软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2018,29(11):3326−3339 [doi: 10.13328/j.cnki.jos.005477]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

云环境下基于多目标的多科学工作流调度算法*

袁友伟[1,2], 鲍泽前[1,2], 俞东进[1], 李万清[1]

[1](杭州电子科技大学 计算机科学与技术学院,浙江 杭州 310018)

[2](复杂系统建模与仿真教育部重点实验室(杭州电子科技大学),浙江 杭州 310018)

通讯作者: 鲍泽前, E-mail:549135624@qq.com

摘 要: 针对现有云环境下的多科学工作流调度算法中存在的未考虑安全调度问题,提出了多科学工作流安全-时间约束费用优化算法 MSW-SDCOA(multi-scientific workflows security-deadline constraint cost optimization algorithm).首先,MSW-SDCOA 基于数据依赖关系压缩科学工作流,减少任务节点数从而节省了调度开销;并通过改进 HEFT(heterogeneous earliest-finish-time)算法形成调度序列,以实现全局多目标优化调度;最后,通过优化 ACO(ant colony optimization)中信息素更新策略和启发式信息,进一步改善费用优化效果.仿真实验表明,MSW-SDCOA 算法在费用优化效果上比 MW-DBS 算法提高了约 14%.

关键词: 安全调度;费用优化;多科学工作流;压缩;分层计算

中图法分类号: TP306

中文引用格式: 袁友伟,鲍泽前,俞东进,李万清.云环境下基于多目标的多科学工作流调度算法.软件学报,2018,29(11):3326−3339. http://www.jos.org.cn/1000-9825/5477.htm

英文引用格式: Yuan YW, Bao ZQ, Yu DJ, Li WQ. Multi-Scientific workflow scheduling algorithm based on multi-objective in cloud environment. Ruan Jian Xue Bao/Journal of Software, 2018,29(11):3326−3339 (in Chinese). http://www.jos.org.cn/1000-9825/5477.htm

Multi-Scientific Workflow Scheduling Algorithm Based on Multi-Objective in Cloud Environment

YUAN You-Wei[1,2], BAO Ze-Qian[1,2], YU Dong-Jin[1], LI Wan-Qing[1]

[1](School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China)

[2](Key Laboratory of Complex Systems Modeling and Simulation (Hangzhou Dianzi University), Ministry of Education, Hangzhou 310018, China)

Abstract: To address the problem that safe scheduling is not taken into consideration in existing multi-scientific scheduling workflow algorithm in cloud environment, this paper proposes a multi-scientific workflows security-deadline constraint cost optimization algorithm (MSW-SDCOA). First, based on data flow dependency, MSW-SDCOA compresses scientific workflow and reduces the number of task nodes to save scheduling cost. Secondly, through optimizing HEFT algorithm, a scheduling sequence is formed to realize overall multi-objective optimization scheduling. Lastly, by optimizing update strategies of pheromone and heuristic information in ant colony optimization (ACO), cost optimization effect is further improved. The simulation experiment results show that the cost optimization effect of MSW-SDCOA algorithm is about 14% better than that of MW-DBS algorithm.

Key words: security scheduling; cost optimization; multi-scientific workflow; compress; hierarchical compute

* 基金项目: 国家自然科学基金(61370218); 浙江省重点高校建设专项资金(GK158800205032)

Foundation item: National Natural Science Foundation of China (61370218); Key University Construction Project of Zhejiang Province of China (GK158800205032)

本文由面向智能制造的业务过程管理与服务技术专题特约编辑王建民教授、刘建勋教授推荐.

收稿时间: 2017-07-19; 修改时间: 2017-09-16; 采用时间: 2017-11-14; jos 在线出版时间: 2017-12-06

CNKI 网络优先出版: 2017-12-06 15:37:24, http://kns.cnki.net/kcms/detail/11.2560.TP.20171206.1536.024.html

科学工作流是工作流应用于科学研究的一种重要机制,可以帮助科学家进行实验仿真和数据分析等工作[1],广泛应用于医药学[2]、物理学[3]、生物学[4]等学科领域.随着科学计算系统的复杂性日益增加,科学工作流应用已成为大数据应用,需要大规模的基础设施,以便在合理的时间内执行,故传统的计算环境很难满足多科学工作流的要求.云计算拥有丰富的存储资源以及高效的计算能力[5],从而为多科学工作流提供了一种新的部署和执行方案.

目前,对于单个科学工作流调度的研究,不论是调度模型还是调度目标的多样性均取得了很大进展,然而针对多科学工作流调度的研究相对较少[6].文献[7]将多个工作流合成一个工作流后,再利用单工作流调度算法来调度合成的工作流,但这种方法存在调度的不公平性问题.文献[8]针对多工作流调度的公平性问题提出了Fairness 算法,但未考虑动态提交多科学工作流的情况.在云服务的科学工作流调度进程中,存在多个不同结构的科学工作流同时提交或在执行过程中动态提交的情况,因此,针对多异构科学工作流的调度算法需要拥有对动态提交的科学工作流进行实时处理的功能.

提供商通常以基于小时的定价模式向用户收费,所以费用是通过运行的小时数计算[9].一项调查显示,云服务最关心的问题之一是安全性[10],这是因为云服务允许用户执行未验证的第三方应用程序,这成为了安全威胁的来源[11],若未考虑安全性,则可能会导致科学工作流调度失败.目前,对于单个科学工作流的研究已有很大进展,但这些研究不能直接运用在多科学工作流的调度上.

云科学工作流调度中,用户主要考虑时间和费用约束,为此,在这两个约束下的云科学工作流调度研究方面产生了一系列成果.

- 基于大多数算法具有高时间复杂度从而难以在现实环境中使用的缺点,文献[12]在考虑用户时间和费用约束下提出了具有二次时间复杂度的启发式调度算法 DBCS(deadline-budget constrained scheduling).
- 针对目前云市场由众多不同的云提供商组成的情况,文献[13]通过修改关键路径算法 PCPA(partial critical paths algorithm),提出了一种多云部分关键路径算法 MCPCP(multi-cloud partial critical paths),旨在满足费用约束下最大限度地减少执行费用.
- 文献[14]在考虑云的特性下,提出元启发式遗传算法,在满足时间约束下,最大程度降低执行费用.
- 此外,还有通过 ACO(ant colony optimization)算法和 PSO(particle swarm optimization)算法等[15,16]启发式算法在时间和费用的约束下进行优化.

上述算法均具有一定的费用优化效果,但是针对具有截止时间的多个科学工作流费用优化的研究相对较少,仍需进一步研究.

在科学工作流调度过程中,安全性已经成为云计算环境下另一个关键因素[17−19].目前,许多云计算环境没有采取相应的安全措施来对抗安全威胁[20],因此,云计算环境下的安全调度问题成为科学工作流领域的研究热点.针对用户私有数据在混合云计算环境存在曝光风险问题,文献[21]在同时考虑用户的时间和费用约束下,提出了一种在工作流调度中保留用户隐私的算法 MHPC(multiterminal cut for privacy in hybrid clouds).出于安全性考虑,存在数据集因隐私原因无法在数据中心中移动,文献[22]对此提出了安全和预算感知工作流调度策略 SABA(security-aware and budget-aware workflow scheduling strategy).为了防止工作流任务在云资源上执行发生错误,文献[23]通过设计任务故障智能预测模型来预测科学工作流应用程序的任务故障,从而实现主动容错.上述算法仅仅研究了单科学工作流的安全措施,有必要针对多科学工作流的安全调度问题做进一步的研究.

针对上述问题,本文是在公有云的环境下,研究单个用户同时提交多异构科学工作流以及动态提交的问题,将关键的安全性和时间约束加入到多科学工作流费用优化模型中,提出了云环境下基于安全性和时间约束的多科学工作流费用优化模型,并提出了针对该模型的费用优化算法 MSW-SDCOA.该算法首先在任务调度之前利用科学工作流的数据依赖性对其进行压缩,从而减少任务节点数,并通过改进 HEFT 算法[24],考虑任务路径长度以及安全开销下分层计算任务节点的权重值形成调度序列,在任务调度过程,优化 ACO 算法中的信息素更新策略和启发式信息,以期望进一步改善多科学工作流的费用优化效果.

本文前言介绍背景和相关工作.第 1 节提出本文的模型.第 2 节讲诉基于本文模型提出的算法.第 3 节给出实验仿真结果和分析.第 4 节给出本文的结论.

1 云环境下基于安全性和时间约束的多科学工作流费用优化模型

1.1 模型描述

云环境下基于安全性和时间约束的多科学工作流费用优化模型中的多科学工作流可描述为

$$W=\{w_i|w_i=(T,E)\},$$

其中,

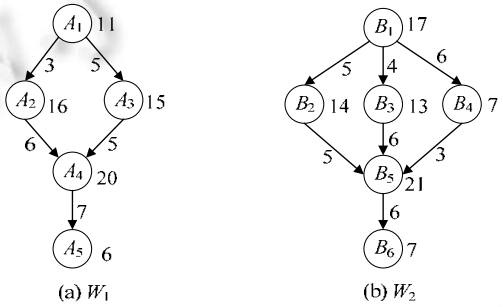
- T 代表一个科学工作流任务的集合,$T=\{t_{i,j}\,|\,t_{i,j}=(I_{t_{i,j}},O_{t_{i,j}},L_{t_{i,j}},A_{t_{i,j}})\}$,$t_{i,j}$ 代表第 i 个工作流的第 j 个任务,$I_{t_{i,j}}$ 代表 $t_{i,j}$ 的输入数据集,$O_{t_{i,j}}$ 代表 $t_{i,j}$ 的输出数据集,$L_{t_{i,j}}$ 代表 $t_{i,j}$ 的指令数大小,$A_{t_{i,j}}$ 代表 $t_{i,j}$ 所在的科学工作流;
- E 用来表示科学工作流任务之间的依赖关系:

$$E=\{(t_{i,j},t_{i,k},d_{t_{i,j},t_{i,k}})\,|\,(t_{i,j},t_{i,k})\in T,d_{t_{i,j},t_{i,k}}\in O_{t_{i,j}}\text{ and }d_{t_{i,j},t_{i,k}}\in I_{t_{i,k}}\}.$$

$(t_{i,j},t_{i,k},d_{t_{i,j},t_{i,k}})$ 代表 $t_{i,j}$ 和 $t_{i,k}$ 的数据依赖关系,$t_{i,j}$ 是 $t_{i,k}$ 的前驱任务,$t_{i,k}$ 是 $t_{i,j}$ 的后继任务,$d_{t_{i,j},t_{i,k}}$ 代表 $t_{i,j}$ 传给 $t_{i,k}$ 的数据集,$L(d_{t_{i,j},t_{i,k}})$ 代表数据集的大小.

$t_{i,j}$ 的前驱任务集记为 $pre(t_{i,j})$,$t_{i,j}$ 的后继任务集记为 $succ(t_{i,j})$.

下面通过图 1(a)和图 1(b)两个子图对应的 w_1 和 w_2 这两个科学工作流实例和表 1 的对应参数信息对本文提出的模型中多科学工作流进行详细的描述.



(a) W_1 (b) W_2

Fig.1 Multi-Scientific workflows instance

图 1 多科学工作流实例

Table 1 Multi-Scientific workflows instance parameter information

表 1 多科学工作流实例参数信息

W	W_1					W_2					
t	A_1	A_2	A_3	A_4	A_5	B_1	B_2	B_3	B_4	B_5	B_5
I	NA	3	5	6, 5	7	NA	5	4	6	5, 6, 3	6
O	3, 5	6	5	7	NA	5, 4, 6	5	6	3	6	NA
L	11	16	15	20	6	17	14	13	7	21	7
pre	NA	A_1	A_1	A_2, A_3	A_4	NA	B_1	B_1	B_1	B_2, B_3, B_4	B_5
$succ$	A_2, A_3	A_4	A_4	A_5	NA	B_2, B_3, B_4	B_5	B_5	B_5	B_6	NA

本文提出的云环境下基于安全性和时间约束的多科学工作流费用优化模型如图 2 所示,主要由科学工作流压缩、分层计算任务节点权重值生成调度序列、生成调度方案、动态调度多科学工作流这 4 分组成.为了更直观地表示出模型中的步骤,分别使用 A,B,C 代表不同的科学工作流.在该模型中,用户提交多个科学工作流,这些科学工作流到达后,首先会将多科学工作流进行压缩,减少其任务节点数,然后通过改进的 HEFT 算法分层计

算各个科学工作流的任务节点的权重值,并按照规则生成调度序列,接着将调度序列作为输入调用改进的 ACO 算法生成调度方案,最后按调度方案进行调度.如果用户在提交多科学工作流后提交了新的科学工作流,则通过动态调度多科学工作流部分实现调度.

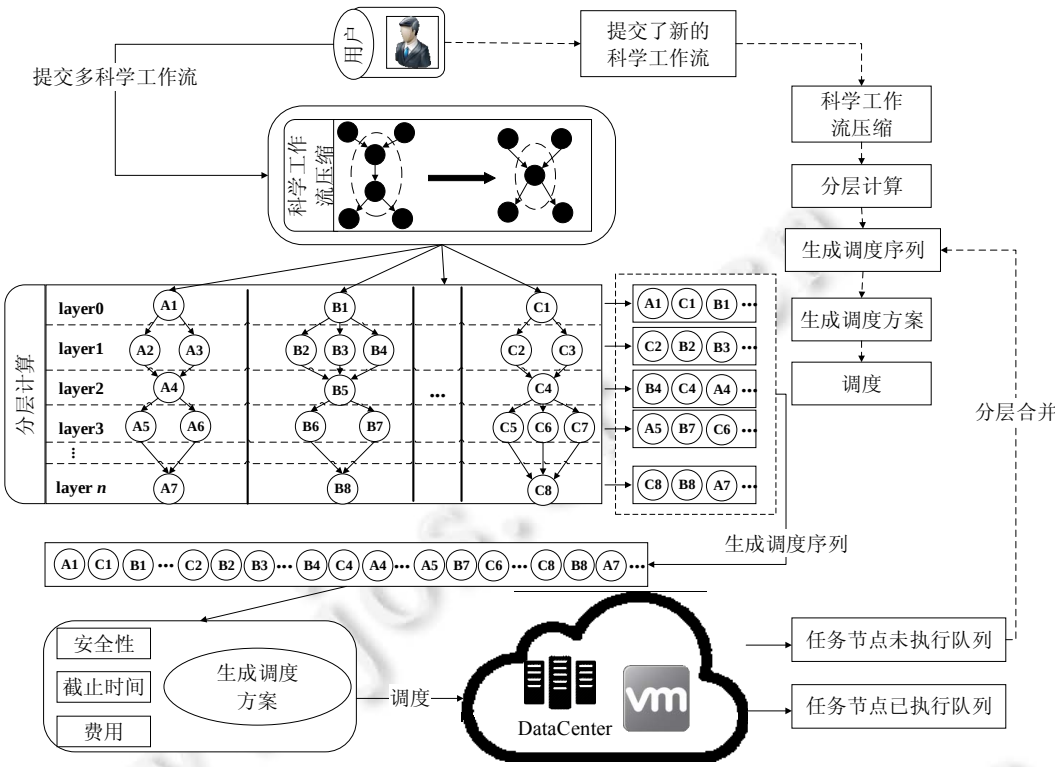


Fig.2 Multi-Scientific workflows scheduling flowchart

图 2 多科学工作流调度流程图

该模型对应解决的问题是:保证安全性和完成时间在用户设定的约束范围内选取租赁费用最小的多科学工作流调度方案.

1.2 多科学工作流安全性计算

在云计算环境中存在多种安全威胁,ARP 欺骗、监听、拒绝服务是云系统的 3 种常见攻击,我们使用 3 种安全服务(即身份验证服务、完整性服务和机密性服务)来防范科学工作流运行时被攻击[25],为了方便描述,我们使用 a,b 和 c 分别表示身份验证服务、完整性服务和机密性服务.

安全性是指多个科学工作流都执行成功的概率,通过计算每个任务节点在虚拟机上接受数据、执行以及生成输出数据的风险率求得.假设每个任务都需要 3 种不同安全级别的安全服务,在接受输入数据时使用身份验证服务,在任务执行时使用完整性服务,在生成输出数据过程中使用机密性服务.例如完整性服务,其中的各个哈希函数性能不同,假设每个哈希函数的安全级别与其时间开销成反比,将每个哈希函数的安全级别分配在 0.18~1 之间[25].例如,我们将安全级别 1 分配给效果最好但实现最慢的哈希函数 TIGER.

任务节点 $t_{i,j}$ 的每种安全服务的风险率如公式(1)所示[26].

$$P(t_{i,j},sl_{t_{i,j}}^{l})=1-\mathrm{e}^{-\lambda^{l}(1-sl_{t_{i,j}}^{l})},l\in\{a,b,c\} \tag{1}$$

其中,$sl_{t_{i,j}}^{l}$ 代表任务 $t_{i,j}$ 获得的安全服务 l 的级别.在云环境下,因为单位时间内受到的不同种类的攻击次数不同,故 λ^{l} 的值是不同的.

考虑任务的 3 种安全服务,任务 $t_{i,j}$ 的风险率 $P(t_{i,j})$定义为[26]

$$P(t_{i,j})=1-\prod_{l\in\{a,b,c\}}(1-P(t_{i,j},sl_{t_{i,j}}^{l})) \tag{2}$$

科学工作流 w_i 的风险率 $P(w_i)$可以通过其任务集合的风险率求得[26]:

$$P(w_i)=1-\prod_{t_{i,j}\in w_i}(1-P(t_{i,j})) \tag{3}$$

多科学工作流 W 的安全性概率 $S(W)$可以通过其科学工作流集合的风险率求得,多科学工作流 W 运行成功的条件为各科学工作流都运行成功,故通过计算科学工作流的风险率可得其成功率,将各科学工作流的成功率相乘即为多科学工作流 W 的安全性概率,所以 $S(W)$可定义为

$$S(W)=\prod_{w_i\in W}(1-P(w_i)) \tag{4}$$

1.3 多科学工作流完成时间计算

假设 $t_{i,j}$ 被分配到虚拟机上 $vm_{t_{i,j}}$ 执行,则 $t_{i,j}$ 开始时间利用如下公式求取:

$$\left.\begin{aligned} st(t_{i,j},vm_{t_{i,j}})&=\max\{t_{i,j}^{avai},avai(vm_{t_{i,j}})\}\\ t_{i,j}^{avai}&=\max_{t_{i,k}\in pre(t_{i,j})}\{ft(t_{i,k},vm_{t_{i,k}})+tt(t_{i,j},t_{i,k})\}\\ tt(t_{i,j},t_{i,k})&=\begin{cases}L(d_{t_{i,k},t_{i,j}})/b(vm_{t_{i,k}},vm_{t_{i,j}}), & vm_{t_{i,j}}!=vm_{t_{i,k}}\\ 0, & vm_{t_{i,j}}=vm_{t_{i,k}}\end{cases}\end{aligned}\right\} \tag{5}$$

其中,$t_{i,j}^{avai}$ 代表 $t_{i,j}$ 所有前驱任务都执行完成并且数据都传输至 $vm_{t_{i,j}}$ 上的时间,$avai(vm_{t_{i,j}})$代表 $vm_{t_{i,j}}$ 执行完 $t_{i,j}$ 之前所有任务后的时间.$t_{i,k}$ 是 $t_{i,j}$ 的后继任务,假设 $t_{i,k}$ 被分配到代表 $t_{i,k}$ 的完成时间,$tt(t_{i,j},t_{i,k})$代表 $t_{i,j}$ 将数据传输至 $t_{i,k}$ 的时间,当 $t_{i,k}$ 和 $t_{i,j}$ 分配的虚拟机相同时,传输时间为 0;否则,传输时间为 $L(d_{t_{i,k},t_{i,j}})$除以两个虚拟机之间的带宽 $b(vm_{t_{i,k}},vm_{t_{i,j}})$.

$t_{i,j}$ 进行安全服务所花费的时间 $st_{t_{i,j}}$ 分为身份验证服务、完整性服务以及机密性服务这 3 个部分,$st_{t_{i,j}}$ 被定义为

$$st_{t_{i,j}}=t(sl_{t_{i,j}}^{a})+t(sl_{t_{i,j}}^{b})+t(sl_{t_{i,j}}^{c}) \tag{6}$$

$t(sl_{t_{i,j}}^{a})$代表身份验证服务所消耗的时间,$t(sl_{t_{i,j}}^{b})$代表完整性服务所消耗的时间,$t(sl_{t_{i,j}}^{c})$代表机密性服务所消耗的时间.身份验证服务所消耗的时间仅与选择的安全服务级别有关,完整性服务和机密性服务所消耗的时间与选择的安全服务级别和保护的数据大小有关.故 $t(sl_{t_{i,j}}^{b})$的计算方法为 $I_{t_{i,j}}/OD(sl_{t_{i,j}}^{b})$,$t(sl_{t_{i,j}}^{c})$的计算方法为 $O_{t_{i,j}}/OD(sl_{t_{i,j}}^{c})$,$OD(sl_{t_{i,j}}^{b})$和 $OD(sl_{t_{i,j}}^{c})$分别代表完整性服务和机密性服务单位时间能处理的数据长度.

$t_{i,j}$ 的完成时间为开始时间加上运行时间以及安全服务所花费的时间,可利用如下公式计算:

$$ft(t_{i,j},vm_{t_{i,j}})=st(t_{i,j},vm_{t_{i,j}})+et(t_{i,j},vm_{t_{i,j}})+st_{t_{i,j}} \tag{7}$$

其中,$t_{i,j}$ 在 $vm_{t_{i,j}}$ 上的执行时间为 $et(t_{i,j},vm_{t_{i,j}})=L_{t_{i,j}}/m_{vm_{t_{i,j}}}$,$m_{vm_{t_{i,j}}}$ 代表 $vm_{t_{i,j}}$ 单位时间可以执行的指令数.多科学工作流最后一个任务结束的时间为整个多科学工作流的完成时间,故多科学工作流的完成时间 mp 为

$$mp=\max_{i\in(1,N),j\in(1,N_i)}\{ft(t_{i,j},vm_{t_{i,j}})\} \tag{8}$$

其中,N 代表科学工作流的个数,N_i 代表科学工作流 i 拥有的任务数.

1.4 多科学工作流费用计算

当前,大部分云平台采用的计费方式为按时间周期计费,例如 Amazon EC2 以小时为单位计费,运行不足一小时按一小时收费,虚拟机之间的数据传输是免费的.因此,多科学工作流执行的费用为各个虚拟机花费的费用之和.总费用 $allc$ 为

$$\left.\begin{aligned} allc &= \sum_{i=1}^{m} price(vm_i) \\ price(vm_i) &= \lceil ft(t_{end}, vm_i) - st(t_{start}, vm_i) \rceil \times P_{vm_i} \end{aligned}\right\} \tag{9}$$

其中,$price(vm_i)$代表虚拟机 vm_i 花费的费用,m 代表虚拟机的个数,t_{end} 代表在虚拟机 vm_i 上最后完成的任务,t_{start} 代表在 vm_i 上最早执行的任务, P_{vm_i} 代表 vm_i 单位时间的费用.

1.5 多科学工作流费用优化模型

本文提出的云环境下基于安全性和时间约束的多科学工作流费用优化模型即为寻找一种在满足多科学工作流的总完成时间低于用户预先设定的截止时间以及安全性大于用户设定的安全性下,最小化租赁费用的调度方案,形式化描述如公式(10):

$$\left.\begin{aligned} &\min allc \\ \text{s.t.}\quad &mp \leqslant deadline \\ &S(W) \geqslant scon \end{aligned}\right\} \tag{10}$$

其中,$deadline$ 代表用户设定的多科学工作流调度的截止时间,$scon$ 代表用户设定的多科学工作流调度的最低安全性.

2 云环境下基于安全性和时间约束的多科学工作流费用优化算法

本文提出的MSW-SDCOA算法的基本思想:首先对多个科学工作流进行压缩;然后,通过改进HEFT算法获得多科学工作流的任务调度序列;最后,通过优化的ACO算法获取任务序列的最优调度方案.

2.1 基于数据依赖的科学工作流压缩

在科学工作流中,每个任务与其前驱任务节点和后继任务节点都存在数据依赖关系,只有其前驱任务节点都运行完成,该任务节点才能运行,如果存在一个任务节点 $t_{i,j}$ 的前驱任务节点只有 $t_{i,k}$,并且 $t_{i,k}$ 的后继任务节点也只有 $t_{i,j}$,那么可以将 $t_{i,j}$ 和 $t_{i,k}$ 合并成一个任务节点.这既避免了 $t_{i,j}$ 和 $t_{i,k}$ 之间地数据传输,又减少了任务节点数,从而减少了任务调度时间以及调度费用.本文中考虑的为单个数据中心,没有考虑数据集隐私问题.

2.2 多科学工作流任务调度序列的生成

科学工作流的分层方法为,将同一深度的任务节点视为一层.本文的多科学工作流任务调度序列生成算法是通过改进 HEFT 算法实现的:首先计算任务节点的权重值;其次,将每一层上任务节点按权重值从低到高进行排序;最后,按层数从低到高将每层的序列进行合并,获得多科学工作流任务调度序列.这充分保证了多科学工作流调度的公平性.

对于每个任务节点权重值的计算,通过考虑 $t_{i,j}$ 到出口任务 t_{exit} 之间的路径长度 $Len(t_{i,j})$以及 $t_{i,j}$ 的安全性开销,使得任务节点的权重计算通过追求局部最优达到全局多目标优化调度.$t_{i,j}$ 的权重计算公式如下:

$$rank(t_{i,j}) = avg\{w(t_{i,j})\} + \max_{t_{i,k} \in succ(t_{i,j})}\{tt(t_{i,j}, t_{i,k}) + rank(t_{i,k}) + Len(t_{i,k})\} + \underset{l \in \{a,b,c\}}{avg}\{t(sl^{l}_{t_{i,j}})\} \tag{11}$$

当$t_{i,j}$是出口任务时$Len(t_{i,j})$=0,权重计算公式如下:

$$rank(t_{i,j}) = avg\{w(t_{i,j})\} + \underset{l \in \{a,b,c\}}{avg}\{t(sl^{l}_{t_{i,j}})\} \tag{12}$$

任务调度序列的生成算法如算法 1 所示.

算法 1. 任务调度序列生成算法.

输入:包含每个科学工作流任务节点信息的队列 *Mqueue*.

输出:工作流任务调度序列 *Q*.

1. get the max depth k //获取多科学工作流的最大层数 k
2. $q_1,q_2,q_3,\ldots,q_k$={};
3. Q={};

4. **for** each task *t* in MQueue **do**
5. **if** $t==t_{exit}$;
6. Calculate the *rank*(*t*) according to Eq.(12);
7. **else**
8. Calculate the *rank*(*t*) according to Eq.(11);
9. *j*==*t.getdepth*(); //获取任务 t 的层数
10. Put task *t* to q_j;
11. **end if**
12. **end for**
13. **for** *i* to *k* **do**
14. *QuickSort*(q_i,1,q_i.*size*()); //将队列中的任务按 *rank*(*t*)的大小递减排序
15. *Q*+=q_i;
16. **end for**
17. **return** *Q*

算法的具体执行过程如下:

(1) 获取多科学工作流的最大层数 *k*,然后初始化 *k* 个空队列以及初始化工作流任务调度序列 *Q* 为空(第 1 行~第 3 行);

(2) 依次读取包含每个科学工作流任务节点信息的队列 *MQueue* 的任务 *t*,判断任务 *t* 是否为出口任务:如果是,则按公式(12)计算该任务的权重值;否则,用公式(11)计算该任务的权重值,然后读取该任务节点所在的层数 *j*,然后将任务 *t* 添加至队列 q_j(第 4 行~第 12 行);

(3) 依次将 *k* 个队列中的任务按权重值递增排序,并将队列添加至 *Q* 中(第 13 行~第 16 行);

(4) 返回工作流任务调度序列 *Q*(第 17 行).

2.3 多科学工作流任务序列的调度

本文中,任务调度算法基于 ACO 算法,该算法是元启发式算法,具有很强的跨领域能力,可以用在科学工作流的调度上.将 ACO 算法用于求解多科学工作流调度问题,需要考虑的算法的求解效率以及调度结果的好坏.

将任务 t_i 分配至每个虚拟机的概率公式如下:

$$\left.\begin{aligned} p_{t_i,vm_j}&=\frac{[\tau_{t_i,vm_j}]^{\alpha}[\eta_{t_i,vm_j}]^{\beta}}{\sum[\tau_{t_i,vm_s}]^{\alpha}[\eta_{t_i,vm_s}]^{\beta}},(vm_j,vm_s)\in allowed_v\\ \eta_{t_i,vm_j}&=1/(et(t_i,vm_j)+st(t_i,vm_j))\end{aligned}\right\} \tag{13}$$

其中,τ为信息素矩阵,η是启发性信息,τ_{t_i,vm_j} 为由任务 t_i 到虚拟机 vm_j 的路径信息素强度,p_{t_i,vm_j} 为蚂蚁将任务 t_i 分配到虚拟机 vm_j 的概率,$allowed_v$ 代表能够使用的虚拟机资源,α为路径权,β为启发性信息的权.

计算出任务 t_i 分配至每个虚拟机的概率之后,通过轮盘赌的方法决定任务 t_i 分配至哪个虚拟机.

为了提高调度的效率与结果,从以下两个方面对 ACO 算法进行优化.

1. 信息素更新策略的优化

信息素的扩散速度应随着蚂蚁迭代次数的增加也随之增加,改进后,计算方式如下:

$$\rho^{*}=[1-e^{\varphi\times(n-N)/N}]\times\rho \tag{14}$$

其中,*n* 是当前蚂蚁的迭代次数;*N* 是设定的迭代次数;ρ是预设定的信息素数量的蒸发系数;φ是常数,代表信息素的总体扩散速度,取值范围为[1,5].

在蚂蚁寻找最优解的过程中,避免出现过早收敛的情况.信息素存在两个阈值$\sigma_{\max}$ 和$\sigma_{\min}$,当信息素 τ_{t_i,vm_j} 大于阈值$\sigma_{\max}$ 时,应相应地减少信息素,防止其过大;当信息素 τ_{t_i,vm_j} 小于阈值$\sigma_{\min}$ 时,应相应地增加信息素,防止其过小.本文的优化方法如下:

$$\tau_{t_i,vm_j}(n+1)=\begin{cases}\rho^{*e^{-0.5}}\times\tau_{t_i,vm_j}(n)+\Delta\tau_{t_i,vm_j}, \text{ if } \tau_{t_i,vm_j}(n)>\sigma_{\max}\\ \rho^{*e^{0.5}}\times\tau_{t_i,vm_j}(n)+\Delta\tau_{t_i,vm_j}, \text{ if } \tau_{t_i,vm_j}(n)<\sigma_{\min}\end{cases} \tag{15}$$

如果在本次迭代中发现最优解,则发现最优解的蚂蚁在和其他蚂蚁一样更新完自己路径上的信息素后,还要再进行一次更新,所有蚂蚁更新公式如下:

$$\left.\begin{aligned}\tau_{t_i,vm_j}(n+1)&=\rho^{*}\times\tau_{t_i,vm_j}(n)+\Delta\tau_{t_i,vm_j}\\ \Delta\tau_{t_i,vm_j}&=\begin{cases}\sum_{k=1}^{m}\Delta\tau_{t_i,vm_j}^{k}+\chi\times\Delta\tau_{t_i,vm_j}^{bs}, \text{ if the Ant find the } bs\\ \sum_{k=1}^{m}\Delta\tau_{t_i,vm_j}^{k}, \qquad \text{others}\end{cases}\\ \Delta\tau_{t_i,vm_j}^{bs}&=\begin{cases}1/bscost, \text{ if } bs \text{ } contains(t_i,vm_j)\\ 0, \qquad \text{others}\end{cases}\\ \Delta\tau_{t_i,vm_j}^{k}&=\frac{q}{\sum L_k}\end{aligned}\right\} \tag{16}$$

其中,m 为蚂蚁个数;n 为迭代次数;$\Delta\tau_{t_i,vm_j}^{k}$ 为蚂蚁 k 由任务节点 t_i 到虚拟机 vm_j 的路径上留下的信息素数量;ρ为路径上信息素数量的蒸发系数;q 为信息素质量系数;χ是一个随机数,在[0,1]中进行选取;$\sum_{k=1}^{m}\Delta\tau_{t_i,vm_j}^{k}$ 是所有蚂蚁共同更新的信息素;bs 为当前最优解;$bscost$ 为当前最优解的费用值;$\sum L_k$ 为蚂蚁 k 的调度方案费用值.

2. 启发性信息的优化

为了增大效率高的虚拟机被选到的概率,修改启发性信息 η_{t_i,vm_j} 如下:

$$\left.\begin{aligned}\gamma&=\frac{(m_{vm_j}/P_{vm_j})\times m}{\sum_{n=1}^{m}m_{vm_j}/P_{vm_j}}\\ \eta_{t_i,vm_j}&=\gamma/(et(t_i,vm_j)+st(t_i,vm_j))\end{aligned}\right\} \tag{17}$$

其中,γ代表 vm_j 在所有虚拟机中的效率值.

多科学动态动态调度策略为:若用户在提交的科学工作流未执行完成的情况下提交了新的科学工作流,需给出新的调度参数,首先对新的科学工作流进行压缩,其次计算新的科学工作流每个任务节点的权重值;未执行完成的科学工作流剩下的节点层数从第 1 层开始计数,然后按层数从低到高将新的科学工作流每一层任务节点依据权重值递增排序的规则合并到还未执行完成的科学工作流所在的每一层任务队列中,生成新的调度序列,最后,利用优化的 ACO 算法进行调度.这样可以避免新的科学工作流因之前科学工作流剩余任务还未执行而得不到调度最终导致执行时间跨度变大的问题.

MSW-SDCOA 具体算法如算法 2 所示.

算法 2. MSW-SDCOA 算法.

输入:$W=\{w_1,w_2,w_3,\ldots,w_N\}$; //科学工作流列表

$VM=\{vm_1,vm_2,vm_3,\ldots,vm_m\}$. //云环境下虚拟机列表

输出:*bestschedule*. //安全性和时间约束下的最优调度方案

1. Workflow compress
2. 通过任务调度序列生成算法生成队列 Q
3. 初始化信息素矩阵τ,初始化蚁群 Ants
4. **for** n to *maxgen* **do** //蚁群开始进行迭代,*maxgen* 是最大迭代数
5. **for** Ant in Ants **do**
6. Choose the *VM* for each task t in queue Q;

7. Calculate the Ant's $S(W)$ according to Eq.(4); //计算安全性
8. Calculate the Ant's *makespan* according to Eq.(8); //计算完成时间
9. Calculate the Ant's *allc* according to Eq.(9); //计算总费用值
10. **if** $S(W)\geqslant scon$ && $makespan\leqslant deadline$ && $allc<bestschedule.allc$
11. *bestschedule=ant.sehedule*;
12. **else**
13. continue;
14. **end if**
15. Update the τ;
16. **end for**
17. **end for**
18. **if** *bestschedule*==null
19. **return** "please modify the scheduling parameters";
20. **else**
21. **return** *bestschedule*;

算法的具体执行过程如下.

(1) 对多科学工作流进行压缩(第 1 行).

(2) 对压缩后的多科学工作流调用任务调度序列生成算法生成 Q(第 2 行).

(3) 为 ACO 算法的运行做初始化工作(第 3 行).

(4) 开始进行迭代,计算每一次迭代中每只蚂蚁调度方案的安全性、运行时间和费用,在安全性和时间约束下判断是否为最优调度方案:如果是,则保留最优方案,每一次迭代过后需要更新信息素矩阵(第 4 行~第 17 行).

(5) 迭代完成后,如果最优调度方案为空,则说明未找到符合安全和时间约束下的调度方案,提醒用户修改调度参数;否则,返回安全和时间约束下的最优调度方案(第 18 行~第 21 行).

3 仿真实验

为了分析和评估本文提出的 MSW-SDCOA 算法的性能,将 MSW-SDCOA 算法与 HEFT-ACO(HEFT 算法加上传统 ACO 算法)、MW-DBS 算法[27]和 CCRH 算法[6]进行对比,本文在 CloudSim[28]平台上进行仿真调度实验.为了能进行公平地对比,HEFT-ACO 和 MW-DBS 算法的安全性计算均采用本文所述的方法进行计算.实验基于以下环境:处理器为 Inter(R) Core(TM)i5-4590,3.30GHz;8GB 内存;Windows 7,64 位操作系统.

3.1 实验环境

本文不考虑数据中心问题,实验采用一个数据中心、10 个不同类型的虚拟机,假设每个虚拟机都拥有 3 种安全性服务,各个虚拟机参数信息见表 2.

Table 2 Virtual machine parameter information

表 2 虚拟机参数信息

编号	MIPS(Hz)	带宽(Mbps)	单价($/h)	编号	MIPS(Hz)	带宽(Mbps)	单价($/h)
1	531	1 200	1.59	6	890	1 334	2.67
2	1 550	785	4.65	7	1 223	1 350	3.67
3	985	980	2.95	8	1 079	1 446	3.24
4	605	558	1.81	9	1 237	980	3.71
5	1 345	899	4.03	10	786	1 043	2.36

实验所用的科学工作流为 Montage,LIGO 和 Epigenomics,通过采用 WorkflowGenerator 生成器[29],生成任务数为 25,50,100 的 Montage 工作流、任务数为 30,50,100 的 LIGO 工作流和任务数为 24,46,100 的 Epigenomics

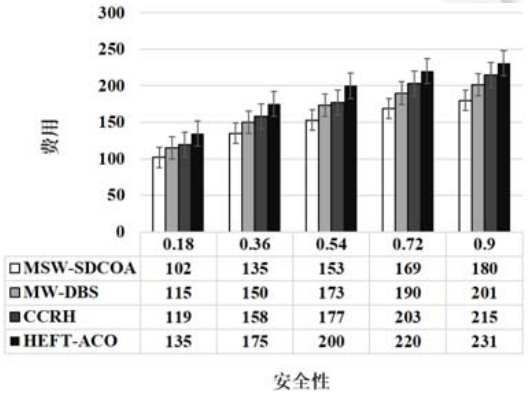
工作流.实验中,多科学工作流组合设定是用户同时提交的多个科学工作流.

对于 3 种安全性服务,实验使用 HMAC-MD5,HMAC-SHA-1 和 CBC-MAC-AES 这 3 种认证方式实现身份服务,使用 7 个哈希函数(MD4,MD5,RIPEMD,RIPEMD-128,SHA-1,RIPEMD-160 和 Tiger)实现完整性服务,使用 8 种加密算法(SEAL,RC4,Blowfish,Knufu/Khafre,RC5,Rijndael,DES 和 IDEA)实现机密性服务[25].3 种安全服务的风险系数参数设定为λ^a=3.5,λ^b=2.5,λ^c=1.5,参数设定参考文献[30,31].

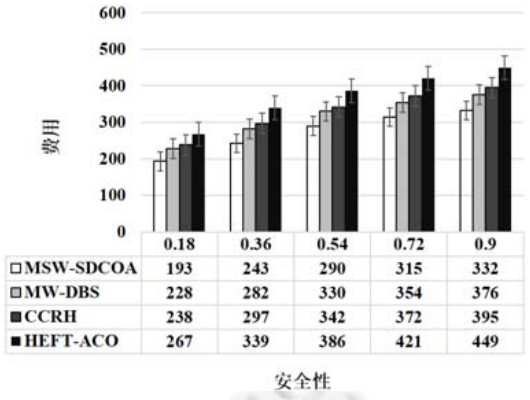
对于优化的 ACO 算法,实验中定义参数ρ=0.65,α=1.1,β=4.0,q=700,φ=3,蚁群数量为 60,迭代次数为 200,参数设定参考文献[30,31].

3.2 实验结果对比

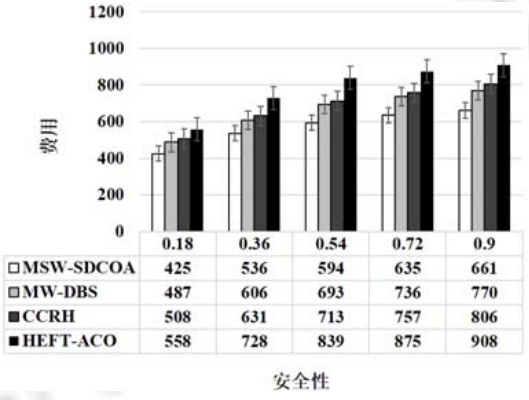
图 3 显示了在截止时间相同的情况下,对比在不同安全性约束下,将 MSW-SDCOA 算法的调度结果与 HEFT-ACO 算法、MW-DBS 算法、CCRH 算法的调度结果进行对比.实验中分别采用了(LIGO-30,Montage-25,Epigenomics-24),(LIGO-50,Montage-50,Epigenomics-46)以及(LIGO-100,Montage-100,Epigenomics-100)这 3 种多科学工作流实例,MSW-SDCOA 算法、HEFT-ACO 算法、MW-DBS 算法和 CCRH 算法在多种不同的多科学工作流实例进行比较,保证实验的准确性.实验采取的截止时间为(t_{min}+t_{max})/2,t_{min} 和 t_{max} 分别代表多科学工作流在虚拟机配置条件不变且没有安全性约束下的最小完成时间和最大完成时间,不同组合的多科学工作流截止时间不同.从图 3 可以看出,对于不同组合的多科学工作流,随着安全性约束的提高,4 种调度算法的费用也在增大.这是因为随着安全级别的提升,任务节点在虚拟机上运行的时间会逐渐增加.



(a) LIGO-30, Montage-25, Epigenomics-24



(b) LIGO-50, Montage-50, Epigenomics-46

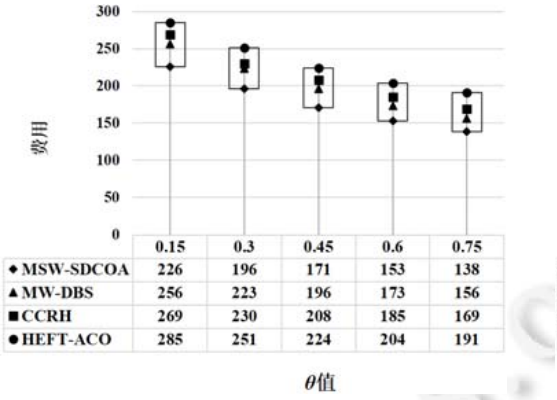


(c) LIGO-100, Montage-100, Epigenomics-100

Fig.3 Scheduling results of different multi-scientific workflows under different security constraints

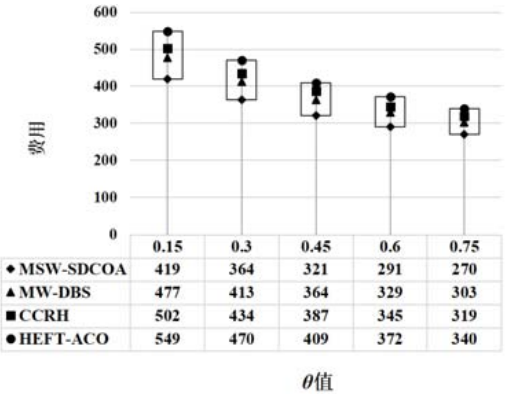
图 3 不同安全性约束下,不同多科学工作流组合调度结果

图 4 显示了在安全性相同的情况下,对比在不同截止时间下,将 MSW-SDCOA 算法的调度结果与 HEFT-ACO 算法、MW-DBS 算法、CCRH 算法的调度结果进行对比.实验中分别采用了(LIGO-30,Montage-25,Epigenomics-24),(LIGO-50,Montage-50,Epigenomics-46)以及(LIGO-100,Montage-100,Epigenomics-100)这 3 种多科学工作流实例.实验中,安全性值设为 0.7,选取最小完成时间和最大完成时间区间中的 5 个数值作为用户截止时间,即 $deadline=t_{min}+\theta\times(t_{max}-t_{min})$,其中,$\theta\in\{0.15,0.3,0.45,0.6,0.75\}$.从图 4 可以看出,随着用户截止时间的提高,4 种调度算法的费用在降低.这是因为随着截止时间的提高,符合安全性和时间约束的调度方案逐渐增多.



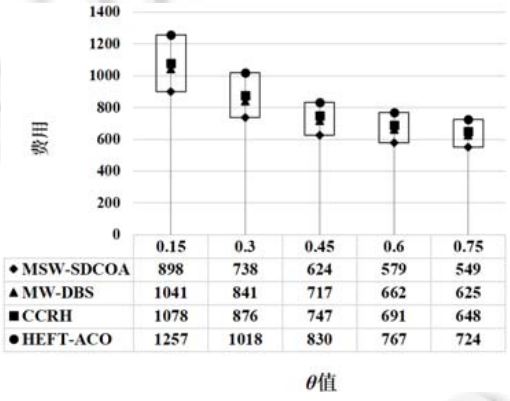
	0.15	0.3	0.45	0.6	0.75
◆MSW-SDCOA	226	196	171	153	138
▲MW-DBS	256	223	196	173	156
■CCRH	269	230	208	185	169
●HEFT-ACO	285	251	224	204	191

(a) LIGO-30, Montage-25, Epigenomics-24



	0.15	0.3	0.45	0.6	0.75
◆MSW-SDCOA	419	364	321	291	270
▲MW-DBS	477	413	364	329	303
■CCRH	502	434	387	345	319
●HEFT-ACO	549	470	409	372	340

(b) LIGO-50, Montage-50, Epigenomics-46



	0.15	0.3	0.45	0.6	0.75
◆MSW-SDCOA	898	738	624	579	549
▲MW-DBS	1041	841	717	662	625
■CCRH	1078	876	747	691	648
●HEFT-ACO	1257	1018	830	767	724

(c) LIGO-100, Montage-100, Epigenomics-100

Fig.4 Scheduling results of different multi-scientific workflows under different deadline constraints

图 4 不同时间约束下,不同多科学工作流组合调度结果

在费用优化方面,从图 3 和图 4 可以看出,MSW-SDCOA 算法所生成的调度方案费用值最低.这是由于 MSW-SDCOA 算法中的任务调度序列生成算法考虑了全局多目标优化,从而生成更加合理的调度序列,并且在信息素的优化中,对于信息素数量的蒸发系数的改进、增加信息素的上下阈值以及发现最优调度方案时的处理策略,避免了 ACO 算法出现过早收敛的情况,以至于增强了 ACO 算法的全局搜索能力,最终能够得到更好的调度结果.

图 5 显示了在安全性和截止时间约束相同的情况下,对比在同时到达的不同多科学工作流个数下的完成时间大小.从图 5 中可以看出,随着多科学工作流个数的增加,各算法运行的时间也增加,但是本文方法的耗时时间低于 HEFT-ACO 算法、MW-DBA 算法和 CCRH 算法.这是由于在初始阶段进行了工作流压缩,使得任务数减少,并且避免了部分任务之间的数据传输,在寻找最优调度方案中,对启发式信息的优化,使效率高的虚拟机被

选到的概率增大,从而减少多科学工作流调度时间.

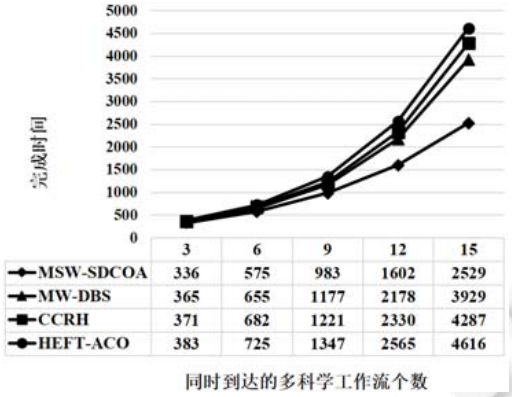


Fig.5 Scheduling time of the number of different scientific workflows

图 5 不同多科学工作流个数下调度完成时间

图 6 显示了 4 种算法对于图 3 和图 4 中 6 个实验的云资源平均利用率,从图中可以看出,本文算法对于云资源的利用率高于 HEFT-ACO 算法、MW-DBA 算法和 CCRH 算法,这体现出在相同的云资源条件下,MSW-SDCOA 算法能够有更好的收益.

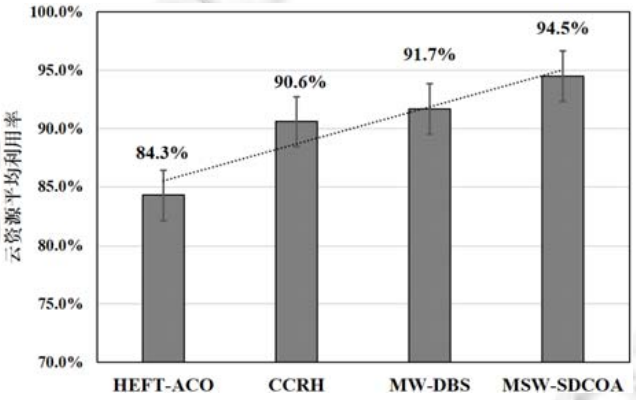


Fig.6 Average utilization of cloud resources

图 6 云资源平均利用率

4 结束语

针对现有多科学工作流费用优化模型未考虑安全调度问题,本文设计了带有安全性和时间约束多科学工作流费用优化模型,并提出了针对该模型的费用优化算法 MSW-SDCOA.MSW-SDCOA 利用数据依赖关系压缩科学工作流,减少了费用开销,并通过改进 HEFT 算法形成调度序列以及优化 ACO 中信息素更新策略和启发式信息,进一步改善费用优化效果.实验中使用了 3 种不同领域下的真实科学工作流,与 HEFT-ACO 和 MW-DBA 算法进行比较,证明了 MSW-SDCOA 算法的有效性.在未来的研究工作中,将考虑多个用户同时提交多个科学工作流的情况以及资源负载均衡对科学工作流调度的影响.

References:

[1] Deelman E, Singh G, Livny M, Berriman B. The cost of doing science on the cloud: The montage example. In: Proc. of the 2008 ACM/IEEE Conf. on Supercomputing. Piscataway: IEEE, 2008. 1–12.

[2] Kannas CC, Kalvari I, Lambrinidis G, Neophytou CM, Savva CG, Kirmitzoglou I, Antoniou Z, Achilleos KG, Scherf D, Pitta CA. LiSIs: An online scientific workflow system for virtual screening. Combinatorial Chemistry & High Throughput Screening, 2015, 18(3):281−295.

[3] Bennett JC, Bhagatwala A, Chen JH, Seshadhri C, Pinar A, Salloum M. Trigger detection for adaptive scientific workflows using percentile sampling. SIAM Journal on Scientific Computing, 2016,38(5):S240−S263.

[4] Pradal C, Artzet S, Chopard J, Dupuis D, Fournier C, Mielewczik M, Nègre V, Neveu P, Parigot D, Valduriez P. InfraPhenoGrid: A scientific workflow infrastructure for plant phenomics on the grid. Future Generation Computer Systems, 2017,67:341−353.

[5] Zhao Y, Li Y, Raicu I, Lu S, Lin C, Zhang Y, Tian W, Xue R. A service framework for scientific workflow management in the cloud. IEEE Trans. on Services Computing, 2015,8(6):930−944.

[6] Wang Y, Jia C, Xu Y. Multiple dags dynamic workflow scheduling based on the primary backup algorithm in cloud computing system. In: Proc. of the Ninth Int'l Conf. on Broadband and Wireless Computing, Communication and Applications. Piscataway: IEEE, 2014. 177−182.

[7] Tischer A, Auton M. Scheduling strategies for mapping application workflows onto the grid. In: Proc. of the 14th IEEE Int'l Symp. on High Performance Distributed Computing. Piscataway: IEEE, 2005. 125−134.

[8] Zhao H, Sakellariou R. Scheduling multiple DAGs onto heterogeneous systems. In: Proc. of the 2006 Parallel and Distributed Processing Symp. Washington: IEEE, 2006. 1−14.

[9] Rodriguez MA, Buyy R. Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. IEEE Trans. on Cloud Computing, 2014,2(2):222−235.

[10] Weins K. Cloud computing trends: 2017 state of the cloud survey. 2017. https://www.rightscale.com/blog/cloud-industry-insights/cloud-computing-trends-2017-state-cloud-survey

[11] Yurcik W, Koenig GA, Meng X, Greenseid J. Cluster security as a unique problem with emergent properties: Issues and techniques. In: Proc. of the 5th Lci Int'l Conf. on Linux Clusters: The Hpc Revolution. 2004. 18−20.

[12] Arabnejad H, Barbosa JG, Prodan R. Low-Time complexity budget-deadline constrained workflow scheduling on heterogeneous resources. Future Generation Computer Systems, 2016,55:29−40.

[13] Lin B, Guo W, Chen G, Xiong N, Li R. Cost-Driven scheduling for deadline-constrained workflow on multi-clouds. In: Proc. of the 2015 IEEE Int'l Parallel and Distributed Processing Symp. Workshop. Piscataway: IEEE, 2015. 1191−1198.

[14] Meena J, Kumar M, Vardhan M. Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint. IEEE Access, 2016,4:5065−5082.

[15] Singh L, Singh S. Deadline and cost based ant colony optimization algorithm for scheduling workflow applications in hybrid cloud. Journal of Scientific & Engineering Research, 2014,5(10):1417−1420.

[16] Verma A, Kaushal S. A hybrid multi-objective particle swarm optimization for scientific workflow scheduling. Parallel Computing, 2017,62:1−19.

[17] Zissis D, Lekkas D. Addressing cloud computing security issues. Future Generation Computer Systems, 2012,28(3):583−592.

[18] Huang Q, Yang Y, Shen M. Secure and efficient data collaboration with hierarchical attribute-based encryption in cloud computing. Future Generation Computer Systems, 2017,72:239−249.

[19] Sookhak M, Yu FR, Khan MK, Xiang Y, Buyya R. Attribute-Based data access control in mobile cloud computing: Taxonomy and open issues. Future Generation Computer Systems, 2017,72:273−287.

[20] Ali M, Khan SU, Vasilakos AV. Security in cloud computing: Opportunities and challenges. Information Sciences, 2015,305: 357−383.

[21] Sharif S, Taheri J, Zomaya AY, Nepal S. Mphc: Preserving privacy for workflow execution in hybrid clouds. In: Proc. of the 2013 Int'l Conf. on Parallel and Distributed Computing, Applications and Technologies. Piscataway: IEEE, 2013. 272−280.

[22] Zeng L, Veeravalli B, Li X. SABA: A security-aware and budget-aware workflow scheduling strategy in clouds. Journal of Parallel and Distributed Computing, 2015,75:141−151.

[23] Bala A, Chana I. Intelligent failure prediction models for scientific workflows. Expert Systems with Applications, 2015,42(3): 980−989.

[24] Topcuoglu H, Hariri S, Wu M. Performance-Effective and low-complexity task scheduling for heterogeneous computing. IEEE Trans. on Parallel and Distributed Systems, 2002,13(3):260−274.

[25] Xie T, Qin X. Scheduling security-critical real-time applications on clusters. IEEE Trans. on Computers, 2006,55(7):864−879.

[26] Tang XY, Li K, Zeng Z, Veeravalli B. A novel security-driven scheduling algorithm for precedence-constrained tasks in heterogeneous distributed systems. IEEE Trans. on Computers, 2011,60(7):1017−1029.

[27] Arabnejad H, Barbosa JG. Multi-Workflow QoS-constrained scheduling for utility computing. In: Proc. of the 2015 IEEE 18th Int'l Conf. on Computational Science and Engineering. Piscataway: IEEE, 2015. 137−144.

[28] Calheiros RN, Ranjan R, Beloglazov A, Rose CAFD, Buyya R. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, 2011,41(1):23−50.

[29] Bharathi S, Chervenak A, Deelman E, Mehta G, Su MH, Vahi K. Characterization of scientific workflows. In: Proc. of the 3rd Workshop on Workflows in Support of Large-Scale Science. Piscataway: IEEE, 2008. 1−10.

[30] Vinay VP, Sridharn R. Taguchi method for parameter design in ACO algorithm for distribution-allocation in a two-stage supply chain. Int'l Journal of Advanced Manufacturing Technology, 2013,64(9-12):1333−1343.

[31] Siemiński A. Ant colony optimization parameter evaluation. Multimedia and Internet Systems: Theory and Practice, 2013,183: 143−153.

袁友伟(1966－),男,湖北潜江人,博士,教授,CCF 专业会员,主要研究领域为云计算,工作流调度.

俞东进(1969－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为软件工程理论和方法,业务过程管理,行业大数据.

鲍泽前(1994－),男,硕士生,主要研究领域为工作流调度,大数据分析.

李万清(1979－),男,博士,副教授,主要研究领域为大数据分析.