计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190400047

云环境下基于 HEDSM 的工作流调度策略

孙 敏 陈中雄 叶侨楠
山西大学计算机与信息技术学院 太原 030006

摘 要 针对传统算法处理云环境中任务调度时出现的寻优性能差以及寻优方案不能满足用户多样性需求的问题,在考虑任务完成时间、完成成本以及资源闲置率3个优化目标的情况下,文中通过模拟启发式算法调度过程(初始化—适应度评估—任务调度—选择),建构了一种层次评估和动态选择模型(Hierarchy Evaluation and Dynamic Selection Model,HEDSM)。在初始化阶段,利用传统的表调度算法(Heterogeneous Earliest Finish Time,HEFT)对工作流任务模型进行预处理,保证任务具有一定的优先级。在适应度评估阶段,从云用户和云服务提供商两个层次构建不同的方案评估模型来同时满足两方面的需求。在任务调度阶段,设置两步调度:1)设置策略集,对任务进行预调度,保证生成的预调度方案继承各个策略的调度优势;2)设置任务迁移策略,对预调度方案进行处理,以此提升算法的寻优性能。在选择阶段,根据不同的评估模型在方案集中选择合适的调度方案。实验利用 WorkflowSim 仿真平台,采用科学工作流实例进行实验,将传统的 Min-Min,Max-Min,FCFS 调度策略以及目前存在的 IMax-Min 和 LWRound_Robin 调度策略作为对比算法,从用户多样性需求和策略改进比(Improve Ratio of Strategy,IROS)两个方面评估算法的调度性能。结果证明,所提算法在保证负载均衡的基础上,缩短了完成时间并降低了完成成本,更适用于复杂多变的云环境下的任务调度。

关键词:云计算;任务调度;工作流;任务迁移;多目标优化
中图法分类号 TP393

Workflow Scheduling Strategy Based on HEDSM Under Cloud Environment

SUN Min,CHEN Zhong-xiong and YE Qiao-nan
School of Computer & Information Technology,Shanxi University,Taiyuan 030006,China

Abstract The traditional algorithm has poor performance and its optimization solution cannot meet the diversity needs of users, when deals with the task scheduling in the cloud environment. Based on three optimization goals:task completion time,completion cost,and resource idle rate,this paper simulates the process of heuristic algorithm (the initialization,fitness assessment,task scheduling and selection stages) to construct a hierarchical evaluation and dynamic selection model(HEDSM). In the initialization phase,in order to ensure that tasks have a certain priority,the workflow task model is preprocessed using the traditional table scheduling algorithm (HEFT). In the fitness assessment phase,in order to meet the need of two aspects,the difficult solution evaluation modelsare constructed from two levels which are cloud users and cloud service providers. In the task scheduling phase, two-step scheduling is set. First,the policy set is setting,the task is pre-scheduled to ensure that the pre-scheduling scheme inherits the scheduling advantages of each strategy. Second,in order to enhance the performance of the algorithm,the task migration policy is setting to process the pre-scheduling plan. In the selection phase,the appropriate scheduling scheme is selected in the solution set according to the evaluation modle. The experiment uses WorkflowSim simulation platform and scientific workflow instance to make comparative analysis. Traditional Min-Min,Max-Min,FCFS scheduling strategies and existing IMax-Min and LWRound_Robin scheduling strategies are as comparison algorthms. The algorithms are evaluated from the diversity of user requirements and the IROS two aspects. The results show that the propsed algorithm improves the complete time and cost,therefore it is more suitable for the complex task scheduling in cloud environment.

Keywords Cloud computing,Task scheduling,Workflow,Task immigration,Multi-objective optimization

1 引言

随着并行计算、分布式计算和网格计算的发展,云计算作为一种新的计算模型应运而生,它以公开的标准和服务为基础,是一种高效、可伸缩且具有高可靠性的网络资源模型[1],通过互联网向用户提供快速、按需和可伸缩的数据存储和网

到稿日期:2019-04-08 返修日期:2019-07-21 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:国家自然科学基金项目(61872226);山西省自然科学基金计划资助项目(201701D121052)
This work was supported by the National Natural Science Foundation of China (61872226) and Natural Science Foundation of Shanxi Province, China (201701D121052).
通信作者:孙敏(476957266@qq.com)

络计算服务。在云环境中，用户使用服务器上部署的多种服务来完成提交的应用[2]，这些应用多以工作流的形式存在，包含若干个任务，每个任务通常对应多个服务以供选择。由于云环境中即付即用的原则，云计算必须充分考虑用户的需求，包括执行时间、费用、可靠性及安全性等。云计算需要对云资源进行合理地分配管理，使得用户提交的任务能够在较短时间内以较低成本完成调度，从而提高用户满意度，同时还要保证数据的安全可靠性。但随着计算服务需求的急剧增长，云环境中数据中心部署的主机资源规模急剧增长，一个数据中心的主机资源数量高达几万台，甚至几十万台[3]，云服务系统需要消耗大量的能量来运行这些主机。高能耗就意味着高成本，如何降低云数据中心资源的闲置率并保证整体资源负载均衡是云计算需要考虑的非常重要的问题。

目前，任务调度算法大致分为两类，即传统算法和启发式算法。传统算法有基于开源云计算框架 Hadoop 实现的 FIFO(First In First Out)、计算能力调度(Capacity scheduler)以及公平调度算法(Fair scheduler)等，亦有最大最小策略(Max-Min)、最小最小策略(Min-Min)以及轮询策略(Round Robin)等[4-5]。启发式算法包括粒子群、蚁群、人工神经网络等[6-7]。本文主要研究的是传统算法对云环境中的任务调度。Alebrahim 等[8]在传统表调度算法的基础上引入一个阈值决策机制，该机制对任务和资源的映射关系进行处理，与传统算法相比，该算法在缩短完成时间方面有显著的效果。Asmaa 等[9]提出了节点分配机制，实验结果表明，其在缩短完成时间方面也具有显著效果。Chitra 等[10]通过综合考虑计算机资源的性能、任务规模大小以及任务之间的依赖关系因素，建立了一种负载评估模型，相较于 Static Round Robin 和 Weighted Round Robin 策略，该模型在任务完成时间和系统资源利用率方面均有所提升。Kaur 等[11]利用任务结构、规模大小等属性对用户提交的任务进行预处理，建立了一种任务约束模型，相较于传统的 Max-Min 策略，提升了系统的资源利用率。Panda 等[12]提出用户在多云环境下有不同的 QoS 目标，而针对多云环境下不确定性的 QoS 目标，与 Min-Min，Max-Min 以及 Smoothing 算法相比，Min-Min 在用户满意度方面做得更好。Ali 等[13]提出了 Resource-Aware Min-Min 算法，该算法通过判定虚拟资源的执行情况，建立节点选择分配机制，相较于传统的 Min-Min 及 Max-Min，减少了等待时间，提高了资源利用率。Babu 等[14]提出一种公平调度策略，该策略试图有效缩短依赖任务的整体在虚拟资源上的处理时间，从而降低并优化任务的完成成本，实验表明该策略在缩短完成时间和减少完成成本方面是有效的。Gupta 等[15]提出了一种基于表调度的任务调度策略，该策略将任务划分为两个阶段进行调度，利用科学工作流实例进行验证分析，该算法在任务完成时间和资源利用率方面有良好的效果。Guo 等[16]提出了一种基于资源模糊聚类的工作流任务调度算法，该算法定义了一组描述在资源系统中处理单元综合性能的特性，采用模糊聚类方法对处理单元网络进行预处理，很大程度上降低了当前任务的执行成本和执行时间。Mahajan 等[17]提出了一种虚拟机负载均衡算法，通过监测虚拟机的分配状态，达到资源的有效利用，实验结果表明，该算法在负载均衡方面有良好的效果。Wang 等[18]通过将 Max-Min 算法融入遗传算法，建立了一种可靠性预测模型，相较于传统的遗传算法，在缩短完成时间和提升用户满意度方面有显著的效果。Choudhary 等[19]重新构建了引力搜索算法框架，在算法内部重新定义了个体更新方法，对调度的两个目标作了正规化处理，取得了一定效果。Rodriguez 等[20]提出一种基于时间截止的资源分配与调度模型，该将其用于粒子群算法，在保证完成时间的同时，提高了用户的满意度。Wang 等[21]为了提高资源利用率和经济效益，建立了一种云资源定价模型，该算法在满足用户需求的情况下，最大化其服务收益。

然而，上述调度算法存在如下问题：1)部分传统算法过分关注于负载问题而忽略了算法的寻优能力，它们通过建立负载均衡模型来实现资源的有效利用；2)各种算法有不同的调度优势，从而产生具有不同性能的方案，而目前只针对单一策略进行规则处理，产生的单一的解并不能满足复杂多变的云环境。

2　云环境下的任务调度模型

2.1　云环境下的任务调度体系结构

云环境下的任务调度体系结构如图 1 所示，该调度结构包括用户提交的任务、任务调度中心和计算机群。其中，任务调度中心是该体系结构的核心模块，包括计算任务生成、任务处理、调度方案评估、策略选择和任务分配五大功能，任务调度中心的重要目标是针对用户提交上来的任务，选择合适的策略，生成最优的调度方案。

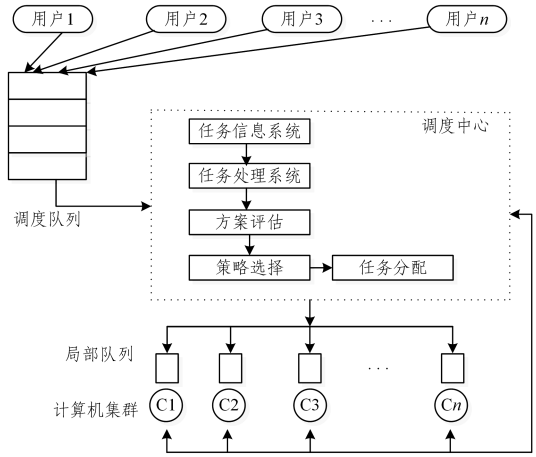

图 1　任务调度体系

Fig. 1　Task scheduling architecture

2.2　云任务模型

云环境下的任务调度问题中，并行的应用程序以多个任务图[22-23]进行表述，单个任务图可以描述出子任务与子任务之间的依赖关系。

具有 5 个子任务的任务图如定义 1 所示。

定义 1　任务图可以表示为一个二元组：$G=\langle T,E\rangle$，其中 $T=\{T_j\mid 1<j<n\}$ 表示任务的集合，$T_j=\{id,pesNumber,length,Parlist,childlist,filelist\}$ 表示一个单独的任务，其中，$pesNumber$ 描述计算机资源的属性，表示须将任务 T_j 分配到有如此配置的计算机资源上；$Parlist$ 代表 T_j 的父任务列表；$childlist$ 代表 T_j 的子任务列表；$filelist$ 代表执行 T_j

需要的输入文件；$E=\{e_{ij}\mid 1\leqslant i\leqslant n, 1\leqslant j\leqslant n\}$表示任务间具有约束关系的边的集合。图 2 给出了一个工作流图。

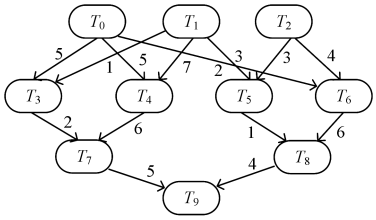

图 2 工作流图

Fig. 2 Workflow

2.3 云计算机系统模型

由于不同客户的计算需求和数据中心设备的不断更新，现有云计算平台的硬件基础设施通常由大量的服务器组成，这些处理机含有不同的计算性能，且一般同一个任务在计算性能高的处理机上运行比在计算性能较低的处理机上运行产生能耗开销更多，因此将进行合理的调度可以减少能耗开销。基于此，本文针对云计算环境进行研究，该云计算环境如定义 2 所示。

定义 2 具有 m 台服务器的云计算系统可定义为一个二元组 $C=\langle S,V\rangle$。其中，$S=\{s_i\mid 1\leqslant i\leqslant m\}$表示服务器集，单个服务器表示为 $s_i=\{id,name,hostlist\}$，服务器上的处理机列表为 $hostlist=\{h_j\mid 1\leqslant i\leqslant n\}$，其中 $h_j=\{hostId,ram,bw,storage,pelist\}$表示列表中的一个处理机，作为一个单独的处理机，$ram$ 代表内存，bw 代表带宽，$storage$ 代表外存，$pelist$ 代表 $mips$ 列表；虚拟机资源集合为 $V=\{V_i\mid 1\leqslant i\leqslant n\}$，单个虚拟机资源表示为 $V_i=\{vmidmips, size, ram, bw, pesNumber\}$，其中 $mips$ 代表虚拟机的执行速度，$size$ 代表外存，$pesNumber$ 代表虚拟机内核数量。

2.4 指标评估模型

定义 3 整个工作流的总执行时间($Makespan$)包括租用的 VM 的启动时间、执行时间和关闭时间，计算如下：

$$Makespan=VM_{boot\text{-}time}+\max_{i=1}^{m}(VM_{time[i]})+VM_{off\text{-}time} \tag{1}$$

虚拟机的执行时间包括分配到虚拟机上的任务队列的执行时间和相关任务的等待时间，计算如下：

$$VM_{time[i]}=\sum_{j=1}^{n}\left[ET_{t_j}^{v_i}+\max_{x\in t_j\cdot parlist} wait(x,t_j)\right]*w_{ij} \tag{2}$$

其中，$ET_{t_j}^{v_i}$代表任务 t_j 分配到虚拟机 v_i 上的执行时间，计算如下：

$$ET_{t_j}^{v_i}=\frac{t_{j.length}}{v_{i.mips}} \tag{3}$$

其中，$wait(x,t_j)$包含两部分，一部分是等待父任务列表中任务 x 完成的时间，另一部分是任务 x 到任务 t_j 的传输时间。W 表示任务与虚拟机的分配约束关系，计算如下：

$$W=\begin{cases}1, & t_j \text{ is on } v_i\\ 0, & \text{other}\end{cases} \tag{4}$$

定义 4 整个工作流的执行成本($Total_{\text{cost}}$)是以 VM 的执行时间为衡量标准，$cost(v_i)$为虚拟资源 v_i 的单位执行成本，其计算式如下：

$$Total_{\text{cost}}=\sum_{i}^{m}VM_{time[i]}*cost(v_i) \tag{5}$$

定义 5 资源闲置率($Resource_{\text{idle}}$)是通过 VM 的闲置时间和虚拟机的执行时间来衡量的，其计算式如下：

$$Resource\_idle_i=\left(\frac{VM_{idleTime[i]}}{VM_{time[i]}}\right) \tag{6}$$

$$Resource_{\text{idle}}=\frac{1}{m}\sum_{i}^{m}Resource\_idle_i \tag{7}$$

本文从用户和云服务提供商两个层面来评估任务调度方案的优劣。

定义 6 用户层次的评估(M_{user})是通过任务的执行时间和执行成本来进行衡量，评估因子 γ 用来决定两者的偏好权重。

由于云用户基数较大，其对于执行时间和执行成本的偏好各有不同，本文实验通过随机设置 γ 值来应对各种不同的云任务调度问题。

$$M_{\text{user}}=\gamma Makespan+(1-\gamma)Total_{\text{cost}} \tag{8}$$

定义 7 云服务提供商层次的评估(M)是通过综合考虑资源闲置率以及用户的需求等因素来选择评估，服务因子 α 用来决定其两者的偏好权重。

每个云服务提供商对于此 α 值的设定各有不同，本文实验通过随机设置 α 值来进行实验。

$$M=\alpha Resource_{\text{idle}}+(1-\alpha)M_{\text{user}} \tag{9}$$

3 采用 HEDSM 进行“云”中工作流调度

目前，针对工作流任务调度问题，众多学者将各种智能算法引入调度中，并就其研究的算法进行相应的创新，有静态调度算法，如遗传算法、粒子群算法；亦有动态调度算法，如先来先服务(FCFS)、短作业优先(SJF)。本文提出的云环境下基于 HEDSM 的工作流调度策略模型如算法 1 所示。

算法 1 基于 HEDSM 的工作流调度策略模型

输入：Workflow 任务，云环境下服务器及虚拟机参数，服务因子 α，评估因子 γ

输出：最优调度方案 S_{best}

1. 设置工作流任务集，记为 Work；
2. 在 Work 中随机选取 k 个任务，记为 Work_G；
3. 根据式(10)对 Work_G 中的 G_i 进行优先级设置，更新 Work_G；
4. 设置策略集 Sty，并对任务集 Work_G 进行预调度，生成调度方案集 Solution1；
5. 利用任务迁移策略对 Solution1 进行任务迁移处理，生成新的 Solution；
6. 根据服务因子 α 和评估因子 γ，利用式(11)计算衡量调度方案的评估值；
7. 根据步骤 6 得到的评估指标在方案集 Solution 中选择合适的调度方案 S_{best}。

3.1 工作流任务初始化

本文针对的是云环境下工作流任务调度，$Work=\{G_i\mid 1\leqslant i\leqslant d\}$，$G_i=\langle T,E\rangle$，即任务之间相互独立，可以并行执行；$G_i$ 中，子任务之间彼此相互依赖，呈现一定的执行顺序，因此需要对 $Work$ 中的 G_i 进行优先级设置，确定从哪个 G_i 的入口任务开始执行。G_i 的优先级可表示为：

$$Rank(G_i)=M_{\text{own}} \tag{10}$$

其中，M_{own}是利用传统的表调度算法 HEFT[24] 的思想来计算

优先级。

3.2　适应度评估

利用式(9)构建评估调度方案的适应度函数：

$$Fitness(fit_i)=\frac{1}{(1+M^i)} \tag{11}$$

根据式(9)、式(11)可知，M 值越小，$Fitness(fit_i)$ 值越高，代表方案的调度效果越好。

3.3　任务调度

3.3.1　预调度阶段

设置策略集，对任务进行预调度，生成预调度方案。策略集中包含先来先服务(FCFS)调度策略、最小最小(Min-Min)调度策略以及最大最小(Max-Min)调度策略[25-26]，这 3 个调度策略在各自分配规则的基础上保证了完成时间最短，同时分别包含各自的优先级，即优先调度先来的任务、优先调度小任务以及优先调度大任务。

3.3.2　任务迁移阶段

设置任务迁移策略，对任务进行迁移处理。该任务迁移针对的是入口任务的迁移，入口任务的执行影响着后续任务的执行，对入口任务进行迁移时，需要判定 3 部分内容：1)虚拟机上是否有合适的空闲时间段方便任务迁移；2)任务迁移之后是否会影响后继任务的执行；3)任务迁移之后的调度方案代表一种新的调度方案，这种新的调度方案相比原方案而言，是否在某一目标值方面表现出较优的性能。对此，定义其模型如下。

对于给定工作流任务 $t\in T$，在满足条件式(12)的基础上，同时满足式(13)－式(15)其中之一，则可对其进行迁移。

$$ET_{t_j}^{v_i}+D_s\leqslant D_f \tag{12}$$

$$Makespan(S_i')'<Makespan(S_i) \tag{13}$$

$$Total_{cost}(S_i')'<Total_{cost}(S_i) \tag{14}$$

$$Resource\_idle(S_i')'<Resource\_idle(S_i) \tag{15}$$

其中，$[D_s,D_f]$代表一个空闲时间段，S_i 代表旧的调度方案，S_i'代表任务迁移后的调度方案。

任务迁移的具体流程如算法 2 所示。

算法 2　任务迁移

1. 采用 FCFS，Min-Min 及 Max-Min 调度策略对工作流任务集 Work 进行预调度，并获得任务的调度方案集 Solution1；
2. 创建一个新的方案集 Solution；
3. while (Solution1 非空) do
4. 　从 Solution1 中取出第一个调度方案 S_i，获得相应的完成时间、完成成本以及资源闲置率；
5. 　遍历任务集 $Work_G$，得到的入口任务按照所分配的虚拟机计算性能升序排列并加入入口任务队列 List 中；
6. 　根据虚拟机的计算性能，对其进行升序排序并加入虚拟机队列 List1；
7. 　While (List 非空) do
8. 　　从 List 中选取一个入口任务 t；
9. 　　for (v∈List1) do
10. 　　if(满足条件式(12)，同时满足迁移条件式(13)－式(15)中之一)
11. 　　将入口任务 t 迁移到虚拟机 v 上执行；
12. 　　将得到的分配方案放入 Solution；
13. 　　end if
14. 　end for
15. 　删除任务；
16. end while
17. end while

3.4　选择

利用式(11)构建不同的评估模型在方案集 Solution 中选择合适的调度方案。任务调度阶段的处理过程保证了生成的方案集拥有各种调度性能，为选择同时满足用户和服务提供商需求的调度方案提供了便利。

3.5　算法示例

本节以图 2 的任务约束以及表 1 所列的任务和虚拟机参数为例。设置 3 个虚拟机 $v_0=2000$ mbits/s，$v_1=1000$ mbits/s，$v_2=500$ mbits/s，虚拟机的单位时间执行成本对应表 1 中价格一行，任务数量为 10。

表 1　DAG 任务的计算开销

Table 1　Calculation overhead of DAG task

	2000	1000	500
价格(元/秒)	0.9	0.5	0.2
任务 t_0/s	6.70	13.39	26.78
任务 t_1/s	6.92	13.83	27.66
任务 t_2/s	6.68	13.36	26.72
任务 t_3/s	5.30	10.59	21.18
任务 t_4/s	5.30	10.59	21.18
任务 t_5/s	5.44	10.88	21.77
任务 t_6/s	5.41	10.81	21.62
任务 t_7/s	4.41	8.81	17.62
任务 t_8/s	6.41	12.81	25.62
任务 t_9/s	6.91	13.81	27.62

图 2 是已经初始化的一个 DAG 图，表 1 是对应的 DAG 任务图在各个虚拟资源上的计算代价。以策略集中的 Min-Min 为例，首先 DAG 任务图经过 Min-Min 策略预调度后，得到图 3 的任务调度情况。

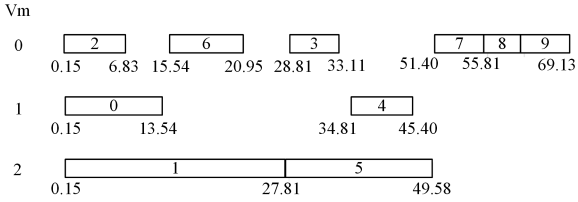

图 3　任务调度图

Fig. 3　Task scheduling

其中，虚拟机的启动时间为 0.15 s，关闭时间为 0.1 s，根据式(1)、式(5)及式(7)得到最终的完成时间为 69.23 s，完成成本为 94.973 元，资源闲置率为 32.3%。

由图 2 可知，入口任务为$\{T_0,T_1,T_2\}$，非入口任务为$\{T_3,T_4,T_5,T_6,T_7,T_8,T_9\}$。对入口任务按照所分配的虚拟资源的计算性能进行排序，得到队列 $list=\{T_1,T_0,T_2\}$，以 T_1 为例，由任务迁移策略可知，在虚拟机 V_1 上存在一个空闲时间段$[D_s\ D_f]=[13.55\ \ 27.82]$，假如把任务 T_1 迁移到虚拟机 V_1 上，T_1 在虚拟机 V_1 上的开始时间为 13.83 s，完成时间为 13.54＋13.83＝27.37＜D_f＝34.81 s。在迁移任务后，如图 4 所示，根据式(1)、式(5)及式(7)计算新的任务调度方

案的情况，得到完成时间为 66.56 s，完成成本为 89.382 s，资源闲置率为 35.18%，满足迁移条件式(12)－式(14)，因此任务 T_1 可以迁移到虚拟机 V_1 上执行。如此往复，得到最终的迁移结果。

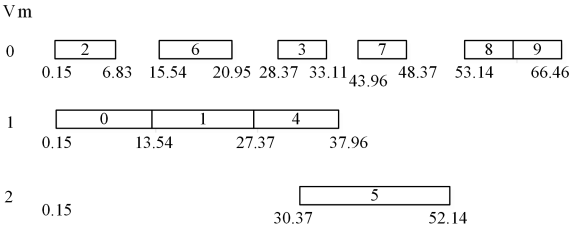

图 4 T_1 任务迁移图

Fig. 4 Graph of T_1 Task scheduling

图 5 是最终的迁移结果，完成时间为 53.17 s，完成成本为 67.958 元，资源闲置率为 21.4%。

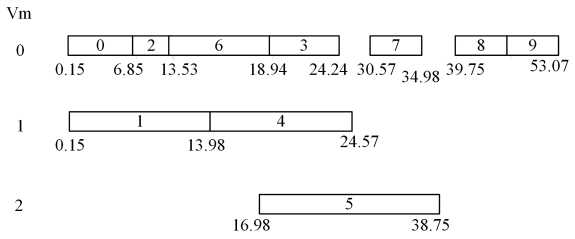

图 5 最终迁移图

Fig. 5 Final migration

4 仿真实验

4.1 仿真环境及参数配置

为了验证 HEDSM 模型的可行性和有效性，实验利用 WorkflowSim 平台模拟云计算资源调度环境，并对其进行对比分析测试。在模拟资源调度时需要进行参数配置，为此，任务采用 Workflow Generator[17-18] 中的科学工作流案例[1)]，如表 2 所列。表 3 列出了实验采用的亚马逊 EC2 实例配置及计价方式。

表 2 工作流类型

Table 2 Workflow Type

Workflow Type	DAX Type			
Montage	25 Node	50 Node	100 Node	1000 Node
CyberShake	30 Node	50 Node	100 Node	1000 Node
Epigenomic	24 Node	46 Node	100 Node	997 Node
LIGO Inspiral	30 Node	50 Node	100 Node	1000 Node
SIPHT	30 Node	60 Node	100 Node	1000 Node

表 3 亚马逊计价模式

Table 3 Pricing model of Amazon

Vm 类型	(memory size, CPU speed)	hourly Priceper
m4. L	8000 MB MEM, 3250 MIPS	$0.120
m4. xL	16000 MB MEM, 6500 MIPS	$0.239
m4. 2xL	32000 MB MEM, 13000 MIPS	$0.479
c4. xL	10000 MB MEM, 10000 MIPS	$0.299
c4. 2xL	15000 MB MEM, 15500 MIPS	$0.419
c4. 2xLe	15000 MB MEM, 22500 MIPS	$0.489
c4. 4xL	30000 MB MEM, 32000 MIPS	$0.838

4.2 性能分析

本文分别从异构有效性和策略改进比(Improvement Ration of Strategy, IROS)两个方面来进行实验对比分析。首先，将所提模型与传统的 FCFS、Min-Min 和 Max-Min[25-26] 进行比较，然后将其与 IMax-Min 和 LWRound_Robin[10-11] 做对比。实验所需的工作流总任务数量设置为[20, 50]，单个任务的子任务数量为[20, 1000]，服务因子 α 和评估因子 γ 随机选择。

4.2.1 异构有效性

定义 8 异构性用于衡量 $Workflow=\{G_i \mid 1\leqslant i\leqslant d\}$ 中不同任务 G 的数量，定义如下：

$$ls=\frac{N_{G\_type}}{N_G} \tag{16}$$

其中，N_{G_type} 代表 G 的种类，N_G 代表 G 的数量。不同类型的用户提交的任务模型不同，本文用 ls 来粗略衡量不同的任务模型。

首先，随机选取任务数量，通过设置不同的 ls 进行模拟实验，ls 的取值范围为[10%, 100%]。

其次，由于不同类型的用户和服务提供商有不同的评估模式，对调度方案有不同的要求，对任务完成时间、完成成本以及资源闲置率这 3 个优化目标有不同的偏好。因此，本文随机选取 10 组(α,γ)，即相当于 10 种不同类型的调度需求，每组实验 50 次。将 HEDSM 模型得到的方案分别与其他策略进行比较，图 6－图 8 展现了其中的 3 组结果。

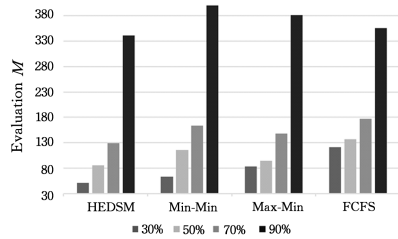

图 6 (α,γ)为(0.5, 0.5)时 HEDSM 与传统算法的对比

Fig. 6 Comparing of HEDSM and traditional strategies when (α,γ) is (0.5, 0.5)

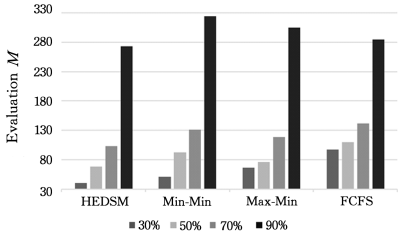

图 7 (α,γ)为(0.4, 0.8)时 HEDSM 与传统算法的对比

Fig. 7 Comparing of HEDSM and traditional strategies when (α,γ) is (0.4, 0.8)

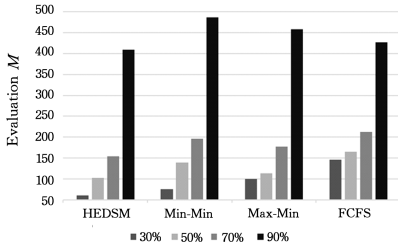

图 8 (α,γ)为(0.7, 0.4)时 HEDSM 与传统算法的对比

Fig. 8 Comparing of HEDSM and traditional strategies when (α,γ) is (0.7, 0.4)

1) https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator

根据式(9)计算得到 M 值，其用来衡量任务调度之后的方案效果，M 值越低，代表的方案效果越好。图 6－图 8 分别设置(α,γ)因子值为(0.5,0.5)，(0.4,0.8)，(0.7,0.4)，在这 3 种不同的调度评估下，分别设置 ls 为 30%，50%，70%，90%时以对比 M 值。如图 6 所示，在(α,γ)因子值为(0.5,0.5)，ls 为 50%时，HEDSM 的 M 值要低于其他策略的 M 值。由此得出以下两个结论。

(1)面对不同类型的调度需求，即(α,γ)值不同时，相较于 FCFS，Min-Min 和 Max-Min 调度算法，HEDSM 仍能寻求到较好的调度方案，这说明该算法模型生成的方案集具有多样性，可以满足各种不同的需求。

(2)面对云环境中不同类型的任务模型，本文提出的算法模型在保证任务执行时间、执行成本以及资源闲置率的同时，都能够很好地完成任务调度。

图 9－图 11 的实验参数设置与图 6－图 8 一致。如图 9 所示，当 ls 值设置为 30%和 50%时，各个策略的 M 值相差无几，调度效果不是很明显。当 ls 值为 70%和 90%时，HEDSM 的 M 值明显比其他两个调度策略的 M 值低，而 IMax-Min 与 LWRound_Robin 策略相比，在不同的 ls 值下，M 值时高时低，调度效果不是很稳定，图 10－图 11 亦是如此。由此可得出以下两个结论。

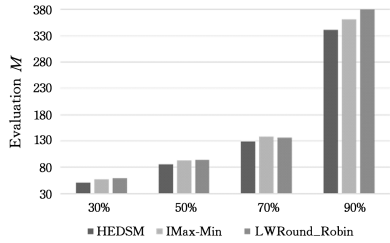

图 9　(α,γ)为(0.5,0.5)

Fig. 9　(α,γ) is (0.5,0.5)

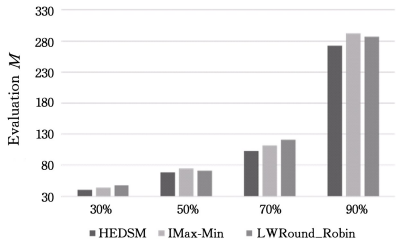

图 10　(α,γ)为(0.4,0.8)

Fig. 10　(α,γ) is (0.4,0.8)

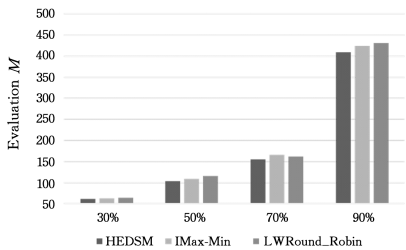

图 11　(α,γ)为(0.7,0.4)

Fig. 11　(α,γ) is (0.7,0.4)

(1)相较于 IMax-Min 与 LWRound_Robin 算法，在面对不同的(α,γ)值和 ls 值时，本文提出的算法模型寻优更加稳定，这说明在本算法模型中设置的策略集是有效的，策略集包含在各个方面具有调度优势的策略，使算法的寻优效果更加稳定。

(2)本实验仅设置了任务模型和用户需求两个不同的影响云环境调度性能的因素，结果证明本算法模型是有效的，这也侧面说明了在面对云环境中其他未知复杂多变的因素时，HEDSM 模型更适用。

4.2.2　策略改进比

定义 9　策略改进比用于衡量提出的 HEDSM 相对于其他 3 个策略的改进程度的。

$$IROS_i=\frac{|H_i-S_i|}{S_i} \tag{17}$$

其中，H 表示 HEDSM 策略；S 表示{Min-Min，Max-Min，FCFS}策略集中的其中一个；$i\in\{Time,Cost,Resource_{idle}\}$，$Time$ 由式(1)计算得到，$Cost$ 由式(5)计算得到，$Resource_{idle}$ 首先通过式(7)计算，再进行正交化得到，保证 3 个优化目标在同一范围内变化。根据式(1)、式(5)和式(7)的定义可得，根据式(1)、式(5)、式(7)计算出来的值越小，则调度方案越佳。

本次实验子任务数量为[500,3 000]，随机进行 20 次实验，表 4－表 6 将 HEDSM 从完成时间、执行成本和资源闲置率 3 个方面分别与 Min-Min，Max-Min 及 FCFS 3 个调度策略进行对比的结果，其中 $IROS_{avg}$ 是 20 次实验结果的平均值。

表 4　Min-Min 和 HEDSM 的对比 IROS 值

Table 4　Value of IROS comparing with Min-Min and HEDSM

	HEDSM	Max-Min	$IROS$%	$IROS_{(avg)}$/%
Time	34.56	53.15	35.0	23.5
	80.62	100.03	19.4	
	121.37	153.33	20.8	
Cost	61.17	61.56	6.3	6.8
	138.74	150.15	7.6	
	200.03	212.41	5.8	
$Resource_{idle}$	61.29	70.84	13.5	28.9
	247.65	372.64	33.5	
	337.74	495.21	31.8	

表 5　Max-Min 和 HEDSM 的对比 IROS 值

Table 5　Value of IROS comparing with Max-Min and HEDSM

	HEDSM	Max-Min	$IROS$%	$IROS_{(avg)}$/%
Time	34.56	99.06	65.1	44.5
	80.62	128.97	37.5	
	121.37	221.37	45.1	
Cost	61.17	63.23	3.2	4.6
	138.74	145.98	5.0	
	200.03	209.50	4.5	
$Resource_{idle}$	61.29	95.28	35.7	43.2
	247.65	478.76	48.3	
	337.74	577.30	41.5	

表 6　FCFS 和 HEDSM 的对比 IROS 值

Table 6　Value of IROS comparing with FCFS and HEDSM

	HEDSM	Max-Min	$IROS$%	$IROS_{(avg)}$/%
Time	34.56	108.52	68.1	51.2
	80.62	156.16	48.4	
	121.37	214.83	43.5	
Cost	61.17	64.87	5.7	6.5
	138.74	149.02	6.9	
	200.03	217.42	8.0	
$Resource_{idle}$	61.29	176.76	65.3	54.5
	247.65	515.93	52.0	
	337.74	755.57	55.3	

以表 4 为例，从云用户的角度分析可得，本文提出的算法在完成时间和完成成本上分别提高了 23.5%和 6.8%；从提供商的角度分析可得，本文提出的算法在资源闲置率上提高了 28.9%。由此，本文提出的算法模型中设置的任务迁移策略是有效的，它能有效地对任务进行迁移处理，使得算法在 3 个目标优化方面都有所改善。

表 7 IMax-Min 和 HEDSM 的对比 IROS 值

Table 7 Value of IROS comparing with IMax-Min and HEDSM

	HEDSM	IMax-Min	$IROS$/%	$IROS_{(avg)}$/%
Time	34.56	38.56	10.4	12.8
	80.62	92.67	13.0	
	121.37	142.83	15.0	
Cost	61.17	62.34	1.8	1.7
	138.74	140.25	1.1	
	200.03	205.87	2.8	
$Resource_{idle}$	61.29	64.65	5.2	7.8
	247.65	263.3	6.7	
	337.74	373.6	9.6	

表 8 LWRound_Robin 和 HEDSM 的对比 IROS 值

Table 8 Value of IROS comparing with LWRound_Robin and HEDSM

	HEDSM	LWRound_Robin	$IROS$/%	$IROS_{(avg)}$/%
Time	34.56	38.31	9.8	11.4
	80.62	90.89	11.3	
	121.37	138.07	12.1	
Cost	61.17	63.19	3.2	4.1
	138.74	145.28	4.5	
	200.03	207.93	3.8	
$Resource_{idle}$	61.29	62.86	2.5	3.7
	247.65	257.43	3.9	
	337.74	355.14	4.9	

将 HEDSM 从完成时间、执行成本、资源闲置率 3 个方面分别与 IMax-Min 和 LWRound_Robin 策略进行对比。表 7 与表 8 分别展现了 *IROS* 平均值以及其中 3 组实验结果。分析可得，相较于 IMax-Min 和 LWRound_Robin 这两个策略，HEDSM 在完成时间上分别缩短了 12.8%和 11.4%，而在执行成本和资源闲置率方面略有提升。在云环境下任务调度过程中，IMax-Min 在执行成本方面显现优势，而 LWRound_Robin 在资源闲置率方面显现优势。由此可得，本文提出的算法模型通过设置策略集，继承了不同策略的调度优势，经过任务迁移，从而生成具有不同优势的调度方案，使得调度性能不仅稳定而且优于其他两者，说明该算法模型是有效的。

4.3 算法复杂度分析

本文设定的任务规模为 S，入口任务规模为 T，子任务规模为 N，虚拟机规模为 M，根据研究问题可得 $M<S<T<N$。传统的 FCFS，Min-Min 和 Max-Min 调度算法均是利用特定的规则实现 N 个子任务到 M 个虚拟机上的映射，即时间复杂度为 $O(M*N)$，而根据 IMax-Min 和 LWRound_Robin[10-11]的算法流程可知，IMax-Min 的时间复杂度为 $O(M*N)+O(M)$，LWRound_Robin 的时间复杂度为 $O(M*N)$。

本文提出的算法的大致流程为：1)任务优先级设置，时间复杂度为 O(S)；2)任务预调度，时间复杂度为 $O(3M*N)$；3)任务迁移阶段，根据迁移的具体流程可知时间复杂度为 $O(3T*M)$；4)选择阶段，在生成的方案集中选择合适调度方案，则时间复杂度为 $O(3T*M)$。算法的时间复杂度为各阶段时间复杂度之和。相较于上述算法，在时间复杂度方面，本文提出的算法计算开销更大。

结束语 针对云环境下的任务调度问题，本文构建了一个 HEDSM 模型，在该模型中设置任务优先级，构建评估系统，建立调度集，并引入任务迁移策略。实验结果证明，该模型具有良好的效果。但与此同时，该模型仍有不足之处：1)在适应度评估上仍需要进一步探索，本文只针对完成时间、完成成本及资源闲置率 3 个方面，并没有考虑其他因素；2)在策略集中，本文只设置了 3 种策略来进行预调度以验证模型的有效性，仍应进一步添加策略集，且应预先设置分类，不仅对提交上来的任务模型进行分类，还应根据任务模型对各个策略进行分类，以便快速找到适合的调度策略来处理提交上来的任务；3)任务迁移策略只是针对入口任务进行迁移来降低资源闲置率，策略仍有近一步改善的可能。以上 3 点就是接下来需要进一步研究的方向。

参 考 文 献

[1] NETJINDA N, SIRINAOVAKUL B, ACHALAKUL T. Cost optimal scheduling in IaaS for dependent workload with particle swarm optimization[J]. The Journal of Supercomputing, 2014, 68(3): 1579-1603.

[2] VAQUERO L M, RODERO M L, CACERES J, et al. A break in the clouds: Towards a cloud definition[J]. ACM SIGCOMM Computer Communication Review, 2008, 39(1): 50-55.

[3] CHEN H, ZHU X, GUO H, et al. Towards energy-efficient scheduling for real-time tasks under uncertain cloudcomputing environment [J]. Journal of Systems and Software, 2015, 99(2): 20-35.

[4] RAJ A, KAUR K, DUTTA U, et al. Enhancement of Hadoop Clusters with Virtualization Using the Capacity Scheduler[C]// Third International Conference on Services in Emerging Markets. IEEE, 2013.

[5] YADAV R K, MISHRA A K, PRAKASH N, et al. An improved round robin scheduling algorithm for CPU scheduling[J]. International Journal on Computer Science and Engineering, 2010, 2(4): 1064-1066.

[6] TRIPATHY B, DASH S, PADHY S K. Dynamic task scheduling using a directed neural network[J]. Journal of Parallel and Distributed Computing, 2015, 75: 101-106.

[7] JENA R K. Multi objective task scheduling in cloud environment using nested PSO framework[J]. Procedia Computer Science, 2015, 57: 1219-1227.

[8] ALEBRAHIM S, AHMAD I. Task scheduling for heterogeneous computing systems[J]. Journal of Supercomputing, 2017, 73(6): 2313-2338.

[9] ATEF A, HAGRAS T, MAHDY Y B, et al. Lower-bound com-

plexity algorithm for task scheduling on heterogeneous grid[J]. Computing，2017.

[10] CHITRA D D，RHYMEND U V. Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Nonpreemptive Dependent Tasks[J]. The Scientific World Journal，2016，2016：1-14.

[11] KAUR N，KAUR K. Improved max-min scheduling algorithm [J]. IOSR Journal of Computer Engineering (IOSR-JCE)，2015，17(3)：42-49.

[12] PANDA S K，JANA P K. Uncertainty-based QoS min-min algorithm for heterogeneous multi-cloud environment[J]. Arabian Journal for Science and Engineering，2016，41(8)：3003-3025.

[13] ALI S A，ALAM M. Resource-Aware Min-Min (RAMM) Algorithm for Resource Allocation in Cloud Computing Environment [J]. arXiv：1803. 00045，2018.

[14] BABU L D D，GUNASEKARAN A，KRISHNA P V. A decision-based pre-emptive fair scheduling strategy to process cloud computing work-flows for sustainable enterprise management [M]. Inderscience Publishers，2017.

[15] GUPTA I，KUMAR M S，JANA P K. Transfer time-aware workflow scheduling for multi-cloud environment[C]// 2016 International Conference on Computing，Communication and Automation (ICCCA). IEEE，2016.

[16] GUO F，YU L，TIAN S，et al. A workflow task scheduling algorithm based on the resources' fuzzy clustering in cloud computing environment[J]. International Journal of Communication Systems，2015，28(6)：1053-1067.

[17] MAHAJAN K，MAKROO A，DAHIYA D. Round robin with server affinity：a VM load balancing algorithm for cloud based infrastructure[J]. Journal of Information Processing Systems，2013，9(3)：379-394.

[18] WANG X，YEO C S，BUYYA R，et al. Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm[J]. Future Generation Computer Systems，2011，27(8)：1124-1134.

[19] CHOUDHARY A，GUPTA I，SINGH V，et al. A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing[J]. Future Generation Computer Systems，2018，83：14-26.

[20] RODRIGUEZ M A，BUYYA R. Deadline based resource provisioningand scheduling algorithm for scientific workflows on clouds[J]. IEEE transactions on Cloud Computing，2014，2(2)：222-235.

[21] LEE Y C，WANG C，ZOMAYA A Y，et al. Profit-driven scheduling for cloud services with data access awareness[J]. Journal of Parallel and Distributed Computing，2012，72(4)：591-602.

[22] JUVE G，CHERVENAK A，DEELMAN E，et al. Characterizing and profiling scientific workflows[J]. Future Generation Computer Systems，2013，29(3)：682-692.

[23] BHARATHI S，CHERVENAK A，DEELMAN E，et al. Characterization of scientific workflows[C]// Third Workshop on Workflows in Support of Large-Scale Science. IEEE，2008：1-10.

[24] TOPCUOGLU H，HARIRI S，WU M. Performance-effective and low-complexity task scheduling for heterogeneous computing[J]. IEEE Transactions on Parallel and Distributed Systems，2002，13(3)：260-274.

[25] GUO P，LI T，LI Q L. A load scheduling algorithm in cloud computing environment[J]. System Engineering Theory and Practice，2014，34(s1)：269-275.

[26] ZHOU Z，HU Z G. Research on Scheduling Algorithms for Integrating Greedy Strategies in Cloud Computing[J]. Small Microcomputer System，2015，36(5)：1024-1027.

SUN Ming，born in 1966，master degree，is a member of China Computer Federation. Her main research interests include cloud computing，intrusion detection，and web collaboration.