云环境下基于 DO-GAPSO 的任务调度算法

孙 敏 陈中雄 卢伟荣
(山西大学计算机与信息技术学院 太原 030006)

摘 要 为了找到合理的云计算任务调度方案,仅从单一方面来优化调度策略已不能满足用户需求,但从多个方面优化调度策略又面临着权重分配问题。针对上述问题,从任务完成时间、任务完成成本、服务质量 3 个方面考虑,提出一种基于遗传与粒子群算法相融合的动态目标任务调度算法,在算法的适应度评价函数建模中引入线性权重动态分配策略。通过 CloudSim 平台进行云环境仿真实验,并将此算法与经典的双适应遗传算法(DFGA)、离散粒子群优化算法(DPSO)进行比较。实验结果表明,在相同的设置条件下,该算法在执行效率、寻优能力等方面优于其他两个算法,是一种云计算环境下有效的任务调度算法。

关键词 云计算,任务调度,惯性权重,粒子群优化,遗传算法
中图法分类号 TP393 **文献标识码** A

Task Scheduling Algorithm Based on DO-GAPSO under Cloud Environment

SUN Min CHEN Zhong-xiong LU Wei-rong
(School of Computer & Information Technology, Shanxi University, Taiyuan 030006, China)

Abstract In order to find reasonable cloud computing task scheduling scheme, the demand of users can not be satisfied by optimizing scheduling strategy from a single aspect, and there are some weight assignment problems in several aspects to optimize scheduling policy. Focusing on the problems, considering the completion time, cost and service quality, an algorithm of a dynamic target based on particle swarm and genetic algorithm(DO-GAPSO) was proposed, a dynamic linear weighting allocation policy was introduced in the fitness of function modeling. Cloud environment simulation experiment was conducted in the CloudSim platform. Under the same condition, discrete particle swarm optimization(DPSO), double fitness genetic algorithm(DFGA) were compared with the proposed algorithm. The experimental results show that the proposed algorithm is better than the other two algorithms in execution efficiency and optimization ability. It is a kind of effective task scheduling algorithm in cloud computing environment.

Keywords Cloud computing, Task scheduling, Inertia weight, Particle swarm optimization, Genetic algorithm

1 引言

云计算是并行计算、分布式计算和网格计算的发展,它以公开的标准和服务为基础,以互联网为中心,为用户提供快速、按需和可伸缩的数据存储和网络计算服务。在云计算环境中,用户提交任务请求,云计算调度中心根据用户请求为其分配资源。由于任务种类多、规模大,如何对云计算任务进行合理调度成为目前云计算系统中亟需解决的问题。

目前,已有研究证明云计算任务调度属于一个 NP 完全问题,由于任务调度策略直接影响到用户任务的执行效率以及云环境下资源的使用效率,众多学者提出将遗传算法、蚁群算法、粒子群算法等智能算法引入到云任务调度中,收到了一定的成效。其中,邬开俊等[3]提出云环境下基于 DPSO 的任务调度算法,该算法在优化过程中动态改变粒子更新速度的权重,一定程度上取得了良好的效果。Khalili 等[10]重点研究了权重策略对粒子群算法的影响,一定程度上减少了完成时间。Mu[6]考虑了任务的完成时间和任务的平均完成时间,并在选择操作中进行了适应度度量。上述的各种启发式算法或目标单一,或收敛早熟,或考虑到多目标但存在优先权的分配问题,严重影响了执行效率。

针对上述问题,文中设计了遗传法算与粒子群算法相融合的动态目标任务调度算法(DO-GAPSO),算法的关键是通过考虑完成时间、成本、服务质量,将线性权重动态分配策略引入算法中作为适应度评价函数,有效地解决了 DFGA 和 DPSO 算法中的不足。

2 云计算任务调度问题的描述

云计算主要利用了虚拟化技术,虚拟化的引入使得云计算与传统分布式的资源调度模式不同。云计算的资源调度模型如图 1 所示。

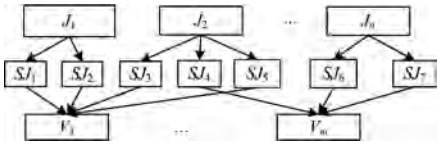

图 1 资源调度模型

本文受山西省自然科学基金项目(201701D121054)资助。

孙 敏(1965—),女,副教授,硕士生导师,CCF 会员,主要研究方向为云计算、Web 智能、协同编辑等,E-mail:476957266@qq.com;陈中雄(1992—),男,硕士生,主要研究方向为云计算、人工智能;卢伟荣(1993—),男,硕士生,主要研究方向为 Web 智能、云计算。

首先，将每个任务划分为若干个相互独立的子任务，系统受到请求时，会分配一定的虚拟资源来应对，每个子任务对应一个虚拟资源节点。

建立如下云计算任务调度模型：将任务划分成 n 个独立的子任务并将其分配到 m 个资源上，其中 $m<n$，任务 $J_j=\{id,pesNumber,length,fileSize,outputSize\}$，任务序列 $J=\{J_1,J_2,\cdots,J_j\}(j=1,2,\cdots,n)$ 代表 n 个任务，虚拟机资源 $V_i=\{vmid,mips,size,ram,bw,pesNumber\}$，虚拟机资源序列 $V=\{V_1,V_2,\cdots,V_i\}(i=1,2,\cdots,m)$。每个 J_j 只能被分配到一个 V_i 上，由此得到 T 和 V 的对应关系矩阵 W 如下：

$$W=\begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{m1} & \cdots & w_{mm} \end{bmatrix} \tag{1}$$

其中，W_{ij} 代表 J_j 和 V_i 的对应关系，$W_{ij}\in\{0,1\}$，$\sum_{i=1}^{m} w_{ij}=1$ $(i\in\{1,2,\cdots,m\},j\in\{1,2,\cdots,n\})$，$W_{ij}=1$ 表示 J_j 在 V_i 上执行，$W_{ij}=0$ 表示 J_j 不在 V_i 上执行。J_j 在 V_i 上完成的期望时间为 ET_{ij}，与矩阵 W 相对应的矩阵 ET 如下：

$$ET=\begin{bmatrix} ET_{11} & \cdots & ET_{1n} \\ \vdots & \ddots & \vdots \\ ET_{m1} & \cdots & ET_{mm} \end{bmatrix} \tag{2}$$

3　采用 DO-GAPSO 进行“云”中任务调度

遗传算法（Genetic Algorithm）和粒子群算法（Particle Swarm Algorithm）均是云任务调度中的经典算法，为了得到总任务的完成时间、任务执行成本和服务质量较好的调度结果，本文在两种融合算法的基础上进行了改进，在变异操作中引入粒子群算法，将线性权重分配策略引入到函数建模中来动态调整适应度，即遗传粒子群算法的动态目标任务调度算法（DO-GAPSO）。

3.1　染色体编码与解码

染色体编码有很多种方式，可以采用二进制编码，也可采用实数编码。本文采用资源-任务的实数编码方式，若一个种群大小为 S，子任务总数为 N，虚拟机资源 V 为 M 个，则初始化种群如下：随机生成 S 个染色体，染色体长度为 N，每个染色体上的基因随机取，取值范围为[1　M]。例如任务总数为 5，虚拟机资源为 3，则染色体的长度为 5，每个基因的取值为[1　3]，则产生如下的一条染色体：

{2,1,3,1,2}

该染色体代表任务 J_1 在虚拟资源 V_2 上执行，任务 J_2 在虚拟资源 V_1 上执行，…，任务 J_5 在虚拟资源 V_2 上执行。之后就是对这个染色体进行解码，得到虚拟机 V 上的任务 J 的分布情况，如下所示：

$V_1:\{2,4\}$，$V_2:\{1,5\}$，$V_3:\{3\}$

根据解码后的序列和 ET（Expected Time）矩阵（ET_{ij} 表示第 j 个任务 J_j 在第 i 个资源 V_i 上执行完成所用的时间）得到完成所有任务的总时间函数为：

$$Time=\max_{1\leqslant i\leqslant m}\sum_{j=1}^{k} ET_{ij} \tag{3}$$

k 为分配到 V_i 上的子任务数，m 为虚拟资源数，$Time$ 为任务的完成时间。

$$Cost=\sum_{i=1}^{m}\sum_{j=1}^{k} ET_{ij} * cost_i \tag{4}$$

其中，$cost_i$ 表示单位时间虚拟资源 V_i 的花费，$Cost$ 代表完成所有任务的花费。

$$VS(j,i)=r_i+c_i \tag{5}$$

$$Serve=\sum_{j=1}^{n} VS(j,i) \tag{6}$$

其中，r_i 表示虚拟资源 V_i 的计算能力；c_i 表示虚拟资源 V_i 的耗能；S_i 表示虚拟资源 V_i 的服务能力；$VS(j,i)$ 表示任务 J_j 分配的虚拟资源 V_i 的服务能力；$Serve$ 表示服务质量。

3.2　函数建模

遗传算法是通过一定的适应度函数来进行下一代的选择进化，从而来寻找问题的最优解，因此适应度的选取相当重要。在任务调度中的一个重要目标是：虽然总任务的完成时间短，但执行任务的执行成本和服务质量也不能忽略。在这种情况下，三者的权重分配尤为重要，适当的权重分配不仅有利于提高算法的收敛速度，而且也可找到实际最优解。为此，在进行适应函数建模的过程中进行以下操作。

第 1 步　利用式（3）、式（4）和式（6）计算得到结果，对结果进行标准化处理，使之在一个可控的范围内。其实现代码如下：

```
//对计算得到的成本 cost 进行标准化
BigDecimal bg1=new BigDecimal(time/cost);
double x1=bg1. setScale(1,BigDecimal. ROUND_UP). doubleValue
();
double Cost=x1 * cost;
//对计算得到的服务质量 serve 进行标准化
BigDecimal bg2=new BigDecimal(time/serve);
double x2=bg2. setScale(1,BigDecimal. ROUND_UP). doubleValue
();
double Serve=x2 * serve;
```

第 2 步　对三者进行权重分配：

$$w_1=0.5*\frac{T_{\max}-T_{\text{cur}}}{T_{\max}}+0.4 \tag{7}$$

$$w_2=w_3=0.5*(1-w_1) \tag{8}$$

其中，$T_{\max}$ 代表种群的迭代次数，T_{cur} 代表个体当前的迭代次数，$w_1+w_2+w_3=1$。通过对三者的权重分配，得到的权重变化范围如下：

$w_1\in[0.4,0.9]$，$w_2\in[0.05,0.3]$，$w_3\in[0.05,0.3]$

第 3 步　函数建模。算法在寻优过程中的大致路线为：在刚开始的迭代中，寻优以完成时间为主要目标，随着种群的不断更新，完成成本和服务质量两个目标被逐渐纳入考虑目标当中。因此，开始时总任务的完成时间所占的比重较最大，其余两者比重较小，随着迭代次数的增加，总任务的完成时间所占比重逐渐减小，从而引起其余两者权重变化，最终三者比重相当。

利用式（3）、式（4）、式（6）、式（7）和式（8）构建适应度函数，其结果如下：

$$F(X)=w_1*Time+w_2*Cost+w_3*Serve \tag{9}$$

函数分析：首先，在此函数中，个体的适应度评价方式随着迭代次数的增加不断变化，最终趋于一个稳定的状态。其次，在算法后续的选择、交叉以及变异融合操作中均要利用式（9）进行个体评价。

3.3　算法操作

3.3.1　选择操作

选择操作是遗传算法对个体适应性的评价方式，优良基因直接通过这种方式传递给下一代。个体进入下一代种群的

概率与适应度函数值直接相关，个体的选择概率计算式如下：

$$P(i)=1-\frac{f(i)}{\sum_{i=1}^{n}f(i)} \tag{10}$$

3.3.2 交叉操作

交叉是遗传算法中最主要的搜索算子，本文采用自适应算法来计算交叉概率[6]。

3.3.3 变异操作

本文在该阶段引入粒子群算法，并利用粒子群体的当前最优解和历史最优解重构变异算子。如此，变异操作分为两步：

第 1 步 通过自适应算法计算出变异概率[6]；

第 2 步 通过粒子群算法构造变异操作：

$$V_{k+1}=wV_k+c_1(P_k-X_k)+c_2(g_k-X_k) \tag{11}$$

$$X_{k+1}=X_k+V_{k+1} \tag{12}$$

3.4 算法实现

```
//种群初始化
Group groups=new Group(100);
while (groups.getTime()<MAXTIME){
    //选择操作
    groups.chooseNewEnt();
    //交叉操作
    groups.intersect();
    //变异操作
    groups.variation(groups.group);
    //种群内部个体更新迭代次数
    groups.Tij();
    //种群的迭代次数
    groups.time++;
    //统计种群迭代中最好的个体
    groups.setGroup();
    //呈现结果
    groups.display();
    }
```

4 仿真实验

4.1 仿真环境及参数配置

为了验证 DO-GAPSO 算法的可行性和有效性，实验利用 Cloudsim 模拟平台进行云计算资源调度环境的模拟，并在相同的条件下对文中提出的 DFGA 算法和 DPSO 算法进行对比分析测试。在进行资源调度模拟时需要进行参数配置，为此，采用 Amazon Elastic Compute Cloud EC2 实例，实例配置详情如表 1 和表 2 所列。

表 1 虚拟资源配置

参数	种类	单位
CPU MIPS	{2500,2000,1000,500}	MIPS
RAM SIZE	{870,1740,1740,613}	MB
Storage Size	1	GB
Bandwidth	1	GB/s
VM Price	{0.20,0.148,0.047,0.02}	$
Core in host	1	PIECE

表 2 物理资源配置

参数	种类	单位
CPU MIPS	{2500,3000,3500}	MIPS
RAM SIZE	4096	MB
Storage Size	1	GB
Bandwidth	1	GB/s
Core in host	2	PIECE

4.2 性能分析

4.2.1 收敛速度

在 DO-GAPSO，DPSO，DFGA 3 种算法中，首先要确定其收敛程度，在迭代次数达到一定程度时，种群内的各个属性达到一个稳定值。为此进行如下实验，参数配置如表 3 所列。

表 3 实验参数

参数指标	类型	注解
种群规模	100	固定
任务数量	610	固定
任务长度	[2500 10000]	随机
物理资源	150	固定
虚拟资源	250	固定
迭代次数	{5,20,30,40,50,70}	

根据以上参数进行仿真实验，设置 6 组迭代次数，每组进行 12 次实验，去除实验中最好和最坏的情况，剩下的求平均值，结果如图 2 所示。

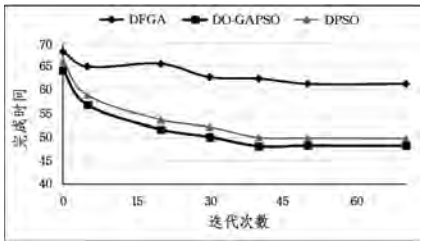

图 2 收敛性能

图 2 示出 DO-GAPSO 和 DPSO 算法的收敛速度大体一致，但明显大于 DFGA 算法。其次，DO-GAPSO 与 DPSO 算法在寻找任务调度最优方案方面明显比 DFGA 算法要好，但 DO-GAPSO 和 DPSO 算法的寻优能力并没有明显的差异。

4.2.2 寻优能力

本文通过任务完成时间、任务完成成本和服务质量 3 个目标来衡量云任务调度方案的优劣。实验参数配置如表 4 所列。

表 4 参数配置

	DO-GAPSO	DPSO	DFGA
种群规模		100	
物理资源		150	
虚拟资源		250	
迭代次数	40	40	50
任务数量		[500 2000]随机选取	

随机设置任务数量，设置 5 组实验，每组 12 次，去掉最好和最坏的情况，求其平均值，结果如图 3 所示。

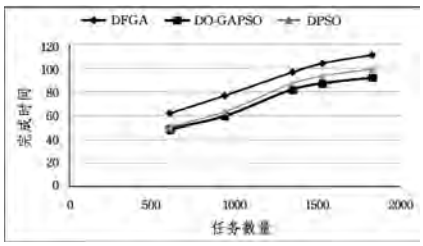

图 3 完成时间

图 3 表明随着任务数量的增加，DO-GAPSO 和 DPSO 算法的完成时间要明显优于 DFGA；其次，随着任务数量的增加，DO-GAPSO 和 DPSO 算法在完成时间方面出现差异，即 DO-GAPSO 要优于 DPSO 算法。

图 4 和图 5 分别对应图 3 中任务数量为 1830 时的服务质量和完成成本；图 6 和图 7 分别对应图 3 中任务数量为

1350 时的服务质量和完成成本。通过图 4－图 7 可以看出，随着云任务的增加，DO-GAPSO 对任务调度方案的寻优能力明显强于 DPSO 和 DFGA。

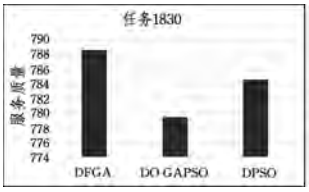

图 4　任务数量为 1830 时的服务质量

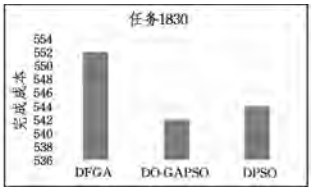

图 5　任务数量为 1830 时的完成成本

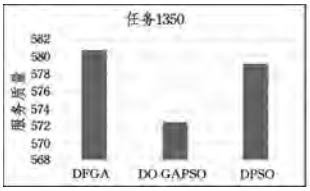

图 6　任务数量为 1350 时的服务质量

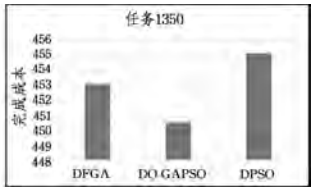

图 7　任务数量为 1350 时的完成成本

以上是任务数量较大时 3 个算法的性能，接下来验证任务数量较少时 3 个算法的性能，参数配置如表 5 所列。

表 5　参数配置

	DO-GAPSO	DPSO	DFGA
种群规模		100	
物理资源		5	
虚拟资源		4	
迭代次数	40	40	50
任务数量		[0 100]随机选取	

任务数量随机选取，设置 3 组实验，每组实验进行 22 次，去掉最好和最坏的情况，其余求其平均值，结果如图 8－图 10 所示。

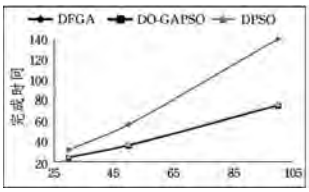

图 8　完成时间

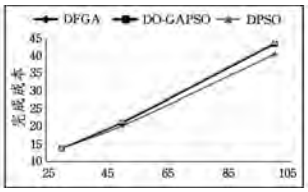

图 9　完成成本

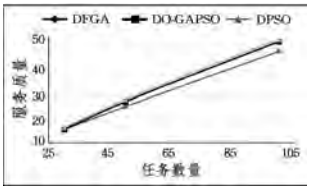

图 10　服务质量

从图 8－图 10 可以看出，DPSO 和 DO-GAPSO 算法的寻优性没有明显差异，但都明显强于 DFGA 算法。

综上所述，在大规模的云任务调度中，在任务完成时间、完成成本、服务质量 3 个方面，DO-GAPSO 算法的寻优性能的整体效果要优于其余两者。

结束语　针对云计算任务调度问题，本文提出一种基于遗传算法与粒子群算法相融合的动态目标任务调度算法。在该算法中，在适应度函数建模中引入线性权重动态分配策略，以此作为选择操作中的个体直接传递给下一代的标准。在交叉和变异操作中引入自适应算法，来决定种群的交叉率和变异率；在变异操作中引入粒子群算法，利用粒子群间的反馈信息影响变异因子，以避免盲目变异产生新个体。通过在 Cloudsim 平台上进行仿真对比实验可验证，DO-GAPSO 算法在云任务调度上的总体性能优于 DGA 和 DPSO 算法。

参 考 文 献

[1] AHMED M，CHOWDHURY A S M R，AHMEDAN M，et al. Advanced survey on cloud computing and state-of-the-art research issues[J]. International Journal of Computer Science Issues，2012，9(1)：201-207.

[2] 封良良，张陶，贾振红，等. 云计算环境下基于改进粒子群的任务调度算法[J]. 计算机工程，2013，39(5)：183-186.

[3] 邬开俊，鲁怀伟. 云环境下基于 DPSO 的任务调度算法[J]. 计算机工程，2014，40(1)：59-62.

[4] 盛小东，李强，刘昭昭. 云环境下基于模板遗传算法的任务调度方法[J]. 计算机应用，2016，36(3)：633-636.

[5] ZHANG D，GUAN Z，LIU X. An adaptive particle swarm optimization algorithm and simulation[C]// IEEE International Conference on Automation & Logistics. 2007：2399-2402.

[6] WU M. Research on Improvement of Task Scheduling Algorithm in Cloud Computing[J]. Applied Mathematics & Information Sciences，2015，9(1)：507-516.

[7] SAVITHA P，REDDY J G. A Review Work On Task Scheduling In Cloud Conputing Using GeneticAlgorithm[J]. International Journal of Scientific Technology Research，2013，2(8)：241-245.

[8] MANDAL T，ACHARYYA S. Optimal Task Scheduling in Cloud Computing Environment：Meta Heuristic Approaches[J]. International Conference on Electrical Information and Communication Technology，2016，1(28)：24-28.

[9] AGARWAL A，JAIN S. Efficient Optimal Algorithm of Task Scheduling in Cloud Computing Environment[J]. International Journal of Computer Trends & Technology，2014，9(7)：344-349.

[10] KHALILI A，BABAMIR S M. Makespan Improvement of PSO-based Dynamic Schedulingin Cloud Environment[J]. Electrical Engineering，2015，7(2)：613-618.

[11] RANI A，GARG K. A Review on Task Scheduling Algorithmin Cloud Computing Environment[J]. International Journal of Scientific & Engineering Research，2016，5(4)：9724-9729.

[12] RAMEZANI F，LU J，TAHERI J，et al. Evolutionary algorithm-based multi-objective task scheduling optimization model in cloud environments[J]. World Wide Web-internet & Web Information Systems，2015，18(6)：1737-1757.

[13] JENA R K. Multi Objective Task Scheduling in Cloud Environment Using Nested PSO Framework[J]. Procedia Computer Science，2015，7(57)：1219-1227.

[14] LAKSHMI R D，SRINIVASU N. A dynamic approach to task scheduling in cloud computing using genetic algorithm[J]. Journal of Theoretical & Applied Information Technology，2016，3(85)：124-135.

[15] KAUR S，VERMA A. An Efficient Approach to Genetic Algorithm for Task Scheduling in Cloud Computing Environment[J]. International Journal of Information Technology & Computer Science，2012，4(10)：159-190.

[16] AKILANDESWARI P，SRIMATHI H. Survey and analysis on Task scheduling in Cloud environment[J]. Indian Journal of Science & Technology，2016，9(37)：974-5645.