基于深度强化学习的智能车间调度方法研究

罗梓珲，江呈羚，刘亮，郑霄龙，马华东
（北京邮电大学计算机学院（国家示范性软件学院），北京 100876）

摘　要：工业物联网的空前繁荣为传统的工业生产制造模式开辟了一条新的道路。智能车间调度是整个生产过程实现全面控制和柔性生产的关键技术之一，要求以最大完工时间最小化分派多道工序和多台机器的生产调度。首先，将车间调度问题定义为马尔可夫决策过程，建立了一个基于指针网络的车间调度模型。其次，将作业调度过程看作是从一个序列到另一个序列的映射，提出了一种基于深度强化学习的车间调度算法。通过分析模型在不同参数设置下的收敛性，确定了最优参数。在不同规模的公共数据集和实际生产数据集上的实验结果表明，所提出的深度强化学习算法能够取得更好的性能。

关键词：工业物联网；智能车间调度；柔性生产；深度强化学习；车间调度方法
中图分类号：TP18
文献标志码：A
doi:10.11959/j.issn.2096−3750.2022.00260

Research on deep reinforcement learning based intelligent shop scheduling method

LUO Zihui, JIANG Chengling, LIU Liang, ZHENG Xiaolong, MA Huadong
School of Computer Science (National Pilot Software Engineering School), Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract: The unprecedented prosperity of the industrial internet of things (IIoT) has opened up a new path for the traditional industrial manufacturing model. Intelligent shop scheduling is one of the key technologies to achieve the overall control and flexible production of the whole production process. It requires an effective plan with a minimum makespan to allocate multiple processes and multiple machines for production scheduling. Firstly, the shop scheduling problem was defined as a Markov decision process (MDP), and a shop scheduling model based on the pointer network was established. Secondly, the job scheduling process was regarded as a mapping from one sequence to another, and a new shop scheduling algorithm based on deep reinforcement learning (DRL) was proposed. By analyzing the convergence of the model under different parameter settings, the optimal parameters were determined. Experimental results on different scales of public data sets and actual production data sets show that the proposed DRL algorithm can obtain better performances.

Key words: IIoT, intelligent shop scheduling, flexible production, deep reinforcement learning, shop scheduling method

0　引言

工业物联网[1]（IIoT, industrial internet of things）作为新一代信息技术与制造业深度融合的产物，通过实现人、机、物的全面互联，为传统的工业生产制造模式开辟了一条新的道路。IIoT 中设想的智能车间如图 1 所示，采用“云–边–端”3 层架构，终端各种类型的传感设备综合感知实时生产数据，并通过无线传感器网络实时传输到边缘服务器，在边缘端使用云端训练好的车间调度模型结合订单、物料

收稿日期：2021−09−22；**修回日期**：2022−01−21
通信作者：刘亮，liangliu@bupt.edu.cn
基金项目：国家自然科学基金资助项目（No.62061146002，No.61632008，No.61921003）；中央高校基本科研业务费资助项目（No.2019XD-A14）
Foundation Items: The National Natural Science Foundation of China (No.62061146002, No.61632008, No.61921003), The Fundamental Research Funds for the Central Universities (No.2019XD-A14)

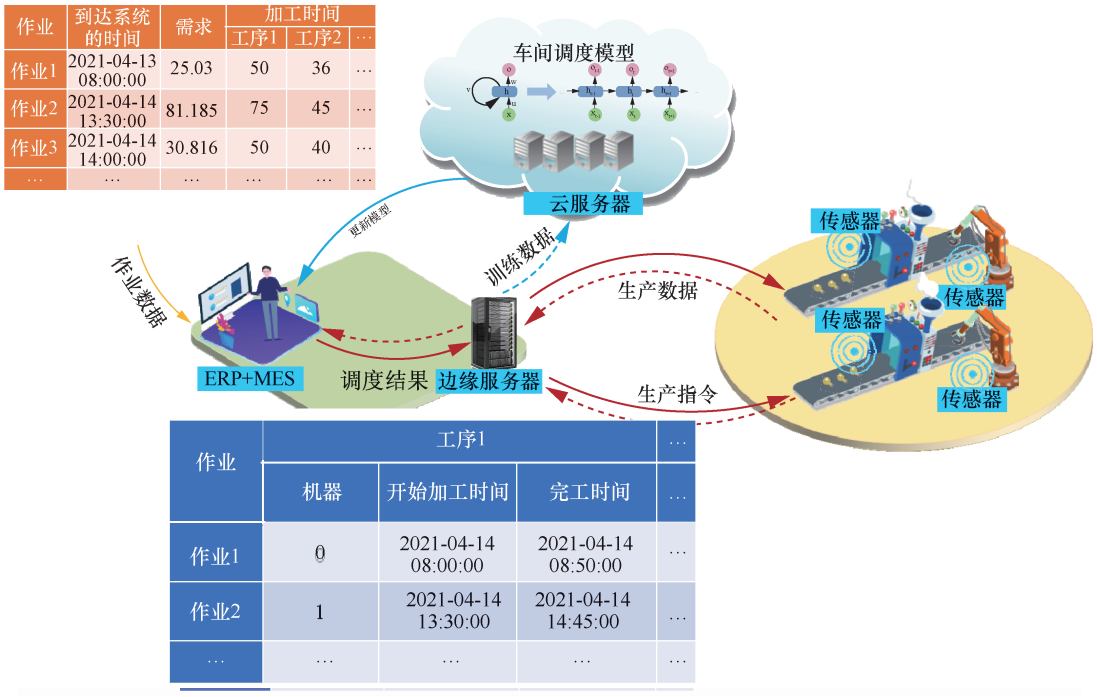

图 1　IIoT 中设想的智能车间

等系统，对等待生产的作业进行快速排产调度，然后将调度结果通过生产指令的方式下发到生产线执行，以实现整个生产过程的自动化、智能化、无人化。图 1 中虚线表示从下层到上层（如端到边、边到云）的数据传输，实线表示从上层到下层的数据传输。显然，在云服务器中利用不断获取的新数据定期训练更新的车间调度模型决定了整个生产计划的性能，能直接影响生产效率，但是现有的车间调度方法还不能很好地满足 IIoT 的愿景。因此，设计一种高效、智能的车间调度方法具有重要的紧迫性和现实意义。

车间调度问题可以看作从一个待调度作业的输入序列到一个调度结果的输出序列的映射。Vinyals 等[2]提出了指针网络（PN, pointer network），利用循环神经网络（RNN, recurrent neural network）和注意力机制来解决序列到序列（Seq2Seq）的建模问题，并应用于求解旅行商问题（TSP, traveling salesman problem）。Ling 等[3]通过全卷积网络（FCN, fully convolutional network）学习从可行域到最优解的映射，在中小规模的 TSP 上取得了不错的效果。然而，PN 和 FCN 需要通过有监督的方式训练模型，其性能在很大程度上取决于标签的质量，但在 IIoT 场景下对大量数据进行手工标注是不现实的，这使得监督学习不适用。区别于有监督学习，强化学习（RL, reinforcement learning）是一种数据驱动的方法，它可以在没有标签的情况下与环境交互学习稳定的策略。Nazari 等[4]采用 RL 提出了一个端到端的框架解决车辆路径问题（VRP, vehicle routing problem），并应用策略梯度算法优化模型参数，证明了在中等规模问题上解决方案的有效性。然而，在车间调度问题中，每个作业都具有多维的动态特征，这是 RL 难以处理的。

近年来，随着人工智能的发展，结合了深度学习提取高维特征能力和 RL 决策学习能力的深度强化学习（DRL, deep reinforcement learning）得到了科研人员的广泛重视，其可以处理高维状态空间和高维动作空间下的决策问题，并自主挖掘问题的特征，只用原始输入而不用人工特征来生成输出，从而实现端到端学习，已经广泛应用于游戏、机器人、自动驾驶和对话系统等领域，成功地解决了各种实际问题[5-7]，但在组合优化领域的研究还很少[8]。受此启发，本文采用基于 DRL 的智能方法解决车间调度问题。本文的模型首先利用接收的作业数据和生产线采集的生产数据在云端服务器进行训练，然后将训练好的模型部署到离车间更近的边缘端服务器，从而可以结合实时的作业数据和生产数据生成相应的动态调度结果，优化产线的生产计划。同时，利用不断获取的数据，可以定期优化更新车间

调度模型。本文的主要贡献包括以下 3 个方面。

1) 将车间调度问题定义为马尔可夫决策过程，建立了一个基于 PN 的车间调度模型，该模型包含一个编码器和一个解码器，可以利用工业物联网中大量的无标签数据更新优化模型。

2) 将作业调度过程看作从一个序列到另一个序列的映射，提出了一种基于 DRL 的智能车间调度算法，该算法包含一个演员（actor）网络和一个评论家（critic）网络，可以适用于不同规模的问题和不同类型的应用场景。

3) 利用公共数据集和实际的生产数据集进行仿真实验分析，与不同类型的算法进行了对比，实验结果验证了算法的有效性。

1　相关工作

流水车间调度问题（FSP, flow-shop scheduling problem）是一个经典的 NP 难（NP-hard）问题[9]，它关注的是每道工序只有一台机器的场景。自 1954 年 Johnson[10]发表了第一篇关于 FSP 的开创性论文以来，车间调度问题在过去近 70 年里得到了相当多的研究。Reza 等[11]对 Johnson 的从早期到 2004 年之间以最大完工时间（makespan）为准则的 FSP 研究进行了完整的综述。Zhang 等[12]在工业 4.0 的视角下对超过 120 篇论文进行了回顾，并讨论了新技术在传统调度转化为智能分布式调度过程中可以发挥的作用。

如果存在至少一道工序包含多台并行机器，则该问题变为混合流水车间调度问题（HFSP, hybrid flow- shop scheduling problem），其同样也是 NP 难问题[13]。HFSP 最早由 Arthanary 等[14]于 1971 年提出，由于其复杂性和在现实世界中的重要应用，它得到了广泛的研究。Ruiz 等[15]回顾了 2010 年之前的相关文献，并简要讨论了 HFSP 的几种变种，每种变种都考虑了不同的假设、约束和目标函数。Tosun 等[16]分析了过去 10 年间的 219 篇有关 HFSP 的文章，他们主要关注的是智能优化方法。Li 等[17]系统总结了实际工程背景下 HFSP 的研究现状，指出了当前研究中存在的问题和可能的解决途径以及未来若干可能的研究方向。

在过去，解决车间调度问题的方法主要分为两大类[18]：精确算法和近似算法。精确算法[19,-20]主要采用基于析取图模型的枚举方法，或基于活动调度生成、混合整数规则模型等的方法，对解决小规模问题时有很好的效果，但由于车间调度问题的 NP 难特性，它们难以求解中大规模问题，无法在实际生产中应用。近似算法包含启发式算法和智能优化算法。基于优先权的规则调度算法[21]、CDS（Campbell-Dudek-Smith）算法[22]等启发式算法由于其易于实现、计算复杂度低，能快速构造求解问题的解，可用于实时调度系统，多年来被学者们广泛研究，但是缺乏优化的全局性，往往不能保证所得到的调度方案解的质量，特别是对于复杂工况下的车间调度问题，缺乏对问题性质的深度挖掘，使得很难设计出高效的启发式算法。近些年，遗传算法[23]（GA, genetic algorithm）、蚁群优化[24]（ACO, ant colony optimization）算法和人工蜂群[25]（ABC, artificial bee colony）算法等智能优化算法或改进的组合算法因能在合理的时间内取得较优解倍受关注，并已经成为了解决车间调度问题的重要手段。尽管智能优化算法在诸多车间调度问题上可以取得近似较优解，但由于在大规模问题上收敛到局部最优点的迭代搜索过程非常耗时，且很少有算法能利用历史信息学习先验知识、快速调整其搜索行为，这导致了算法的性能严重依赖于设计者的经验，因而在大规模问题上还有很大的提升空间。

Cunha 等[26]对作业车间调度、进化算法和深度强化学习的研究现状进行了综述，提出了一种利用 DRL 解决作业车间调度问题的新体系结构。Liu 等[27]提出了一种结合异步更新和深度确定性策略梯度的并行训练方法来训练包含演员网络和评论家网络的模型，Han 等[28]基于析取图调度提出了一种 DRL 框架可直接根据输入的制造状态学习行为策略，Wang 等[29]采用近端策略优化寻找最优调度策略得到动态作业车间调度问题的最优解。Luo[30]将深度 Q-网络与提出的 6 个组合调度规则结合，解决了在新作业插入下以总时延最小为目标的动态柔性作业车间调度问题。Zhang 等[31]提出了一种基于图神经网络的方案来嵌入求解实际作业车间调度问题，验证了 DRL 智能体（agent）训练学习得到的策略网络是与大小无关的，能有效支持大规模实例的泛化。王凌等[32]以最小化最大完工时间为目标，提出了求解 FSP 的一种基于 DRL 与迭代贪婪算法的框架。Luo 等[33]提出了一种包含勘探回路和开发回路的双环深度 Q-网络的方法解决作业随机到达情况下的作业车间调度问题。总的来说，现有采用 DRL 方法的工作只关注于每道工序仅包含一台机器的作业或流水车间调度问题，对于状态空间

和动作空间更为复杂的每道工序包含多台并行机器的 HFSP 还需要更进一步的研究。

2 问题描述和数学表达

2.1 符号解释

（1）索引

i：作业索引，$i \in \{1,2,\cdots,N\}$；

k：工序索引，$k \in \{1,2,\cdots,K\}$；

m：工序 k 的机器索引，$m \in \{1,2,\cdots,M_k\}$。

（2）参数

N：作业总数量；

K：工序总数量；

M_k：工序 k 的机器总数量；

rt_i：作业 i 到达系统的时间；

st_{ik}：作业 i 在工序 k 的可开始加工时间；

st_{ikm}：作业 i 在工序 k 的机器 m 上的开始加工时间；

pt_{ikm}：作业 k 在工序 k 的机器 m 上的加工时间；

ct_{ik}：作业 i 在工序 k 的完工时间；

ct_{ikm}：作业 i 在工序 k 的机器 m 上的完工时间；

$C_{\max}$：最大完工时间。

（3）决策变量

$$X_{ikm}=\begin{cases}1, & \text{工序}k\text{时作业}i\text{在机器}m\text{上加工}\\ 0, & \text{其他}\end{cases}$$

$$S_{ikt}=\begin{cases}1, & \text{工序}k\text{上作业}i\text{在}t\text{时刻时正在加工}\\ 0, & \text{其他}\end{cases}$$

$$O_{jikm}=\begin{cases}1, & \text{作业}j\text{在工序}k\text{机器}m\text{上加工的}\\ & \text{分派次序在作业}i\text{之前}\\ 0, & \text{其他}\end{cases}$$

$$\mathrm{mask}_i=\begin{cases}1, & \text{作业}i\text{等待被分派到一台机器}\\ 0, & \text{作业}i\text{已经被分派到一台机器}\end{cases}$$

2.2 问题描述

本文主要考虑了在工业物联网中一类典型的车间调度问题，其广泛存在于炼钢、纺织、机械、半导体等行业。具体来说，给定 N 个作业的集合 $J=\{1,2,\cdots,N\}$ 和 K 道工序，每道工序 k 有 $M_k(M_k \geqslant 1)$ 台机器 $m(1 \leqslant m \leqslant M_k)$，每个作业 $i(1 \leqslant i \leqslant N)$ 需要在每道工序 k 依次加工。FSP 的各道工序 k 只有一台机器，HFSP 的各道工序 k 有一台或多台机器且必须有不少于一道工序拥有两台及以上机器，各个作业必须且只需要在每道工序的任意一台机器上加工一次。已知作业 i 在各道工序机器 m 的加工时间 pt_{ikm} 和作业到达系统的时间 rt_i（即在第一道工序的可开始加工时间 st_{i1}），因此，在 FSP 中本文只需要确定作业 i 在各道工序 st_{i1} 的加工次序即可计算每个作业 i 在各道工序 i 的开始加工时间 st_{ik} 和完工时间 ct_{ik}，而在 HFSP 包含多台机器的工序，本文还需要确定作业 i 的加工机器，才能计算在各道工序 k 上机器 m 的开始加工时间 st_{ikm} 和完工时间 ct_{ikm}。

在实际生产中，有的车间调度问题不属于标准的 FSP 或 HFSP，其中有些作业在加工时存在工序跳跃现象，这时为了使所有作业在加工时都能符合标准模式，本文可以假设那些需要跳跃的工序也需要进行加工并将加工时间设置为 0，同时为了减少对其他作业加工次序的影响，可以设置虚拟的专用机器用于加工这些作业跳跃的工序。

2.3 问题的数学表达

本文合理地假设每个作业 i 到达系统的时间 rt_i 是已知的，那么作业 i 在第一道工序的开始加工时间 st_{i1} 不能早于 rt_i，可表示为

$$\mathrm{st}_{i1} \geqslant \mathrm{rt}_i, \forall i \tag{1}$$

任意作业 i 只有在前一道工序 $k-1$ 的加工完成后才能开始下一道工序 k 的加工，并且本文假设缓冲区容量为无穷大且在开始加工前允许等待，即分派到机器 m 的作业 i 在机器非空闲状态时需要等待。因此，作业 i 在工序 k 的可开始加工时间 st_{ik} 应等于该作业在上一道工序 $k-1$ 的完工时间与已经分派到同一台机器的其他作业的完工时间的最大值，即

$$\mathrm{st}_{ik}=\max(\mathrm{ct}_{i(k-1)}, O_{jikm}\mathrm{ct}_{jk}), \forall i \neq j, k=2,3,\cdots,K \tag{2}$$

本文假设作业 i 在相邻工序无转运等额外时间且不能中断，在各道工序 k 的机器 m 上的加工时间 pt_{ikm} 是已知且确定的。因此，作业 i 在工序 k 的完工时间 ct_{ik} 等于该工序的可开始加工时间 ct_{ik} 加上分派到该工序待加工机器 m 的加工时间，可表示为

$$\mathrm{ct}_{ik}=\mathrm{st}_{ik}+\mathrm{pt}_{ikm}, \forall i,k \tag{3}$$

同时，任意作业 i 在任一工序 k 只能在一台机器上加工，且最多只能加工一次，如果该作业在该工序存在工序跳跃，则采用虚拟机器进行加工时间为 0 的虚拟加工，可以忽略不计，即

$$\sum_{m=1}^{M_k} X_{ikm}=1, \forall i,k \tag{4}$$

$$\sum_{t=1}^{T} S_{ikt} = 1, \forall i,k \tag{5}$$

最后，任一工序 k 的机器 m 在同一时刻最多只能加工一个作业，即

$$S_{ikt} X_{ikm} \sum_{j=1}^{N} \sum_{t=\mathrm{st}_{ik}}^{\mathrm{ct}_{ik}} S_{jkt} X_{jkm} = 0, \forall i,j,k,t,m, i \neq j \tag{6}$$

因此，任意作业 i 在工序 k 分派的机器 m 上加工的完工时间 ct_{ikm} 等于该作业在该道工序的完工时间，开始加工时间 st_{ikm} 等于该作业在该道工序的完工时间 ct_{ikm} 与加工时间 pt_{ikm} 的差值，即

$$\mathrm{ct}_{ikm} = \mathrm{ct}_{ik}, \forall i,k,m \tag{7}$$

$$\mathrm{st}_{ikm} = \mathrm{ct}_{ikm} - \mathrm{pt}_{ikm}, \forall i,k,m \tag{8}$$

本文的优化目标是让最迟完工的作业尽早完成加工，以尽快使所有作业完成交货，即最小化最后一道工序的最大完工时间，因此，整个问题可以表示为一个混合整数规划模型

$$\min C_{\max} = \max_{1 \leqslant i \leqslant N} \mathrm{ct}_{iK} \tag{9}$$

$$\text{s.t. } (1) \sim (6)$$

3　车间调度模型和算法

3.1　马尔可夫决策过程

本文将车间调度问题定义为马尔可夫决策过程（MDP, Mark decision process）进行建模。MDP 是一个四元组$(S,A,P_{s,a},R)$，其中 S 表示状态集，A 表示动作集，$P:S\times A\mapsto[0,1)$ 表示在状态 s 下采取动作 a 的状态转移矩阵，以及 $R:S\times A\times S'\mapsto R$ 表示在状态 s 下采取动作 a 后转移到状态 s' 获取的期望（立即）奖励。具体来说，在每个时间步 t，agent 通过观察当前的环境状态 s_t 并采取动作 a_t，agent 根据自己采取的动作获取相应的奖励 R。然后环境状态从 s_t 转换到 s_{t+1}，agent 不断地与环境进行交互并采取动作，以获得最大的累积奖励。

3.1.1　状态

车间调度模型在时间步 t 时的状态 s_t 为元组 $(\mathrm{st}_{ikm},\mathrm{pt}_{ikm},\mathrm{ct}_{ikm})$，也是 DRL agent 的输入。作业 i 在工序 k 上机器 m 的加工时间 pt_{ikm} 系统输入已知，它是作业 i 的静态特征，在任何时候都是不变的。作业 i 在各工序 k 上的动态特征开始加工时 st_{ikm} 和完工时间 ct_{ikm} 可由式(7)和式(8)计算。本文将每一个作业定义为 $j_i=\{f_i,d_i^t\}$，其中 fi 表示作业 i 的静态特征，d_i^t 表示作业 i 在时间步 t 时的动态特征。当一道工序中的所有作业都转移到下一道工序时，状态集 S_t 将转移到下一个状态集 S_t'。

3.1.2　动作

在车间调度模型中，一个动作 $a_t \in A_t$ 是时间步 t 时在状态 S_t 下对工序 k 上的一个作业 i 确定一台机器 m 进行加工。因此，在每道工序 k，需要确定所有作业相对应的待加工机器，每台机器 m 上确定作业的次序就是分派的加工次序。因为任意作业 i 在 t 时只能选择一台机器且只能在该道工序上加工一次，所以 FSP 和 HFSP 在每道工序的动作空间的最大大小分别为 $|N|$ 和 $|NM_k|$。本文设置了一个虚拟作业 j_0（加工时间为 0，mask_0=1），工序 k 上的机器按编号依次做出动作，每次可选择一个正常作业节点或虚拟作业节点，并记录动作次数，若为正常作业，则将其 mask 值置为 0，如此循环往复，直到所有正常作业的 mask 值为 0 或动作次数达到了最大动作空间为止。输入序列和输出序列的特定示例如图 2 所示，在包含两台并行机器的工序给定一个待加工作业的输入序列 $\{j_0,j_1,j_2,j_3,j_4,j_5,j_6\}$，输出序列为 $\{j_2,j_0,j_1,j_o,j_3\}$ 和 $\{j_6,j_4,j_0,j_5\}$，分别表示机器 m_1 上分派的待加工的作业和次序为 $\{j_2,j_1,j_3\}$，机器 m_2 上分派的待加工的作业和次序为 $\{j_6,j_4,j_5\}$。

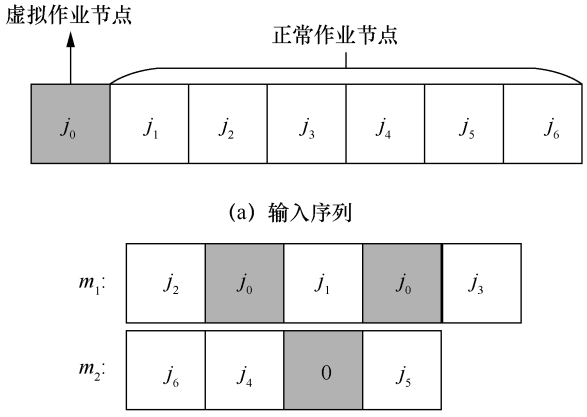

图 2　输入序列和输出序列的特定示例

3.1.3　奖励

奖励反映了 agent 在当前环境状态下所采取动作的质量。本文的最终目标是逐步学习作业调度，使目标值 $C_{\max}$ 最小化。目标值越小，agent 做出的调度决策越好，对 agent 的奖励就越大，所以对车间调度模型的奖励函数 R 设计为

$$R=\begin{cases}-C_{\max}, \text{在最大动作空间内所有} \\ \qquad \text{作业都被分派完} \\ -\infty, \text{其他}\end{cases} \tag{10}$$

3.1.4 策略

对于状态 s_t，一个随机策略 $\pi(a_t|s_t)$ 输出 A_t 中动作的分布。最优策略 $\pi^*(a_t|s_t)$ 肯定会产生最优解。因此，本文的目标是使 $\pi(a_t|s_t)$ 尽可能接近 $\pi^*(a_t|s_t)$。

3.2 车间调度模型总体架构

车间调度模型总体架构如图 3 所示，它基于指针网络，包括一个编码器和一个解码器。

（1）编码器

在 PN 中，编码器是基于 RNN 实现的。然而，RNN 只有在输入序列的排列顺序起到传递不同的信息时才有意义，但是，在本文的输入序列中，作业的顺序没有任何意义，因为任何的随机排列都包含与原始输入相同的信息。因此，车间调度模型省略了编码器中的 RNN，而是直接使用一维卷积层作为嵌入层来将静态特征 $f=\{f_i,i=0,1,2,\cdots,N\}$ 映射到矩阵 $\overline{f}=\{\overline{f}_i,i=0,1,2,\cdots,N\}$，从而降低计算复杂度的同时不降低效率[5]。

（2）解码器

解码器包括一个长短期记忆（LSTM, long short-term memory）网络，一个注意力机制和一个掩膜矩阵。在每个时间步 t 处，LSTM 获取其隐藏状态 h_{t-1} 和输入 j_t（即时间步 $t-1$ 的输出 y_{t-1}），输出隐藏状态 h_t。将编码器的输出矩阵 $\overline{f}$、输入序列中所有作业节点的动态特征 d_t 和 LSTM 的隐藏状态 h_t 输入注意力机制，然后利用掩膜矩阵得到各作业节点的概率分布。最后，概率最高的作业节点被选中作为在时间步 t 的输出。agent 在时间步 t 做出选择后，作业的动态特征将从 d_t 更新为 d_{t+1}，掩膜矩阵也将相应更新，并作为下一个时间步 t+1 的输入。

在每个时间步 t 处，注意力机制通过计算输入序列中每个作业节点处概率分布作为输入序列中作业节点的指针。引入与输入序列长度相同的掩膜矩阵约束 agent 的决策。掩膜矩阵的每个元素都与输入序列中的作业节点一一对应，其每个元素的值为 0 或 1。对应于虚拟作业节点的元素的值始终为 1，使 agent 能够在任何情况下都能选择虚拟作业节点。当 agent 在时间步 t 选择了正常作业节点 i 后，其掩膜矩阵对应的元素的值置为 0。结合掩膜矩阵，采用式(11)计算时间步 t 时各作业节点的最终概率分布，其中 v_a 和 w_a 为可训练变量。最后选取概率最高的作业节点作为输出节点。

$$P(y_t|Y_{t-1},S_t)=\text{Softmax}\left(v_a\tanh\left(w_a\left[\overline{f};h_t\right]\right)\right) \tag{11}$$

3.3 基于 DRL 的车间调度算法

本文采用基于DRL的车间调度算法对模型进行训练。它包含两个网络：演员网络和评论家网络。演员网络用于预测输入序列中每个作业节点在时间步 t 处的输出概率，假设其参数为 θ。评论家网络用于计算每个输入序列的期望奖励，假设其参数为 φ。

基于 DRL 的车间调度算法如算法 1 所示。首先，对每个作业的静态特征数据进行归一化，以加快训练速度，并随机初始化演员网络和评论家网络的参数。在每个 epoch（训练轮次），重置梯度和每个作业第一道工序的可开始加工时间和完工时间，并从训练集中随机抽取 J 个实例（每个实例

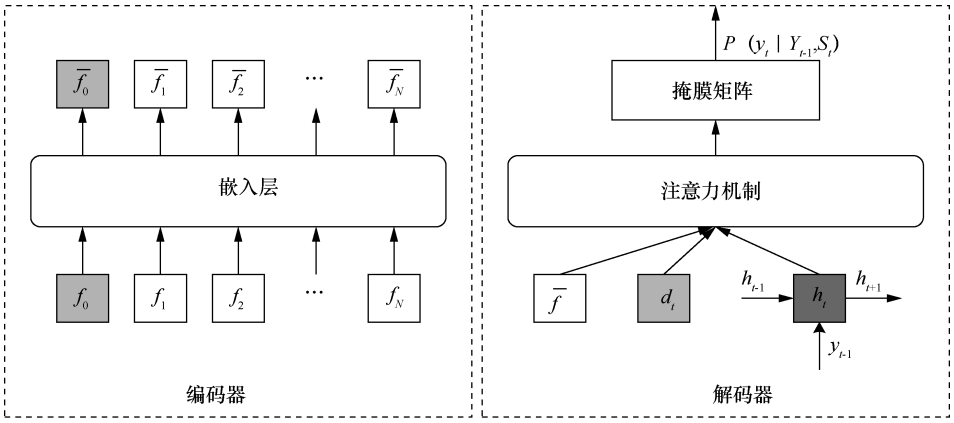

图 3　车间调度模型总体架构

是 N 个作业的集合）。在每道工序的时间步 0 处，对于每个实例 j，初始化时间步 t、掩膜矩阵和动作计数器，构造输入序列，利用式(2)和式(3)更新每个作业 i 的可开始加工时间 st_{ik} 和完工时间 ct_{ik}。演员网络观察当前环境状态 S_t^j，根据式(11)选择输出节点 y_t，然后相应地更新环境状态、掩膜矩阵和时间步长和动作次数。当所有的正常作业在实例 j 中被分派到相应的机器或动作次数达到最大动作空间时，进入下一道工序。直到所有工序都完成分派后，演员网络和评论家网络分别计算实际奖励 R_j 和期望奖励 $V(S_0^j;\varphi)$。结束 J 个实例的调度后，根据式(12)和式(13)计算并更新演员网络和评论家网络的梯度。重复以上步骤，直到达到最大 epoch 数。最后，经过训练的模型可以用于后续的车间调度任务。

$$\mathrm{d}\theta \leftarrow \frac{1}{J}\sum_{j=1}^{J}\left(R_j - V\left(S_0^j;\varphi\right)\right)\nabla_\theta \log P\left(Y_j|S_0^j\right) \tag{12}$$

$$\mathrm{d}\varphi \leftarrow \frac{1}{J}\sum_{j=1}^{J}\nabla_\varphi\left(R_j - V\left(S_0^j;\varphi\right)\right)^2 \tag{13}$$

其中，S_0^j 表示第 j 个实例在时间步 t=0 时的环境状态，Y_j 表示关于 S_0^j 的输出序列，R_j 表示第 j 个实例实际获得奖励，$P(Y_j|S_0^j)$ 表示每个作业节点的概率分布，$V(S_0^j;\varphi)$ 表示第 j 个实例关于 S_0^j 的期望奖励。

算法 1 基于 DRL 的车间调度算法

输入：所有作业的静态特征 f_i 和动态特征 d_i^t

输出：每个作业 i 在各道工序 k 相应机器 m 上的开始加工时间 st_{ikm} 和完工时间 ct_{ikm}

归一化输入的静态特征：$\hat{x} = \dfrac{x - x_{\min}}{x_{\max} - x_{\min}}$；

随机初始化演员网络的参数 θ 和和评论家网络的参数 φ；

for epoch $= 1,2,\cdots$ do

　重置梯度：$\mathrm{d}\theta \leftarrow 0$，$\mathrm{d}\varphi \leftarrow 0$；

　重置时间：$\mathrm{st}_{i1} = \mathrm{rt}_i$，$\mathrm{ct}_{i1} = \mathrm{st}_{i1} + \mathrm{pt}_i$；

　从训练集中随机抽取 J 个实例；

　for $k = 1,2,\cdots,K$ do

　　初始化时间步 $t \leftarrow 0$，掩膜矩阵 $\mathrm{mask} = [1,1,\cdots,1]$，$\mathrm{count} = 0$；

　　利用式（2）和式（3）更新可开始加工时间 st_{ik} 和完工时间 ct_{ik}；

　　if （$\mathrm{count} \leqslant NM_k$ && $\mathrm{mask} != [1,0,\cdots,0]$）

　　then

　　for $m = 1,2,\cdots,M_k$ do

　　观察状态 S_t^j，根据 $P\left(y_t|Y_{t-1},S_t\right)$ 选择输出节点 y_t；

　　更新状态 $S_t^j \leftarrow S_{t+1}^j$，$\mathrm{mask}_i = 0$，时间步 $t \leftarrow t+1$，$\mathrm{count}++$；

　　end for

　　end if

　　利用式（7）和式（8）计算每个作业 i 在分派到的机器 m 上的 st_{ikm} 和 ct_{ikm}；

　　根据式（10）计算 R_j，$V\left(S_0^j;\varphi\right)$；

　end for

　计算 $\mathrm{d}\theta$ 和 $\mathrm{d}\varphi$ 并分别更新演员网络和评论家网络；

end for

3.4　时间复杂度分析

Agent 包括动作选择和网络训练，从算法 1 中可以看出选择一次动作的时间复杂度 $\mathrm{TC}_{\mathrm{act}}$ 为 $O(KNM_k)$，其中 K 表示工序数量，N 表示作业数量，M_k 表示每道工序可加工的机器数量（FSP 中 M_k=1）。对于批尺寸为 b 的批，训练一次的时间复杂度为

$$\mathrm{TC}_{\mathrm{NN}} = b\left(|S|n_{\mathrm{hid}} + n_{\mathrm{hid}}|A|\right) = O(N) \tag{14}$$

其中，状态空间 $|S|$ 和动作空间 $|A|$ 的大小分别为网络输入层和输出层的节点数，n_{hid} 为隐藏层节点数。

环境中的计算包括更新状态和反馈奖励，由算法 1 可知环境计算的时间复杂度 $\mathrm{TC}_{\mathrm{env}}$ 与 $\mathrm{TC}_{\mathrm{act}}$ 相同。J 个训练集和 e 个 epoch 条件下的训练过程的时间复杂度 $\mathrm{TC}_{\mathrm{train}}$ 为：

$$\mathrm{TC}_{\mathrm{train}} = eJN(\mathrm{TC}_{\mathrm{act}} + \mathrm{TC}_{\mathrm{NN}} + \mathrm{TC}_{\mathrm{env}}) = O(eKN^2M_k) \tag{15}$$

由于测试过程只需要在每次选择动作时调用网络的结果，所以测试过程的时间复杂度 $\mathrm{TC}_{\mathrm{test}}$ 为：

$$\mathrm{TC}_{\mathrm{test}} = N\left(\mathrm{TC}_{\mathrm{act}} + \frac{1}{b}\mathrm{TC}_{\mathrm{NN}} + \mathrm{TC}_{\mathrm{env}}\right) = O(KN^2M_k) \tag{16}$$

算法的训练时间和测试时间都随着作业数量 N、工序数量 K 和每道工序可加工机器数量 M_k 的增加而多项式地增加。虽然 b、n_{hid}、J 和 e 等常量也直接影响算法的训练时间，但这并不影响算法在大规模场景下的可扩展性，特别是训练过程是在有足

够计算资源的云服务器中离线运行，只需要根据收集的产线数据定期更新即可。

在下一节中对比分析了基于优先权的规则调度算法（先进先出（FIFO, first in first out）、后进先出（LIFO, last in first out）、最长加工时间作业优先（LPT, longest processing time first）和最短加工时间作业优先（SPT, shortest processing time first））和智能优化算法（ACO、GA、ABC 和一种人工蜂群和禁忌搜索相结合的混合算法（ABC-TS, artificial bee colony and tabu search））。因为这些算法不需要提前训练，因此我们只分析在车间调度问题中测试过程的时间复杂度。

基于优先权的规则调度算法是按照作业到达系统时间 rt_i 的大小或在每道工序加工时间 pt_{ikm} 的大小按顺序地将作业安排在每道工序当前总完工时间最少的机器上，因为其在每道工序利用快速排序方法排序的时间复杂度为 $O(N\log N)$，在每道工序选择加工机器的时间复杂度为 $O(N\log NM_k)$，所以其时间复杂度为 $O(KN\log NM_k)$。

智能优化算法通过模仿生物界的进化机理和群体协作行为搜索构造问题的解，ACO 借助信息素和正反馈现象来找到出发点到目的点的最短路径，其时间复杂度为 $O(IKAN^2M_k)$，其中 I 表示迭代次数，A 表示参与搜索的蚂蚁数量；GA 把问题的解表示成染色体，首先给出一群初始染色体并将其按适者生存的原则选择出比较适合环境的染色体进行复制，再通过交叉、变异过程产生更适应环境的新一代染色体群，按此过程进行一代一代地进化，最后就会收敛到最适应环境的一个染色体上，其时间复杂度为 $O(IKGN^2\log NM_k)$，其中 G 表示初始种群大小；ACO-TS 则在 ACO 的基础上通过添加禁忌表过滤掉蚂蚁已经走过的路程以快速到达目的地，其时间复杂度为 $O(IKAN^2\log NM_k)$。

4 实验结果与分析

4.1 实验设置

FSP 实验在 Taillard 的 FSP 公共基准数据集[34]上进行测试，其中包含了不同作业和工序规模的实例，每种规模分别有 10 个实例。由于空间的限制，与文献[32]相同，本文选取了每一规模下 10 个实例中的最后一个实例进行性能测试。

HFSP 由于没有可用的基准数据集，本文使用了 NISCO 宽厚板卷厂的真实生产数据。包括 4 道生产工序，前 3 道工序具有 3 台并行机器，最后一个工序具有一台机器。为了便于评估，作业每道工序的处理时间是过去一年历史数据的平均值。为了更好地评估算法的性能，符合实际生产情况，本文分别设置了 50、100、200 个作业和（3,3,3,1）、（2,2,2,1）、（1,1,1,1）台机器的实例，模拟了每道工序的机器可能由于故障和维护而停机的情况。需要注意的是，由于 NISCO 只提供了 4 道工序的生产数据，所以本文的实验采用了固定的工序数进行实验比较。

本文使用开源 Python 库 TensorFlow 实现车间调度模型，在一个 NVIDIA Tesla P100-PCIE 16 GB GPU 的高性能计算集群上训练该模型，并在联想 Intel(R) Core(TM) i7-6700 CPU @ 3.40 GHz 8 GB 内存的 PC 上进行评估。在模型训练阶段，本文为每种规模生成 50 000 个实例，并将它们用于 400 个 epoch 的训练中，激活函数为 tanh。选择 50 000 是因为笔者想要有一个对不同规模的稳定配置，这个数字可以更大也可以更小，或者也可以实时生成实例，只要确保每种规模的训练实例与测试实例具有相同的数据分布即可。

4.2 收敛性分析

对 100×10 规模在不同隐藏层神经元数量、学习率、批尺寸和不同作业数量的 FSP 下对模型的收敛性进行了控制变量分析。

首先，本文分析了隐藏层中神经元数分别为 8、32、64、128 和 256 下模型的收敛性。不同神经元数量下模型的收敛性如图 4 所示，这里 Reward 为数据归一化后得到的奖励值。在一定范围内，神经元数量越多，收敛速度越快。但当神经元数为 256 时，由于神经元数过大，模型在 200 个 epoch 后才收敛。因此，隐藏层神经元数设置为 128，收敛速度最快。

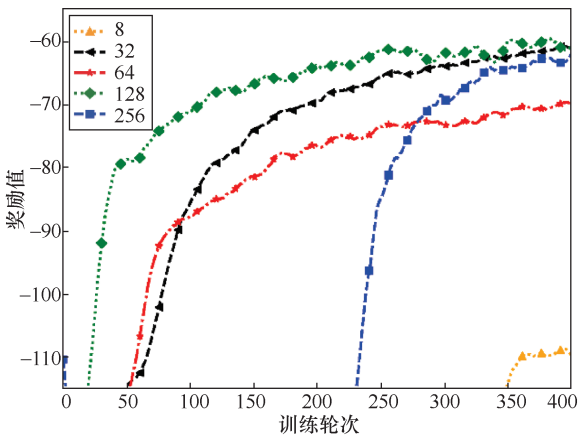

图 4 不同神经元数量下模型的收敛性

其次，本文分析了学习率值分别为 0.01、0.001、0.000 1 和 0.000 01 下模型的收敛性。不同学习率下模型的收敛性如图 5 所示，当学习率为 0.000 01 时，学习率太小，收敛时间很长（400 epoch 之后）。如果学习率为 0.01，明显值太大导致无法收敛。因此，学习率设置为 0.000 1。

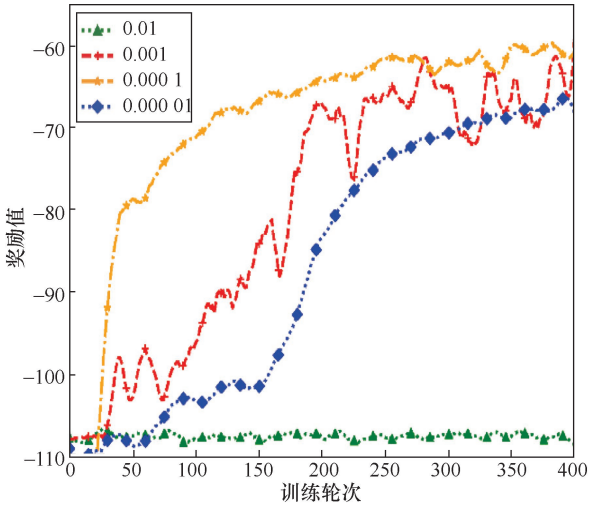

图 5　不同学习率下模型的收敛性

再次，本文分析了批尺寸分别为 1、8、32、64 和 128 下模型的收敛性。不同批尺寸下模型的收敛性如图 6 所示，批尺寸越小，模型的梯度振荡幅度越大，不利于模型的收敛。批尺寸越大，模型的梯度振荡幅度越小，但其达到相同 Reward 的 epoch 数量增多。当批尺寸为 32 时，梯度振荡幅度较小，模型收敛速度快。因此，批尺寸大小设置为 32。

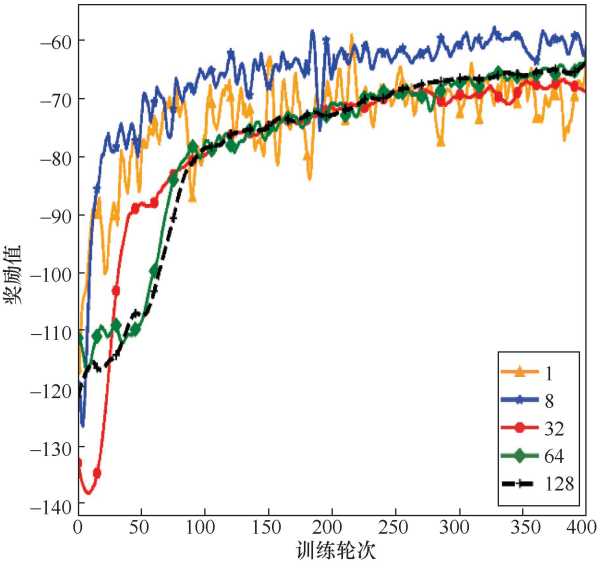

图 6　不同批尺寸下模型的收敛性

最后，本文分析了作业总数量和机器总数量分别为 20×10、50×10、100×100、200×10 时 FSP 的收敛效果。不同规模下模型的收敛性如图 7 所示，问题规模越大，收敛速度越慢。

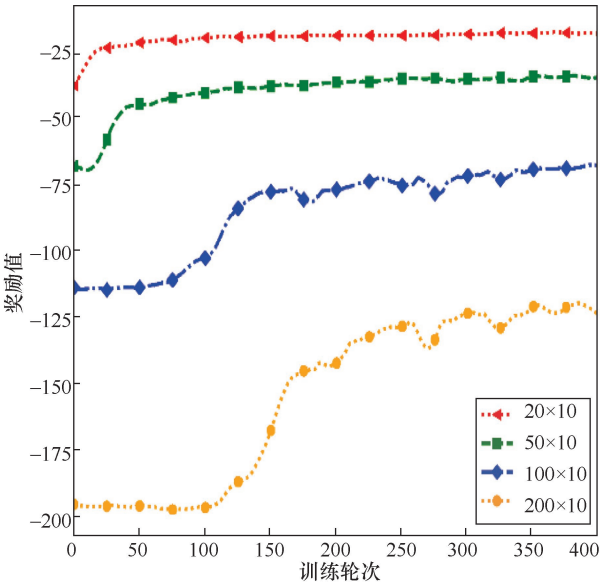

图 7　不同规模下模型的收敛性

4.3　与其他算法的性能对比

为了评估提出的基于 DRL 的算法的性能，本文将其与常见的基于优先权的规则调度算法[21]和改进的智能优化算法进行了比较，包括 FIFO、LIFO、LPT、SPT、ACO、GA、ABC-TS。特别的，针对 FSP，本文还比较了文献[32]中的 DRL_IG 算法。为了尽可能地消除算法的不确定性影响和考虑实际运行的时间成本，与文献[32]相同，本文将每个实例运行 10 次并取平均值（保留整数位）。

FSP 实例结果见表 1，对于 FSP 场景，DRL_IG 算法和本文提出的基于 DRL 的智能车间调度算法优于基于优先权的规则调度算法和智能优化算法，特别是对于作业总数量和机器总数量分别为 50×10、100×10 和 200×10 等中大规模场景，基于 DRL 的方法性能提升较为明显。与 DRL_IG 算法相比较，本文的算法虽然在性能上没有显著提升，但 DRL_IG 算法先将模型离线训练了 48 h，然后在 IG（iterated greedy）搜索环节迭代 8 000 次，网络训练和运行的时间成本都较高，无法满足更大规模的实时动态环境。

HFSP 实例结果见表 2，对于作业总数量为 50

的小规模 HFSP 场景，本文提出的方法优于基于优先权的规则调度算法，性能与智能优化算法相当，这是因为当作业、工序和各道工序可加工机器数量都不太多时智能优化算法也能优化到较好结果，同时，由于 NISCO 宽厚板卷厂的作业在各道工序机器上的加工时间差异并不是太大，因此相比于基于优先权的规则调度算法在数值上没有显著提升。在作业总数量为 100 和 200 的中大规模场景上，也都取得了比其他对比算法更好的结果。特别的，从相同作业、不同加工机器数量的对比结果也能发现，当可用加工机器数量增多时，能显著提升生产效率。

由此可见，本文的方法可以用于解决不同生产环境下不同作业、工序和加工机器数量的车间调度问题，且可以自行学习参数和提取特征，不需要特别地手动调整，这解决了传统智能优化算法在应用到不同问题场景之前，需要凭借经验进行大量手动调整的问题，也不再需要在算力资源和迭代优化次数有限的情况下，多次运行算法取最优值以尽量避免其陷入局部最优而带来的额外时间代价的问题。

5 结束语

本文建立了一个基于指针网络的智能车间调度模型，并提出了一种基于 DRL 的智能车间调度算法，该算法能够充分利用工业物联网中大量的无标签数据学习和更新策略。本文分析了该模型在不同参数设置下的收敛性，并与多种方法进行比较。实验表明，本文的方法能够适用于不同规模的问题和场景，并更快地得到更好的解，尤其是对于中大规模问题具有更强的实用性。未来，我们将重点研究利用 DRL 方法结合工业物联网中的作业批处理问题来解决更广泛的智能车间调度问题。

表 1 FSP 实例结果

$N \times K$	FIFO	LIFO	LPT	SPT	ACO	GA	ABC-TS	DRL_IG	Ours
20×5	1 404	1 513	1 681	1 339	1 190	1 212	1 187	1 108	1 108
20×10	2 051	2 103	2 059	1 833	1 709	1 741	1 654	1 601	1 594
50×5	3 188	3 397	3 869	3 089	2 869	2 864	2 805	2 782	2 782
50×10	3 845	3 945	4 088	3 927	3 450	3 436	3 426	3 100	3 091
100×5	6 157	5 897	6 440	5 821	5 558	5 563	5 453	5 322	5 328
100×10	6 930	6 915	7 494	6 564	6 313	6 288	6 256	5 864	5 845
200×10	12 274	12 379	13 359	12 155	11 361	11 899	11 336	10 716	10 976
平均统计	5 121	5 164	5 570	4 961	4 636	4 715	4 588	4 356	4 355

表 2 HFSP 实例结果

N	$M_k \times K$	FIFO	LIFO	LPT	SPT	ACO	GA	ABC-TS	Ours
50	(1, 1, 1, 1)	2 163	2 164	2 170	2 162	2 162	2 162	2 162	2 162
50	(2, 2, 2, 1)	1 122	1 117	1 118	1 123	1 109	1 109	1 109	1 109
50	(3, 3, 3, 1)	774	765	773	787	761	761	761	761
100	(1, 1, 1, 1)	6 514	6 508	6 544	6 537	6 475	6 474	6 474	6 454
100	(2, 2, 2, 1)	3 321	3 337	3 343	3 375	3 308	3 302	3 303	3 266
100	(3, 3, 3, 1)	2 278	2 251	2 269	2 301	2 217	2 226	2 213	2 204
200	(1, 1, 1, 1)	12 867	12 887	12 916	12 909	12 849	12 849	12 849	12 587
200	(2, 2, 2, 1)	6 578	6 492	6 516	6 522	6 472	6 464	6 463	6 415
200	(3, 3, 3, 1)	4 474	4 361	4 390	4 435	4 338	4 339	4 332	4 332
平均统计		4 455	4 431	4 449	4 461	4 410	4 409	4 407	4 365

参考文献：

[1] GILCHRIST A. Industry 4.0: The industrial internet of things[M]. Berkeley, CA: Apress, 2016.

[2] VINYALS O, FORTUNATO M, JAITLY N. Pointer networks[J]. CoRR, 2015: abs/1506.03134.

[3] LING Z X, TAO X Y, ZHANG Y, et al. Solving optimization problems through fully convolutional networks: an application to the traveling salesman problem[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2021, 51(12): 7475-7485.

[4] NAZARI M, OROOJLOOY A, SNYDER L V, et al. Reinforcement learning for solving the vehicle routing problem[J]. CoRR, 2018: abs/1802.04240.

[5] BELLO I, PHAM H, LE Q V, et al. Neural combinatorial optimization with reinforcement learning[C]//Proceeding of 5th International Conference on Learning Representations. Toulon: 2017: 1-13.

[6] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Playing atari with deep reinforcement learning[J]. CoRR, 2013: abs/1312.5602.

[7] ZHANG C, SONG W, CAO Z G, et al. Learning to dispatch for job shop scheduling via deep reinforcement learning[EB]. 2020.

[8] 刘建伟, 高峰, 罗雄麟. 基于值函数和策略梯度的深度强化学习综述[J]. 计算机学报, 2019, 42(6): 1406-1438.
LIU J W, GAO F, LUO X L. Survey of deep reinforcement learning based on value function and policy gradient[J]. Chinese Journal of Computers, 2019, 42(6): 1406-1438.

[9] GAREY M R, JOHNSON D S, SETHI R. The complexity of flowshop and jobshop scheduling[J]. Mathematics of Operations Research, 1976, 1(2): 117-129.

[10] JOHNSON S M. Optimal two-and three-stage production schedules with setup times included[J]. Naval Research Logistics Quarterly, 1954, 1(1): 61-68.

[11] REZA HEJAZI S, SAGHAFIAN S. Flowshop-scheduling problems with makespan criterion: a review[J]. International Journal of Production Research, 2005, 43(14): 2895-2929.

[12] ZHANG J, DING G F, ZOU Y S, et al. Review of job shop scheduling research and its new perspectives under Industry 4.0[J]. Journal of Intelligent Manufacturing, 2019, 30(4): 1809-1830.

[13] HARTMANIS J. Computers and intractability: a guide to the theory of np-completeness (Michael R. garey and David S. Johnson)[J]. Siam Review, 1982, 24(1): 90-91.

[14] ARTHANARY T S. An extension of two machine sequencing problem[J]. Opsearch, 1971(8): 10-22.

[15] RUIZ R, VÁZQUEZ-RODRÍGUEZ J A. The hybrid flow shop scheduling problem[J]. European Journal of Operational Research, 2010, 205(1): 1-18.

[16] TOSUN Ö, MARICHELVAM M K, TOSUN N. A literature review on hybrid flow shop scheduling[J]. International Journal of Advanced Operations Management, 2020, 12(2): 156.

[17] 李颖俐, 李新宇, 高亮. 混合流水车间调度问题研究综述[J]. 中国机械工程, 2020, 31(23): 2798-2813, 2828.
LI Y L, LI X Y, GAO L. Review on hybrid flow shop scheduling problems[J]. China Mechanical Engineering, 2020, 31(23): 2798-2813, 2828.

[18] 夏柱昌, 刘芳, 公茂果, 等. 基于记忆库拉马克进化算法的作业车间调度[J]. 软件学报, 2010, 21(12): 3082-3093.
XIA Z C, LIU F, GONG M G, et al. Memory based Lamarckian evolutionary algorithm for job shop scheduling problem[J]. Journal of Software, 2010, 21(12): 3082-3093.

[19] REN T, WANG X Y, LIU T Y, et al. Exact and metaheuristic algorithms for flow-shop scheduling problems with release dates[J]. Engineering Optimization, 2021: 1-17.

[20] HIDRI L, ELKOSANTINI S, M MABKHOT M. Exact and heuristic procedures for the two-center hybrid flow shop scheduling problem with transportation times[J]. IEEE Access, 2018, 6: 21788-21801.

[21] HUNSUCKER J L, SHAH J R. Comparative performance analysis of priority rules in a constrained flow shop with multiple processors environment[J]. European Journal of Operational Research, 1994, 72(1): 102-114.

[22] CAMPBELL H G, DUDEK R A, SMITH M L. A heuristic algorithm for thenJob, mMachine sequencing problem[J]. Management Science, 1970, 16(10): B-630.

[23] CHEN W, HAO Y F. Genetic algorithm-based design and simulation of manufacturing flow shop scheduling[J]. International Journal of Simulation Modelling, 2018, 17(4): 702-711.

[24] ENGIN O, GÜÇLÜ A. A new hybrid ant colony optimization algorithm for solving the no-wait flow shop scheduling problems[J]. Applied Soft Computing, 2018(72): 166-176.

[25] LI X T, MA S J. Multiobjective discrete artificial bee colony algorithm for multiobjective permutation flow shop scheduling problem with sequence dependent setup times[J]. IEEE Transactions on Engineering Management, 2017, 64(2): 149-165.

[26] CUNHA B, MADUREIRA A M, FONSECA B, et al. Deep reinforcement learning as a job shop scheduling solver: a literature review[C]//Hybrid Intelligent Systems. 2020.

[27] LIU C L, CHANG C C, TSENG C J. Actor-critic deep reinforcement learning for solving job shop scheduling problems[J]. IEEE Access, 2020(8): 71752-71762.

[28] HAN B A, YANG J J. Research on adaptive job shop scheduling problems based on dueling double DQN[J]. IEEE Access, 2020, 8: 186474-186495.

[29] WANG L B, HU X, WANG Y, et al. Dynamic job-shop scheduling in smart manufacturing using deep reinforcement learning[J]. Computer Networks, 2021, 190: 107969.

[30] LUO S. Dynamic scheduling for flexible job shop with new job insertions by deep reinforcement learning[J]. Applied Soft Computing, 2020, 91: 106208.

[31] ZHANG C, SONG W, CAO Z G, et al. Learning to dispatch for job shop scheduling via deep reinforcement learning[J]. Advances in Neural Information Processing Systems, 2020, 33.

[32] 王凌, 潘子肖. 基于深度强化学习与迭代贪婪的流水车间调度优化[J]. 控制与决策,2021, 36(11): 2609-2617.
WANG L, PAN Z X. Scheduling optimization for flow-shop based on deep reinforcement learning and iterative greedy method[J]. Control and Decision, 2021, 36(11): 2609-2617.

[33] LUO B, WANG S B, YANG B, et al. An improved deep reinforcement learning approach for the dynamic job shop scheduling problem with random job arrivals[J]. Journal of Physics: Conference Series, 2021, 1848(1): 012029.

[34] TAILLARD E. Benchmarks for basic scheduling problems[J]. European Journal of Operational Research, 1993, 64(2): 278-285.

[作者简介]

罗梓珲（1996− ），男，北京邮电大学博士生，主要研究方向为工业物联网、边缘计算。

江呈羚（1997− ），女，北京邮电大学硕士生，主要研究方向为智能优化调度、深度强化学习

刘亮（1982− ），男，北京邮电大学教授，主要研究方向为物联网、智能感知技术。

郑霄龙（1989− ），男，北京邮电大学副教授，主要研究方向为物联网、无线网络、普适计算。

马华东（1964− ），男，北京邮电大学教授，主要研究方向为多媒体系统与网络、物联网与传感网、视频理解与大数据分析。