带截止期约束的复杂依赖任务的智能调度优化*

王 佳[1,2,3]，潘仟喜[1,2,3]，杨文忠[1,2,3]

(1. 新疆大学 计算机科学与技术学院，新疆 乌鲁木齐 830017；2. 丝路多语言认知计算国际合作联合实验室，新疆 乌鲁木齐 830017；
3. 新疆多语种信息技术重点实验室，新疆 乌鲁木齐 830017)

摘　要： 农作物分类、种植面积估计、需水量/蓄水量的预测等是农业领域水资源高效利用的重要任务. 当前天-地协同水资源管理中，不同设备（天基卫星、空基遥感、地面观测）所产生的异构数据间紧密关联. 用户提交的不同任务通常由一系列相互依赖的子任务构成，且要求在给定的截止期前完成. 故提出一种基于图注意力网络和元学习的调度优化策略，通过均衡子任务间的传输时间和计算时间以最小化所有作业的完工时间. 为更好提取异构数据间的紧密关联，采用改进多头注意力机制的图注意力网络有效提取子任务间的依赖和关联关系. 同时，利用指数平滑改进的元学习方法进行网络参数优化以提高模型的适应性. 与现有深度学习调度算法相比，所提调度策略在截止期前完成的作业数量比率平均提升7.52%.

关键词： 图注意力网络；元学习；截止期; 依赖任务；智能调度

DOI： 10.13568/j.cnki.651094.651316. 2024.12.20.0006

中图分类号： TP391.7　**文献标识码：** A　**文章编号：** 2096-7675(2025)04-0416-09

引文格式： 王佳，潘仟喜，杨文忠. 带截止期约束的复杂依赖任务的智能调度优化[J]. 新疆大学学报(自然科学版中英文)，2025，42(4)：416-424.

英文引文格式： WANG Jia，PAN Qianxi，YANG Wenzhong. A smart scheduling optimization for dependent tasks with deadlines[J]. Journal of Xinjiang University(Natural Science Edition in Chinese and English)，2025，42(4)：416-424.

A Smart Scheduling Optimization for Dependent Tasks with Deadlines

WANG Jia[1,2,3], PAN Qianxi[1,2,3], YANG Wenzhong[1,2,3]

(*1. School of Computer Science and Technology, Xinjiang University, Urumqi Xinjiang 830017, China; 2. Joint International Research Laboratory of Silk Road Multilingual Cognitive Computing, Urumqi Xinjiang 830017, China;*
3. Xinjiang Key Laboratory of Multilingual Information Technology, Urumqi Xinjiang 830017, China)

Abstract： Crop classification, planting areas estimation and water demand/storage prediction are critical for optimizing water resource management in agriculture. In the water management based on sky-ground collaboration, heterogeneous data from various devices such as satellite, airborne remote sensing and ground observations, are closely depended. In general, tasks required by users always contain several dependent subtasks, and tasks are required to be finished before deadline. In this paper, we introduce a smart scheduling based on graph attention network and meta-learning for tasks with deadlines, which minimizes makespan by the balance between the data transmission time and computation time of subtasks. An enhanced multi-head attention mechanism in graph attention networks is designed to extract associations among heterogeneous data. Additionally, an exponentially smoothed meta-learning approach is designed to optimize parameters of strategies. Compared to existing deep learning-based scheduling algorithms, the proposed strategy improves the average proportion of tasks completed before deadlines by 7.52%.

Key words： graph attention network; meta-learning; deadline; dependent tasks; smart scheduling

---

* **收稿日期：** 2024-12-20

基金项目： 新疆维吾尔自治区自然科学基金“带截止期约束的流作业动态智能调度优化”（2023D01C20）；国家自然科学基金“面向新疆农业多模态气象大数据的云模式灾害精准预测”（62363032）；新疆维吾尔自治区教育厅培育类项目“基于多模态多因素数据的交通流量预测及调度研究”（XJEDU2022P011）；新疆维吾尔自治区天山英才科技创新团队“面向农业的天地协同水资源时空精准调度研究及应用创新团队”（2023TSYCTD0012）.

作者简介： 王佳（1987—），女，博士，副教授，主要从事云计算和大数据中的调度优化研究，E-mail：jw1024@xju.edu.cn.

0　引 言

农业领域的数智化发展进程中，精准数据分析和处理，在农作物分类、种植面积估计、需水量和蓄水量预测等方面具有重要作用[1]. 随着不同观测设备（如天基卫星、空基遥感和地面观测等）所产生的异构数据持续增加[2]，现有数据处理技术无法完成有效分析. 同时，上述任务均由一系列相互依赖的子任务构成，即后继任务的开始依赖于前驱任务的完成以及二者间数据的传输. 此外，用户提交的不同任务通常给予截止期约束以避免决策延误. 鉴于Apache Spark[3]等大数据框架和云计算技术[4]在大数据分析处理方面的显著优势，从农业实际问题出发，本文考虑面向截止期约束的Spark异构工作流调度优化，从而有效满足复杂数据的处理需求.

以农作物分类任务为例，其包含异构农作物数据收集、农作物数据预处理、深度网络特征提取、模型训练等一系列相互依赖的子任务. 各子任务间均存在大量的中间数据处理，结合Apache Spark框架中前驱子任务执行完成后才开始中间数据传输的特点，极大增加了后继子任务等待时间，从而延长任务完成时间，导致任务无法在截止期前完成. 同时，各子任务中均需处理来自于不同设备的多类异构数据，数据间的关联关系导致子任务间复杂的关联关系，如何有效提取数据及任务间的关联关系至关重要. 此外，不同子任务需由各类相同或不同服务器资源完成，如数据预处理可由CPU、内存资源完成，而模型训练需由GPU、内存资源完成. 也就是说，不同子任务间存在同类资源竞争问题，如何保证异构资源的合理分配以满足不同任务的截止期需求极为重要. 综上所述，面向截止期约束的Spark异构工作流调度优化是现阶段农业领域数据发展待需解决的问题.

现有Spark调度优化算法可分为基于传统方法的调度策略和基于深度学习的调度策略[5]. 基于传统方法的调度策略，依赖启发式调度规则等将不同任务调度至合适资源，以提高资源利用率及任务处理效率，如公平调度器[6]和容量调度器[7]等. Tang等提出一种启发式成本效益任务调度策略，通过建模工作流为I/O数据感知的有向无环图，使用动态任务租赁账单周期共享方法及任务子截止期松弛，降低截止期约束的工作流执行成本[8]. Hussain等提出截止期约束的成本感知工作流调度算法，在私有云优先执行最多数量的工作流任务，并将未调度任务转移至公共云，以满足任务优先级和截止期需求[9]. Medara等设计了基于异构最早完成时间的能耗效率启发式调度方法，通过最大化资源利用率，有效解决工作流截止期约束[10]. 考虑虚拟机实例的异构性、任务优先级约束、传输时间和性能不确定性等，Rajput等提出一种新的启发式方法，在混合云网络中完成Spark工作流的调度，从而降低租赁成本[11]. Islam等利用混合云中不同虚拟机实例的定价，优化本地和云资源的虚拟机使用成本，以最大化作业截止期限的满足率[12]. 基于传统方法的调度策略在任务调度性能、资源利用率等方面取得一定进展，但尚未考虑多源异构数据的复杂关联及任务执行与数据传输的过长时间. Mirza等认为人工智能技术对当前大数据计算框架调度性能提升有巨大推动[13]，基于深度学习的调度策略逐渐成为研究热点.

基于深度学习的调度策略将深度学习技术引入任务调度中，利用网络模型从数据中学习调度决策，从而通过精准数据分析和模型自适应性动态调整调度策略，能更好处理复杂关联的异构数据及任务. 通过充分考虑空间结构关系和任务时间特性，Li等提出基于图注意神经网络和多智能体深度强化学习的分布式调度算法，有效解决了时间敏感性任务调度问题[14]. 结合长短期记忆网络，Zhang等利用深度强化学习在云计算资源上动态调度大规模工作负载，减少了资源消耗和任务等待时间[15]. Lee等提出基于深度强化学习的优先级分配模型，通过图卷积网络处理有向无环图任务在多处理器系统中的调度，以最小化任务的完成时间[16]. Gu等通过智能资源需求估计，设计高效的GPU资源管理平台Liquid，提升了作业执行效率[17]. Capel等提出基于“模型平均化”技术的方法，显著提升Apache Spark在随机梯度下降和潜在狄利克雷分配模型训练中的性能，从而解决了Spark在分布式机器学习任务中执行缓慢的问题[18]. 基于深度学习的调度策略着重采用深度强化学习模型解决作业调度过程中任务与资源间的有效动态匹配，来优化任务执行时间或资源消耗等，较多忽略了任务截止期约束和Spark中任务执行与数据传输的过长时间，对基于截止期约束的异构多源复杂工作流调度少有研究.

为解决面向截止期约束的Spark异构工作流调度问题，本文提出基于图注意力网络和元学习的调度优化策略. 考虑异构数据及子任务间的复杂关联，设计基于依赖系数的图注意力网络，更准确识别和提取异构数据间的关系. 同时，为提升调度策略中网络参数的学习效率和模型的动态适应性，引入指数平滑特性的元学习方法.

1　所提方法

考虑农业领域大数据分析处理任务的截止期约束、异构数据及任务间的复杂关联，提出基于图注意力网络

和元学习的调度优化策略（Meta GAT-Scheduler, MGS），其框架如图1所示. 假定有N个任务请求，每个任务均可用有向无环图（Directed Acyclic Graph, DAG）表示. 各任务用TASK DAG表示，分别进入基于依赖系数的图注意力网络模块进行任务及各子任务间复杂关联关系的学习. 依赖系数通过子任务间的历史交互、子任务优先级差异和计算资源的共享频率等因素综合计算得到，提升了子任务间依赖特征的提取及表达. 通过图注意力网络模块得到第i个任务T_i中第j个子任务ST_{ij}的特征表示hd_{ij}，以及第j个子任务ST_{ij}和第k个子任务ST_{ik}间的关联权重α_{ijk}，结合资源R_r（$1 \le r \le M$）的特征表示F_{R_r}，依次进入基于指数平滑的元学习模块，以获取较优的调度决策，其中M为整体资源数量. 各子任务ST_{ij}的特征表示hd_{ij}融合后形成各任务T_i的特征表示hd_i，与F_{R_r}通过线性网络计算得到当前时刻策略$\pi(t)$下的损失函数. 为提高模型鲁棒性，引入指数平滑因子以平滑调整每次迭代的参数更新量，从而得到最优的调度策略.

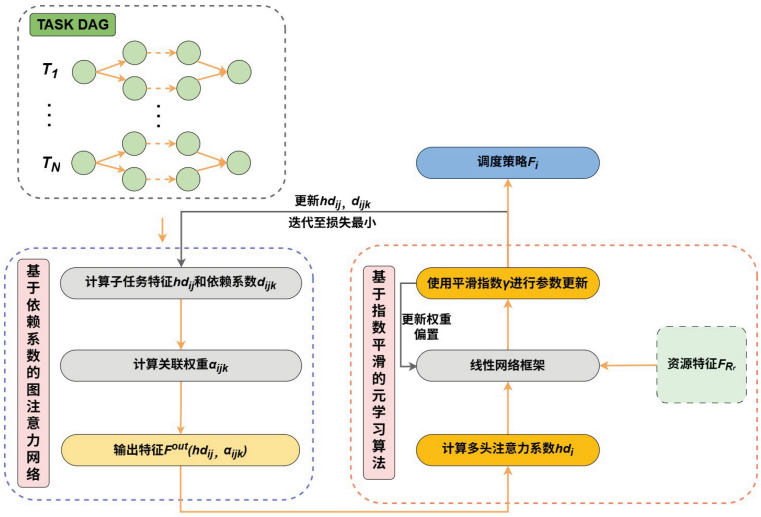

图 1 MGS框架

1.1 基于依赖系数的图注意力网络模块

考虑多头注意力机制在节点依赖关系提取中的优越性[19−21]，图注意力网络被广泛应用于特征提取. 现有多头注意力机制通过并行运行多个注意力机制，以获取不同任务间的关联关系，忽略了任务调度过程中任务间的历史交互、计算资源间的资源竞争概率等信息，无法精准描述任务与资源间的匹配关系. 本文提出基于依赖系数的图注意力网络模块，以捕获任务间的历史交互、任务优先级和计算资源间的资源竞争概率，从而得到准确的各任务内各子任务特征表示和子任务间关联关系.

针对任务T_i中子任务ST_{ij}，其对应的子任务输入特征表示为$F^{in}(ST_{jd}, ST_{js}, ST_{ij})$，其中$ST_{jd}$和$ST_{js}$分别代表子任务$ST_{ij}$的前驱和后继任务. 设$d_{ijk}$为子任务$ST_{ij}$和子任务$ST_{ik}$间的依赖系数，其中$ST_{ik} \in \{ST_{jd}, ST_{js}\}$. 假定$f_{ijk}$、$p_{ijk}$和$c_{ijk}$分别表示$ST_{ij}$和$ST_{ik}$间的历史交互信息、优先级差异度和资源竞争概率，且$d_{ijk} = [f_{ijk}, p_{ijk}, c_{ijk}]$. f_{ijk}和c_{ijk}均由基于指数平滑特性的元学习模块得出. p_{ijk}为ST_{ij}和ST_{ik}优先级差值的绝对值，ST_{ij}的优先级根据TASK DAG的拓扑排序计算得出. 不同任务T_i中各子任务ST_{ij}所得到的$F^{in}(ST_{jd}, ST_{js}, ST_{ij})$和$d_{ijk}$分别进入对应的图注意力网络模块得到其输出特征表示$F^{out}(hd_{ij}, \alpha_{ijk})$，其中$hd_{ij}$为融合依赖系数后的子任务信息、$\alpha_{ijk}$为子任务间的关联权重. 由于图注意力网络的输入包含$F^{in}(ST_{jd}, ST_{js}, ST_{ij})$和$d_{ijk}$，相应的图注意力网络计算过程需进行相应修改，具体描述如下.

设e_{ijk}为子任务ST_{ij}和子任务ST_{ik}间的注意力分数，a^T和W分别为映射向量和特征增维共享参数，其计算公式为

$$e_{ijk} = LeakyReLu(a^T Concat(WF^{in}(ST_{jd}, ST_{js}, ST_{ij}), WF^{in}(ST_{kd}, ST_{ks}, ST_{ik}), f(d_{ijk}))), \tag{1}$$

式中：$f(d_{ijk})$为特征增维映射，将d_{ijk}映射为与$WF^{in}(ST_{jd}, ST_{js}, ST_{ij})$相同的维度. 任务调度过程中，子任务间的历史交互信息及资源竞争情况实时变化，对子任务特征提取影响较大. 设$g(d_{ijk})$为依赖系数的权重，根据依赖系数动态调整注意力分数对特征提取的影响，ST_{ij}和ST_{ik}的注意力权重α_{ijk}计算公式为$\alpha_{ijk} = \frac{\exp(e_{ijk} \cdot g(d_{ijk}))}{\sum_{v \in N_{T_i}} \exp(e_{ijv} \cdot g(d_{ijv}))}$，

其中N_{T_i}为任务T_i所包含的子任务数量. 假定$\zeta(\cdot)$表示加权聚合函数，依据α_{ijk}、$F^{in}(ST_{jd},ST_{js},ST_{ij})$和$d_{ijk}$可得融合依赖系数后的子任务信息$hd_{ij}$，即$hd_{ij}=\zeta(\alpha_{ijk},F^{in}(ST_{jd},ST_{js},ST_{ij}),d_{ijk})$. 综上所述，得到子任务$ST_{ij}$的输出特征表示$F^{out}(hd_{ij},\alpha_{ijk})$.

1.2　基于指数平滑的元学习模块

鉴于元学习在参数调整过程中的优越性[22−23]，本文在调度策略生成部分采用元学习方法，以提高调度策略的准确性和实时性. 现有元学习采用插值法进行参数调整，极易导致参数的不稳定性及目标函数的波动性，不易于模型收敛. 考虑指数平滑方法在时间依赖任务中的适用性，提出基于指数平滑的元学习模块，以快速得到精准调度策略.

假设基于指数平滑的元学习模块的参数为θ，初始化为θ_0. θ_0由τ样本数据学习得到，τ设为5. 根据基于依赖系数的图注意力网络模块所得各子任务ST_{ij}的输出特征表示$F^{out}(hd_{ij},\alpha_{ijk})$，采用多头注意力机制综合得到各任务$T_i$的特征表示$F'(hd_i,\alpha_i)$，其中$hd_i$和$\alpha_i$分别表示任务$T_i$的任务信息和关联权重，具体计算方式为

$$hd_i=Concat(\sigma^u\sum_{j\in N_{T_i}}\alpha_{ij}\cdot hd_{ij}),\quad 1\le u\le N, \tag{2}$$

式中：$\sigma^u\sum_{j\in N_{T_i}}\alpha_{ij}\cdot hd_{ij}$是第$u$个注意力机制，共$N$个注意力机制进行融合. 根据式 (2) 可从多个任务角度得到多任务间的依赖关系，从而得出更准确的任务特征表示.

若资源R_r的特征表示为$F_{R_r}=[G,M_{use},I_{ocu}]$，其中$G$、$M_{use}$和$I_{ocu}$分别表示各资源的GPU数量、内存使用量和当前任务的内存占用量. 结合上述得到的任务特征表示$F'(hd_i,\alpha_i)$，得到任务T_i的调度策略表征$F_i=Concat(F_{R_r},F')$. 所有F_i经过网络结构后，获取当前时刻t策略$\pi(t)$下的任务完成时间C_i，即$C_i=f(\theta,\pi(t),F_i)$，其中$f(\cdot)$为网络结构对应的映射函数. 考虑任务的截止期约束，本文的损失函数定义为

$$L(\theta,\pi(t))=\sum_{i\in N}\max(0,\frac{C_i-D_i}{|C_i-D_i|})\log P_i+(1-\max(0,\frac{C_i-D_i}{|C_i-D_i|}))\log(1-P_i), \tag{3}$$

式中：D_i和P_i分别表示任务T_i的截止期和在截止期前完成的概率，且$P_i=soft\max C_i$.

考虑指数平滑方法在时间依赖任务中的适用性，采用指数平滑方法以降低原有插值法的波动性，则参数θ的更新公式为$\theta=\gamma\theta+(1-\gamma)(\theta+\beta\Delta\theta)$，其中：$\gamma$为平滑因子，根据经验设为0.9，以保证模型的稳定性；β为元学习率；$\Delta\theta$为相邻两个θ的差值. 同时，基于指数平滑的元学习模块得到f_{ijk}和c_{ijk}用于更新依赖系数. 经过多次迭代至收敛，得到最终调度决策. 假定A_{ST}为各任务的平均子任务数量，S_T和B分别表示迭代次数和批量大小，则算法整体复杂度为$O(S_T(A_{ST}BD+P))$，其中D和P分别表示任务特征表征维度和模型参数量.

2　实验分析

2.1　实验配置

本文算法以及所有比较算法均采用python3.6.15 torch1.10.1编程框架进行编码，相应的硬件配置为Xeon(R) Gold 6148 CPU和RTX 3080×2 GPU.

选择的数据集为TPC-H[24]和Alibaba cluster-trace-v2018[25]. 两种数据集不同任务中存在相互依赖的子任务，且不同任务中的子任务数量不同. TPC-H是经典决策支持系统的基准测试数据集，主要模拟供应商与采购商之间的订单交易关系，从而评估数据库在复杂查询和数据分析场景下的性能. TPC-H数据集中任务之间存在多层级的依赖关系，如订单、客户、供应商等以雪花型结构组织. Alibaba cluster-trace-v2018是阿里巴巴发布的生产集群数据集，记录了4 000台服务器上在线应用和离线计算任务的运行情况，包含任务的资源使用、依赖关系以及任务的生命周期等信息. 任务之间的依赖关系以有向无环图形式表示，反映了实际生产环境中任务的复杂性和动态性. 考虑任务规模对调度性能的影响，本文选择TPC-H和Alibaba cluster-trace-v2018数据集中数据规模为2 GB、20 GB、50 GB、80 GB、100 GB的任务进行实验. 不同数据规模的任务类别均为22.

考虑任务的截止期约束，评价指标设为截止期达成率$R=\frac{N-N_d}{N}\times 100\%$. 其中$N_d$为截止期前完成的任务数量. 截止期达成率衡量调度策略在规定时间内完成任务的能力，较大的R表明当前策略能够使得更多的任务在截止期前完成. 为进一步表征所提算法的资源占用情况，通过记录算法运行过程中每个时刻t的显存占用量VM_t，从

而得到算法运行时间T范围内的平均显存占用率$U=\frac{1}{T}\sum_{t=1}^{T}\frac{VM_t}{VM}$，其中$VM$为总的显存容量. 越小的$U$表明当前策略所占用的显存量越少，有较低的资源占用量.

2.2 消融实验

由于基于依赖系数的图注意力网络模块和基于指数平滑的元学习模块对MGS算法的性能有较大影响，考虑图注意力网络（Graph Attention Networks, GAT）和元学习方法（Meta-Learning, ML）以及所提改进方法对MGS算法进行消融实验. 假定DGAT和ESML分别为本文所提基于依赖系数的图注意力网络模块（Dependency Graph Attention Networks, DGAT）和基于指数平滑的元学习模块（Exponentially Smoothed Meta-Learning, ESML），则共有GAT-ML、GAT-ESML、DGAT-ML和DGAT-ESML四种消融算法. 其中DGAT-ESML为本文所提MGS方法，即采用基于依赖系数的图注意力网络模块和基于指数平滑的元学习模块进行任务调度优化.

图2给出TPC-H数据集中不同消融算法在不同任务规模下的截止期到达率，横坐标表示任务规模，纵坐标表示截止期到达率. 由图2可知，不同任务规模下，GAT-ML的截止期到达率均为同类任务规模的最低值，主要原因在于GAT所得任务特征缺乏子任务间的依赖关联，无法得到准确的任务表征；同时，采用插值法的ML导致任务性能波动较大，无法较快收敛，导致任务完成时间较长，未能满足截止期依赖. DGAT-ML在不同任务规模下的截止期到达率明显优于GAT-ML，进一步表明DGAT可以通过子任务间的依赖系数得到较优的任务表征，从而得到较优的调度策略. GAT-ESML在不同任务规模下的性能略优于GAT-ML，表明采用指数平滑的元学习方法在时间依赖的调度策略学习中较插值法有一定的优势. 在不同任务规模中，本文所提策略MGS在同类任务规模中可得到最高的截止期到达率，表明DGAT方法所得综合依赖系数的任务表征能够学习到更全面的任务间依赖关系以及资源间的相互竞争关系，更有利于后续的任务调度结果生成；同时，采用ESML策略进行模型的参数更新，能够加快模型收敛速度，降低任务调度时间，保证较多任务能在截止期前完成. 随着任务规模的不断增加，所有消融算法的截止期到达率均在降低，而本文所提MGS算法下降速度最慢，表明该算法较其他消融策略更适用于大规模任务的调度优化. 图3给出Alibaba cluster-trace-v2018数据集中不同消融算法在不同任务规模下的截止期到达率，各消融算法在Alibaba cluster-trace-v2018数据集中的表现如同TPC-H数据集.

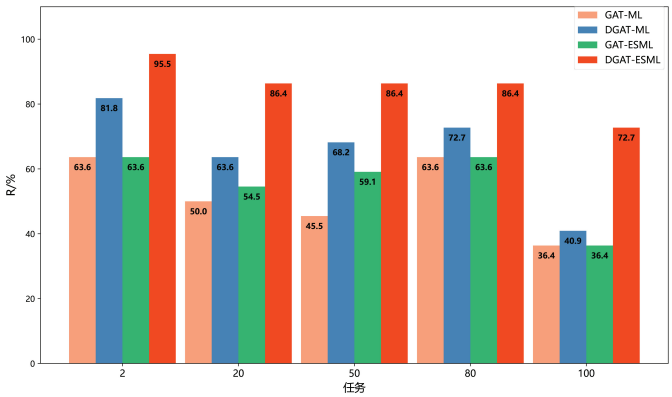

图 2 TPC-H数据集中不同消融算法在不同任务规模下的截止期到达率

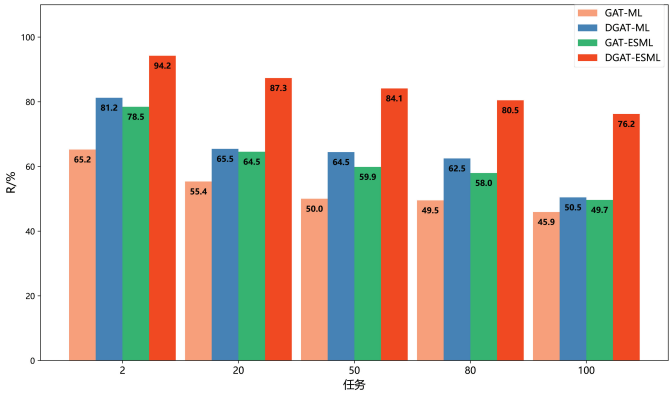

图 3 Alibaba cluster-trace-v2018数据集中不同消融算法在不同任务规模下的截止期到达率

图4、图5分别给出不同消融算法在TPC-H和Alibaba cluster-trace-v2018数据集中的显存占用率，其中横坐标为任务规模、纵坐标为U. 随着任务规模的增加，占用率呈增长趋势. DGAT-ESML的显存占用率在两个数据集中最低，主要原因在于DGAT-ESML采用依赖系数进行子任务间关联关系的表征，并采用指数平滑元学习策略动态调整模型参数，二者共同提高调度效率、减少作业完成时间，从而降低资源占用率. GAT-ML的显存占用率在两个数据集中最高，主要原因在于GAT-ML未考虑子任务间的依赖关系，导致任务调度不合理，从而出现大量时间段的显存占用. DGAT-ML采用依赖系数对子任务间的关联关系进行表征再进行调度优化，其显存占用率较GAT-ML有所下降，同时插值法元学习所导致的参数波动极易造成资源浪费. GAT-ESML采用指数平滑元学习策略替代插值法，能够动态调整参数以减弱资源占用的波动性，从而降低显存占用率，其U值低于DGAT-ML. 随着任务规模扩大，DGAT-ESML的显存占用率增速较GAT-ML低，验证了所提策略对大规模任务的调度优化能力. 通过消融实验，表明DGAT-ESML在保证较多任务在截止期前完成的同时，显著降低了复杂依赖场景下的资源占用量.

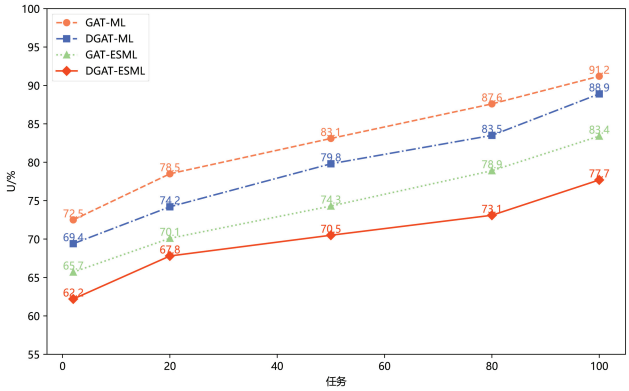

图 4　TPC-H数据集中不同消融算法在不同任务规模下的显存占用率

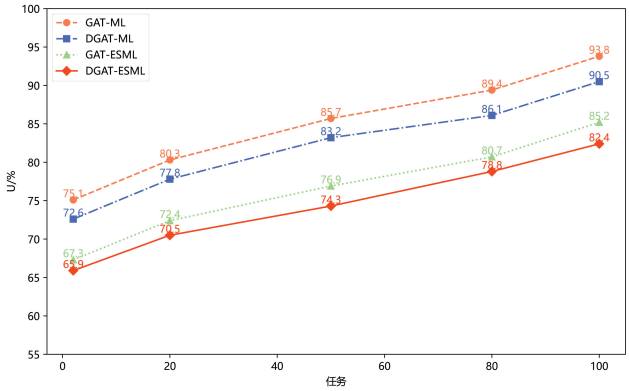

图 5　Alibaba cluster-trace-v2018数据集中不同消融算法在不同任务规模下的显存占用率

2.3　算法比较

为验证所提MGS算法的有效性，与现有深度学习算法GAACO[26]、DRL-LSTM[27]和RATS-HM[28]进行比较. 由于GAACO、DRL-LSTM和RATS-HM无法解决带截止期约束的复杂依赖任务的调度优化，对其进行简单修改，使其适用于本文所需解决的问题.

由表1可知，任意数据集的所有任务规模中，RATS-HM所得截止期到达率最低（在TPC-H和Alibaba cluster-trace-v2018数据集中的平均截止期到达率分别为75.9%和75.6%），其调度策略是基于启发式规则给出的，而固定的启发式规则无法适用于所有任务规模所有依赖关系的任务调度，难以精准完成复杂依赖关联的任务调度. GAACO和DRL-LSTM在两个数据集不同任务规模中的算法性能相似，二者均采用时间序列深度学习网络结构进行任务特征表征，进而对任务与资源进行匹配，他们通过较为精准的任务表征后所得调度策略性能优于RATS-HM. MGS算法所得截止期到达率高于其他算法，其平均截止期到达率可达85%左右，主要原因在于MGS综合考虑了子任务间的依赖关系、任务间的关联关系以提取任务特征，从而根据高效的任务特征得到精

准的调度策略；在模型参数更新过程中，采用基于指数平滑的元学习模块，更适用于时间序列依赖的任务调度，同时提高模型的整体收敛速度.

表 1 比较算法在TPC-H和Alibaba cluster-trace-v2018数据集不同任务规模中的截止期到达率

任务规模		MGS/%	GAACO/%	DRL-LSTM/%	RATS-HM/%
TPC-H	2 GB	**95.5**	91.0	86.4	86.4
	20 GB	**86.4**	79.6	81.8	79.6
	50 GB	**86.4**	79.6	77.3	75.0
	80 GB	**86.4**	77.2	76.5	72.7
	100 GB	**72.7**	70.0	68.2	65.9
	平均	**85.5**	79.5	78.0	75.9
Alibaba cluster-trace-v2018	2 GB	**94.2**	89.5	87.2	85.6
	20 GB	**87.3**	81.6	80.1	79.3
	50 GB	**84.1**	78.4	76.2	74.9
	80 GB	**80.5**	73.5	72.4	71.2
	100 GB	**76.2**	70.1	69.6	67.0
	平均	**84.5**	78.6	77.1	75.6

注：加粗字体表示性能最优值，下同

由表2可知，所提MGS算法在两个数据集不同任务规模中的表现均为最好，占用的显存资源最少，在TPC-H和Alibaba cluster-trace-v2018数据集中其平均显存占用量为70.3%和74.4%. MGS采用依赖系数表征子任务间关联关系的同时，还采用指数平滑元学习策略对模型参数进行动态调整，优化子任务与资源的匹配，改善了任务完成时间，也降低了资源占用量. 基于启发式规则的RATS-HM能够较好完成任务调度，但复杂的子任务间依赖关系难以表征，从而无法避免较多资源竞争关系，导致较长的任务完成时间，同时增加了资源占用量. GAACO融合了深度学习网络结构与进化搜索策略，复杂的网络结构模型导致显存开销快速上升，增加了显存占用量. 而DRL-LSTM在所有数据集不同任务规模中所得显存占用量最大，主要原因在于其采用LSTM网络结构进行时序特征提取时需缓存较多历史状态信息，模型参数量和中间状态存储需求使显存占用率始终处于较高水平. 此外，由任务规模2 GB到100 GB的显存占用量可知，所有比较算法的显存占用量持续增加. 由于基于依赖系数的关联关系表征可准确提取资源与任务间的特征，同时基于平滑指数的元学习可动态更新模型参数，减少中间状态的存储需求，所提MGS算法的增加速率最小.

表 2 比较算法在TPC-H和Alibaba cluster-trace-v2018数据集不同任务规模中的显存占用率

任务规模		MGS/%	GAACO/%	DRL-LSTM/%	RATS-HM/%
TPC-H	2 GB	**62.2**	68.5	73.2	65.1
	20 GB	**67.8**	75.3	80.7	78.4
	50 GB	**70.5**	82.1	86.9	85.2
	80 GB	**73.1**	87.6	90.4	89.7
	100 GB	**77.7**	92.3	94.8	93.5
	平均	**70.3**	81.2	85.2	82.4
Alibaba cluster-trace-v2018	2 GB	**65.9**	70.2	75.8	68.4
	20 GB	**70.5**	77.6	83.1	81.3
	50 GB	**74.3**	85.4	88.7	87.9
	80 GB	**78.8**	90.1	92.6	92.0
	100 GB	**82.4**	94.5	96.2	95.8
	平均	**74.4**	83.6	87.3	85.1

考虑算法执行时间对调度优化策略的重要影响，表3给出所有比较算法在TPC-H和Alibaba cluster-trace-v2018数据集不同任务规模中的算法执行时间. 由表3可知，随着任务规模的不断增加，所有对比算法的执行时间均增加. MGS算法的平均执行时间最小（在TPC-H和Alibaba cluster-trace-v2018数据集中的平均执行时间

分别为45.6 s和50.6 s），主要原因在于该算法借鉴了元学习方法，能够通过采样学习得到较好的参数初始值，同时使用指数平滑策略对其进行改进，降低了模型的迭代次数，从而减少执行时间. 此外，RATS-HM算法执行时间较GAACO和DRL-LSTM少（在TPC-H和Alibaba cluster-trace-v2018数据集中的平均执行时间分别减少2.4 s和3.0 s），这是因为启发式规则所得通常是较优解，不需要花费较多时间计算最优解；而基于深度网络结构的GAACO和DRL-LSTM需要得到最优解，从而花费较多的执行时间；表1中RATS-HM、GAACO和DRL-LSTM的性能亦可得到相应结论. 表3中，两个数据集的任务规模为2 GB时，DRL-LSTM算法的执行时间最长；任务规模上升为20 GB时，MGS的执行时间最长；任务规模为50 GB、80 GB和100 GB时，GAACO的执行时间最长. 主要原因在于DRL-LSTM算法为两个深度网络结构的融合，策略的执行时间明显长于其他算法，故在任务规模较小时，所得算法执行时间最长. 中等任务规模时，MGS需要通过大量计算来融合子任务、任务间的复杂依赖关系，从而得到任务和资源的精准特征表示，占用了较长时间，但由表1可知，MGS算法所得解是最优的. GAACO算法融合了深度学习网络和进化学习算法，在较大任务规模时进化学习算法的执行时间较长，导致时间增加. 结合表1、表3可知，本文所提MGS算法能够在较短时间内得到最优的调度策略.

表 3　比较算法在TPC-H和Alibaba cluster-trace-v2018数据集不同任务规模中的算法执行时间

任务规模		MGS/s	GAACO/s	DRL-LSTM/s	RATS-HM/s
TPC-H	2 GB	**17**	20	25	19
	20 GB	40	39	**36**	**36**
	50 GB	**50**	62	54	52
	80 GB	**57**	71	67	67
	100 GB	**64**	85	89	88
	平均	**45.6**	55.4	54.2	52.4
Alibaba cluster-trace-v2018	2 GB	**19**	22	28	21
	20 GB	43	41	39	**37**
	50 GB	**58**	66	61	60
	80 GB	**63**	77	74	74
	100 GB	**70**	90	84	84
	平均	**50.6**	59.2	57.2	55.2

3　结论和未来工作

针对带截止期约束的复杂依赖任务的调度优化，考虑依赖子任务间的传输时间和计算时间均衡化以及模型参数学习效率问题，提出基于图注意力网络和元学习的调度优化策略. 通过设计基于依赖系数的图注意力模块来综合子任务间、任务间的特征表示和相互依赖关系，以有效提取异构数据间的紧密关联；同时利用指数平滑改进的元学习方法进行网络参数优化，避免原有插值计算过程导致的模型波动性，以更好适应时序关联的任务调度，从而提高模型的适应性. 通过实验比较，所提调度策略在截止期前完成的作业数量比率平均提升7.52%. 鉴于人工智能算法对调度优化的重大影响，本文仅在任务特征提取层面引入深度强化学习思想，后续在任务与资源的匹配阶段还要进一步尝试人工智能策略，以得到更大的性能提升.

参考文献：

[1] KARUNATHILAKE E M B M，LE A T，HEO S，et al. The path to smart farming：Innovations and opportunities in precision agriculture[J]. Agriculture，2023，13(8)：1593.

[2] KADKHODAEI H，MOGHADAM A M E，DEHGHAN M. Big data classification using heterogeneous ensemble classifiers in Apache Spark based on MapReduce paradigm[J]. Expert Systems with Applications，2021，183：115369.

[3] ABDEL-FATTAH M A，OTHMAN N A，GOHER N. Predicting chronic kidney disease using hybrid machine learning based on Apache Spark[J]. Computational Intelligence and Neuroscience，2022，2022：9898831.

[4] DEBAUCHE O，MAHMOUDI S，MANNEBACK P，et al. Cloud and distributed architectures for data management in agriculture 4.0：Review and future trends[J]. Journal of King Saud University - Computer and Information Sciences，2022，34(9)：7494-7514.

[5] SUDNICINA K. Task-in-pod scheduling support for Kubernetes and Apache Spark stack[D]. Amsterdam：Vrije University Amsterdam，2024.

[6] LATTUADA M, BARBIERATO E, GIANNITI E, et al. Optimal resource allocation of cloud-based Spark applications[J]. IEEE Transactions on Cloud Computing, 2022, 10(2): 1301-1316.

[7] SANTOS J, WANG C, WAUTERS T, et al. Diktyo: Network-aware scheduling in container-based clouds[J]. IEEE Transactions on Network and Service Management, 2023, 20(4): 4461-4477.

[8] TANG X Y, CAO W B, TANG H Y, et al. Cost-efficient workflow scheduling algorithm for applications with deadline constraint on heterogeneous clouds[J]. IEEE Transactions on Parallel and Distributed Systems, 2022, 33(9): 2079-2092.

[9] HUSSAIN M, LUO M X, HUSSAIN A, et al. Deadline-constrained cost-aware workflow scheduling in hybrid cloud[J]. Simulation Modelling Practice and Theory, 2023, 129: 102819.

[10] MEDARA R, SINGH R S, SOMPALLI M. Energy and cost aware workflow scheduling in clouds with deadline constraint[J]. Concurrency and Computation: Practice and Experience, 2022, 34(13): e6922.

[11] RAJPUT K Y, LI X P, ZHANG J Q, et al. A novel scheduling approach for Spark workflow tasks with deadline and uncertain performance in multi-cloud networks[J]. IEEE Transactions on Cloud Computing, 2024, 12(4): 1145-1157.

[12] ISLAM M T, WU H M, KARUNASEKERA S, et al. SLA-based scheduling of Spark jobs in hybrid cloud computing environments[J]. IEEE Transactions on Computers, 2022, 71(5): 1117-1132.

[13] MIRZA N M, ALI A, MUSA N S, et al. Enhancing task management in Apache Spark through energy-efficient data segregation and time-based scheduling[J]. IEEE Access, 2024, 12: 105080-105095.

[14] LI Y N, LI J B, PANG J J. A graph attention mechanism-based multiagent reinforcement-learning method for task scheduling in edge computing[J]. Electronics, 2022, 11(9): 1357.

[15] ZHANG X D, LI X P, DU H A, et al. Task scheduling for Spark applications with data affinity on heterogeneous clusters[J]. IEEE Internet of Things Journal, 2022, 9(21): 21792-21801.

[16] LEE H, CHO S, JANG Y, et al. A global DAG task scheduler using deep reinforcement learning and graph convolution network[J]. IEEE Access, 2021, 9: 158548-158561.

[17] GU R, CHEN Y Q, LIU S, et al. Liquid: Intelligent resource estimation and network-efficient scheduling for deep learning jobs on distributed GPU clusters[J]. IEEE Transactions on Parallel and Distributed Systems, 2022, 33(11): 2808-2820.

[18] CAPEL M I, SALGUERO-HIDALGO A, HOLGADO-TERRIZA J A. Parallel PSO for efficient neural network training using GPGPU and Apache Spark in edge computing sets[J]. Algorithms, 2024, 17(9): 378.

[19] CHEN J, CHEN H P. Edge-featured graph attention network[EB/OL]. 2021: 2101.07671. https://arxiv.org/abs/2101.07671v1.

[20] LEE S H, JI F, TAY W P. SGAT: Simplicial graph attention network[EB/OL]. 2022: 2207.11761. https://arxiv.org/abs/2207.11761v1.

[21] THOST V, CHEN J. Directed acyclic graph neural networks[EB/OL]. 2021: 2101.07965. https://arxiv.org/abs/2101.07965v3.

[22] NICHOL A, ACHIAM J, SCHULMAN J. On first-order meta-learning algorithms[EB/OL]. 2018: 1803.02999. https://arxiv.org/abs/1803.02999v3.

[23] BECHTLE S, MOLCHANOV A, CHEBOTAR Y, et al. Meta learning via learned loss[C]//2020 25th International Conference on Pattern Recognition (ICPR). January 10-15, 2021. Milan, Italy. IEEE, 2021: 4161-4168.

[24] TPC. The TPC-H benchmarks[EB/OL]. (2017-11-01)[2024-11-20]. https://www.tpc.org/tpch/.

[25] Alibaba. Cluster data collected from production clusters in Alibaba[EB/OL]. (2021-09-02)[2024-11-20]. https://developer.aliyun.com/article/789333.

[26] ZHANG Y F, LIU B, GONG Y L, et al. Application of machine learning optimization in cloud computing resource scheduling and management[C]//Proceedings of the 5th International Conference on Computer Information and Big Data Applications. Wuhan, China. ACM, 2024: 171-175.

[27] RJOUB G, BENTAHAR J, ABDEL WAHAB O, et al. Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems[J]. Concurrency and Computation: Practice and Experience, 2021, 33(23): e5919.

[28] BAL P K, MOHAPATRA S K, DAS T K, et al. A joint resource allocation, security with efficient task scheduling in cloud computing using hybrid machine learning techniques[J]. Sensors, 2022, 22(3): 1242.

责任编辑：张自强