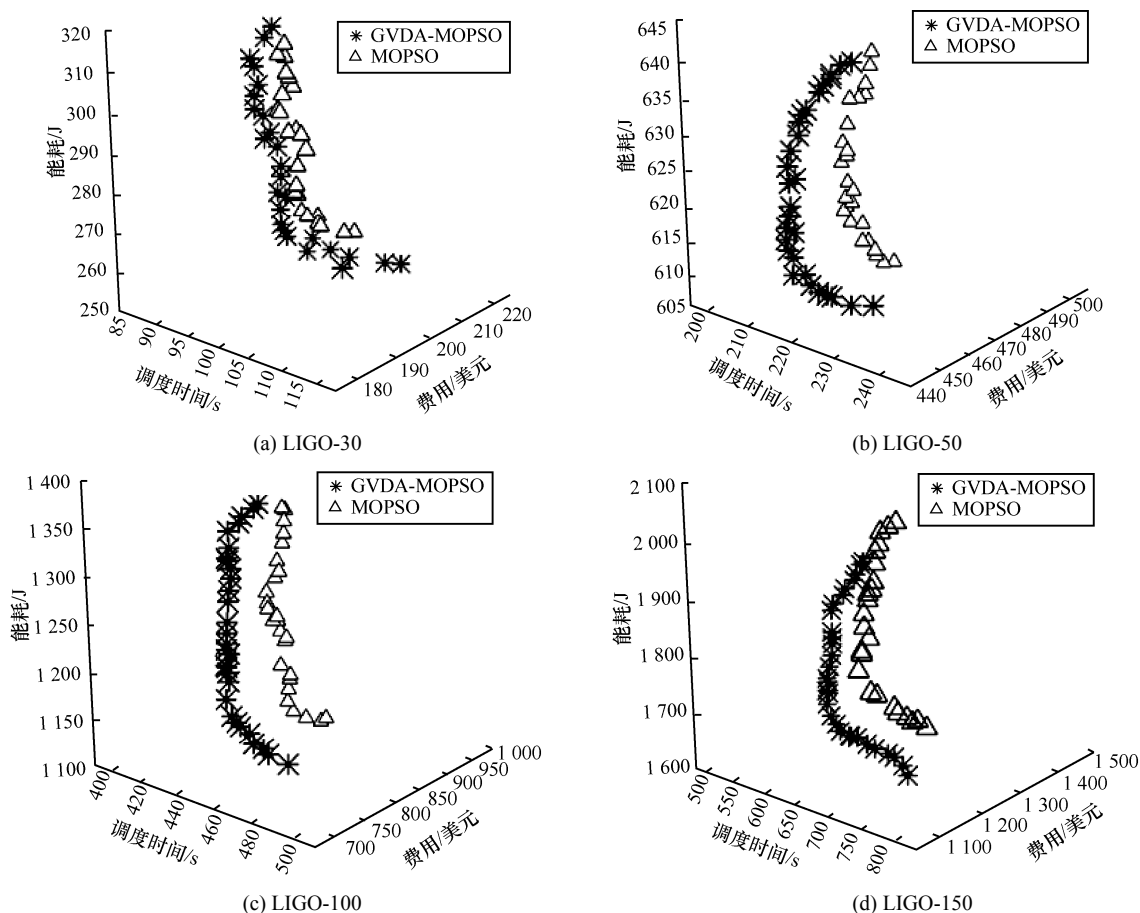
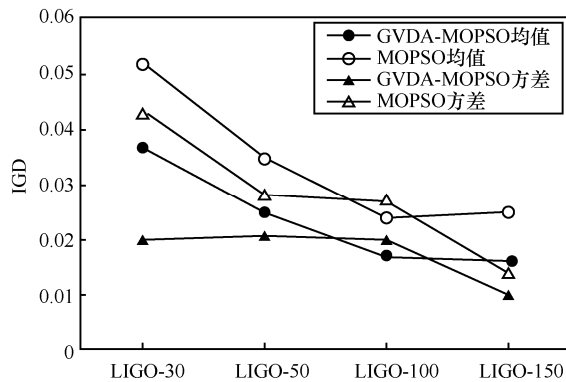
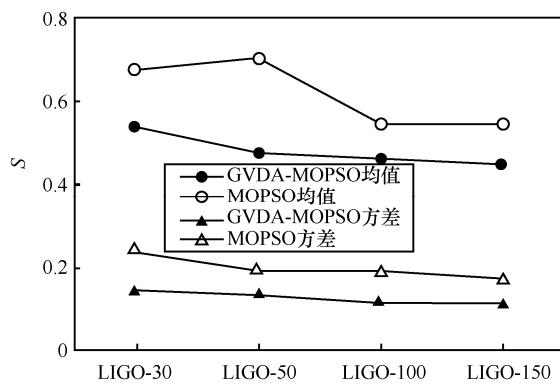


图7 两种算法得到的最优 workflow 调度解集



(a) IGD 指标的均值和方差对比



(b) S 指标的均值和方差对比

图8 两种算法的IGD和S对比

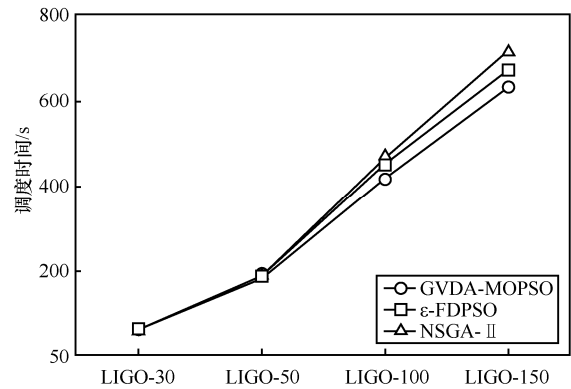
种群大小为 150, c_1 由 2.5 递减至 0.5, c_2 由 0.5 递增至 2.5, 惯性权重 ω 由 0.9 递减至 0.2。

(2) NSGA-II 种群大小为 150, 变异概率为 0.3, 交叉概率为 0.8。

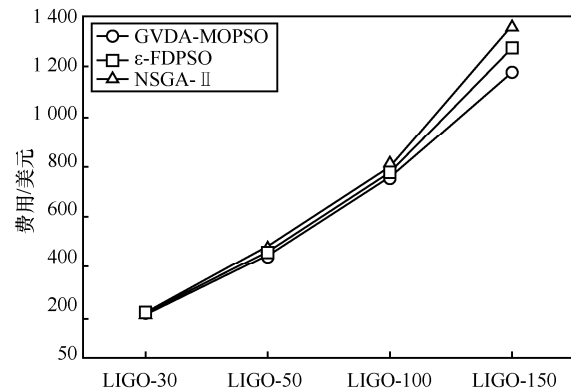
当 LIGO 任务数为 30、50、100、150 时, 3 种算法在本文实验环境下各运行 150 次, 取得调度方案时间、费用和能耗指标的平均值, 对比结果如图 9 所示。

从图 9 可以看出, GVDA-MOPSO 生成的调度方案在时间、费用和能耗 3 项指标上的平均值均低于 ϵ -FDPSO 和 NSGA-II, 表明 GVDA-MOPSO 生成的调度方案优于其他两种算法。同时, 当任务数由 30 增加至 150 时, 在任务数较多的情况下, GVDA-MOPSO 生成调度方案明显优于 ϵ -FDPSO 和 NSGA-II。这是因为在任务数较多时, GVDA-MOPSO 算法对解空间的探索范

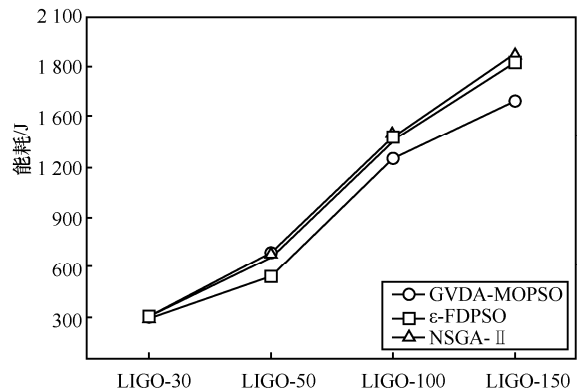
围扩大, 更能体现本算法动态调整搜索策略的优势, 使得生成的最优工作流调度解集较好。



(a) 调度时间对比



(b) 费用对比



(c) 能耗对比

图9 3种方案在时间、费用、能耗方面的对比

5 结束语

为了解决粒子群优化算法应用于云工作流调度的多目标问题时存在开发与开采难以平衡的问



题, 本文提出了基于格分布方差的多目标云 workflow 调度算法和差粒子自学习策略。采用目标空间转换策略, 利用格分布方差评估当前调度解空间的多样性程度, 并以此动态调整进化策略。通过仿真实验, 与 MOPSO、 ϵ -FDPSO 和 NSGA-II 算法进行比较, 证明了 GVDA-MOPSO 算法的有效性。考虑到大规模的应用场景, 本文将在现有研究的基础上, 进一步结合资源的负载均衡^[27]进行深入研究。

参考文献:

- [1] MASDARI M, VALIKARDAN S, SHAHI Z, et al. Towards workflow scheduling in cloud computing: a comprehensive analysis[J]. *Journal of Network and Computer Applications*, 2016, 66(5): 64-82.
- [2] AWADA U, KEQIU L, YANMING S. Energy consumption in cloud computing data centers[J]. *International Journal of Cloud Computing and Services Science*, 2014, 3(3).
- [3] 袁景凌, 钟珞, 杨光, 等. 绿色数据中心不完备能耗大数据填补及分类算法研究[J]. *计算机学报*, 2015, 38(12): 2499-2516.
YUAN J L, ZHONG L, YANG G, et al. Towards filling and classification of incomplete energy big data for green data centers[J]. *Chinese Journal of Computers*, 2015, 38(12): 2499-2516.
- [4] 张鹏, 王桂玲, 徐学辉. 云计算环境下适于 workflow 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.
- [5] 沈尧, 秦小麟, 鲍芝峰. 一种云环境中数据流的高效多目标调度方法[J]. *软件学报*, 2017, 28(3): 579-597.
SHEN Y, QIN X L, BAO Z F. Effective multi-objective scheduling strategy of dataflow in cloud[J]. *Journal of Software*, 2017, 28(3): 579-597.
- [6] 王琳杰. 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[J]. *电信科学*, 2016, 32(9): 113-119.
WANG L J. Task scheduling scheme based on improved particle swarm optimization with biological symbiosis mechanism in cloud computing[J]. *Telecommunications Science*, 2016, 32(9): 113-119.
- [7] TOPCUOGLU H, HARIRI S, WU M. Performance-effective and low-complexity task scheduling for heterogeneous computing[J]. *IEEE Transactions on Parallel Distributed Systems*, 2002, 13(3): 260-274.
- [8] ABRISHAMI S, NAGHIBZADEH M, EPEMA D H. Deadline-constrained workflow scheduling algorithm for infrastructure as a service cloud[J]. *Future Generation Computer Systems*, 2013, 29(1): 158-169.
- [9] ZHENG W, SAKELLARIOU R. Budget-deadline constrained workflow planning for admission control[J]. *Journal of Grid Compute*, 2013, 11(4): 633-651.
- [10] ISRAEL C, JAVID T, RAJIV R, et al. GA-ETI: an enhanced genetic algorithm for the scheduling of scientific workflows in cloud environments[J]. *Future Generation Computer Systems*, 2018: 257-271.
- [11] MEENA J, KUMAR M, VARDHAN M. Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint[J]. *IEEE Access*, 2016(4): 5065-5082.
- [12] SINGH L, SINGH S. Deadline and cost based ant colony optimization algorithm for scheduling workflow applications in hybrid cloud[J]. *Journal of Scientific & Engineering Research*, 2014, 5(10): 1417-1420.
- [13] RODRIGUEZ M A, BUYYA R. Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds[J]. *IEEE Transactions on Cloud Computing*, 2014, 2(2): 222-235.
- [14] YASSA S, CHELOUAH R, KADIMA H, et al. Multi-objective approach for energy-aware workflow scheduling in cloud computing environment[J]. *The Science World Journal*, 2013: 1-13.
- [15] DENG G S, LI Z M, ZHAO Y M, et al. MOEA/D for energy-aware scheduling on heterogeneous computing systems[C]//10th International Conference on Bio-Inspired Computing, Theories and Applications(BIC-TA 2015), Sep 25-28, 2015, Hefei, China. Heidelberg: Springer, 2015: 94-106.
- [16] VERMAA, KAUSHAL S. A hybrid multi-objective particle swarm optimization for scientific workflow scheduling[J]. *Parallel Computing*, 2017(62): 1-19.
- [17] NESMACHNOW S, ITURRIAGA S, DORRONSORO B, et al. Multiobjective energy-aware workflow scheduling in distributed datacenters[C]//6th International Conference on High Performance Computer Applications(ISUM 2015), March 9-13, 2015, Mexico City, Mexico. Heidelberg: Springer, 2015.
- [18] DURILLO J J, NAE V, PRODAN R. Multi-objective energy-efficient workflow scheduling using list-based heuristics[J]. *Future Generation Computer Systems*, 2014(36): 221-236.
- [19] PRZYSTALKA P, KATUNIN A. Multi-objective meta-evolution method for large-scale optimization problems[M]//Recent advances in computational optimization. Heidelberg: Springer, 2016: 165-182.
- [20] LEE Y C, ZOMAYAA Y. Energy conscious scheduling for distributed computing systems under different operating conditions[J]. *IEEE Transactions on Parallel Distributed System*, 2011, 22(8): 1374-1381.
- [21] GARG R, SINGHAK. Multi-objective workflow grid scheduling using ϵ -fuzzy dominance sort based discrete particle swarm optimization[J]. *Journal of Supercomputing*, 2014, 68(2): 709-732.
- [22] 胡旺, YEN G G, 张鑫. 基于 Pareto 熵的多目标粒子群优化算法[J]. *软件学报*, 2014, 25(5): 1025-1050.

- HU W, YEN G G, ZHANG X. Multi-objective particle swarm optimization based on Pareto entropy[J]. Journal of Software, 2014, 25(5): 1025-1050.
- [23] PADMAVENI K, ARAVINDHAR D J. Hybrid memetic and particle swarm optimization for multi objective scientific workflows in cloud[C]//IEEE International Conference on Cloud Computing in Emerging Markets, Oct 19-21, 2016, Bangalore, India. Piscataway: IEEE Press, 2016: 66-72.
- [24] DOGAN A, OZGUNER F. Biobjective scheduling algorithms for execution time-reliability trade-off in heterogeneous computing systems[J]. Journal of Computer, 2013, 48(3): 300-314.
- [25] CHEN W, DEELMAN E. WorkflowSim: a tool kit for simulating scientific workflows in distributed environment[C]//International Conference on E-Science, Oct 8-12, 2012, Chicago, IL, USA. Washington DC: IEEE Computer Society, 2012: 1-8.
- [26] LI H, ZHANG Q F. Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2008, 13(2): 284-302.
- [27] 包晓安, 魏雪, 陈磊, 等. 基于 mean-variance 的服务集群负载均衡方法[J]. 电信科学, 2017, 33(1): 6-13.
- BAO X A, WEI X, CHEN L, et al. Load balancing method of service cluster based on mean-variance[J]. Telecommunications Science, 2017, 33(1): 6-13.

[作者简介]



包晓安 (1973—), 男, 浙江理工大学教授, 主要研究方向为云计算、自适应软件和智能信息处理。



曹云棣 (1990—), 女, 浙江理工大学硕士生, 主要研究方向为云计算、智能计算与分布式处理。



张娜 (1977—), 女, 浙江理工大学副教授, 主要研究方向为分布式数据处理、软件工程。



钱俊彦 (1973—), 男, 桂林电子科技大学教授, 主要研究方向为软件工程、软件分析和云计算。



曹建文 (1969—), 男, 中国科学院软件研究所博士生导师, 主要研究方向为高性能并行软件与算法的研究与开发。