计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.201000102

边缘环境下基于模糊理论的科学工作流调度研究

林潮伟[1,2]　林　兵[2,3]　陈　星[1,2]
1 福州大学数学与计算机科学学院　福州 350108
2 福建省网络计算与智能信息处理重点实验室　福州 350108
3 福建师范大学物理与能源学院　福州 350117
(cwlin1998@foxmail.com)

摘　要　作为一种新型计算范式,边缘计算已成为解决大规模科学应用程序的重要途径。针对边缘环境下的科学工作流调度问题,考虑到任务计算过程中的服务器执行性能波动和数据传输过程中的带宽波动造成的不确定性,文中基于模糊理论,使用三角模糊数表示任务计算时间和数据传输时间,同时提出一种基于遗传算法算子的自适应离散模糊粒子群优化算法(Adaptive Discrete Fuzzy GA-based Particle Swarm Optimization,ADFGA-PSO),目的是在满足工作流截止日期约束的前提下,降低其模糊执行代价。该方法引入遗传算法的两点交叉算子以及关于任务优先级的邻域变异算子和关于服务器编号的自适应多点变异算子,避免粒子陷入局部最优,有效提高算法的搜索性能。实验结果表明,与其他调度策略相比,基于 ADFGA-PSO 的调度策略能够更加有效地降低边缘环境下带截止日期约束的科学工作流的模糊执行代价。

关键词:边缘计算;工作流调度;不确定性;三角模糊数;遗传算子
中图法分类号　TP338

Study on Scientific Workflow Scheduling Based on Fuzzy Theory Under Edge Environment

LIN Chao-wei[1,2],LIN Bing[2,3] and CHEN Xing[1,2]
1 College of Mathematics and Computer Science,Fuzhou University,Fuzhou 350108,China
2 Fujian Provincial Key Laboratory of Networking Computing and Intelligent Information Processing,Fuzhou 350108,China
3 College of Physics and Energy,Fujian Normal University,Fuzhou 350117,China

Abstract　As a novel computing paradigm,edge computing has become a significant approach to solve large-scale scientific applications. Aiming at scientific workflow scheduling under edge environment,task computation time and data transmission time are uncertain due to the fluctuation of server processing performance and bandwidth,respectively. In order to help capture and reflect the uncertainty during workflow execution,task computation time and data transmission time are represented as triangular fuzzy numbers (TFN),based on fuzzy theory. Simultaneously,an adaptive discrete fuzzy GA-based particle swarm optimization (ADFGA-PSO) is proposed to minimize fuzzy execution cost of workflow while satisfying deadline constraint. Besides,two-point crossover operator,neighborhood mutation and adaptive multipoint mutation operator of genetic algorithm (GA) are introduced to avoid particles being trapped in local optimum. Experimental results show that,compared with others,scheduling strategy based on ADFGA-PSO can more effectively reduce fuzzy execution cost in regard to deadline-constrained scientific workflow scheduling under edge environment.

Keywords　Edge computing,Workflow scheduling,Uncertainty,Triangular fuzzy numbers,Genetic operators

1　引言

在生物信息学、天文学和物理学等其他科学领域[1],科学工作流(下文简称为工作流)被广泛用于大规模科学应用程序的建模。工作流通常是数据密集型或计算密集型的,由成百上千个数据相互依赖的任务组成。事实上,工作流调度至关重要,其结果的好坏将直接影响科学应用系统的运行性能[2]。由于工作流自身结构的复杂性以及任务之间的数据依赖关系,即使在高性能计算环境下,在合理的时间范围内完成工作流的执行仍然是一项挑战。

就云计算环境下的工作流调度而言,由于用户终端与云端资源之间的物理距离较远,任务之间的大规模数据传输

到稿日期:2020-10-20　返修日期:2021-03-05　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家重点研发计划(2018YFB1004800);福建省自然科学基金杰青项目(2020J06014)
This work was supported by the National Key R&D Program of China(2018YFB1004800) and Natural Science Foundation of Fujian Province for Distinguished Young Scholars(2020J06014).
通信作者:林兵(WheelLX@163.com)

往往存在着响应时延高、带宽压力大等问题。在传统云计算架构的基础上，边缘计算技术通过在网络边缘部署一定的计算和存储能力，来增强移动网络的计算能力，从而为用户提供低时延和高带宽的网络服务。通过将数据密集型任务调度到边缘，将计算密集型任务调度到云端[3]，不仅满足了计算密集型科学应用程序的需求，同时有效地降低了数据传输延迟。尽管如此，云端虚拟机和边缘服务器（下文统称为服务器）之间往往是异构的，服务器的处理能力、传输带宽以及在负载和能耗方面产生的代价效益存在差异[4]，因此需要一个高性价比的调度策略，使得在满足截止日期约束的同时，最大限度降低工作流的执行代价。

另外，现有关于工作流调度的研究主要集中在确定性环境下，服务器的 CPU、带宽等其他因素相对稳定[5-7]。然而，在实际调度过程中，服务器的执行性能波动和带宽波动等因素都会对工作流调度产生不确定的影响，任务的计算时间和任务之间的数据传输时间难以预估，且不可忽视。当前相关工作主要针对模糊作业车间调度[8-9]，尚缺乏对于大规模科学工作流模糊调度问题的深入研究，因此本文主要研究边缘环境下基于模糊理论的工作流调度问题。受文献[8-9]的启发，本文使用三角模糊数来表示边缘环境下任务计算时间和数据传输时间的不确定性，同时提出一种基于遗传算法算子的自适应离散模糊粒子群优化算法（ADFGA-PSO），在尽可能满足截止日期的条件下，最小化工作流在边缘环境下的执行代价。本文的主要贡献包括以下 3 个部分：

1）考虑任务计算过程中的服务器执行性能波动和数据传输过程中的带宽波动对工作流调度造成的不确定性。

2）采用二维离散粒子来编码工作流的模糊调度策略，并引入遗传算法的两点交叉算子以及关于任务优先级的邻域变异算子和关于服务器编号的自适应多点变异算子，有效提高 ADFGA-PSO 的搜索性能，避免粒子陷入局部最优。

3）综合考虑由任务计算和数据传输引起的科学工作流模糊执行代价，设计了一种基于 ADFGA-PSO 的带截止日期约束的科学工作流代价驱动调度策略。

2 相关工作

科学工作流由一系列存在数据依赖关系的计算任务组成，常用于模拟和分析现实世界中复杂的大规模科学应用程序[10]。工作流调度至关重要，其结果将直接影响科学应用程序的性能。

由于边缘计算技术能够有效降低工作流调度的系统时延[11]，边缘环境下工作流调度或卸载已引起广泛研究。Xie 等[12]设计了一种新型的定向非局部收敛粒子群优化算法，以达到同时优化工作流的完成时间和执行代价的目标。Huang 等[13]提出了一种安全且高能效的计算卸载策略，其目标是在风险概率和截止日期的约束下优化工作流的能耗，实验表明，该策略能够为移动应用程序实现安全性和高能源效率。Peng 等[14]开发出一种基于 Krill-Herd 的算法，来评估移动边缘计算环境中的资源可靠性，结果表明该方法在成功率和完成时间方面均明显优于传统方法。Lin 等[15]根据自动驾驶中实时推理任务的差异和各个时隙中边缘节点的变化，设计了一种在边缘环境中的工作流调度策略，同时建立马尔可夫决策过程来描述问题模型，并提出基于模拟退火算法的 Q-learning 算法，其实验结果从有效性、可行性、探索性和收敛性 4 个方面展示了该算法的性能。然而，现有关于边缘环境下工作流调度的研究，较少考虑在截止日期约束下对任务计算和数据传输引起的工作流执行代价进行优化。

传统的工作流调度研究主要集中于确定性环境。然而，在实际的工作流调度中，服务器的性能和带宽存在波动，边缘环境的不确定性是必然存在的。现有不确定性计算环境下的模糊调度研究主要面向智能制造系统。Lei[16]使用三角模糊数表示模糊完成时间，用梯形模糊数表示模糊交货期，同时提出了一种改进的模糊取大运算，研究关于客户满意度的带可用性约束的模糊作业车间调度问题。Sun 等[8]使用三角模糊数表示处理时间，同时提出一种模糊化方法，将经典数据集中的处理时间模糊化为三角模糊数，研究模糊柔性作业车间调度问题。Fortemps[17]将不确定的持续时间表示为六点模糊数，并以最小化模糊完成时间为目标，建立模糊调度模型。然而，模糊不确定性边缘环境下的工作流调度仍然是一个亟待解决的问题。

因此，边缘环境下考虑模糊任务计算代价和模糊数据传输代价的工作流调度更具实际意义。本文将研究模糊不确定性边缘环境下基于截止日期约束的代价驱动工作流调度问题。

3 问题定义

边缘计算环境下的工作流调度框架主要包括三个部分：边缘环境、带截止日期约束的工作流以及代价驱动调度器。

3.1 确定性边缘环境下的工作流调度

边缘计算环境 $\boldsymbol{S}=\{\boldsymbol{S}_{\text{cloud}},\boldsymbol{S}_{\text{edge}}\}$ 由云和边缘组成，云 $\boldsymbol{S}_{\text{cloud}}$ 包含 n 个云服务器 $\{s_1,s_2,\cdots,s_n\}$，边缘 $\boldsymbol{S}_{\text{edge}}$ 包含 m 个边缘服务器 $\{s_{n+1},s_{n+2},\cdots,s_{n+m}\}$，其中服务器 s_i 可表示为：

$$s_i=(t_i^{\text{boot}},p_i,c_i^{\text{com}},f_i) \tag{1}$$

其中，t_i^{boot} 表示服务器 s_i 的初始化时间；p_i 表示服务器 s_i 的单位要价时间；c_i^{com} 表示服务器 s_i 在 p_i 时间内的单位计算代价；$f_i=\{0,1\}$ 表示服务器 s_i 所属平台的类型，当 $f_i=0$ 时，s_i 属于云，当 $f_i=1$ 时，s_i 属于边缘。

边缘环境下服务器 s_i 和 s_j 之间的带宽 $b_{i,j}$ 表示如下：

$$b_{i,j}=(\beta_{i,j},c_{i,j}^{\text{tran}}) \tag{2}$$

其中，$\beta_{i,j}$ 表示带宽 $b_{i,j}$ 的值，$c_{i,j}^{\text{tran}}$ 表示从服务器 s_i 传输 1GB 数据到服务器 s_j 产生的代价。

工作流可以用有向无环图 $\boldsymbol{W}=(\boldsymbol{V},\boldsymbol{D})$ 来表示，$\boldsymbol{V}$ 表示包含 l 个任务的顶点集合 $\{v_1,v_2,\cdots,v_l\}$，$\boldsymbol{D}$ 表示任务之间的数据依赖集合 $\{d_{1,2},d_{1,3},\cdots,d_{i,j},\cdots\}$，$\forall i\neq j$。对于 $d_{i,j}=\langle v_i,v_j\rangle$，任务 v_j 是任务 v_i 的后继节点，任务 v_i 是任务 v_j 的前驱节点，$d_{i,j}$ 的值为任务 v_i 传输到任务 v_j 的数据量。每个工作流 $\boldsymbol{W}$ 都有一个对应的截止日期 $D(\boldsymbol{W})$，在某个调度策略中，若工作流能够在相应的截止日期前被执行完成，则称该调度策略为一个可行解。

本文研究主要关注工作流的任务计算时间 t_{com} 和数据传输时间 t_{tran}。在不同的调度策略下，工作流中任务计算和数据传输的时间是不同的，所产生的执行性价比也是不同的。

假设任务 v_i 在服务器 s_j 上执行所需的计算时间为 $t_{com}(v_i,s_j)$，这里考虑串行处理模型[18]，即一个任务只能在一个服务器上处理，且一个服务器同时只能处理一个任务。对于数据依赖边 $d_{k,l}$，其数据传输时间 $t_{tran}(d_{k,l},s_r,s_t)$，即数据 $d_{k,l}$ 从服务器 s_r 传输到服务器 s_t 的时间为：

$$t_{tran}(d_{k,l},s_r,s_t)=\frac{d_{k,l}}{\beta_{r,t}} \tag{3}$$

工作流的调度策略可以被定义为：

$$\boldsymbol{\Psi}=(\boldsymbol{W},\boldsymbol{S},\boldsymbol{M},t_{total},c_{total}) \tag{4}$$

其中，$\boldsymbol{M}=\{(v_i,s_j)\bigcup(d_{k,l},s_r,s_t)\mid v_i\in\boldsymbol{V},d_{k,l}\in\boldsymbol{D},s_j,s_r,s_t\in\boldsymbol{S}\}$ 表示工作流 $\boldsymbol{W}=(\boldsymbol{V},\boldsymbol{D})$ 对应于边缘环境 $\boldsymbol{S}$ 的映射关系，t_{total} 表示工作流的完成时间，c_{total} 表示工作流的执行代价。

在某个调度策略 $\boldsymbol{\Psi}$ 中，一旦映射 $\boldsymbol{M}$ 被确定，每个任务执行的服务器以及任务之间的数据传输也随之确定。因此，每个任务 v_i 都有其相应的开始时间 $t_{start}(v_i)$ 和完成时间 $t_{end}(v_i)$。工作流的完成时间 t_{total} 由下式计算得到。

$$t_{total}=\max_{v_i\in\boldsymbol{V}}\{t_{end}(v_i)\} \tag{5}$$

同样地，每个服务器 s_i 也有相应的开启时间 $t_{on}(s_i)$ 以及关闭时间 $t_{off}(s_i)$。工作流的任务计算代价 c_{com} 和数据传输代价 c_{tran} 的计算如下：

$$c_{com}=\sum_{i=1}^{|S|}c_i^{com}\left\lceil\frac{t_{off}(s_i)-t_{on}(s_i)}{p_i}\right\rceil \tag{6}$$

$$c_{tran}=\sum_{v_j\in V}\sum_{v_k\in V}c_{r,t}^{tran}d_{j,k},(v_j,s_r),(v_k,s_t)\in\boldsymbol{M} \tag{7}$$

其中，当 v_k 和 v_j 在同一服务器上被执行时，$c_{r,t}^{tran}=0$，此时不产生数据传输代价。

由于工作流执行过程中数据存储、资源监控等代价与上述代价相比可以忽略不计，从而只考虑任务计算代价和数据传输代价。因此，工作流的执行代价可以表示为：

$$c_{total}=c_{com}+c_{tran} \tag{8}$$

综上所述，对于边缘环境下的工作流调度问题，代价驱动调度器的目标是在完成时间 t_{total} 满足工作流截止日期 $D(\boldsymbol{W})$ 约束的同时，最小化工作流的执行代价 c_{total}。因此，该问题可形式化表示为：

$$\begin{cases}\min\ c_{total}\\ \text{s.t. }\ t_{total}\leqslant D(\boldsymbol{W})\end{cases} \tag{9}$$

3.2 不确定性边缘环境下的工作流调度

上述问题定义中，我们总是假定工作流执行过程中任务计算时间和数据传输时间是确定的。然而，在实际的边缘环境中，由于任务计算过程中的服务器执行性能波动以及数据传输过程中的带宽波动，工作流的执行过程是不确定的。为了刻画工作流执行过程的不确定性，本节引入模糊理论[19]，用三角模糊数来表示任务计算时间和数据传输时间。

三角模糊数 $\tilde{t}=(t^l,t^m,t^u)$ 的隶属函数 $\mu_{\tilde{t}}(x)$ 如图 1 所示。

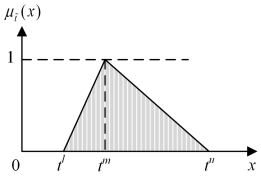


图 1 三角模糊数的隶属函数

Fig. 1 Membership function of a TFN

其中，顶点 t^m 表示最可能的执行时间，左右端点 t^l 和 t^u 表示执行时间的可能范围。当 $t^l=t^m=t^u$ 时，$\tilde{t}$ 即实数。

本文统一用“$\tilde{\tau}$”表示变量 τ 在不确定性边缘环境下对应的模糊变量。基于不确定性的概念，在边缘环境下的工作流调度问题中，工作流的执行代价和完成时间均为三角模糊数，表示为 $\tilde{c}_{total}$ 和 $\tilde{t}_{total}$。基于式(9)，该问题可形式化表示为：

$$\begin{cases}\min\ \tilde{c}_{total}\\ \text{s.t. }\ \tilde{t}_{total}\leqslant D(\boldsymbol{W})\end{cases} \tag{10}$$

对于目标函数 $\tilde{c}_{total}$，其函数值是一个三角模糊数，其优劣由均值 $mean(\tilde{c}_{total})$ 与标准差 $std(\tilde{c}_{total})$ 共同决定。Palacios 等提出一种比较准则[20]来最小化两个目标值的线性组合，称作 Palacios 准则。因此，目标函数可转化为：

$$\min\ \tilde{c}_{total}=(c^l,c^m,c^u)\Rightarrow\min\ Z(\tilde{c}_{total})=mean(\tilde{c}_{total})+\eta\cdot std(\tilde{c}_{total}) \tag{11}$$

其中，$\eta\geqslant 0$ 为标准差 $std(\tilde{c}_{total})$ 的权重。

基于模糊事件概率测度的概念，Lee 等定义了模糊集分别在均匀分布和比例分布两种情形下的均值和标准差[21]。而工作流的模糊执行代价 $\tilde{c}_{total}$，是一种基于比例分布的情形，其均值和标准差分别由下式给出：

$$mean(\tilde{c}_{total})=\frac{\int x\cdot\mu_{\tilde{c}_{total}}^2(x)\mathrm{d}x}{\int\mu_{\tilde{c}_{total}}^2(x)\mathrm{d}x}=\frac{c^l+2c^m+c^u}{4} \tag{12}$$

$$\begin{aligned}std(\tilde{c}_{total})&=\left[\frac{\int x^2\mu_{\tilde{c}_{total}}^2(x)\mathrm{d}x}{\int\mu_{\tilde{c}_{total}}^2(x)\mathrm{d}x}-[mean(\tilde{c}_{total})]^2\right]^{1/2}\\&=\left[\frac{2(c^l-c^m)^2+(c^l-c^u)^2+2(c^m-c^u)^2}{80}\right]^{1/2}\end{aligned} \tag{13}$$

对于约束条件 $\tilde{t}_{total}\leqslant D(\boldsymbol{W})$，基于时效性考虑，对于一个可行调度策略，在最坏情况下工作流的完成时间，即 $\tilde{t}_{total}$ 的上界 t^u 也应该满足截止日期约束。因此，约束条件可转化为：

$$\text{s.t. }\ \tilde{t}_{total}=(t^l,t^m,t^u)\leqslant D(\boldsymbol{W})\Rightarrow\text{s.t. }\ t^u\leqslant D(\boldsymbol{W}) \tag{14}$$

综上所述，边缘环境下基于模糊理论的工作流调度问题可形式化表示为：

$$\begin{aligned}&\min\ Z(\tilde{c}_{total})\\&\text{s.t. }\ t^u\leqslant D(\boldsymbol{W})\end{aligned} \tag{15}$$

3.3 工作流调度中的模糊理论

3.3.1 数据集的模糊化

在工作流的实际调度过程中，其任务计算时间和数据传输时间更可能大于预估时间[22]。基于 Sun 等[8]所采用的方法，本文提出一种更加符合实际的模糊化方法来刻画任务计算时间和数据传输时间的不确定性。

对于预估时间 t，其对应的模糊时间 $\tilde{t}=(t^l,t^m,t^u)$ 的参数定义如下：t^m 的值取为 t；t^l 和 t^u 分别从区间 $[\delta_1 t,t]$ 与 $[2t-t^l,\delta_2 t]$ 中随机选取，其中，$\delta_2-1>1-\delta_1>0$。

此时，构造得到的三角模糊数 $\tilde{t}$ 满足 $t^u-t^m\geqslant t^m-t^l$，其均值 $mean(\tilde{t})$ 如式(16)所示。也就是说，在模糊事件的概率测度意义下，其均值 $mean(\tilde{t})$ 更可能大于预估时间 t。

$$mean(\tilde{t})=\frac{t^l+2t^m+t^u}{4}\geqslant t^m=t \tag{16}$$

3.3.2 数据集的模糊化

在不确定性边缘环境下,构建工作流的模糊调度策略时,模糊数的一些运算需要重新定义,如加法运算、比较运算、取大运算和数乘运算。值得注意的是,对于实数 t,我们将其看作特殊的模糊数 $\tilde{t}=(t,t,t)$进行运算。

加法运算被用于计算任务的模糊完成时间。对于两个三角模糊数 $\tilde{r}=(r^l,r^m,r^u)$和 $\tilde{t}=(t^l,t^m,t^u)$,根据模糊理论中关于两个模糊数加法的定义,其加法运算由下式给出:

$$\tilde{r}+\tilde{t}=(r^l+t^l,r^m+t^m,r^u+t^u) \tag{17}$$

比较运算被用于比较模糊完成时间的长短,从而得到取大运算的结果。本文采用 Sakawa 等[23]提出的比较准则来比较 $\tilde{r}=(r^l,r^m,r^u)$和 $\tilde{t}=(t^l,t^m,t^u)$,具体如下:

1)若 $c_1(\tilde{r})=(r^l+2r^m+r^u)/4>c_1(\tilde{t})$,则 $\tilde{r}>\tilde{t}$;

2)若 $c_1(\tilde{r})=c_1(\tilde{t})$且 $c_2(\tilde{r})=r^m>c_2(\tilde{t})$,则 $\tilde{r}>\tilde{t}$;

3)若 $c_1(\tilde{r})=c_1(\tilde{t})$,$c_2(\tilde{r})=c_3(\tilde{t})$且 $c_3(\tilde{r})=r^u-r^l>c_3(\tilde{t})$,则 $\tilde{r}>\tilde{t}$。

取大运算被用于计算前驱任务的模糊完成时间和当前服务器的模糊完成时间的最大值,从而确定任务的模糊开始时间。对于两个三角模糊数 $\tilde{r}=(r^l,r^m,r^u)$和 $\tilde{t}=(t^l,t^m,t^u)$,其取大运算 $\tilde{r}\vee\tilde{t}$ 的隶属函数 $\mu_{\tilde{r}\vee\tilde{t}}(z)$的定义如下:

$$\begin{aligned}\mu_{\tilde{r}\vee\tilde{t}}(z)&=\sup_{z=x\vee y}\min(\mu_{\tilde{r}}(x),\mu_{\tilde{t}}(y))\\&\triangleq\bigvee_{z=x\vee y}(\mu_{\tilde{r}}(x)\wedge\mu_{\tilde{t}}(y))\end{aligned} \tag{18}$$

本文采用 Lei 准则[16]近似计算两个三角模糊数的最大值。对于 $\tilde{r}=(r^l,r^m,r^u)$和 $\tilde{t}=(t^l,t^m,t^u)$,$\tilde{r}\vee\tilde{t}$ 近似如下:

$$\tilde{r}\vee\tilde{t}\cong\begin{cases}\tilde{r},\tilde{r}\geqslant\tilde{t}\\\tilde{t},\tilde{r}<\tilde{t}\end{cases} \tag{19}$$

数乘运算被用于计算服务器的模糊计算代价和模糊传输代价。一般地,对于三角模糊数 $\tilde{t}=(t^l,t^m,t^u)$,其数乘运算由下式给出:

$$\lambda\cdot\tilde{t}=(\lambda t^l,\lambda t^m,\lambda t^u),\forall\lambda\in\mathbb{R} \tag{20}$$

另外,三角模糊数与实数之间的除法运算可以等价为数乘运算,即:

$$\tilde{t}\div\kappa\triangleq\lambda\cdot\tilde{t},\lambda=1/\kappa,\forall\kappa\in\mathbb{R} \tag{21}$$

4 基于 ADFGA-PSO 算法的调度策略

本节首先介绍传统 PSO 算法,然后具体阐述 ADFGA-PSO 算法的主要内容。

4.1 PSO 算法

PSO 算法是一种基于种群社会行为的群智能优化技术,最早由 Kennedy 等[24]于 1995 年提出,其核心思想是基于对鸟群觅食行为的研究,粒子被用于模拟鸟群中的个体,对应于优化问题的一个候选解,其运动速度可根据自身当前情况、粒子个体最优位置和种群全局最优位置进行调整,粒子运动过程即为问题搜索过程。为了评价每个粒子的好坏,引入适应度函数来评估粒子的质量。每个粒子通过学习自身和其他粒子的经验,不断更新自身的速度和位置,以获得更好的适应度,其更新方式如下:

$$V_i^{t+1}=w\cdot V_i^t+c_1r_1(pBest_i^t-X_i^t)+c_2r_2(gBest^t-X_i^t) \tag{22}$$

$$X_i^{t+1}=X_i^t+V_i^{t+1} \tag{23}$$

其中,t 表示当前迭代次数;V_i^t 和 X_i^t 分别表示第 i 个粒子经过 t 次迭代后的速度和位置;$pBest_i^t$ 和 $gBest^t$ 分别表示第 i 个粒子的个体最优位置和种群全局最优位置;w 是惯性因子,决定算法的搜索能力,从而影响算法的收敛性;c_1 和 c_2 是学习因子,分别体现粒子对个体认知部分和社会认知部分的学习能力;r_1 和 r_2 是取值于[0,1]区间的随机数,用于加强算法在迭代过程中的随机搜索能力。

4.2 ADFGA-PSO 算法

4.2.1 问题编码

为了使 PSO 算法更好地解决工作流调度这一离散型优化问题,受文献[25]的启发,本文提出一种新型编码方式,即优先级和服务器嵌套组成的二维离散粒子来编码工作流调度策略。粒子群中的每个粒子对应于工作流在不确定性边缘环境下的一种潜在调度策略,用 P_i^t 表示第 t 次迭代中粒子群的第 i 个粒子,如式(24)所示:

$$P_i^t=((\phi_{i1},s_{i1})^t,(\phi_{i2},s_{i2})^t,\cdots,(\phi_{i|V|},s_{i|V|})^t) \tag{24}$$

对于二元组$(\phi_{ij},s_{ij})^t$,ϕ_{ij} 表示第 j 个任务的优先级编码,该编码为一实数,其值越大,表明该任务在待执行队列中的优先级越高,若存在编码值相同的任务,则最先进入待执行队列的任务优先级更高;s_{ij} 表示第 j 个任务的服务器编码,该编码为一整数,它的值代表不同的服务器编号。图 2 展示了包含 8 个任务的工作流调度策略所对应的编码粒子,以任务 3-5 为例,若任务 3-5 同时在待执行队列中,则根据任务优先级大小,优先为任务 4 分配第 1 个服务器;然后依次为任务 5 和任务 3 分配相应的第 3 个服务器。

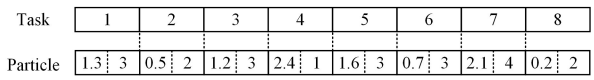


图 2 对应于工作流调度策略的编码粒子

Fig. 2 Encoded particle corresponding to workflow scheduling strategy

4.2.2 模糊适应度函数

本文研究旨在模糊完成时间 $\tilde{t}_{\text{total}}$满足截止日期约束的同时,最小化工作流的模糊执行代价 $\tilde{c}_{\text{total}}$。由此可见,本文提出的编码策略可能存在不满足截止日期约束的不可行解。因此,用于比较两个候选解优劣性的模糊适应度函数,必须根据以下 3 种不同情况进行区分定义。

1)两个粒子都是可行解。选择模糊执行代价 $\tilde{c}_{\text{total}}$较低的粒子,其模糊适应度函数定义为:

$$F(P_i^t)=\tilde{c}_{\text{total}}(P_i^t)\Rightarrow Z(\tilde{c}_{\text{total}}(P_i^t)) \tag{25}$$

2)一个粒子是可行解,另一个粒子是不可行解。显然要选择可行解,其模糊适应度函数定义为:

$$\begin{aligned}F(P_i^t)&=\begin{cases}\tilde{c}_{\text{total}}(P_i^t), & \tilde{t}_{\text{total}}(P_i^t)\leqslant D(\boldsymbol{W})\\\infty, & \tilde{t}_{\text{total}}(P_i^t)>D(\boldsymbol{W})\end{cases}\\&\Rightarrow\begin{cases}Z(\tilde{c}_{\text{total}}(P_i^t)), & t^u(P_i^t)\leqslant D(\boldsymbol{W})\\\infty, & t^u(P_i^t)>D(\boldsymbol{W})\end{cases}\end{aligned} \tag{26}$$

3)两个粒子都是不可行解。选择模糊完成时间 $\tilde{t}_{\text{total}}$较小的粒子,因为该粒子经过进化后更有可能成为可行解,其模糊适应度函数定义为:

$$F(P_i^t)=\tilde{t}_{\text{total}}(P_i^t)\Rightarrow t^u(P_i^t) \tag{27}$$

4.2.3 粒子的更新策略

在迭代过程中，每个粒子的更新受其自身当前情况、粒子个体最优位置和种群全局最优位置的影响[26]。为了避免传统 PSO 算法过早收敛的缺陷，ADFGA-PSO 引入遗传算法的变异算子和交叉算子，对式(22)相应部分进行更新。在第 $t+1$ 次迭代中，第 i 个粒子的更新方式如式(28)所示，其中，$\odot$ 和 $\otimes$ 分别表示遗传算法的变异算子和交叉算子。

$$P_i^{t+1}=c_2\otimes(c_1\otimes(w\odot P_i^t,pBest_i^t),gBest^t) \tag{28}$$

对于惯性部分，引入遗传算法中的变异算子对式(22)的相应部分进行更新，其更新方式如下：

$$A_i^{t+1}=w\odot P_i^t,r<w \tag{29}$$

其中，r 是取值于[0,1]区间的随机数，仅当 $r<w$ 时，对粒子 P_i^t 执行双变异算子 $\odot$。双变异算子具体由以下两个部分组成。

对于任务优先级，邻域变异算子[27]随机选取粒子的 3 个分位，基于 3 个分位的优先级编码，生成该粒子的邻域，并选择其中任一粒子作为变异后的邻域粒子，如图 3 所示。对于服务器编号，自适应多点变异算子随机选取邻域粒子(以图 3 中的第二个邻域粒子为例)的 k 个分位，分别对每个分位的服务器编码在[1,|$\boldsymbol{S}$|]内进行变异，生成新的编码粒子，如图 4 所示。

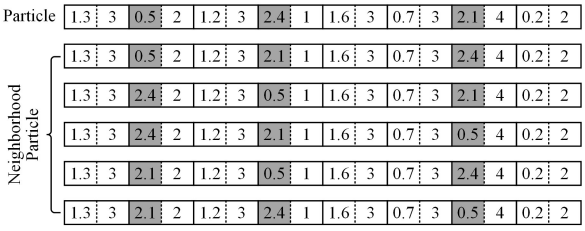


图 3 关于任务优先级的邻域变异算子

Fig. 3 Neighborhood mutation operator for task order

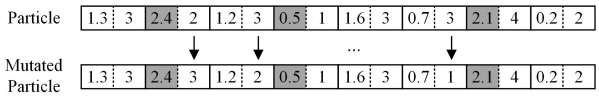


图 4 关于服务器编号的自适应多点变异算子

Fig. 4 Adaptive multipoint mutation operator for server number

对于个体认知和社会认知部分，引入遗传算法中的交叉算子对式(22)的相应部分进行更新，其更新方式如式下：

$$B_i^{t+1}=c_1\otimes(A_i^{t+1},pBest_i^t),r_1<c_1 \tag{30}$$

$$P_i^{t+1}=c_2\otimes(B_i^{t+1},gBest^t),r_2<c_2 \tag{31}$$

其中，r_1 和 r_2 是取值于[0,1]区间的随机数，仅当 $r_1<c_1$(或 $r_2<c_2$)时，对 A_i^{t+1}(B_i^{t+1})执行两点交叉算子 $\otimes$，随机选取待更新粒子中的 2 个编码分位，然后将 2 个分位之间的编码与 $pBest$(或 $gBest$)中对应编码进行交叉，如图 5 所示。

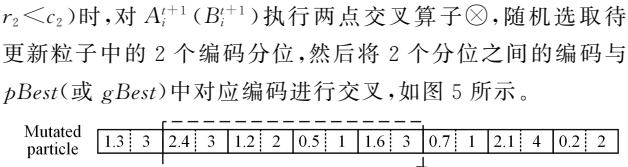


图 5 个体(或社会)认知部分的两点交叉算子

Fig. 5 Two-point crossover operator for individual cognition and social cognition

4.2.4 参数调整

对于惯性因子，本文提出一种新的调整机制，能够根据当前粒子 P_i^t 的质量，即与全局最优粒子 $gBest^t$ 之间的差异，自适应调整惯性因子 w 的值，从而增强 ADFGA-PSO 算法的搜索能力。惯性因子 w 的调整方式如下式所示：

$$\begin{aligned}w&=w_{\max}-(w_{\max}-w_{\min})\times\exp\left(\frac{d(P_i^t)}{d(P_i^t)-1.01}\right)\\d(P_i^t)&=\frac{div(gBest^t,P_i^t)}{|P_i^t|}\end{aligned} \tag{32}$$

其中，$w_{\max}$ 和 $w_{\min}$ 分别表示 w 的最大值和最小值，$|P_i^t|$ 表示粒子 P_i^t 的编码空间大小，$div(gBest^t,P_i^t)$ 表示当前粒子和全局最优粒子之间不同编码值的位数。当 $div(gBest^t,P_i^t)$ 较小时，意味着 $gBest^t$ 与 P_i^t 之间的差异较小，应倾向于增强粒子的局部搜索能力，以提高算法的收敛效果，因此减小惯性因子 w 的大小；反之，则增大惯性因子，提高粒子的全局搜索能力，扩大对解空间的搜索。

对于惯性部分所采用的自适应多点变异算子，其变异位数 k 随惯性因子 w 的变化进行正反馈调整，如式(33)所示：

$$k=k_{\min}+(k_{\max}-k_{\min})\cdot\frac{w-w_{\min}}{w_{\max}-w_{\min}} \tag{33}$$

其中，$k_{\max}$ 和 $k_{\min}$ 分别表示变异位数 k 的最大值和最小值。当惯性因子 w 较大时，变异位数 k 增加，提高变异算子的变异能力，从而增强算法的全局搜索能力；反之，则减少变异位数 k，只保留一定的变异能力，以维持种群在进化过程中的多样性，从而增强算法的局部搜索能力。

随着算法的不断迭代，学习因子 c_1，c_2 采用线性增减的方式进行动态调整[28]，具体的更新策略不再赘述。

4.2.5 基于 ADFGA-PSO 的工作流模糊调度策略

首先，对于 4.2.1 节中定义的编码粒子，给出不确定性边缘环境下该粒子到工作流模糊调度策略的映射，具体如算法 1 所示。

算法 1 编码粒子 P 到工作流模糊调度策略 $\boldsymbol{\Psi}$ 的映射

输入：$\mathbf{W}$，$\mathbf{S}$，P

输出：$\boldsymbol{\Psi}=(\mathbf{W},\mathbf{S},\mathbf{M},\tilde{t}_{total},\tilde{c}_{total})$

1. 初始化：$\mathbf{M}\leftarrow$null，$\tilde{c}_{tran}\leftarrow(0,0,0)$；
2. 计算 $\tilde{t}_{com}[|\mathbf{V}|\times|\mathbf{S}|]$，$\tilde{t}_{tran}[|\mathbf{D}|,|\mathbf{S}|\times|\mathbf{S}|]$；
3. for i=1 to i=$|\mathbf{V}|$ do
4. $\quad\mathbf{M}=\mathbf{M}\cup(v_i,s_j)$；// 基于编码粒子 P
5. $\quad$if v_i是入任务 then // v_i不存在父任务
6. $\quad\quad$if s_j is off then
7. $\quad\quad\quad$启动 s_j；
8. $\quad\quad\quad\tilde{t}_{off}(s_j)=\tilde{t}_j^{boot}$，$\tilde{t}_{on}(s_j)=(0,0,0)$；
9. $\quad\quad$end if
10. $\quad\quad\tilde{t}_{start}(v_i)=\tilde{t}_{off}(s_j)$；
11. $\quad$else
12. $\quad\quad$maxT=(0,0,0)；
13. $\quad\quad$foreach parent v_p of v_i do
14. $\quad\quad\quad$maxT=maxT $\vee$ ($\tilde{t}_{end}(v_p)+\tilde{t}_{tran}(d_{p,i},s_q,s_j)$)；//($v_p$，$s_q$)，($v_i$，$s_j$)$\in\mathbf{M}$
15. $\quad\quad\quad\tilde{c}_{tran}+=$fuzzy($c_{q,j}^{tran}\cdot d_{p,i}$)；//fuzzy(*)为模糊化函数

16. end for
17. if s_j is off then
18. 启动 s_j；
19. $\tilde{t}_{off}(s_j)=maxT\vee\tilde{t}_j^{boot}$；
20. $\tilde{t}_{on}(s_j)=\tilde{t}_{off}(s_j)-\tilde{t}_j^{boot}$；
21. end if
22. $\tilde{t}_{start}(v_i)=maxT\vee\tilde{t}_{off}(s_j)$；
23. end if
24. $\tilde{t}_{end}(v_i)=\tilde{t}_{start}(v_i)+\tilde{t}_{com}(v_i,s_j)$；
25. end for
26. 基于式(5)和式(8)计算 $\tilde{t}_{total}$，$\tilde{c}_{total}$；
27. if $\tilde{t}_{total}>D(\boldsymbol{W})$ then
28. 将 P 设置为不可行粒子；
29. return $\boldsymbol{\Psi}=\phi$；
30. end if
31. return $\boldsymbol{\Psi}=(\boldsymbol{W},\boldsymbol{S},\boldsymbol{M},\tilde{t}_{total},\tilde{c}_{total})$.

另外，基于 ADFGA-PSO 的工作流模糊调度策略，其核心算法 ADFGA-PSO 的主要流程包含以下 6 个步骤。

1)初始化 ADFGA-PSO 算法的相关参数，如种群大小 S_{pop}、最大迭代次数 $iter_{max}$、惯性因子 w、变异位数 k 和学习因子 c_1，c_2 等，并随机生成初始种群。

2)根据算法 1，基于式(25)－式(27)，计算每个粒子的适应度。每个粒子的初始状态设置为其个体最优粒子，而初始种群中适应度最小的粒子设置为当前的全局最优粒子。

3)根据式(28)－式(31)，引入遗传算法中的变异算子和交叉算子更新粒子的编码，并计算更新后粒子的适应度。

4)若更新后粒子的适应度小于其个体最优粒子，则将当前粒子设置为其个体最优粒子；否则，直接执行步骤 6。

5)若更新后粒子的适应度小于种群的全局最优粒子，则将当前粒子设置为种群全局最优粒子。

6)检查是否满足算法终止条件，即是否达到最大迭代次数。若满足，则终止算法；否则，返回至步骤 3。

5 实验仿真与结果

为了验证基于 ADFGA-PSO 的调度策略对截止日期约束下最小化工作流模糊执行代价的有效性，本节进行实验仿真和结果分析。ADFGA-PSO 和所有对比算法均在 Python 3.7 环境下实现，并在 16 GB RAM 和 3.60 GHz Intel (R) Core(TM) i5-7200U CPU 的 64 位 Win10 系统下运行。基于文献[28]，ADFGA-PSO 的参数设置如下：$S_{pop}=100$，$iter_{max}=1\,000$，$w_{max}=0.9$，$w_{min}=0.4$，$c_1^{start}=0.9$，$c_1^{end}=0.2$，$c_2^{start}=0.4$，$c_2^{end}=0.9$，$k_{max}=|\boldsymbol{V}|/10$，$k_{min}=1$。

5.1 实验设置

用于测试的工作流模型来自 Bharathi 等[29]对 5 个不同科学领域进行深入研究得到的 5 种科学工作流：地震科学的 CyberShake、生物基因学的 Epigenomics、重力物理学的 LIGO、天文学的 Montage 以及生物信息学的 SIPHT。每种工作流具有不同的结构、任务数量等，其计算需求和数据传输量等相关信息均被存储在 xml 文件中。针对每个科学领域，本文选取 3 种规模的工作流：微型(约 30 个任务)、小型(约 50 个任务)和中型(约 100 个任务)。

边缘环境下存在 3 个云服务器(s_1，s_2，s_3)和 2 个边缘服务器(s_4，s_5)，分别具有特定的任务计算能力和单位时间计算代价。假定云服务器 s_3 的计算能力最强，工作流中各任务在 s_3 上的计算时间直接由相应的 xml 文件得到。同时，以 s_3 为基准，云服务器 s_1(s_2)的计算能力大约为 s_3 的 1/2(1/4)，边缘服务器 s_4(s_5)的计算能力大约为 s_3 的 1/8(1/10)。另外，设定云服务器 s_3 的单位时间计算代价为 15.5 \$/h，其余服务器的单位时间计算代价与其计算能力近似成正比。另外，假定服务器的初始化时间 $\tilde{t}_i^{boot}=(97\,s,97\,s,97\,s)$，$\forall i\in[1,|\boldsymbol{S}|]$。

目前主流的商业云服务，如 Amazon EC2，通常以 60 s 或 1 h 作为单位要价时间 p_i，根据实际租赁时间进行按需付费，根据工作流的不同规模大小，本文选取不同的单位要价时间：微型和小型工作流以 60 s 为单位，中型工作流以 1 h 为单位。

服务器 s_i 和 s_j 之间的带宽 $b_{i,j}=(\beta_{i,j},c_{i,j}^{tran})$，根据二者所属平台的类型 f_i 和 f_j，按表 1 进行设置。

表 1 服务器 s_i 和 s_j 之间的带宽

Table 1 Bandwidth between s_i and s_j

f_i	↔	f_j	$\beta_{i,j}$/(MB/s)	$c_{i,j}^{tran}$(\$/GB)
0	↔	0	2.5	0.4
0	↔	1	1.0	0.16
1	↔	1	12.5	0.8

另外，第 3.3.1 节已详细介绍如何将任务计算时间 t_{com} 和数据传输时间 t_{tran} 模糊化为三角模糊数，其中参数 δ_1 和 δ_2 分别取为 0.85 和 1.2。为了更好地比较各算法的性能，对模糊执行代价和模糊完成时间进行去模糊化处理，具体方法详见第 3.2 节，其中 η 取 1。

最后，每个工作流都有一个对应的截止日期作为约束条件，并以此测试算法对于工作流调度问题的性能。针对每个工作流 $\boldsymbol{W}$，定义其截止日期 $D(\boldsymbol{W})$ 如下：

$$D(\boldsymbol{W})=1.5*HEFT(\boldsymbol{W}) \tag{34}$$

其中，$HEFT(\boldsymbol{W})$ 表示基于 HEFT 算法[30]的调度策略下工作流 $\boldsymbol{W}$ 的执行时间。

5.2 对比算法

本文提出的 ADFGA-PSO 基于传统 PSO 算法，引入遗传算法中的变异和交叉算子分别对粒子的惯性部分和个体(社会)认知部分进行更新，以期提升算法的搜索性能，避免粒子陷入局部最优。在本次实验中，为了测试 ADFGA-PSO 在不确定性边缘环境下的调度性能，我们采用以下 3 种算法作为本文的对比算法，其中下述算法中的“模糊”指在原算法的基础上，使其适用于不确定性边缘环境，调度过程中的任务计算时间和数据传输时间被模糊化为三角模糊数。

模糊粒子群优化 (Fuzzy Particle Swarm Optimization, FPSO)：该方法基于传统 PSO 框架[21]，采用与 ADFGA-PSO 类似的二维离散编码，任务的优先级和服务器编码均采用传统的连续编码值，并将服务器编码值四舍五入的结果作为服务器编号。另外，粒子的更新策略以及相关参数的设置参照文献[25]。

模糊遗传算法(Fuzzy Genetic Algorithm,FGA):该方法基于 GA 的更新策略[28],采用与 ADFGA-PSO 相同的优先级-服务器嵌套编码,通过二元锦标赛选择、两点交叉和交换变异算子对当前种群中个体的编码进行更新,并采用精英保留策略,将当前种群中的精英个体,即适应度值最高的个体,完整地复制到下一代,不断对染色体进行更新,最终输出种群的精英个体作为最优解。其中,FGA 的交叉概率和变异概率分别取 0.8 和 0.1。

模糊随机算法(Fuzzy Random Algorithm,FRA):该方法基于随机搜索策略[29],采用与 ADFGA-PSO 相同的编码方案,同时对问题的解空间进行随机搜索,从而生成种群中每个粒子的优先级和服务器编码,并将粒子映射到对应的调度策略,计算粒子的适应度值,记录随机搜索过程中的最优解,每次迭代之间互不影响,最终输出种群的最优解,即最优调度策略。

5.3 实验结果

为了比较 ADFGA-PSO 和其他对比算法在边缘环境下基于模糊理论的工作流调度性能,对不同类型不同规模的工作流分别进行 10 组独立重复实验,并分析 ADFGA-PSO 在工作流调度中的模糊执行代价方面的优越性。针对不同规模的大小工作流,表 2－表 4 记录了基于不同算法的调度策略下工作流模糊执行代价的最优值和平均值及其适应度(单位:$),并对各算法的最优解用加粗的字体表示,不可行解用“*”标注。

表 2 基于不同算法的调度策略对于微型工作流的模糊执行代价

Table 2 Fuzzy execution cost of different algorithms for tiny workflows

Workflows	Algorithms	Optimal fuzzy execution cost and its fitness	Meanfuzzy execution cost and its fitness
CyberShake	ADFGA-PSO	**(8.94,8.98,9.08),9.02**	**(11.25,11.48,12.14),11.74**
	FPSO	(11.08,11.19,11.60),11.36	(13.42,13.73,14.66),14.09
	FGA	(9.86,9.93,10.22),10.05	(12.18,12.49,13.31),12.81
	FRA	(17.96,18.66,20.65),19.43	(21.07,21.96,24.42),22.91
Epigenomics	ADFGA-PSO	**(146.64,154.25,174.53),162.05**	**(151.50,159.47,181.36),167.92**
	FPSO	(150.71,157.72,173.19),163.51	(157.01,164.49,177.16),169.03
	FGA	(162.41,166.14,177.28),170.49	(165.85,171.71,181.42),175.17
	FRA	(168.93,174.57,186.42),178.98	(173.47,180.76,198.27),187.40
LIGO	ADFGA-PSO	**(63.14,64.08,66.81),65.14**	**(63.77,65.81,70.11),67.41**
	FPSO	(64.80,67.05,72.19),68.98	(67.28,69.06,73.61),70.80
	FGA	(64.59,66.27,70.48),67.88	(66.90,68.27,72.09),69.75
	FRA	(79.77,81.74,86.06),83.36*	(87.39,89.05,93.78),90.88
Montage	ADFGA-PSO	**(3.89,3.99,4.17),4.05**	**(4.11,4.24,4.48),4.32**
	FPSO	(4.79,4.92,5.33),5.08	(5.44,5.64,6.16),5.84
	FGA	(4.79,5.00,5.32),5.11	(5.52,5.76,6.30),5.96
	FRA	(12.45,13.07,14.53),13.62	(13.76,14.49,16.28),15.17
SIPHT	ADFGA-PSO	(56.26,58.05,64.33),60.54	**(56.58,58.38,64.79),60.93**
	FPSO	**(58.51,59.79,61.88),60.53**	(59.41,60.90,63.78),61.95
	FGA	(60.69,60.84,61.31),61.02	(62.21,63.41,65.96),64.36
	FRA	(68.62,69.60,72.03),70.52	(70.62,71.72,74.62),72.84

表 3 基于不同算法的调度策略对于小型工作流的模糊执行代价

Table 3 Fuzzy execution cost of different algorithms for small workflows

Workflows	Algorithms	Optimal fuzzy execution cost and its fitness	Meanfuzzy execution cost and its fitness
CyberShake	ADFGA-PSO	**(16.50,16.71,17.32),16.94**	**(19.77,20.31,21.67),20.83**
	FPSO	(20.20,20.73,22.48),21.42	(22.61,23.36,25.37),24.13
	FGA	(18.72,19.25,20.29),19.63	(22.54,23.40,25.57),24.23
	FRA	35.50,36.83,40.38),38.20	(37.83,39.41,43.37),40.92
Epigenomics	ADFGA-PSO	**(345.83,363.89,411.18),382.04**	**(376.84,386.03,410.86),395.60**
	FPSO	(391.26,398.33,418.60),406.20	(407.82,416.06,435.23),423.30
	FGA	(363.74,379.11,428.36),398.48	(388.50,397.94,425.33),408.59
	FRA	(496.64,505.48,523.12),511.98*	(564.06,568.60,579.76),572.85
LIGO	ADFGA-PSO	**(107.82,111.24,118.70),114.03**	**(113.38,116.20,122.40),118.52**
	FPSO	(120.41,123.28,131.47),126.46	(124.62,127.64,134.89),130.39
	FGA	(114.56,116.30,120.01),117.68	(123.37,126.19,132.77),128.68
	FRA	(152.70,154.54,160.22),156.76*	(164.98,168.11,175.94),171.09
Montage	ADFGA-PSO	**(14.04,14.74,16.51),15.41**	**(14.81,15.63,17.55),16.35**
	FPSO	(15.19,15.97,17.73),16.63	(17.79,18.70,21.08),19.61
	FGA	(15.18,15.98,17.69),16.62	(16.21,17.10,19.18),17.89
	FRA	(27.20,28.72,32.28),30.06*	(32.01,33.87,38.39),35.58
SIPHT	ADFGA-PSO	**(121.53,125.81,131.61),127.79**	**(127.25,130.62,137.58),133.20**
	FPSO	(126.75,129.94,138.62),133.29	(131.67,135.11,143.00),138.08
	FGA	(116.28,121.37,138.58),128.18	(130.06,131.97,139.32),134.92
	FRA	(158.74,164.08,172.71),167.14	(169.13,172.17,178.76),174.63

表 4 基于不同算法的调度策略对于中型工作流的模糊执行代价

Table 4 Fuzzy execution cost of different algorithms for medium workflows

Workflows	Algorithms	Optimal fuzzy execution cost and its fitness	Meanfuzzy execution cost and its fitness
CyberShake	ADFGA-PSO	**(44.17,45.38,48.34),46.51**	**(47.44,48.76,52.23),50.09**
	FPSO	(45.97,47.64,51.34),49.03	(49.40,51.19,55.32),52.75
	FGA	(50.24,51.94,56.05),53.50	(52.92,54.64,58.70),56.18
	FRA	(93.39,96.68,105.27),99.97 *	(98.46,101.74,110.10),104.94
Epigenomics	ADFGA-PSO	**(3386.73,3466.08,3589.69),3509.62**	**(3504.47,3596.05,3774.37),3661.50**
	FPSO	(3626.87,3666.34,3815.99),3726.23	(3977.96,4007.34,4085.84),4037.54
	FGA	(3554.42,3610.30,3761.11),3668.39	(3898.24,3962.29,4105.97),4016.23
	FRA	(5627.38,5650.88,5724.76),5679.87 *	(7282.51,7315.38,7398.61),7347.17
LIGO	ADFGA-PSO	**(211.42,212.46,218.45),214.94**	**(217.55,219.01,224.18),221.06**
	FPSO	(241.59,243.66,263.88),252.26	(255.12,257.34,263.90),259.90
	FGA	(217.03,218.52,227.99),222.46	(242.46,243.76,250.33),246.45
	FRA	(370.15,371.98,376.40),373.66 *	(407.01,410.40,416.49),412.60
Montage	ADFGA-PSO	**(33.12,34.10,36.36),34.95**	**(35.63,36.56,38.82),37.42**
	FPSO	(49.53,51.48,56.65),53.47	(56.11,58.42,63.99),60.53
	FGA	(35.43,36.28,38.23),37.01	(44.48,45.86,49.01),47.05
	FRA	(76.85,80.90,90.94),84.72 *	(80.16,84.64,95.54),88.78
SIPHT	ADFGA-PSO	**(179.68,179.88,183.85),181.61**	**(196.88,198.53,207.24),202.11**
	FPSO	(202.81,211.04,222.32),214.91	(217.00,221.85,231.43),225.37
	FGA	(183.41,185.18,187.36),185.91	(197.15,199.97,208.19),203.17
	FRA	(286.66,289.82,298.31),293.09	(310.68,315.97,324.83),319.14

微型工作流 10 次重复实验的调度结果如表 2 所列。对于微型工作流而言,除 SIPHT 的最优代价外,ADFGA-PSO 获得所有的最优解;而 FPSO 和 FGA 性能次之;FRA 效果最差,同时存在不可行解的产生。这是由于 ADFGA-PSO 改进了传统 PSO 的编码方式,并引入 GA 的交叉算子和变异算子,从而避免了粒子陷入局部最优,得到更优的模糊调度策略。另外,工作流模糊调度问题的解空间大小一般是指数级的,而 FRA 采用的随机策略搜索效率较低,在有限的种群规模和搜索次数下,难以搜索到高质量解,甚至无法得到可行解。

小型工作流 10 次重复实验的调度结果如表 3 所列。ADFGA-PSO 在所有工作流上均得到最优代价和平均代价的最优解。此外,ADFGA-PSO 的最优代价优于 FPSO 的比例最高达 26.4%,最高优于 FGA 15.9%,最高优于 FRA 125.5%,均为工作流 CyberShake 的调度结果;而其平均代价分别优于 FPSO,FGA 和 FRA,最高达 19.9%(Montage),16.3%(CyberShake)和 117.6%(Montage)。由此看出,ADFGA-PSO 对工作流 CyberShake 和 Montage 的调度性能优于其他对比算法。值得注意的是,这两种类型的工作流具有类似的计算密集型结构,包含大量数据聚合型和数据分区型任务[26],也就是说,ADFGA-PSO 对计算密集型工作流的调度具有更佳的性能。

中型工作流 10 次重复实验的调度结果如表 4 所列。与小型工作流相同,ADFGA-PSO 对于所有工作流均得到最优代价和平均代价的最优解,同时在不同程度上优于其他算法。而 FRA 随着任务规模的增大,其性能越来越差,几乎无法得到可行的调度策略。由此看出,ADFGA-PSO 在任务规模较大的工作流上具有更好的调度性能和鲁棒性。

ADFGA-PSO 引入 GA 的变异算子和交叉算子,使得粒子能够有效地跳出局部最优;根据粒子质量对惯性因子 w 进行自适应调整的策略,使得算法具备较强的搜索性能;对于变异位数 k 的自适应调整,设计关于 w 的正反馈机制,促进了惯性因子 w 对算法搜索性能的提高。综上所述,相较于对比算法,基于 ADFGA-PSO 的调度策略能够得到更好的模糊执行代价,表现出更佳的性能。

结束语 针对边缘环境下科学工作流的不确定性调度,基于模糊理论,本文将任务计算时间和数据传输时间表示为三角模糊数,提出了一种基于 ADFGA-PSO 的代价驱动调度策略,目的是最小化由任务计算和数据传输引起的工作流执行代价,同时满足截止日期约束。本文对 5 种不同规模的科学工作流进行调度实验,结果表明,与其他对比算法相比,ADFGA-PSO 可以获得几乎所有类型科学工作流的最优模糊执行代价。

未来工作中,为了更加直观地反映实际情况,将对服务器的执行性能波动和带宽波动进行直接建模,以充实工作流的不确定性调度模型。

参 考 文 献

[1] NASCIMENTO A,OLIMPIO V,SILVA V,et al. A Reinforcement Learning Scheduling Strategy for Parallel Cloud-Based Workflows[C]// 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). 2019:817-824.

[2] HAN P,DU C,CHEN J,et al. Cost and Makespan Scheduling of Workflows in Clouds Using List Multiobjective Optimization Technique[J/OL]. Journal of Systems Architecture. https://www. sciencedirect. com/science/article/abs/pii/S1383762120301296.

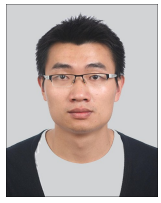
[3] LI Y,LUO J,JIN J,et al. An Effective Model for Edge-Side Collaborative Storage in Data-Intensive Edge Computing[C]// 2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design (CSCWD). 2018:92-97.

[4] KO H,LEE J,PACK S. Spatial and Temporal Computation Offloading Decision Algorithm in Edge Cloud-Enabled Heterogeneous Networks[J]. IEEE Access,2018,6:18920-18932.

[5] ZHANG L X,ZHOU L Q,WEN H,et al. Energy Efficient Scheduling Algorithm of Workflows with Cost Constraint in Heterogeneous Cloud Computing Systems [J]. Computer Science,2020,47(8):112-118.

[6] MA Y Y, ZHENG W B, MA Y, et al. Multi-workflow Offloading Method Based on Deep Reinforcement Learning and Probabilistic Performance-aware in Edge Computing Environment [J]. Computer Science, 2021, 48(1): 40-48.
[7] LI J, ZHANG Y P, PANG L, et al. Joint Resource Allocation and Task Scheduling in Mobile Edge Computing[J]. Journal of Chongqing University of Technology (Natural Science), 2020, 34(11): 156-163.
[8] SUN L, LIN L, GEN M, et al. A Hybrid Cooperative Coevolution Algorithm for Fuzzy Flexible Job Shop Scheduling [J]. IEEE Transactions on Fuzzy Systems, 2019, 27(5): 1008-1022.
[9] GAO D, WANG G, PEDRYCZ W. Solving Fuzzy Job-shop Scheduling Problem Using DE Algorithm Improved by a Selection Mechanism [J]. IEEE Transactions on Fuzzy Systems, 2020, 28(12): 3265-3275.
[10] SAHNI J, VIDYARTHI D P. A Cost-Effective Deadline-Constrained Dynamic Scheduling Algorithm for Scientific Workflows in a Cloud Environment[J]. IEEE Transactions on Cloud Computing, 2018, 6(1): 2-18.
[11] SHI W, ZHANG X. Edge Computing: State-of-the-Art and Future Directions[J]. Journal of Computer Research & Development, 2019, 56(1): 69-89.
[12] XIE Y, ZHU Y, WANG Y, et al. A novel directional and non-local-convergent particle swarm optimization based workflow scheduling in cloud-edge environment [J]. Future Generation Computer Systems, 2019, 97(AUG.): 361-378.
[13] HUANG B, LI Z, TANG P, et al. Security modeling and efficient computation offloading for service workflow in mobile edge computing [J]. Future Generation Computer Systems, 2019, 97(AUG.): 755-774.
[14] PENG Q, JIANG H, CHEN M, et al. Reliability-aware and Deadline-constrained workflow scheduling in Mobile Edge Computing[C]// 2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC). 2019: 236-241.
[15] LIN K, LIN B, CHEN X, et al. A Time-Driven Workflow Scheduling Strategy for Reasoning Tasks of Autonomous Driving in Edge Environment[C]// 2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom). 2019: 124-131.
[16] LEI D. Fuzzy job shop scheduling problem with availability constraints[J]. Computers Industrial Engineering, 2010, 58(4): 610-617.
[17] FORTEMPS P. Jobshop scheduling with imprecise durations: a fuzzy approach[J]. IEEE Transactions on Fuzzy Systems, 1997, 5(4): 557-569.
[18] MATTES A, TAVERA F, OPHEY A, et al. Parallel and serial task processing in the PRP paradigm: a drift-diffusion model approach[J]. Psychological Research, 2021(85): 1529-1552.
[19] ZADEH L A. Fuzzy Sets[J]. Information Control, 1965, 8(3): 338-353.
[20] PALACIOS J J, GONZÁLEZ-RODRÍGUEZ I, VELA C R, et al. Coevolutionary makespan optimisation through different ranking methods for the fuzzy flexible job shop[J]. Fuzzy Sets and Systems, 2015, 278: 81-97.
[21] LEE E S, LI R J. Comparison of fuzzy numbers based on the probability measure of fuzzy events[J]. Computers & Mathematics with Applications, 1988, 15(10): 887-896.
[22] PALACIOS J J, GONZÁLEZ M A, VELA C R, et al. Genetic tabu search for the fuzzy flexible job shop problem[J]. Computers & Operations Research, 2015, 54: 74-89.
[23] SAKAWA M, KUBOTA R. Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms [J]. European Journal of Operational Research, 2000, 120(2): 393-407.
[24] KENNEDY J, EBERHART R. Particle swarm optimization [C]// ICNN95-International Conference on Neural Networks. 1995.
[25] 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.
[26] LI H, YANG D, SU W, et al. An Overall Distribution Particle Swarm Optimization MPPT Algorithm for Photovoltaic System Under Partial Shading [J]. IEEE Transactions on Industrial Electronics, 2019, 66(1): 265-275.
[27] LI X, GAO L. An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem[J]. International Journal of Production Economics, 2016, 174(Apr.): 93-110.
[28] SHI Y. A Modified Particle Swarm Optimizer[C]// Proceedings of IEEE ICEC Conference. 1998.
[29] BHARATHI S, CHERVENAK A, DEELMAN E, et al. Characterization of scientific workflows[C]// Workflows in Support of Large-Scale Science. 2008.
[30] TOPCUOGLU H, HARIRI S, WU M Y. Performance effective and low-complexity task scheduling for heterogeneous computing[J]. IEEE Transactions on Parallel and Distributed Systems, 2002, 13(3): 260-274.
[31] CUI L, ZHANG J, YUE L, et al. A Genetic Algorithm Based Data Replica Placement Strategy for Scientific Applications in Clouds[J]. IEEE Transactions on Services Computing, 2018, 11(4): 727-739.
[32] ZHOU B, XIE S S, WANG F, et al. Multi-step predictive compensated intelligent control for aero-engine wireless networked system with random scheduling [J]. Journal of the Franklin Institute-Engineering and Applied Mathematics, 2020, 357(10): 6154-6174.

LIN Chao-wei, born in 1998, postgraduate. His main research interests include workflow scheduling, computational intelligence and its applications, and fuzzy theory.



LIN Bing, born in 1986, Ph.D, lecturer, postgraduate supervisor, is a member of China Computer Federation. His main research interests include parallel and distributed computing, computational intelligence and its applications, and fuzzy theory.

(责任编辑:李亚辉)