

云环境下基于关键路径划分集群的调度算法

任丰玲¹, 于 凯¹, 陈 威², 于 炯³

(1 新疆财经大学 计算机科学与工程学院, 乌鲁木齐 830012; 2 江苏科技大学
张家港校区基础部, 张家港 215600; 3 新疆大学 研究生院, 乌鲁木齐 830046)

摘 要: 通过对云环境下多 DAG 工作流调度算法进行研究, 提出基于关键路径划分集群的多 DAG 工作流调度算法. 该算法充分考虑了 DAG 工作流的截止期限, 设计了新的综合评价指标, 基于关键路径划分集群, 在保证 DAG 工作流在截止期限内完成的前提下, 提高多个 DAG 工作流之间的公平程度, 缩短 DAG 工作流的延迟执行时间, 降低 DAG 工作流的执行时间跨度. 实验表明, 算法在公平性、延迟执行时间、执行时间跨度这三方面都有所改善.

关键词: 关键路径; 集群; 惩罚值; 截止期限; 时间窗口

中图分类号: TP301.6

文献标识码: A

文章编号: 1000-7180(2018)03-0042-05

Partitioning Cluster Based on the Critical Path of Multiple DAGs Scheduling

REN Feng-Ling¹, YU Kai¹, CHEN Wei², YU Jiong³

(1 School of Computer Science and Engineering, Xinjiang University of Finance and Economics, Urumqi 830012; 2 Campus of Zhangjiagang, Jiangsu University of Science and Technology, Zhangjiagang 215600; 3 School of Software, Xinjiang University, Urumqi 830046)

Abstract: Through the analysis of multiple DAGs workflow scheduling on the cloud, we propose multiple DAGs workflow scheduling based on critical path partitioning cluster algorithm. it takes into account the deadline of DAG workflow, designed a new comprehensive evaluation index, Partitioning Cluster Based on the Critical Path, ensuring DAGs completed before the deadline, improving the fairness between multiple DAGs, shortening the delay of execution time, reducing execution time span. Experiments show that this algorithm can improved in the aspect of fairness, time delay and the execution time span.

Key words: critical path; cluster; fine; deadline; time window

1 引言

随着云计算服务及建设的快速发展^[1], 云环境下工作流系统的研究与应用也受到更多的关注. 其中, 多 DAG (directed acyclic graph) 工作流的动态调度算法, 由于其固有的对动态变化的环境和动态到达的工作流的自适应性, 因而能够更好的处理多 DAG 工作流^[2]的调度问题. 已有的多 DAG 工作流

调度算法多是从工作流的周转时间、延迟时间、工作流的执行时间跨度、费用、资源的利用率、吞吐量等这些指标来考虑调度问题的, 忽略了 DAG 工作流对截止期限的要求. 如果不能在工作流的截止期限内调度完所有任务, 则其它的评价指标就没有意义. 本文主要考虑有截止期限和无截止期限的工作流在异构环境中的调度问题, 采用基于关键路径划分集群的方法, 把不同时间到达的多个 DAG 工作流分

收稿日期: 2017-04-20; 修回日期: 2017-05-22

基金项目: 国家自然科学基金项目(71561025); 新疆维吾尔自治区高校科研项目(XJEDU20161038); 2017 年新疆维吾尔自治区重点人文社科研究基重点项目; 维吾尔自治区教育厅规划课题(142018); 新疆财经大学校级博士基金(2015BS004).

解到不同的集群中,根据集群不同的优先等级,集中选择优势资源,优先调度执行有截止期限的 DAG 中关键路径上的任务,从而达到满足用户对任务截止期限的要求,对无截止期限的 DAG,一般情况下,要求费用较少,我们则选择费用低,可靠性高的资源与之匹配. 最终实现减少工作流的执行时间跨度,提高资源利用率,优化资源负载这三个目标^[3].

2 多 DAG 工作流调度算法

2.1 问题描述

异构非共享环境下,多个具有截止期限约束的 DAG 工作流调度问题的描述^[4]如下:工作流集合 N_{wf} ,资源集合 N_R ,工作流 $wf = \{wf_i | 1 \leq i \leq N_{wf}\}$, wf_i 可用含有 Nt_i 个任务的 DAG 来表示,每个任务用 $Task_{ij}$ ($1 \leq j \leq Nt_i$) 表示. 我们用三元组 (ts_i, tc_i, d_i) 来表示一个工作流, ts_i 是工作流的开始执行时间, tc_i 是工作流的完成时间, d_i 是工作流完成的弹性截止期限,计划完成时间 $T = \max d_i$, $ts_i \leq T$. 因为资源是非共享的,对任一资源 R_k ,定义时间窗口 $[\omega_{kl}; \omega_{kl}]$, $1 \leq l \leq Nw_k$, $0 \leq \omega_{kl}, \omega_{kl} \leq T$, $\omega_{kl} \leq \omega_{kl}$. 调度的目标就是在已知每个任务的开始执行时间 ts_i 和完成时间 tc_i 的情况下,把任务集和资源集进行匹配.

2.2 综合评价指标 U

云环境下多 DAG 工作流的调度结果通常用如下指标来衡量:整体执行时间跨度^[5]或接近最优执行时间跨度的速率^[6],这些不能体现是否满足截止期限;工作流的平均执行时间跨度^[7],该指标不适用于评价多个不同的工作流.

这里给出新的评价指标 U ,假定某个时间段内资源是固定的,用户对工作流的截止期限可变,若工作流 wf_i 改变截止期限,则有惩罚值 $Fine = \max\{tc_i - d_i, 0\}$. 若第 i 个工作流中所有的任务都未被分配资源,则有最大惩罚值 $Fine_i^{\max}$,用平均惩罚值 $\overline{Fine}$ 来衡量用户对截止期限的满意程度

$$\overline{Fine} = \frac{1}{N_{wf}} \cdot \sum_{i=1}^{N_{wf}} \frac{Fine_i}{Fine_i^{\max}} \quad (1)$$

此外,为了衡量资源分配的公平程度,给出

$$Fairness = 1 - \max_{i,j} \frac{|Fine_i - Fine_j|}{Fine_i^{\max}} \quad (1 \leq i, j \leq N_{wf}) \quad (2)$$

则综合评价指标

$$U = \rho_1 \cdot Fairness + \rho_2 \cdot (1 - \overline{Fine}), \quad \rho_1, \rho_2 \in [0, 1], \rho_1 + \rho_2 = 1 \quad (3)$$

2.3 基于关键路径划分集群的调度算法

基于任务的多 DAG 工作流调度算法 (MDW-T),是采用独立集群对合并后的工作流进行划分,算法具体步骤概括如下:

(1) 把工作流分为根节点集合和孩子节点集合,并计算根节点集合中的任务权值.

(2) 调度根节点集合中的任务,计算孩子节点集合中任务的权值,并更新两个集合.

(3) 找出有最大权值的任务,按照一定比例改变孩子节点集合中任务的截止期限.

(4) 循环执行第二步,直到获得所有任务的调度顺序队列.

基于工作流的多 DAG 工作流调度算法 (MDW-W),具体步骤概括如下:

(1) 把工作流分为已调度完的集合 U 和将要被调度的集合 L ,初始化 L ,定义时间窗口 I .

(2) 计算并根据 L 中工作流的优先级,把工作流分配到各个资源对应的时间窗口中执行,更新集合 U 、集合 L 和时间窗口 I .

(3) 循环执行第二步,直到调度完所有的工作流.

综上所述,提出基于关键路径划分集群的多 DAG 工作流调度算法(简称为 MDW-CPC),对新到达的每个 DAG 工作流提取关键路径,并立即分配资源进行调度,对非关键路径上的任务,对其进行分组,每一组即为一个集群,任意集群 c 用如下因素来描述:(1) 截止期限 $d(c) = \max d_{ij}$ (2) 开始时间 $ts(c) = \min ts_{ij}$ (3) 权重 $w(c) = \sum \overline{t_{ij}}$ 是 $task_{ij}$ 的平均执行时间, $task_{ij} \in c$ (4) 长度 $l(c) = d(c) - ts(c)$.

每个集群用一个长方形表示,长方形的宽表示集群的权重,高表示集群执行时间的长度,如下图 1(a) 所示,从图 1(a) 可以看出初始集群的权重有较大的差别,并且有相近子截止期限的任务在不同的集群中,重新调整划分集群后理想的结果如图 1(b) 所示. 算法 MDW-CPC 划分集群的过程其实就是寻找集群的平均权重 $\overline{\Delta w}$ 和集群长度之和 L 之间的平衡,使 $f(\text{set}C)$ 的值最小, $f(\text{set}C) = \overline{\Delta w} / \overline{\Delta w_{\max}} + L / L_{\max}$, $\text{set}C = \{C_1 \cdots C_k \cdots C_{N_c}\}$ 是当前集群的集合,其中 $\overline{\Delta w_{\max}} = \sum_k w(C_k) / 2$, $L_{\max} = \sum_{i,j} \overline{t_{ij}} Task_{ij} \in \text{set}(C)$

图 2 给出划分合并集群的实例,长方形下边的数字表权重,里面的数字表长度.

初始工作流中每一个任务表示一个集群,集群的下标按子截止期限升序排列. 计算比较当前集群

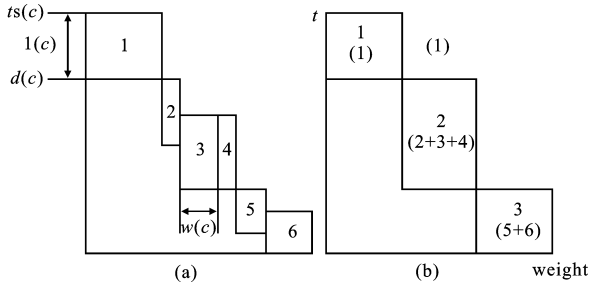


图1 集群实例

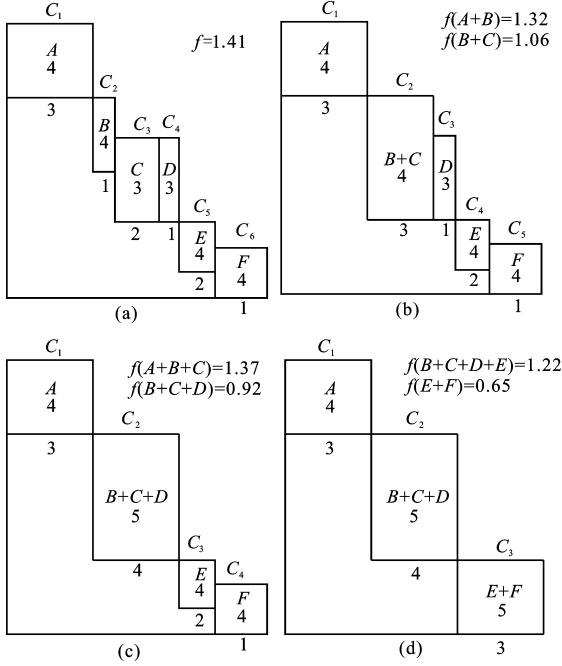


图2 合并集群的具体过程

和其前驱集群、后继集群的 f 值, f 值小的若干个集群可以合并为一个新的集群, 当前集群的下标变化范围从 2 到 $N_c - 1$. 图 2(a) ~ (d) 中, 初始集群 C_1 为当前集群, $f = 1.44$, C_1 无前驱集群, f 值无可比性, 即 C_1 未参与任何合并, 则当前集群的下标自动增加, C_2 变为当前集群(C_2 包含任务 B), $f(A+B) = 1.32$, $f(B+C) = 1.06$, 合并 B 和 C , f 值下降更快, 因此, 集群 C_1 仍然只包含任务 A , 集群 C_2 包含任务 B 和 C , 当前集群为 C_2 , $f(A+B+C) = 1.37$, $f(B+C+D) = 0.92$, 所以合并 B 、 C 、 D 为集群 C_2 . 因为 $f(A+B+C+D) = 1.48$, $f(B+C+D+E) = 1.22$, 所以当前集群变为 C_3 , $f(E+F) = 0.65$, 则合并 E 和 F 为集群 C_3 . 至此, 合并后的集群有大致相等的权重和长度. 集群合并之后, 则对其进行排序, 集群排序依据公式

$$M(c) = \rho_{wf} \cdot \frac{w(c)/I(c)}{wI_{\max}} + \rho_d \cdot \frac{d_{\min}}{d(c)} \quad (4)$$

ρ_{wf} 、 ρ_d 是权重系数, $wI_{\max} = \max \frac{w(c)}{I(c)}$, $d_{\min} = \min d(c)$, $c \in CS$ 优先选择 $M(c)$ 值大的集群进行资源分配.

综上所述, 我们给出 MDW-CPC 算法的伪代码如下所示.

```
While ( DAGnew ≠ ∅ ) do /*当新DAG到达*/
    提取 DAGnew 中所有工作流的关键路径, 生成关键
    路径集群, 关键路径集群按照其对应的工作流的截
    止期限升序排列. 对非关键路径上的任务, 计算
    f(setC) 的值, 并根据该值合并集群, 直到所得
    的集群有大致相等的权重和长度.
    If ( C ∈ CPS ) /*c表示当前集群, cps表示
        关键路径集群组成的集合*/
        立即分配可靠性高、速度快的优势资源进
        行调度, 利用每个资源对应的时间窗口,
        使关键路径集群得到可靠、有效的资源分
        配.
    Else
        计算 M(C) 的值, 并按照该值由大到小的
        顺序对非关键路径上的集群进行排序并调
        度. 所有集群内的任务均按照子截止期限
        升序排列并按照MDW-T算法进行调度.直
        到调度完 DAGnew 中所有的工作流.
```

End While

3 仿真实验与分析

实验部分主要对 MDW-T, MDW-W, MDW-C, MDW-CPC 这四个算法从以下三方面进行比较: (1) 在设置不同个数的时间窗口时, U 值、惩罚值、公平性等的分布情况; (2) 在工作流集合中含有不同数量工作流的情况下, U 值、工作流的平均延迟时间的分布情况; (3) 多个具有不同数量任务的工作流在不同的通信运算比下, 执行时间跨度的分布情况.

实验过程中, 所用的工作流和资源的各项参数如表 1 所示.

表1 工作流和资源的各项参数

参数	参数值
工作流集合大小	20;40;60;80;100
任务计算代价	15 000-216 000
工作流中的任务数	10;20;30;40;50
DAG 的宽度	0.1;0.2;0.8
DAG 的深度	0.2;0.8
DAG 的变化规则	0.2;0.8
DAG 的增长速度	1,2,4
资源个数	5~50
时间窗口的扩展系数	0.25;0.5;0.75
时间窗口个数	1~10

四个算法在不同个数的时间窗口时,其 U 值、惩罚值、公平性的分布如图 3 所示.

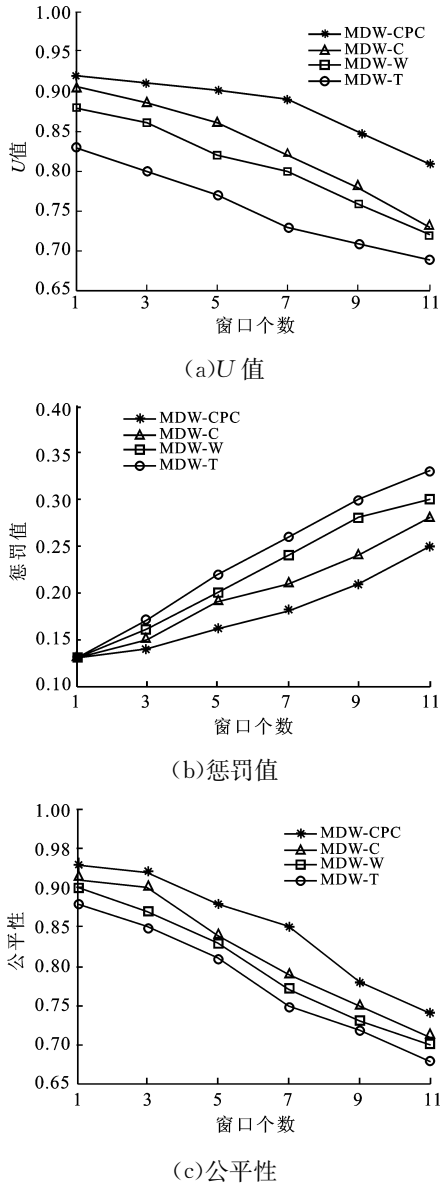


图 3 四个算法的 U 值、惩罚值、公平性的对比

从图 3(a)可知,在设置不同个数的时间窗口时,MDW-CPC 算法的综合评价指标 U 的值都要略大于 MDW-T 算法,说明 MDW-CPC 能更好的同时满足多个 DAG 对截止期限的要求,但随着时间窗口数量的增加,两个算法的 U 值都呈下降趋势.相对而言,MDW-CPC 能更充分的利用资源,但处于等待分配资源的任务也越多,因此,在设置时间窗口数量的同时,不仅要考虑资源利用率,还要考虑任务截止期限的紧急程度.

从图 3(b)可知,随着时间窗口数量的增加,四个算法的惩罚值都呈上升趋势,MDW-CPC 算法的惩罚值略低于其它三个算法.因为随着时间窗口数

量的增加,可改变截止期限的工作流相应的等待调度时间也会增大,从而导致惩罚值呈上升趋势.从图 4(c)可知,随着时间窗口数量的增加,四个算法的公平性都呈下降趋势,MDW-CPC 算法的公平性略高于其它三个算法.

在工作流集合中含有不同数量工作流的情况下,MDW-CPC 算法较其它三个算法而言,其综合评价指标 U 以及工作流的平均延迟时间的分布如表 2 和表 3 所示.

表 2 不同工作流个数的情况下 U 值的对比

工作流 个数	U 值			
	MDW-T	MDW-W	MDW-C	MDW-CPC
20	0.94	0.96	0.97	1
40	1	1	1	1
60	1	1	1	1
80	0.93	0.98	1	1
100	1	1	1	1
120	0.99	0.99	1	1

表 3 不同工作流个数的情况下平均延迟时间

工作流 个数	U 值			
	MDW-T	MDW-W	MDW-C	MDW-CPC
20	0.015	0	0	0
40	0.029 7	0	0.001	0
60	0.03	0	0	0
80	0.17	0.07	0	0
100	0.004 8	0.049	0.034	0.026
120	0.10	0.09	0.07	0.04

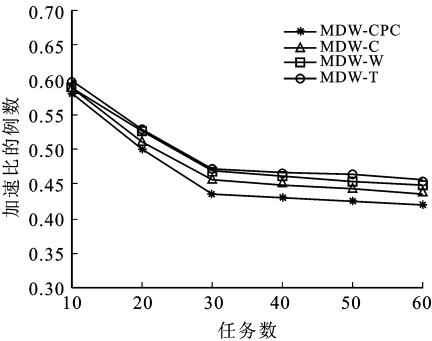
从表 2 可看出,随着集合中工作流个数的增加,MDW-T,MDW-W,MDW-C 三个算法的 U 值未呈现出规律性变化,但都趋近于 1,MDW-CPC 处于最优状态都为 1.因此,推断 MDW-CPC 算法的 U 值与集合中工作流的个数没有太大的关系,相对比较稳定.

从表 3 可看出,随着集合中工作流个数的增加,MDW-T,MDW-W,MDW-C 这三个算法的平均延迟时间并未呈现出规律性变化,MDW-CPC 算法的平均延迟时间有增大趋势,但都略小于其它三个算法.

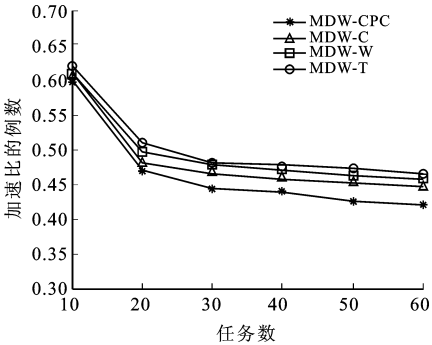
多个有不同数量任务的工作流在不同的通信运算比(CCR)下,加速比的分布如图 4 所示.

从图 4 可看出,对于计算密集型的工作流和一般工作流而言,MDW-CPC 算法的加速比的倒数都要略优于其它三个算法,加速比的倒数值相差大概在 5%左右.对于数据密集型工作流,MDW-CPC 算法的加速比的倒数明显优于其它三个算法,加速比的倒数值相差大概在 10%左右.说明 MDW-CPC 算法更适用于数据密集型工作流的调度.

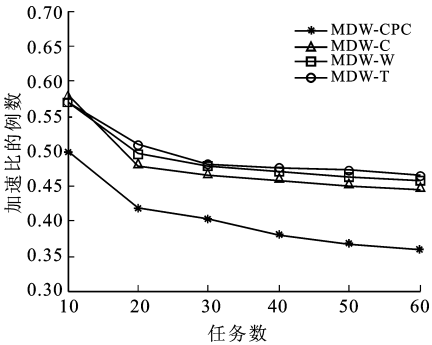
实验结果反映出,基于关键路径划分集群的多 DAG 调度算法,在保证 DAG 工作流在截止期限内完



(a) ccr=0.5



(b) ccr=1



(c) ccr=5

图4 不同 ccr 工作流加速比的倒数

成的前提下,使多个 DAG 之间的公平程度、延迟执行时间以及执行时间跨度这三个性能都有所改善。

4 结束语

本文针对已有算法在调度过程中存在的问题,结合实际,提出基于关键路径划分集群的多 DAG 工作流调度算法.该算法充分考虑了 DAG 工作流的截止期限,基于关键路径划分集群,设计了新的综合评价指标 U,在保证 DAG 工作流在截止期限内完成的前提下,提高多个 DAG 工作流之间的公平程度,缩短 DAG 工作流的延迟执行时间,降低 DAG 工作流的执行时间跨度.仿真实验结果表明,MDW-CPC 算法在设置不同个数的时间窗口时,其 U 值要高于其它三个算法;在包含不同个数工作流

的集合中,其 U 值、工作流的平均延迟执行时间都略优于其它三个算法;在不同的通信运算比和包含不同个数任务的工作流中,其执行时间跨度都有所减少,对于数据密集型的工作流,其执行时间跨度减少的更为明显。

在实际应用中,由于资源和用户提交的任务是动态变化的,因此未来我们将从资源备份、任务确认、资源的有效利用率、资源负载均衡等方面来考虑云环境下多 DAG 工作流的调度。

参考文献:

[1] Abrishami S, Mahmoud Naghibzadeh, Dick Epema. Deadline-constrained workflow scheduling algorithms for IaaS Clouds[J]. Future Generation Computer Systems, 2013, 29 (1):158-169.

[2] Epema D H J, Naghibzadeh M,Deadline-constrained workflow schedulin algorithmsforinfrastructure as a service clouds[J]. Future Generation Computer Systems, 2013, 29 (2):118-129.

[3] Klavdiya Bochenina, Nikolay Butakov, Alexander Boukhanovsky. Static scheduling of multiple workflows with soft deadlines in non-dedicatedheterogeneous environments[J]. Future Generation Computer Systems, 2016 (55):51-61.

[4] Bochenina K. A comparative study of scheduling algorithms for the multiple deadline-constrained workflows in heterogeneous computing systems with time windows[J]. Procedia Comput. Sci. 2014 (29):509-522.

[5] Bittencourt L F, Madeira E R M. Towards the scheduling of multiple workflows on computational grids [J]. Journal of Grid Computing, 2010,8(3):419-441.

[6] Hiraes Carbajal A, Tchernykh A, Yahyapour R, et al. Multiple workflow scheduling strategies with user run time estimates on a grid[J]. Journal of Grid Computing , 2012 , 10 (2) :325-346.

[7] Boregowda U, Chakravarthy V. Multiple DAG applications scheduling on a cluster of processors[C] // Comput. Sci, inf. Technol, CSIT 2014. China, Nanjing, 2014:63-73.

作者简介:

任丰玲 女,(1986-),硕士,讲师.研究方向为网格计算、分布式计算. E-mail: xin-ruo-zhi-shui@qq. com.

于 凯 男,(1974-),博士,副教授,硕士生导师.研究方向为云计算、大数据与社会网络;复杂网络与信息传播.

陈 威 男,(1986-),讲师.研究方向为生存分析、数理统计.

于 炯 男,(1964-),教授,博士生导师.研究方向为网络安全、网络计算、分布式计算.