计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.210800154

云中满足截止时间约束且优化成本的工作流调度策略

王子健[1] 卢政昊[1,2] 潘纪奎[1,2] 孙福权[1]
1 东北大学秦皇岛分校数学与统计学院 河北 秦皇岛 066000
2 东北大学信息科学与工程学院 沈阳 110000
(stxywzj@163.com)

摘 要 云环境中的工作流调度是如今最具挑战性的问题之一。它关注于在指定的服务质量需求下,以将相互依赖的任务映射到虚拟机的方式来执行工作流应用程序。云服务提供商以不同的价格提供不同性能的虚拟机。同样的工作流,配置不同的虚拟机,会产生不同的完工时间以及货币成本。云中工作流调度的主要问题之一是在满足用户给定的截止时间约束的前提下,找到一种更廉价的调度方法。为解决上述问题提出了一种云环境中满足截止时间约束且优化成本的工作流调度策略 DCCO。它基于 δ-alap 对截止时间进行分配,并考虑了两个任务可能被分配到同一虚拟机的情况。实验结果表明,相比于其他经典调度算法,在不同类型工作流测试下,DCCO 具有最高的成功率,且满足截止时间约束,同时可以优化执行成本。

关键词:云环境;工作流调度;截止日期;成本;优化
中图法分类号 TP393

Workflow Scheduling Strategy for Deadline Constrained and Cost Optimization in Cloud

WANG Zi-jian[1], LU Zheng-hao[1,2], PAN Ji-kui[1,2] and SUN Fu-quan[1]
1 School of Mathematics and Statistics, Northeastern University at Qinhuangdao, Qinhuangdao, Hebei 066000, China
2 School of Information Science and Engineering, Northeastern University, Shenyang 110000, China

Abstract Workflow scheduling in cloud is one of the most challenging issues today. It focuses on executing workflow applications with interdependent tasks mapped to virtual machines under specified quality of service requirements. Cloud service providers offer virtual machines with different performances at different prices. The same workflow with different virtual machines can result in different makespan and cost. One of the main problems of workflow scheduling in cloud is to find a cheaper scheduling method on the premise of meeting the deadline. The proposed deadline constrained cost optimization algorithm for workflow scheduling in cloud DCCO can solve the above problems. It assigns deadlines based on δ-alap and also considers cases where two tasks may be assigned to the same virtual machine. Experiments show that compared with other classical scheduling algorithms, DCCO has the highest success rate under different types of workflow tests, meets the deadline constraint, and can optimize the execution cost.

Keywords Cloud, Workflow scheduling, Deadline, Cost, Optimization

1 引言

工作流经常被用于生物信息学、天文学和物理学等大规模建模科学问题[1]。这样的工作流对数据和计算的需求不断增长,因此需要一个高性能的计算环境,以便在合理的时间内执行工作流[2]。多年来,人们对网格和集群等环境中的工作流调度进行了广泛的研究[3]。然而,随着云计算的出现,人们需要开发新的方法来应对云中的工作流调度[4]。如今,工作流调度是云计算中一个被广泛研究的主题[5],它可以大大提高云计算的整体性能[6]。优化工作流的关键是对工作流中任务的调度,这是一个 NP 难问题[7]。在云中,服务提供商以不同的价格提供不同性能的资源[8],资源配置不足将不可避免地损害服务性能,而资源配置过多可能会增加不必要的成本[9]。而且通常来说,性能较好的资源比性能较差的资源运行速度更快,完工时间更短,但是价格也更昂贵。因此,同样的工作流,配置不同的资源,会产生不同的完工时间以及成本。那么,对于特定的工作流应用程序,在满足用户给定的截止时间约束的前提下,如何优化执行成本,是目前面临的重要问题。

本文提出了一种云环境中满足截止时间约束且优化成本的工作流调度策略 DCCO。该策略由分配截止时间、任务排序、虚拟机选择 3 个阶段组成。通过三阶段式的任务调度过程,有效地控制了工作流的完成时间并优化了成本。实验结果表明,DCCO 具有最高的成功率,且满足截止时间约束,同时可以优化执行成本。

2 相关研究

云中的工作流调度是 NP 难问题,为了解决这一问题,Zhu 等[10]提出了一种两步工作流调度算法,该算法可以在确保满足工作流的截止时间约束下,降低任务执行成本。Saeid

基金项目:国家重点研发计划(2018YFB1402800)
This work was supported by the National Key R&D Program of China(2018YFB1402800).
通信作者:孙福权(m17806286850@163.com)

等[11]基于 PCP 算法，针对费用优先的原则提出了 IC-PCP 调度算法，但是该算法没有为部分关键路径的任务分配子截止时间，使得最终得到的解还有进一步的改进空间。Verma 等[12]提出了 BCHGA 算法，以在预算约束下优化执行成本和数据传输成本。Jian 等[13]以最大程度地减少截止时间约束下的工作流调度成本并平衡资源为目的，同时考虑了任务执行的时间成本和不同任务之间数据传输的时间成本。Bilgaiyan 等[14]提出的基于 CSO 的解决方案，该方案减少了迭代次数，并在可用资源上提供了合理的负载平衡。Zhou 等[15]提出了一种基于 ACO 的解决方案，以减少在云中找到计算资源的时间。Gogulan 等[16]提出的多信息素算法可以减少完工时间和资源利用率。Wu 等[17]提出了优先影响调度算法，该算法的重心为工作流优先级和任务权重，能够显著提高调度成功率，并防止对不太重要的工作流的支出。Zhang[18]提出了截止时间及成本约束的云工作流实时响应调度算法 PWHEFT 和基于任务完成延迟的动态云工作流调度策略。Gao[19]提出了单个工作流的成本优化方案 WMFCO 和多个工作流的成本优化映射方案 MWMFCO 。Yu 等[20]提出了一种满足截止时间约束的工作流调度代价最优化遗传算法 CODC-GA。Chen 等[21]针对异构云环境下科学工作流调度的代价优化问题，提出了一种基于约束关键路径的代价优化调度算法。

综合以上分析可以看出，云中工作流调度如果无法满足用户给定的截止时间约束，将会导致成本过高。本文提出的满足截止时间约束下的云中工作流成本优化方法 DCCO，基于 δ-alap 对截止时间进行分配，并考虑了两个任务可能被分配到同一虚拟机的情况。

3 问题描述

3.1 工作流模型

一个工作流可以用有向无环图 $DAG=(V,E)$ 来表示。其中 $V=\{t_1,t_2,\cdots,t_n\}$ 是有向无环图中节点的集合，每个节点都代表一个任务，一个任务也可以被理解成一段独立的且不能被并行执行的程序。对于指定的任务 t_i，用 w_i 代表该任务的计算量。E 是有向无环图中边的集合，E 中的一条边 $e_{ij}=\{t_i,t_j\}$ 代表了任务 t_i 和 t_j 之间的一种依赖关系，即任务 t_j 只能在任务 t_i 执行结束后才能开始执行。此时 t_i 和 t_j 分别被称为父任务和子任务；若一个任务没有父任务，则称之为起始任务；若一个任务没有子任务则称之为终止任务。如果任务之间存在数据传输，则需要为 e_{ij} 添加一个属性 $data_{ij}$ 来代表由 t_i 到 t_j 的数据传输量。此时任务 t_j 只能在 t_i 执行结束且完成数据传输后才开始执行。有些工作流可能同时拥有多个起始节点或终止节点，大多数调度算法并不适用于这种情况。为了一般化工作流模型，可以在工作流的开始处和结束处分别添加两个执行时间为零且不传输任何数据的虚拟任务 t_{entry} 和 t_{exit}。图 1 给出了一个工作流示例。

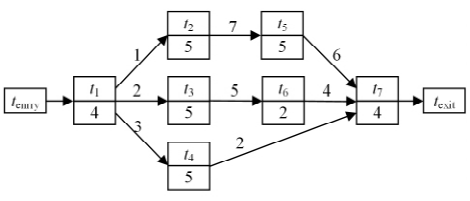


图 1 工作流示例图

Fig. 1 Example of workflow

3.2 云资源模型

云提供者通过虚拟化技术将云平台中的各类资源虚拟化为资源池，以便进行统一的管理并为云使用者提供服务。用 $R=\{r_1,r_2,\cdots,r_m\}$ 代表云提供者可提供的服务类型的集合。不同类型的服务配置了不同的计算和存储资源，因此具有不同的计算能力和使用成本。用 $I=\{vm_1,vm_2,\cdots,vm_n\}$ 代表具体虚拟机实例。虚拟机是工作流中任务的执行平台。工作流中的一个任务只能被调度到唯一的虚拟机上执行，而一个虚拟机可以按顺序执行一组任务。I 中的每一个虚拟机都关联了 R 中的一个特定的服务类型。虚拟机需要通过租用的方式使用，本文采用的定价方式为即付即用模式，即根据虚拟机被租用的单元时间数量来进行收费。当租用的虚拟机没有使用完一个完整的单元时间时，也会按照一个单元时长的费用对其进行收费。如图 2 所示，虚拟机的单元时长设置为 T，虚拟机在 0 时刻被租赁，3.2T 时刻被释放，那么用户需支付 4T 时间的开销。对于服务类型为 r_k 的虚拟机 vm_l，我们使用 c_l 和 v_l 分别表示其在每个时间间隔 T 的租用成本和它的计算能力。

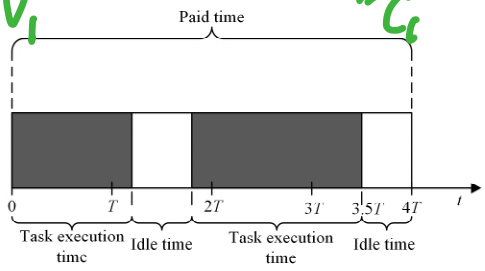


图 2 付费时间段示例图

Fig. 2 Example of a paid time period

3.3 调度模型

调度的目的是将工作流中的任务分配到具体的服务中并按照既定的顺序执行。一次调度的结果可以被描述为〈I, M〉。I 是用来执行工作流中任务的虚拟机的集合，其含义在 3.2 节已有描述。M 是任务到虚拟机的映射集合，每一个映射可使用 $m=\langle t_i,vm_l,ST_{il},FT_{il}\rangle$ 进行描述，其含义为任务 t_i 被分配到虚拟机 vm_l 中执行，在 ST_{il} 时间执行开始，在 FT_{il} 时间执行结束。当工作流中的任务均已调度完毕，此时可计算虚拟机 vm_l 的租用开始时间 LST_l 和租用结束时间 LFT_l。租用开始时间表示该虚拟机上第一个任务开始接收数据的时间，租用结束时间表示该虚拟机上最后一个任务执行完毕且完成数据传输的时间。

服务类型为 r_k 的虚拟机 vm_l 从 LST_l（租用开始时间）租赁到 LFT_l（租用结束时间）的成本如式(1)所示：

$$EC_l=\lceil(LFT_l-LST_l)/T\rceil\times c_l \tag{1}$$

当任务 t_i（计算量为 w_i）分配给虚拟机 vm_l 时，执行时间如式(2)所示：

$$ET_i=w_i/v_l \tag{2}$$

本文假设所有虚拟机都位于相同的物理区域，因此虚拟机之间的平均带宽（bw）大致相等，且内部数据传输是免费的。因此，依赖项 $e_{i,j}$ 的数据传输时间（$TT_{i,j}$）的计算方法如式(3)所示，且当两个任务在同一个虚拟机上执行时，$TT_{i,j}$ 将变为零：

$$TT_{i,j}=\begin{cases}data_{i,j}/bw, & \text{if } vm_i\neq vm_j\\ 0, & \text{otherwise}\end{cases} \tag{3}$$

虚拟机 vm_l 可以开始执行任务 t_i 的最早时间为 RT_l，则在 vm_l 上任务 t_i 的开始时间 $ST_{i,l}$ 和结束时间 $FT_{i,l}$ 的计算式如下：

$$ST_{i,l}=\max\{RT_l,\max_{t_j\in t_i's\ parents}\{FT_{j,?}+TT_{j,i}\}\} \tag{4}$$

$$FT_{i,l}=ST_{i,l}+ET_{i,l} \tag{5}$$

整个工作流的执行时间 $makespan(I,M)$，可以通过式(6)计算得到：

$$makespan(R,M)=\max_{i\in T}\{FT_i\} \tag{6}$$

同时，其总成本 $cost(I,M)$ 如式(7)所示：

$$cost(R,M)=\sum_{l=1}^{|S|}EC_l \tag{7}$$

设 D 为用户指定的截止时间。在此基础上，我们可以定义云中工作流应用程序的最后期限约束成本优化问题为：

$$\begin{aligned}&\min\ cost(R,M)\\&\text{s.t.}\ \ makespan(R,M)\leqslant D\end{aligned} \tag{8}$$

4 算法设计

解决截止时间约束的成本优化问题通常包括以下 3 个步骤：首先将整个工作流的截止时间分配给每个任务从而形成子截止时间；然后按照某种方式对工作流中的任务进行排序；最后按顺序为每个任务选择合适的虚拟机。本文提出的算法 DCCO 中，使用了一种基于 δ-alap 分配截止时间的方式和排序方式。在虚拟机选择的步骤中，依次为每个任务分配满足其子截止时间而且总成本增量最低的虚拟机；若没有满足子截止时间的虚拟机，则选择能令其最早完成的虚拟机。

4.1 基于 δ-alap 的截止时间分配

ALAP(As-Late-As-Possible)是在不影响工作流的关键路径的前提下，任务的执行开始时间可以延迟多久的度量。任务的可延后执行时间 alap 表示在不影响工作流的关键路径的前提下，该任务可以延迟到最晚执行的长度。任务 t_i 的 $alap_i$ 值可以递归定义为式(9)：

$$alap_i=\min_{t_j\in t_i's\ children}\{alap_j-data_{i,j}/bw\}-ET_i \tag{9}$$

由于在实际情况中，把任务 t_i 分配给不同的虚拟机会产生不同的执行时间 ET_i，所以精确的 $alap_i$ 值是无法求得的。本文采用一种近似的方式来求得 $alap_i$，把 t_i 分配给执行速度最快的虚拟机 vm^*，其服务类型为 r^*，计算能力为 v^*，如式(10)所示：

$$alap_i=\min_{t_j\in t_i's\ children}\{alap_j-data_{i,j}/bw\}-w_i/v^* \tag{10}$$

将任务的子截止时间按照任务执行时间与任务可延后执行时间之和占关键路径的比例设置是一种有效的截止时间分配方法。计算任务 t_i 的子截止时间 sd_i 的方法如式(11)所示：

$$sd_i=D\times\frac{alap_i+w_i/v^*}{CPL} \tag{11}$$

令 bl_i 表示任务 t_i 到出口任务 t_{exit} 的最长路径，如式(12)所示：

$$bl_i=\max_{t_j\in t_i's\ children}\{bl_j+data_{i,j}/bw\}+w_i/v^* \tag{12}$$

工作流的关键路径长度 CPL 就是入口任务到出口任务之间的最长路径，那么 CPL 的计算式如式(13)所示：

$$CPL=bl_{entry} \tag{13}$$

这种方法本质上是依靠排序属性进行调度，类似于传统的工作流截止时间分配方法。然而，这种方法没有考虑到一种真实情况，即两个任务都部署在同一虚拟机上时，它们之间的数据传输时间为零。

如图 3 所示，假设图中工作流的截止时间是 100.1，这种分配方法将 t_1，t_2，t_3，t_4 的子截止时间分别设置为 1，1，99，100.1。虽然每个任务的执行时间是相同的，但是 t_3 的子截止时间比其他任务大得多，这可能会导致即使把 t_3 分配给执行速度最快的虚拟机，其他 3 个任务也不能按时完成。

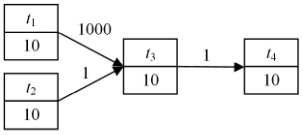


图 3 工作流示例

Fig. 3 An example of workflow

在虚拟机选择阶段，如果两个任务之间的传输时间比其他任务之间大得多，那么它们很有可能被分配到相同的虚拟机上，最终的截止时间分配也将会更加合理。因此，我们引入任务的 δ-alap 值，来对截止时间进行分配，如式(14)所示：

$$\delta\text{-}alap_i=\min_{t_j\in t_i's\ children}\{alap_j-\delta_j\times data_{i,j}/bw\}-w_i/v^* \tag{14}$$

其中，δ_j 表示计算 $\delta\text{-}alap_j$ 时是否考虑到 t_j 的数据传输时间的布尔变量，其值由式(15)决定：

$$\delta_j=\begin{cases}0, & \dfrac{w_j/va^*}{data_{i,j}/bw+w_j/va^*}<rand()\\ 1, & \text{otherwise}\end{cases} \tag{15}$$

其中，$rand()$ 是一个返回值为[0,1)内随机数的函数；$data_{i,j}/bw$ 越大，δ_j 返回 0 的概率越大，反之亦然。

基于 δ-alap 的截止时间分配通过式(16)计算每个任务的子截止时间：

$$\delta\text{-}sd_i=D\times\frac{\delta\text{-}alap_i+w_i/va^*}{CPL} \tag{16}$$

4.2 任务排序和虚拟机选择

尽管有多种任务排序方法，但本文选择 δ-alap 来对任务进行升序排序，因为它考虑到了数据传输时间可能变为零这一事实。

具体的虚拟机选择算法如算法 1 所示。

算法 1 DCCO 中的虚拟机选择算法

```
Input：任务列表 L
Output：调度结果 S=〈I,M〉
1. I ← Ø,M ← Ø,
2. for t_i：L then
3.     vm_l ← 满足 δ-sd_i 且执行 t_i 代价增量最小的虚拟机
4.     if vm_l == NULL then
5.         vm_l ← 最早完成 t_i 的虚拟机
6.         while δ-sd_i 不满足 && vm_l 不是最快类型的虚拟机 then
7.             将 vm_l 的类型更新成更快的级别
8.             更新 vm_l 上所有任务的完成时间
9.     end while
10.    end if
11. 分别通过式(4)和式(5)计算 ST_i,l 和 FT_i,l
12. M ← M ∪ {< t_i,vm_l,ST_i,l,FT_i,l >}
13. if vm_l ∉ I then
14.    I ← I ∪ { vm_l }
15. end if
16. end for
17. return 〈I,M〉
```

候选虚拟机 I 包括已在解决方案中使用过的所有虚拟

机。虚拟机选择的第一标准是选择满足子截止时间并使总成本增量最低的虚拟机(第 3 行)。注意,这个增量并不直接计算为在 vm_l 上运行 t_i 的成本,而是添加 t_i 后 vm_l 的成本与添加 t_i 前 vm_l 的执行成本之差。

当没有虚拟机可以满足子截止时间时,选择虚拟机的标准是从 I 中选择任务完成时间最短的虚拟机(第 5 行)。此外,如果所选的虚拟机不是最快的类型,我们将尝试将其类型设置为更快的级别,并更新部署在其上的每个任务的完成时间(第 6—9 行)。这样可以增加解决方案中满足整个截止时间的可能性。

此外,I 中每个虚拟机 vm_l 的租赁期从部署在 vm_l 上的第一个任务 t_u 开始接收来自其父任务的数据时开始,到部署在 vm_l 上的最后一个任务 t_v 完成向其子任务传输数据时结束,如下所示:

$$LST_l = ST_{u,l} - \max_{t_l \in t_u\text{'s parents}} \{TT_{l,u}\} \tag{17}$$

$$LFT_l = FT_{v,l} + \max_{t_l \in t_v\text{'s children}} \{TT_{v,l}\} \tag{18}$$

5 仿真实验

5.1 实验配置

本文使用文献[22]的实验环境进行实验,使用 4 种不同科学领域的工作流应用程序进行性能评估,包括地震学中的 CyberShake、信息生物学中的 Epigenomics、引力物理学中的 LIGO 和天文学中的 Montage[22]。这 4 种科学工作流的应用领域与计算特征不同,均包含着不同的结构特征,具体结构如图 4 所示。

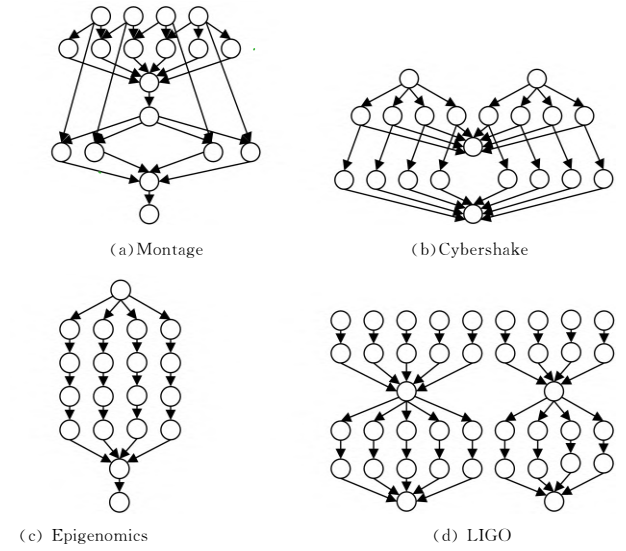


图 4 4 种工作流的结构特征

Fig. 4 Structural characteristics of four workflows

Montage 工作流任务偏好于 I/O 密集型任务,对 CPU 计算能力要求较低;LIGO 工作流任务偏好于计算密集型任务;Epigenomics 工作流任务以计算密集型为主,且对内存要求较高;CyberShake 工作流任务以数据密集型为主,且对计算能力和内存存储均有较高的要求。这些工作流的更多细节可以在文献[23]中找到。4 种工作流结构代表着完全不同的计算任务类型,通过这种多类型工作流的测试可以观察算法在适应不同任务类型下的鲁棒性,进而论证算法是有效可行的。

由于在实际数据中心进行重复实验非常困难,因此我们使用仿真方法来评估所提方法的有效性和效率。这使我们能够对广泛的应用程序配置进行大量的统计实验。假设数据中心提供 9 种不同类型的虚拟机,每种虚拟机具有不同的处理速度和成本,如表 1 所列。我们用服务与标准服务处理速度以及成本的倍数来表示该服务的处理速度和成本。虚拟机之间的平均带宽和收费时间间隔与 Amazon EC2 相同,分别为 20 MBps 和 1 小时。

表 1 可用虚拟机类型的处理速度和成本

Table 1 Processing speed and cost of available virtual machine types

类型	处理速度	成本
1	1.0	0.120
2	1.5	0.195
3	2.0	0.280
4	2.5	0.375
5	3.0	0.480
6	3.5	0.595
7	4.0	0.720
8	4.5	0.855
9	5.0	1.000

为了测试算法性能,本文以一种灵活的方式改变工作流的截止时间约束从而观察算法的适应性。设置方法如式(19)所示:

$$D = M_F + (M_C - M_F) \times \lambda \tag{19}$$

其中,D 为截止时间,M_F 为将所有虚拟机均配置为执行速度最快的虚拟机时工作流的完成时间,M_C 为将所有虚拟机均配置为最廉价虚拟机时工作流的完成时间,引入 $\lambda(\lambda \in [0,1])$ 来设置截止时间的松散度。在实验中,可通过 λ 的增长观察算法对最后期限的松散度的依赖性和适应性,从而使算法能够适应不同的应用场景。

5.2 实验分析

为了评估每种方法得到满足截止时间约束的有效解的能力,我们将 λ 依次取值为 0.005,0.01,0.03,0.05,0.1,0.3,0.5 来进行实验。不同配置下每种方法获得有效解的平均成功率如图 5 所示。

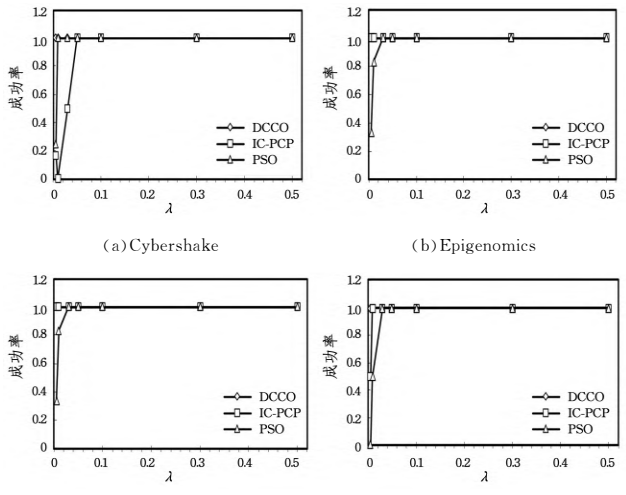


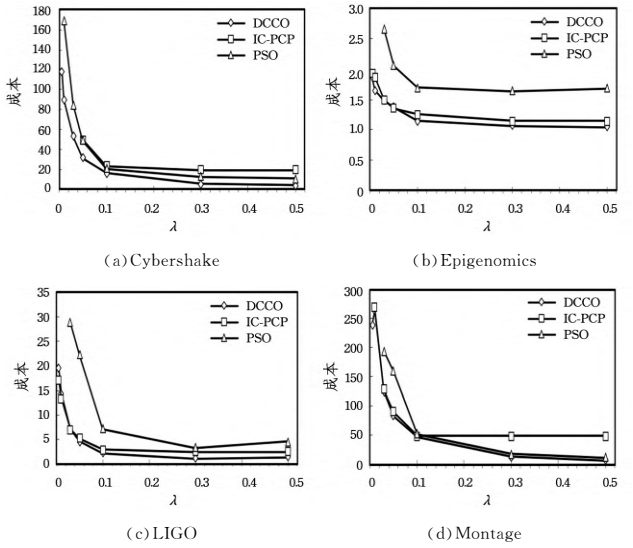
图 5 不同 λ 时每种方法的成功率

Fig. 5 Success rate of each method with different λ

我们观察到,当 λ 变大时,每种方法的成功率都随之增高。当 $\lambda=0.05$ 时,所有方法的成功率都为 100%。IC-PCP 在 LIGO 上的成功率是 100%,而在 Cybershake 上几乎是最差的。对于每种类型的工作流,当 $\lambda=0.005$ 时,PSO 的成功

率均低于 0.6。DCCO 表现最好,对于所有 λ 和所有工作流,它的成功率均达到 100%。

在图 5 中,当 λ 很小时,并不是所有的方法都能得到有效解,而在不满足截止时间约束的情况下比较成本是没有意义的。图 6 给出了每种方法在其成功率为 100%时的成本。



(a)Cybershake (b)Epigenomics

(c)LIGO (d)Montage

图 6 不同 λ 时每种方法的成本

Fig. 6 Cost of each method with different λ

如图 6 所示,成本随着 λ 的增加而降低。在特定的工作流类型和 λ 值下,PSO 和 IC-PCP 的成本高低是不同的。例如,在图 6(c)中,IC-PCP 的成本比 PSO 低,但在图 6(a)中,$\lambda > 0.05$ 时,IC-PCP 的成本比 PSO 高。

虽然在图 6(d)中 $\lambda > 0.3$ 时,PSO 的成本最低,但在大多数情况下,它的成本都高于其他算法。而每一种工作流下 DCCO 的成本都很有竞争力,在这 3 种方法中平均成本最低。其中,DCCO 的平均成本比 IC-PCP 低 27.1%,比 PSO 低 41.3%。

5.3 算法时间复杂度分析

为了计算各任务的 δ-alap,DCCO 首先需要遍历所有节点和边。对于有 n 个任务的工作流,其复杂度为 $O(n^2)$。之后需要按照 δ-alap 升序排列所有任务,从而形成调度列表。这一过程的复杂度为 $O(n*\log n)$。在虚拟机选择阶段,每个任务要遍历的虚拟机数量为 $|I|+T$(算法 1 第 3 行),其中 $|I|$ 为已选择的虚拟机数量,T 为云平台提供的虚拟机类型。因此为一个给定任务选择虚拟机的复杂度约为 $O((n+T))$,T 为常量。若虚拟机不能满足需求,则最多需要进行 T 次虚拟机升级(算法 1 第 4—10 行),因此整个虚拟机选择阶段的复杂度(算法 1 第 2—16 行)为 $O(n*(n+T*2))$。由此得出整个算法的复杂度约为 $O(n^2)$。

结束语 为了满足在用户给定的截止时间约束下优化执行成本,本文提出了一种新的调度策略 DCCO。通过分配截止时间、任务排序以及虚拟机选择 3 个阶段的优化,在满足截止时间约束的前提下,实现了成本的优化。通过工作流应用的仿真测试,验证了策略的性能。实验结果表明,DCCO 具有最高的成功率,且满足截止时间约束,同时可以优化执行成本。

参 考 文 献

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

[2] ZHOU N,LIN W,FENG W,et al. Budget-deadline constrained approach for scientific workflows scheduling in a cloud environment[C]// 2017 IEEE International Conference on Computational Science and Engineering(CSE) and IEEE International Conference on Embedded and Ubiquitous Computing(EUC). Guangzhou,China,2017,7-14.

[3] ALKHANAK E N,LEE S P,REZAEI R,et al. Cost optimization approaches for scientific workflow scheduling in cloud and grid computing: A review,classifications,and open issues[J]. Journal of Systems and Software,2016,10(1),3-52.

[4] MARIA A R,RAJKUMAR B. Deadline Based Resource Provisioning and Scheduling Algorithm for Scientific Workflows on Clouds[J]. IEEE Transactions on Cloud Computing,2014,2(2):222-235.

[5] ISMAYILOV G,TOPCUOGLU H R. Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing[J]. Future Generation Computer Systems,2020,102(4):307-322.

[6] SAHAR S,REIHANEH K,SOMAYE G B,et al. Improved many-objective particle swarm optimization algorithm for scientific workflow scheduling in cloud computing[J]. Computers & Industrial Engineering,2020,147(2):1-23.

[7] MBOULA J E N,KAMLA V C,CLEMENTIN T D. Cost-time trade-off efficient workflow scheduling in cloud[J]. Simulation Modelling Practice and Theory,2020,103(2):102-122.

[8] ZHANG L,ZHOU L,SALAH A. Efficient scientific workflow scheduling for deadline-constrained parallel tasks in cloud computing environments[J].. Information Sciences,2020,531(3):31-46.

[9] KAUR S,BAGGA P,HANS R,et al. Quality of Service(QoS) Aware Workflow Scheduling(WFS) in Cloud Computing: A Systematic Review[J]. Arabian Journal for Science and Engineering,2018,3(1):1-31.

[10] ZHU M M,CAO F,WU C Q. High-Throughput Scientific Workflow Scheduling under Deadline Constraint in Clouds[J]. Journal of Communications,2014,9(4):312-321.

[11] SAEID A,MAHMOUD N,DICK H E,et al. Deadline-constrained workflow scheduling algorithms for Infrastructure as a Service Cloud[J]. Future Generation Computer Systems,2013,29(5):158-169.

[12] VERMA A. Budget constrained priority based genetic algorithm for workflow scheduling in cloud [C]// Communication & Computing. Chandigarh,India,2013,216-222.

[13] JIAN C,WANG Y,TAO M,et al. Time-Constrained Workflow Scheduling In Cloud Environment Using Simulation Annealing Algorithm[J]. Journal of Engineering Science & Technology Review,2013,6(5):33-37.

[14] BILGAIYAN S,SAGNIKA S,DAS M. Workflow scheduling in cloud computing environment using Cat Swarm Optimization [C]// 4th IEEE International Advance Computing(IACC). Busan,Korea,2014:680-685.

[15] ZHOU Y,HUANG X. Scheduling Workflow in Cloud Computing Based on Ant Colony Optimization Algorithm[C]// Sixth International Conference on Business Intelligence & Financial

Engineering. Hangzhou, China: 2013: 57-61.

[16] GOGULAN R, KAVITHA A, KUMAR U K. An Multiple Pheromone Algorithm for Cloud Scheduling With Various QoS Requirements[J]. International Journal of Computer ence Issues, 2012, 9(3): 66-70.

[17] WU H, TANG Z, LI R. A priority constrained scheduling strategy of multiple workflows for cloud computing[C]// International Conference on Advanced Communication Technology. IEEE, 2012: 1086-1089.

[18] ZHANG X. Scheduling of cloud workflow on budget and deadline constraints[D]. Chongqing: Chongqing University, 2017.

[19] GAO T Y. Research on algorithms of minimizing financial cost of cloud workflow under deadline constraints[D]. Xi'an: Northwest University, 2019.

[20] YU K J, ZHANG J Z. Cloud workflow scheduling genetic algorithm of cost optimization under deadline constraint[J]. Computer Engineering and Design, 2018, 39(7): 1938-1945.

[21] CHEN Y T, PEI S J, MIAO H. Cost—optimized scheduling algorithm for cloud scientific workflow with deadline constraint [J]. Journal of Frontier of Computer Science ang Technology, 2019, 13(8)1307-1318.

[22] WU Q, FUYUKI I, QINGSHENG Z, et al. Deadline-Constrained Cost Optimization Approaches for Workflow Scheduling in Clouds[J]. IEEE Transactions on Parallel and Distributed Systems, 2017, 28(12): 99-109.

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

WANG Zi-jian, born in 1968, master. His main research interest is workflow scheduling.

SUN Fu-quan, born in 1964, Ph.D, professor. His main research interests include cloud resource scheduling and allocation and big data analysis.