Journal of System Simulation

Volume 36 | Issue 7 Article 9

7-15-2024

Real-time Scheduling Method for Dynamic Flexible Job Shop Scheduling

Quan Jiang
School of Computer Science and Technology, Xidian University, Xi'an 710065, China,
jq18890952@163.com

Jingxuan Wei
School of Computer Science and Technology, Xidian University, Xi'an 710065, China, wjx@xidian.edu.cn

Real-time Scheduling Method for Dynamic Flexible Job Shop Scheduling

Abstract

Abstract: A multi-objective dynamic flexible job shop scheduling problem model with machine breakdown and random jobs arrival is constructed to address the interference of dynamic events in manufacturing processing on the scheduling scheme, and a real-time scheduling method with multiobjective proximal policy optimization (MPPO) algorithm is proposed. The MPPO algorithm trains two agents, routing agent (RA) and sequencing agent (SA), for real-time scheduling and real-time processing of dynamic events. It employs a linear combination of weight vectors and reward vectors as reward signals and stores the agents' parameters for each weight vector to optimize multiple objectives. The required state information, scheduling rules, and reward signals are defined for the two agents in conjunction with the objective functions. A comparison with nine combinations of scheduling rules for dynamic scheduling problems of different scales verifies that the MPPO algorithm-trained agents have learned an appropriate scheduling policy, which can guarantee the performance of real-time scheduling and optimize all objectives.

Keywords

dynamic scheduling, flexible job shop scheduling, reinforcement learning, multi-agent, multi-objective optimization

Recommended Citation

Jiang Quan, Wei Jingxuan. Real-time Scheduling Method for Dynamic Flexible Job Shop Scheduling [J]. Journal of System Simulation, 2024, 36(7): 1609-1620.

第 36 卷第 7 期 系统仿真学报© Vol. 36 No. 7
2024 年 7 月 Journal of System Simulation Jul. 2024

用于动态柔性作业车间调度的实时调度方法

蒋权，魏静萱*
(西安电子科技大学 计算机科学与技术学院，陕西 西安 710065)

摘要：针对制造加工中的动态事件对调度方案的干扰，构建了带有机器故障和随机工件到达的多目标动态柔性作业车间调度问题模型，提出多目标近端策略优化(multi-objective proximal policy optimization, MPPO)的实时调度方法。*MPPO 算法训练了RA(routing agent)和SA(sequencing agent)两个智能体以实现实时调度并实时处理动态事件；采用权重向量与奖励向量线性组合作为奖励信号，并保存每个权重向量的智能体参数以优化多个目标；结合目标函数为两个智能体定义了所需的状态信息、调度规则、奖励信号。*在不同规模的动态调度问题下与9种调度规则组合进行对比，验证了MPPO算法训练的智能体学习到了合适的调度策略，能够保证实时调度的表现且能优化所有目标。

关键词：动态调度；柔性作业车间调度；强化学习；多智能体；多目标优化
中图分类号：TP391.4 文献标志码：A 文章编号：1004-731X(2024)07-1609-12
DOI: 10.16182/j.issn1004731x.joss.23-0385
引用格式: 蒋权, 魏静萱. 用于动态柔性作业车间调度的实时调度方法[J]. 系统仿真学报, 2024, 36(7): 1609-1620.
Reference format: Jiang Quan, Wei Jingxuan. Real-time Scheduling Method for Dynamic Flexible Job Shop Scheduling [J]. Journal of System Simulation, 2024, 36(7): 1609-1620.

Real-time Scheduling Method for Dynamic Flexible Job Shop Scheduling

Jiang Quan, *Wei Jingxuan**
(School of Computer Science and Technology, Xidian University, Xi'an 710065, China)

Abstract: A multi-objective dynamic flexible job shop scheduling problem model with machine breakdown and random jobs arrival is constructed to address the interference of dynamic events in manufacturing processing on the scheduling scheme, and a real-time scheduling method with multi-objective proximal policy optimization (MPPO) algorithm is proposed. *The MPPO algorithm trains two agents, routing agent (RA) and sequencing agent (SA), for real-time scheduling and real-time processing of dynamic events. It employs a linear combination of weight vectors and reward vectors as reward signals and stores the agents' parameters for each weight vector to optimize multiple objectives. The required state information, scheduling rules, and reward signals are defined for the two agents in conjunction with the objective functions.* A comparison with nine combinations of scheduling rules for dynamic scheduling problems of different scales verifies that the MPPO algorithm-trained agents have learned an appropriate scheduling policy, which can guarantee the performance of real-time scheduling and optimize all objectives.

Keywords: dynamic scheduling; flexible job shop scheduling; reinforcement learning; multi-agent; multi-objective optimization

收稿日期：2023-04-06 修回日期：2023-05-29
基金项目：国家自然科学基金(62272367)
第一作者：蒋权(1999-)，男，硕士生，研究方向为动态调度与智能优化算法。E-mail：jq18890952@163.com
通讯作者：魏静萱(1981-)，女，副教授，博士，研究方向为智能计算与多目标优化。E-mail：wjx@xidian.edu.cn

0 引言

调度问题在生产加工领域普遍存在，经常需要决定各个工件的加工顺序和加工机器，以达到一定的生产目的，如生产成本最低。柔性作业车间调度问题(flexible job shop problem, FJSP)是NP-hard问题，属于生产加工领域的一种典型调度问题，在半导体加工、钢材加工、纺织品加工等领域广泛应用。然而，在现实生产加工中，存在各种动态因素，如机器故障、订单取消、加工时间波动等动态事件，会导致原调度方案次优或不可行。因此，研究生产加工中的动态柔性作业车间调度问题(dynamic FJSP, DFJSP)更符合现实生产环境。

目前，解决动态调度的方法主要分为预测反应式调度和反应式调度两大类[1]。预测反应式调度方法将动态问题分解为一系列静态子问题，在不考虑动态事件的情况下搜索到一个初始调度方案，当动态事件发生时，根据当前的加工信息生成新的调度方案，按新调度方案进行加工直到下一重调度时间点。文献[2]针对带有机器故障的动态调度问题，采用加权法和非支配排序基因算法生成预调度方案，为兼顾调度的稳定性与鲁棒性，提出一种混合纳什均衡策略得到动态调度方案。文献[3]针对具有实时订单接受和条件预防性维护的动态调度问题，设计了四种检查策略，找到每个机器的最佳维护计划，构建了具有三种响应方法的自适应重调度策略池，提出一种混合多目标进化算法进行求解。文献[4]针对带有机器故障的动态调度问题，提出了一种基于工件分类的调整方法以提高重调度的稳定性，采用遗传算法和变邻域搜索算法进行求解。此类方法虽然能够获得较高质量的解，但是需要耗费大量的时间重新搜索新的调度方案，难以满足DFJSP实时调度的要求。

反应式调度方法不生成完整的调度方案，而是根据实时的信息来进行决策，主要采用调度规则进行调度。但是调度规则的性能完全由生产环境和目标函数所决定，没有一个调度规则在所有条件下能达到最优[5]，因此，基于反应式调度方法的关键问题是如何根据实时状态选择合适的调度规则。文献[6]提出基于随机森林的方法从最优调度提取分配规则。文献[7]采用集成学习方法集成四个分类器的表现，来选择合适的调度规则。但是这些基于监督学习方法，需要获得训练样本的最优调度方案，对于大规模的动态调度问题最优调度方案是难以获得的。

近些年来，基于马尔可夫决策的深度强化学习(deep reinforcement learning, DRL)方法在旅行商问题上超越了传统方法及求解器的性能[8-9]，鉴于DRL实时决策的优势，一些研究工作已将DRL方法用于解决同样作为组合优化问题的作业车间调度问题。文献[10]将析取图作为输入，采用注意力机制作为图表示学习模块提取状态信息，采用D3QPN算法将提取到的状态信息映射为最合适的调度规则以最小化完工时间。文献[11]以PPO算法为基础提出含有三个智能体的多层次DRL算法，目标智能体在三个目标函数中选择一个进行优化，而工件和机器智能体分别输出工件选择规则和机器选择规则进行重调度。文献[12]针对最小化动态调度的累计延迟时间问题，提出多层次和分布式结构来实现动态调度的实时控制，采用代理奖励构造技巧来提升D2QN算法的训练效果。文献[13]设计了一种可变长的状态信息和动作空间，以训练智能体直接输出工件优先级，从而决定每个重调度时间点的加工工件以最小化累计延迟时间，采用RNN和GRU来处理变长的状态信息以保证计算出来的工件优先级是全局最优。

但是大多数基于DRL的研究考虑的都是单目标的动态问题，而在实际生产调度问题中往往需要考虑多个生产目标。因此，本文考虑在含有随机机器故障和随机工件到达的条件下，以生产加工的完工时间、延迟时间和加工能耗为目标构建多目标动态柔性作业车间调度模型，然后提出一种多目标近端策略优化算法(multi-objective

proximal policy optimization, MPPO) 来求解。MPPO 算法训练 RA(routing agent) 和 SA (sequencing agent)两个智能体进行实时调度并实时处理动态事件，与多目标进化算法和超启发式方法等重调度方法相比，MPPO 算法能够满足实时调度的要求，且训练好的智能体能直接用于解决不同规模的动态调度问题。

本文的主要贡献和工作为：①考虑在随机机器故障和随机工件到达条件下，构建了含有 3 个生产目标的 DFJSP 模型。②针对 DFJSP 两个子问题，提出包含 SA 和 RA 的多智能体结构实现实时调度，并针对 PPO 不能同时考虑多个目标的问题，提出适用于多目标问题的 MPPO 算法训练 SA 和 RA 学习合适的调度策略以优化多个目标。③结合 DFJSP 的目标函数分别为 SA 和 RA 智能体定义了调度时刻的状态信息、调度规则集合以及奖励向量。④仿真实验证明了 MPPO 算法在多样性和收敛性上优于调度规则组合。

1 问题描述及数学模型

本文研究的 DFJSP 问题可描述为如下形式：有 n 个工件需要进行加工，$N=\{N_1,N_2,\cdots,N_n\}$，其中包含了初始时刻已到达车间的 n_f 个工件和随机到达的 n_s 个新工件；有 m 个加工机器 $M=\{M_1,M_2,\cdots,M_m\}$ 可以使用；工件的到达时间表示为 $A_i(i\in N)$；工件的优先级表示为 $PR_i(i\in N)$；工件的交付日期紧急度为 $ddt_i(i\in N)$；每个工件 $i(i\in N)$ 包含 n_i 个需要依次进行加工的工序 $O_i=\{O_{i1},O_{i2},\cdots,O_{in_i}\}$，其中 O_{ij} 表示工件 i 的第 j 个工序，当 O_{ij} 完成加工后，它的后续工序 O_{ij+1} 才能进行加工；O_{ij} 只能在部分可用机器上选择一个进行加工，O_{ij} 的可用机器集合 $M_{i,j}$ 包含 k 个机器 $M_{ij}=\{M_{ij1},M_{ij2},\cdots,M_{ijk}\}$，$M_{ijk}\in M_{ij}$；$O_{ij}$ 在机器 M_{ijk} 的加工时间表示为 P_{ijk}，加工能耗表示为 E_{ijk}；O_{ij} 在机器 M_{ijk} 的开始加工时间和完成时间分别表示为 S_{ijk}、C_{ijk}，第 i 个工件的完成时间表示为 C_i。每个机器 M_k 会随机发生故障，M_k 发生故障的时间点集合和所需要的修复时间集合分别为 B_k，R_k，第 r 次发生故障的时间为 B_{kr}，需要的修复时间为 R_{kr}。

3 个目标函数将同时被考虑，分别为完工时间 $C_{\max}$、延迟时间 TD、加工能耗 EC。为了便于建立可信的数学模型，做出如下假设：①在任何时刻一个工件只能在一个机器上被加工；②在任何时刻一个机器只能加工一个工件；③除非机器故障，当工件的加工开始后不能被打断；④属于同一个工件的所有工序必须按顺序依次进行加工；⑤工件在机器之间的转运时间忽略不计；⑥只考虑机器在加工状态下的能耗。

目标函数的表达式为

$$\min\begin{cases}C_{\max}=\max\{C_i,i\in N\}\\ TD=\sum\limits_{i\in N}PR_i\cdot\max(C_i-D_i,0)\\ EC=\sum\limits_{i\in N}\sum\limits_{j\in O_i}\sum\limits_{k\in M_{i,j}}(X_{ijk}\cdot E_{ijk})\end{cases}$$

约束条件如下：

工序 O_{ij} 只能选择一个机器加工：

$$\sum_{k\in M_{ij}}X_{ijk}=1\ \forall i,j \tag{1}$$

如果 O_{ij} 没有分配给机器 k，则在机器 k 上的开始时间和结束时间均为 0：

$$S_{ijk}+C_{ijk}\leqslant X_{ijk}\cdot L\ \ \forall i,j,k\ L\text{为较大的正整数} \tag{2}$$

O_{ij} 在机器 k 上的完工时间和开始时间之差至少为所需要的加工时间：

$$C_{ijk}\geqslant S_{ijk}+P_{ijk}-\left(1-X_{ijk}\right)\cdot L\ \ \forall i,j,k \tag{3}$$

开始加工时间至少不小于到达时间：

$$S_{ijk}\geqslant A_i\ \ \forall i,j,k \tag{4}$$

O_{ij} 和 $O_{i'j'}$ 不能同时在共有的可用加工机器 k 上进行加工：

$$S_{ijk}\geqslant C_{i'j'k}-Y_{iji'j'k}\cdot L\ \ \forall i<i',j,j',k\in(M_{ij}\cap M_{i'j'}) \tag{5}$$

$$S_{i'j'k}\geqslant C_{ijk}-(1-Y_{iji'j'k})\cdot L\quad \forall i<i',j,j',k\in(M_{ij}\cap M_{i'j'}) \tag{6}$$

同一工件的工序按次序进行加工：

$$\sum_{k\in M_{ij}} S_{ijk} = \sum_{k\in M_{ij}} C_{i,j-1,k} \quad \forall i, \forall j \in (2,\cdots,n_i) \tag{7}$$

工件的完成时间为

$$C_i \geqslant C_{in_i k} \quad \forall i,k \tag{8}$$

不能在机器故障修复时进行加工：

$$(S_{ijk}-B_{kr})\cdot(S_{ijk}-B_{kr}-R_{kr})\geqslant 0 \quad \forall i,j,k,r \tag{9}$$

$$(C_{ijk}-B_{kr})\cdot(C_{ijk}-B_{kr}-R_{kr}-P_{ijk})\geqslant 0 \quad \forall i,j,k,r \tag{10}$$

工件 i 预期完成时间为

$$D_i = A_i + ddt_i \cdot \sum_{j\in O_i} \frac{\sum_{k\in M_{i,j}} P_{ijk}}{|M_{ij}|} \quad \forall i,j,k \tag{11}$$

以上约束中：$X_{ijk}\in\{0,1\}$，如果 O_{ij} 在机器 M_{ijk} 上加工为1，否则为0；$Y_{iji'j'k}\in\{0,1\}$，如果 O_{ij} 在机器 M_{ijk} 上先于 $O_{i'j'}$ 加工为1，否则为0。

2 MPPO 算法

2.1 算法框架

由于本文研究的 DFJSP 包含 2 个子问题：机器上的工件加工顺序选择和工件的加工机器选择，为实现动态调度的实时控制，采用 SA 和 RA 两个智能体分别解决上述 2 个子问题。在时刻 t 若机器 M_k 完成工件加工，SA 需要在其等待队列 Q_k 中选择下一加工工件，根据 t 时刻调度状态 s_t，SA 输出 a_t 即相应的调度规则，在队列中选择工件 j_n 进行加工，得到本次调度的奖励 r_t，进入下一状态 s_{t+1}，并判断所有工件是否完成调度即 *done* 是否为真。如图 1 所示，当机器 M 完成加工且等待队列非空时，根据 SA 选择的调度规则，机器 M 在队列中选择下一工件进行加工。同样在 RA 满足调度条件时，根据当前加工状态，RA 选择合适的调度规则，从而将工件分配给加工机器 M。

每次 SA 和 RA 完成调度后都要将 5 元组(s_t, a_t, r_t, s_{t+1}, *done*)存储到经验池，当经验池数量达到设定的批量大小后 SA 和 RA 需要进行学习，本文采用强化学习的一种基线算法 PPO(proximal policy optimization)[14]作为 SA 和 RA 的参数更新算法。而对于 t 时刻的动态事件处理如下所示：

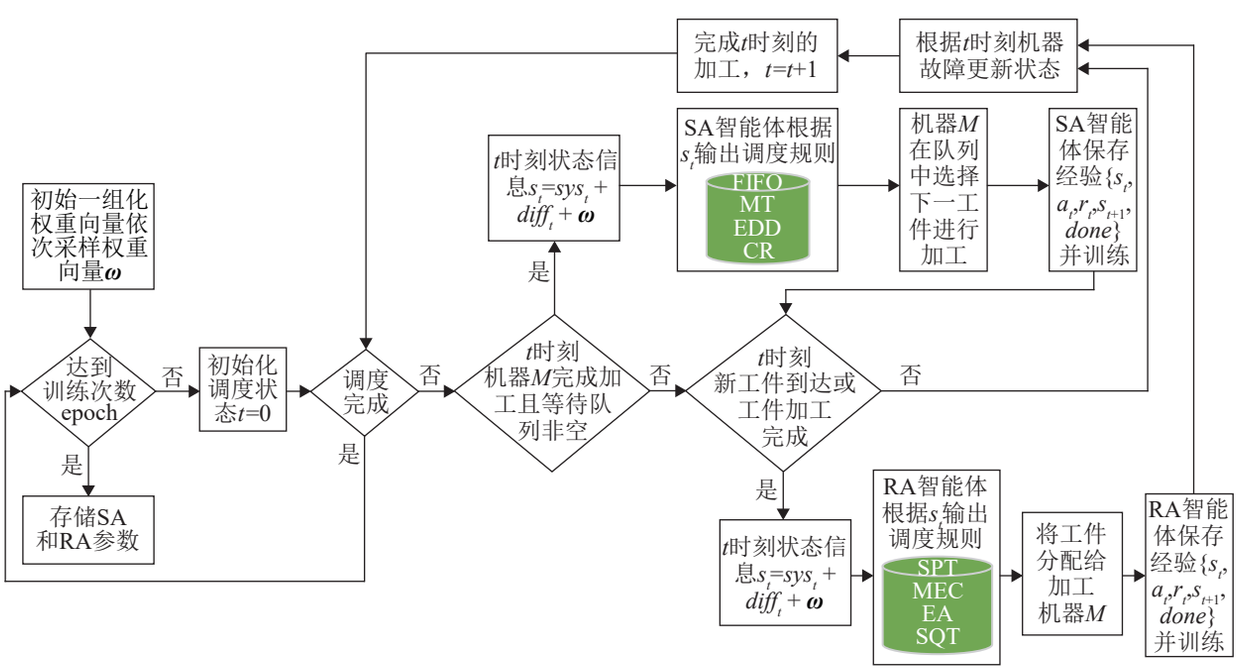


图 1 MPPO 框架

Fig. 1 MPPO framework

(1) 新工件到达：通过RA智能体为新工件分配加工机器。

(2) 机器故障：如果t时刻机器工作，打断工件的加工，更新机器的可用时间和工件的预估完成时间，图2中，t=40时M_3正在进行工序$O_{1,2}$的加工，M_3故障打断$O_{1,2}$的加工，故障修复时间R=10，t=50时M_3完成修复继续进行$O_{1,2}$的加工，t=65时M_3完成$O_{1,2}$的加工，SA分配新工序$O_{2,3}$进行加工。如果t时刻机器空闲，根据故障的修复时间更新机器的可用时间，图2中，t=50时M_1故障经过10个单位时间的修复后在t=60时恢复可用。

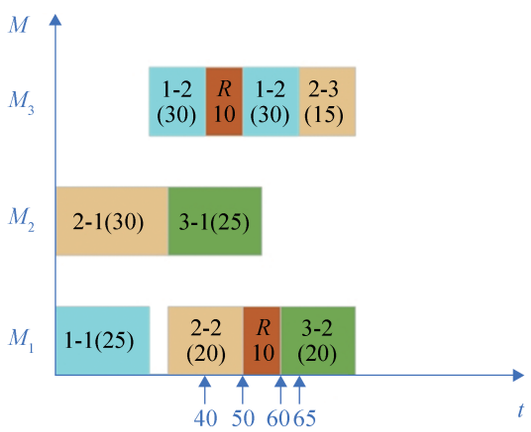


图2　机器故障处理

Fig. 2　Machine breakdown handling

由于强化学习方法是通过与环境互动获得奖励信号，从而训练智能体学习到相应的行为策略[15]，奖励信号与目标函数相关，而PPO算法以单个奖励进行训练，但在本文所研究的问题中需要同时考虑3个目标，为解决PPO算法不适用的问题，提出应用于本文所研究DFJSP的MPPO算法。

首先，借鉴多目标问题分解方法的思想，生成一组均匀分布权重向量，每个向量$\boldsymbol{\omega}_i$=(ω_1, ω_2, ω_3)，ω_1、ω_2、ω_3分别表示3个目标对应奖励信号$\boldsymbol{r}$=(r_1, r_2, r_3)的权重系数，为保证所有目标都能得到优化，保证ω_1、ω_2、ω_3≥0.1且$\omega_1+\omega_2+\omega_3$=1，最终的奖励$r=\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{r}$。而不同的权重向量$\boldsymbol{\omega}_i$会导致计算出来的$r$波动比较大，从而导致训练不稳定和网络结构的参数被覆盖的问题，本文采用文献[16]所提方法，对每个$\boldsymbol{\omega}_i$训练一定次数后保存$\boldsymbol{\omega}_i$对应的模型参数，并切换到与$\boldsymbol{\omega}_i$欧氏距离最小的下一权重向量$\boldsymbol{\omega}_{i+1}$继续进行训练，直到所有权重向量对应的智能体参数都被保存，算法框架如图1所示。

测试阶段流程也保持一致，依次加载每个权重向量对应的SA和RA参数进行实时调度得到对应解。RA和SA进行调度和学习需要进行状态信息、调度规则和奖励信号设计。

2.2 状态信息设计

状态信息设计是强化学习算法的一个重要环节，智能体根据当前状态的信息做出动作与环境互动，状态信息应该保证全面且不冗余。在t时刻调度状态信息s_t包含三部分：系统状态信息sys_t、当前系统状态sys_t与上一调度时刻系统状态sys_{t-1}的差值$diff_t$=sys_t-sys_{t-1}、采样的权重向量$\boldsymbol{\omega}_i$。系统状态信息sys_t与目标函数相关，sys_t={$C_{\max}$, C_{mean}, TD_{mean}, TD_{std}, EC_{mean}, EC_{std}}。

当前完工时间：

$$C_{\max}=\max\{C_i\},i\in N$$

所有机器上的平均完工时间：

$$C_{\mathrm{mean}}=\frac{\sum_{k\in M}\max\{C_{ijk}\}}{m},\forall i,j$$

所有工件的平均延迟时间：

$$TD_{\mathrm{mean}}=\frac{\sum_{i\in N}TD_i}{n}$$

所有工件的延迟时间方差：

$$TD_{\mathrm{std}}=\frac{\sum_{i\in N}(TD_i-TD_{\mathrm{mean}})^2}{n}$$

所有机器的平均加工能耗：

$$EC_{\mathrm{mean}}=\frac{\sum_{k\in M}EC_k}{m}$$

所有机器的加工能耗方差：

$$EC_{\mathrm{std}}=\frac{\sum_{k\in M}(EC_k-EC_{\mathrm{mean}})^2}{m}$$

2.3 调度规则设计

调度规则便于使用能够快速的对动态事件做出反应，被广泛应用于动态调度问题，但是单个调度规则往往只能优化单个目标，因此本文采用多个调度规则作为候选调度规则集合，由智能体根据不同的加工状态在候选调度规则集合中选择合适的一个，候选调度规则集合的设计应保证调度规则排序属性的多样性以及能够优化所有目标函数。

在t时刻机器k完成工件的加工后，SA需要在机器k的队列$Q_k(t)$中选择下一加工工件，考虑本文研究问题目标函数，采用先进先出(first in first out, FIFO)、最大延迟时间(maximum tardiness, MT)、最早预期完成时间(earliest due date, EDD)、关键率(critical ratio, CR)[12]作为SA的候选调度规则集合SA_{rule}={*FIFO, MT, EDD, CR*}，如下所示：

(1) FIFO：argmin$C_{i,j\text{-}1,k}(t)$, $i\in Q_k(t)$, $C_{i,j\text{-}1,k}$为队列工件i的上一工序完成时间即当前工序的入队时间。

(2) MT：argmax$TD_i(t)$, $i\in Q_k(t)$。

(3) EDD：argmin$D_i(t)$, $i\in Q_k(t)$。

(4) CR：计算工件i的预期紧急度和剩余工序加工时间：

$$TTD_i=D_i-A_i$$

$$rm_i(t)=\sum_{j\in O_i(t)}\frac{\sum_{k\in M_{ij}}P_{ijk}}{|M_{ij}|}$$

式中：$Q_i(t)$为t时刻工件i的剩余工序集合。如果$TD_i(t)=0,\ \forall i\in Q_k(t)$，即队列工件都未延期，关键率$cr_i(t)=TTD_i/rm_i(t)$，否则$cr_i(t)=TD_i(t)/rm_i(t)$，argmin$cr_i(t)$, $i\in Q_k(t)$。

类似的，RA需要为工件i当前工序O_{ij}在可用加工机器集合M_{ij}中选择一个加工机器，考虑本文研究问题的目标函数，采用4种启发式调度规则分别为最短加工时间(shortest process time, SPT)、加工能耗最小(minimal energy cost, MEC)、可用时间最早(earliest available, EA)、最短队列时间(shortest queue time, SQT)作为RA的候选调度规则集合RA_{rule}={*SPT, MEC, EA, SQT*}，如下所示：

(1) SPT：argminP_{ijk}, $k\in M_{ij}$。

(2) MEC：argminE_{ijk}, $k\in M_{ij}$。

(3) EA：argmin$ava_k(t)$, $k\in M_{ij}$，$ava_k(t)$为机器k在t时刻时的最早可用时间。

(4) SQT：argmin$qn_k(t)$, $k\in M_{ij}$，$qn_k(t)$为机器k在t时刻的队列工件需要的加工时间总和。

2.4 奖励信号设计

奖励信号与目标函数密切相关，当智能体在当前状态下采取动作后，环境将给予相应的反馈，以评估智能体在当前状态下采取动作的好坏，从而学习到相应的行为策略使得目标函数达到最优。而SA和RA两个智能体分别解决不同的子问题，因此需要分别为SA和RA设计与完工时间、延迟时间和加工能耗相对应的奖励信号。t时刻，SA在队列工件中$Q_k(t)$选择了工件i获得奖励r，SA的奖励r计算如下所示：

(1) 完工时间对应的奖励r_1：计算出平均机器利用率$U_{\text{ave}}(t)$，如果$U_{\text{ave}}(t)>U_{\text{ave}}(t-1)$，$r_1=1$，如果$U_{\text{ave}}(t)=U_{\text{ave}}(t-1)$，$r_1=0$，否则$r_1=-1$。

(2) 延迟时间对应的奖励r_2：计算出工件的平均延迟时间$TD_{\text{ave}}(t)$，如果$TD_{\text{ave}}(t)<TD_{ave}(t-1)$，$r_2=1$，如果$TD_{\text{ave}}(t)=TD_{\text{ave}}(t-1)$，$r_2=0$，否则$r_2=-1$。

(3) 加工能耗对应的奖励的r_3：因为SA负责决定队列中工件的加工顺序，因此SA的奖励信号只与完工时间$C_{\max}$、延迟时间TD相关，因此默认与加工能耗EC相关的r_3为0。

将当前权重系数$\boldsymbol{\omega}_i$与r进行内积得到SA的奖励信号$r=\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{r}$，$\boldsymbol{\omega}_i=(\omega_1, \omega_2, \omega_3)$，$\boldsymbol{r}=(r_1, r_2, r_3)$。

t时刻，RA为工件i当前工序O_{ij}分配了加工机器k获得奖励r，RA的奖励信号计算如下所示：

(1) 完工时间对应的奖励r_1：同样，因为RA负责为工件i分配加工机器，最终的加工顺序是由SA决定的，因此为了便于RA学习到合适的调度策略将与$C_{\max}$相关的r_1默认设为0。

(2) 延迟时间对应的奖励 r_2：计算工件 i 在机器 k 上的预估延迟时间为

$ETD_k(t)=PR_i\cdot\min(0, D_i\text{-}ava_k(t)\text{-}rm_i(t))$

得到在 M_{ij} 上的平均预估延迟时间为

$$ETD_{\text{ave}}(t)=\frac{\sum_{k\in M_{ij}} ETD_k(t)}{|M_{ij}|}$$

如果 $ETD_k(t)< ETD_{\text{ave}}(t)$，$r_2$=1，如果 $ETD_k(t)= ETD_{\text{ave}}(t)$，$r_2$=0，否则 r_2=−1。

(3) 加工能耗对应的奖励的 r_3：计算 t 时刻的当前加工能耗 $EC(t)$ 和累计加工时间 $SP(t)$，然后得到加工时间能耗比为

$$ratio(t)=\frac{SP(t)}{EC(t)}$$

如果 $ratio(t)>ratio(t-1)$，r_3=1，如果 $ratio(t)=ratio(t-1)$，r_3=0，否则 r_3=−1。

同样得到 RA 的奖励信号 $r=\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{r}$，$\boldsymbol{\omega}_i=(\omega_1, \omega_2, \omega_3)$，$\boldsymbol{r}=(r_1, r_2, r_3)$。

3 仿真实验

3.1 动态柔性作业车间调度模拟

为了验证本文所提方法的有效性，综合文献[11,17]的方法对本文所研究的动态调度问题进行模拟，包含的数学模型参数如表 1 所示，其中 $randi[n, m]$ 为 n 与 m 之间的随机整数，$randn[n, m]$ 为 n 与 m 之间的随机浮点数，$round()$ 为向上取整数。

表 1 DFJSP 模型参数
Table 1 DFJSP model parameters

参数	含义	值
m	加工机器数量	{5,10,20,30}
n_f	初始工件数量	1.5 m 或 2 m
n_s	新工件数量	2 m 或 3 m
PR_i	工件优先级	$randi[1, 3]$
ddt_i	工件的截止日期紧急度	$randn[1.0, 1.5]$
n_i	工件所包含的工序数	$n_i= randi[m//2, m]$
M_{ij}	可用加工机器集合大小	$randi[1, m]$
P_{ijk}	加工时间	$randi[20, 50]$
E_{ijk}	加工能耗	$round(randi[0, 5])+40\text{-}P_{ijk}/2)$
A_i	新工件的到达时间	服从指数分布 $\exp(1/\lambda_{\text{new}})$, $\lambda_{\text{new}}=randi[25, 100]$
B_k	机器的故障时间	服从指数分布 $\exp(1/\lambda_{\text{MTBF}})$, λ_{MTBF}=500
R_k	机器故障修复时间	服从指数分布 $\exp(1/\lambda_{\text{MTTR}})$, λ_{MTTR}=50

3.2 评价指标

为验证本文所提算法的有效性，采用 3 个通用的多目标优化问题评价指标 GD[18]、IGD[19]、Spread[20] 用来评价算法的优劣，GD 和 Spread 分别用来评价算法的收敛性和多样性，IGD 同时评价算法的收敛性和多样性。

$$GD(A,P)=\frac{\sqrt{\sum_{i=1}^{|A|} d_{i,A,P}^2}}{|A|}$$

式中：P 为真实的帕累托最优解前沿；A 为评估算法获得的最优帕累托前沿；$d_{i,A,P}$ 为 A 中第 i 个解与 P 中解的最短欧氏距离。

$$IGD(P,A)=\frac{\sum_{i=1}^{|P|} d_{i,P,A}}{|P|}$$

式中：$d_{i,P,A}$ 为 P 中第 i 个解与 A 中所有解的欧氏距离最小值。

$$Spread=\frac{\sum_{j=1}^{n_0} d_{j,A,P}^{\text{e}}+\sum_{i=1}^{|A|}\left|d_{i,A,A}-\bar{d}_{A,A}\right|}{\sum_{j=1}^{n_0} d_{j,A,P}^{\text{e}}+|A|\cdot\bar{d}_{A,A}}$$

式中：$d_{i,A,A}$为A中第i个解到A中最邻近解的欧氏距离；$\bar{d}_{A,A}$为$d_{i,A,A}$的均值；$d^{e}_{j,A,P}$为A和P中第j个目标函数的极值解的欧氏距离；n_0为目标函数个数。

当GD、IGD、Spread值越小值越小，表明算法的收敛性和多样性越好。在本文动态调度问题中，真实的帕累托前沿P是未知的，因此采用所有对比算法的解组成的帕累托前沿作为P的近似。

3.3 算法参数取值及实现细节

为验证训练后智能体的泛化性，本文采用m=30, n_f=20, n_s=40生成训练数据，将训练好的RA和SA参数用于解决其他规模的调度问题。每个智能体包含的*actor*, $actor_{old}$, *critic*都采用多层感知机实现，感知机的隐藏层包含的神经元数量分别为[32, 64, 32]，感知机之间采用的激活函数为tanh，生成100个均匀分布的权重向量，删除不满足约束的权重向量后，余下的权重向量个数*wsize*=28，根据文献[14]给出的参数取值，折扣系数γ=0.9，学习率lr=0.001，超参数ε=0.2。单个权重向量训练次数E的取值范围为{5, 10, 15}，经验池批量大小BS的取值范围为{64, 128, 256}，单次采样学习次数S取值范围为{5, 10, 15}。由于E、BS、S参数存在3^3种组合，为减少实验次数采用正交设计方法[21]构建出$\boldsymbol{L}_9(3^3)$正交矩阵包含9种组合的参数，随机生成的m=40、n_f=20、n_s=60的动态调度问题作为测试问题，采用f作为参数组合的3个评价指标的均值：

$$f=\frac{\sum_{v\in GD,IGD,spread}\frac{v-v_{\min}}{v_{\max}-v_{\min}}}{3}$$

式中：$v_{\max}$和$v_{\min}$分别为9种组合参数中该评价指标的最大值和最小值。

得到$\boldsymbol{L}_9(3^3)$正交矩阵及评价指标均值如表2所示，表3计算出了参数在不同水平取值下的f_{mean}和重要性等级，参数BS最重要，E和S的重要性依次递减。图3给出了3个参数在不同取值下的f_{mean}变化趋势，可以看出：E、BS和S分别取15、128和10时f_{mean}最小。

表2 $\boldsymbol{L}_9(3^3)$正交矩阵及指标均值
Table 2 $\boldsymbol{L}_9(3^3)$ orthogonal matrix and indicator mean values

编号	E	BS	S	f_{mean}
1	5	64	5	0.809 89
2	5	128	10	0.310 65
3	5	256	15	0.488 57
4	10	64	10	0.534 64
5	10	128	15	0.218 04
6	10	256	5	0.549 34
7	15	64	15	0.390 70
8	15	128	5	0.227 75
9	15	256	10	0.183 09

表3 参数的f_{mean}值及重要性等级
Table 3 f_{mean} values and importance levels of the parameters

取值水平	E	BS	S
1	0.536 37	0.578 41	0.528 99
2	0.434 00	0.252 14	0.342 79
3	0.267 18	0.407 00	0.365 77
∇f_{mean}	0.269 19	0.326 27	0.186 10
重要性等级	2	1	3

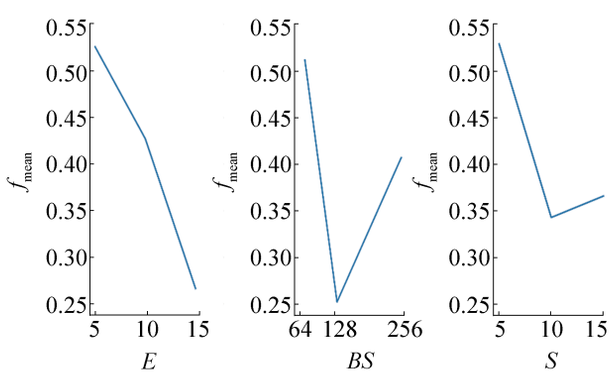


图3 f_{mean}变化趋势
Fig. 3 Trend of f_{mean}

3.4 与调度规则组合对比

本文将MPPO的实验结果与采用的调度规则进行对比，选择单个RA或单个SA调度规则和随机调度规则的组合作为对比方法，为了验证RA和SA智能体学习到了可行的调度规则选择策略，增加随机RA和随机SA调度规则作对比，因此包含

https://dc-china-simulation.researchcommons.org/journal/vol36/iss7/9
DOI: 10.16182/j.issn1004731x.joss.23-0385

FIFO+R、MT+R、EDD+R、CR+R、R+SPT、R+MEC、R+EA、R+SQT、R+R，一共9种对比调度规则组合，R表示随机选取调度规则，如FIFO+R为FIFO与随机SA调度规则的组合，同时为保证得到的解数目相同，调度规则组合运行的次数与本文MPPO生成的权重向量个数相同。

实验结果如表4~6所示，黑色加粗为所有算法中取得的最好值，如在n_f=7、m=5、n_s=10时，R+SQT组合在GD评价指标上取得了最优值0.021。在GD评价指标上，R+SQT在4个测试问题上取得最好表现，但是在所有测试问题的IGD和Spread评价指标上R+SQT都未取得最优值，说明R+SQT取得的帕累托最优解在单个目标函数上取值较小，难以被其余方法取得的解所支配，因此计算出来的GD指标值较小，但是R+SQT取得的帕累托最优解不具备多样性，因此计算出来的IGD和Spread值较大。而MPPO算法在GD评价指标的3个测试问题上取得了最好表现，并且在IGD和Spread指标上取得的最优次数最高，因此证明了MPPO算法的解具备较好的收敛性和多样性。

为了进一步进行分析，图4~6给出了三种规模下MPPO算法与调度规则组合取得的帕累托最优解分布图，不同颜色和标记代表不同的算法取得的帕累托最优解，在图4中9种调度规则组合获得的帕累托最优解分布具有一定的均匀性，而随着问题规模的增长在图5~6中调度规则组合的帕累托最优解分布紧密。

表4 帕累托最优解GD值
Table 4 Pareto optimal solution GD values

n_f	m	n_s	MPPO	FIFO+R	MT+R	EDD+R	CR+R	R+SPT	R+MEC	R+EA	R+SQT	R+R
7	5	10	0.036	0.045	0.039	0.048	0.039	0.027	0.023	0.047	**0.021**	0.044
10	5	15	0.052	0.066	0.067	0.06	0.067	0.050	**0.019**	0.064	0.031	0.071
15	10	20	**0.022**	0.029	0.028	0.033	0.027	0.058	0.024	0.042	0.026	0.031
20	10	30	0.022	0.047	0.031	0.034	0.033	0.014	0.017	0.056	**0.007**	0.036
30	20	40	0.018	0.033	0.028	0.035	0.032	0.017	0.014	0.059	**0.005**	0.03
40	20	60	**0.018**	0.040	0.027	0.028	0.026	0.019	0.022	0.063	0.020	0.026
45	30	60	**0.014**	0.018	0.020	0.024	0.023	0.018	0.019	0.054	0.026	0.019
60	30	90	0.019	0.037	0.024	0.027	0.026	0.005	0.009	0.056	**0.003**	0.028

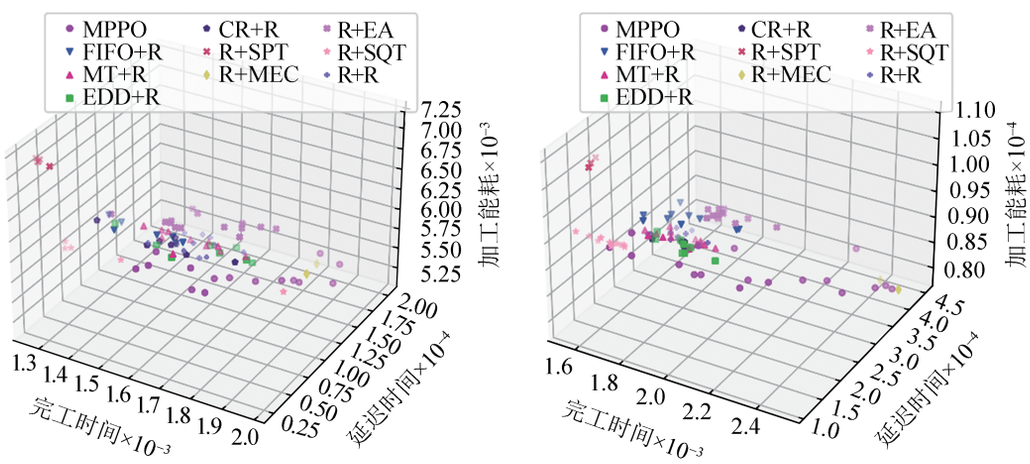
表5 帕累托最优解IGD值
Table 5 Pareto optimal solution IGD values

n_f	m	n_s	MPPO	FIFO+R	MT+R	EDD+R	CR+R	R+SPT	R+MEC	R+EA	R+SQT	R+R
7	5	10	0.040	0.035	**0.034**	0.039	0.034	0.131	0.131	0.040	0.045	0.042
10	5	15	0.099	0.093	0.111	0.096	**0.085**	0.244	0.364	0.136	0.138	0.096
15	10	20	**0.058**	0.080	0.082	0.072	0.069	0.152	0.175	0.073	0.095	0.082
20	10	30	**0.026**	0.068	0.069	0.070	0.073	0.15	0.176	0.075	0.095	0.068
30	20	40	**0.024**	0.072	0.084	0.073	0.082	0.163	0.186	0.083	0.110	0.08
40	20	60	**0.023**	0.071	0.069	0.07	0.067	0.133	0.162	0.078	0.087	0.075
45	30	60	**0.015**	0.071	0.072	0.066	0.068	0.142	0.173	0.070	0.089	0.071
60	30	90	**0.026**	0.085	0.086	0.081	0.087	0.158	0.188	0.084	0.109	0.081

表6 帕累托最优解Spread值

Table 6 Pareto optimal solution Spread values

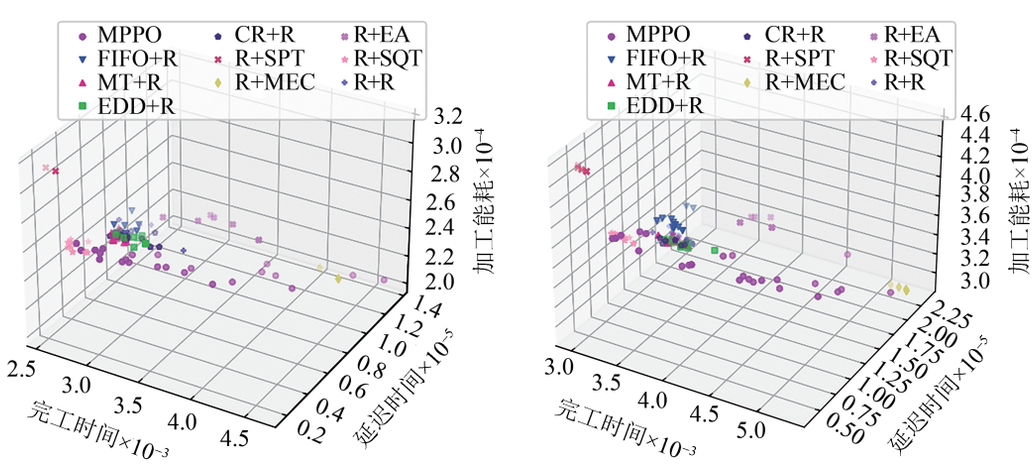
n_f	m	n_s	MPPO	FIFO+R	MT+R	EDD+R	CR+R	R+SPT	R+MEC	R+EA	R+SQT	R+R
7	5	10	0.681	0.725	0.670	0.673	**0.660**	0.976	1.019	0.753	0.684	0.740
10	5	15	0.800	**0.649**	0.666	0.682	0.682	1.012	0.994	0.812	0.897	0.760
15	10	20	0.710	0.715	**0.707**	0.711	0.780	0.944	0.932	0.774	0.811	0.773
20	10	30	0.747	0.788	0.791	0.805	0.837	0.966	0.926	0.78	0.885	**0.745**
30	20	40	**0.742**	0.889	0.887	0.799	0.807	0.942	0.952	0.857	0.901	0.870
40	20	60	**0.798**	0.828	0.849	0.860	0.879	0.981	0.946	0.878	0.940	0.894
45	30	60	**0.785**	0.856	0.880	0.826	0.836	0.984	0.939	0.823	0.906	0.854
60	30	90	**0.822**	0.899	0.884	0.888	0.872	0.996	0.965	0.887	0.959	0.888



(a) n_f=15, m=10, n_s=20 (b) n_f=20, m=10, n_s=30

图4 小规模问题上的帕累托前沿分布图

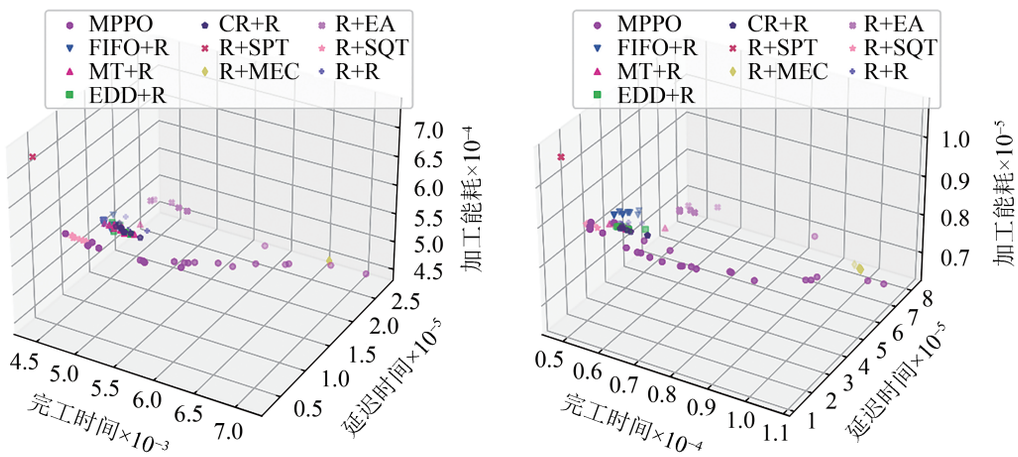
Fig. 4 Pareto frontier distribution on small scale problems



(a) n_f=30, m=20, n_s=40 (b) n_f=40, m=20, n_s=60

图5 中规模问题上的帕累托前沿分布图

Fig. 5 Pareto frontier distribution on medium scale problems



(a) n_f=45, m=30, n_s=60　　(b) n_f=60, m=30, n_s=90

图6　大规模问题上的帕累托前沿分布图

Fig. 6　Pareto frontier distribution on big scale problems

在图4~6中，R+SPT组合的解都分布在最左上方完工时间和延迟时间较小，而加工能耗最大的位置，而R+EA组合的解则分布在完工时间和加工能耗较小，而延迟时间最大的位置，而其余组合获得的解也类似，难以优化所有的目标函数，因为单个调度规则的功能比较单一只能优化单个目标，即便与另一随机调度规则进行组合，随着问题规模的增大也难以保证解的均匀性。

而R+R组合只在表6中的第4个问题Spread评价指标上取得了最优值0.745，在图4~6中的帕累托最优解的分布也不具备性和多样性，说明完全随机的调度规则组合对收敛性和多样性都无法保证，而随着问题规模的增长可以看出MPPO算法获得的解在收敛性和多样性上都优于9种调度规则组合，证明了MPPO算法训练得到的SA和RA智能体学习到了合适的调度策略，能够根据实时加工状态选择不同的调度规则进行调度并保证调度方案的表现。

4　结论

本文针对带有机器故障和随机工件到达的动态调度问题，提出MPPO实时调度方法。MPPO算法训练了SA和RA两个智能体以实现实时调度，通过触发智能体调度和更新实时状态的方式对动态事件进行处理。为优化多个目标，采用权重向量与奖励向量线性组合作为奖励信号，对每个权重向量依次进行训练并保存对应的智能体参数。然后结合目标函数分别为SA和RA定义了状态信息、调度规则和奖励信号。仿真实验证明了MPPO算法的表现在多样性和收敛性上都要优于调度规则组合，而单一调度规则组合难以保证解的多样性。在未来的研究工作中会考虑采取更高效的训练方法优化多个目标；更通用的状态信息以便于迁移到不同动态调度问题；采用更高级网络结构提取状态信息。

参考文献：

[1] Luo Shu. Dynamic Scheduling for Flexible Job Shop with New Job Insertions by Deep Reinforcement Learning[J]. Applied Soft Computing, 2020, 91: 106208.

[2] 尤一琛, 王艳, 纪志成. 基于博弈论的柔性作业车间动态调度研究[J]. 系统仿真学报, 2021, 33(11): 2579-2588.
You Yichen, Wang Yan, Ji Zhicheng. Research on Flexible Job-shop Dynamic Scheduling Based on Game Theory[J]. Journal of System Simulation, 2021, 33(11): 2579-2588.

[3] An Youjun, Chen Xiaohui, Gao Kaizhou, et al. A Hybrid Multi-objective Evolutionary Algorithm for Solving an Adaptive Flexible Job-shop Rescheduling Problem with Real-time Order Acceptance and Condition-based Preventive Maintenance[J]. Expert Systems with

Applications, 2023, 212: 118711.

[4] Wen Xiaoyu, Lian Xiaonan, Qian Yunjie, et al. Dynamic Scheduling Method for Integrated Process Planning and Scheduling Problem with Machine Fault[J]. Robotics and Computer-Integrated Manufacturing, 2022, 77: 102334.

[5] Oliver Holthaus, Chandrasekharan Rajendran. Efficient Dispatching Rules for Scheduling in a Job Shop[J]. International Journal of Production Economics, 1997, 48 (1): 87-105.

[6] Jun S, Lee S, Chun H. Learning Dispatching Rules Using Random Forest in Flexible Job Shop Scheduling Problems[J]. International Journal of Production Research, 2019, 57(10): 3290-3310.

[7] Paolo Priore, Ponte B, Javier Puente, et al. Learning-based Scheduling of Flexible Manufacturing Systems Using Ensemble Methods[J]. Computers & Industrial Engineering, 2018, 126: 282-291.

[8] Paulo Roberto do O da Costa, Jason Rhuggenaath, Zhang Yingqian, et al. Learning 2-opt Heuristics for the Traveling Salesman Problem via Deep Reinforcement Learning[C]//Proceedings of the 12th Asian Conference on Machine Learning. Chia Laguna Resort, Sardinia, Italy: PMLR, 2020: 465-480.

[9] Lu Hao, Zhang Xingwen, Yang Shuang. A Learning-based Iterative Method for Solving Vehicle Routing Problems[C]//ICLR 2020 Conference Blind Submission. New York, USA: ICLR, 2020.

[10] Zeng Yunhui, Liao Zijun, Dai Yuanzhi, et al. Hybrid Intelligence for Dynamic Job-shop Scheduling with Deep Reinforcement Learning and Attention Mechanism [EB/OL]. (2022-01-03) [2023-03-20]. https://arxiv. org/abs/2201.00548.

[11] Luo Shu, Zhang Linxuan, Fan Yushun. Real-time Scheduling for Dynamic Partial-no-wait Multiobjective Flexible Job Shop by Deep Reinforcement Learning[J]. IEEE Transactions on Automation Science and Engineering, 2022, 19(4): 3020-3038.

[12] Liu Renke, Piplani R, Carlos Toro. Deep Reinforcement Learning for Dynamic Scheduling of a Flexible Job Shop [J]. International Journal of Production Research, 2022, 60(13): 4049-4069.

[13] Li Funing, Sebastian Lang, Hong Bingyuan, et al. A Two-stage RNN-based Deep Reinforcement Learning Approach for Solving the Parallel Machine Scheduling Problem with Due Dates and Family Setups[J]. Journal of Intelligent Manufacturing, 2024, 35(3): 1107-1140.

[14] Schulman J, Wolski F, Dhariwal P, et al. Proximal Policy Optimization Algorithms[EB/OL]. (2017-08-28) [2023-03-14]. https://arxiv.org/abs/1707.06347.

[15] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level Control Through Deep Reinforcement Learning[J]. Nature, 2015, 518(7540): 529-533.

[16] Li Kaiwen, Zhang Tao, Wang Rui. Deep Reinforcement Learning for Multiobjective Optimization[J]. IEEE Transactions on Cybernetics, 2021, 51(6): 3103-3114.

[17] Shen Xiaoning, Yao Xin. Mathematical Modeling and Multi-objective Evolutionary Algorithms Applied to Dynamic Flexible Job Shop Scheduling Problems[J]. Information Sciences, 2015, 298: 198-224.

[18] Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results[J]. Evolutionary Computation, 2000, 8 (2): 173-195.

[19] Zitzler E, Thiele L. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach[J]. IEEE Transactions on Evolutionary Computation, 1999, 3(4): 257-271.

[20] Deb K, Pratap A, Agarwal S, et al. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182-197.

[21] Leung Y W, Wang Yuping. An Orthogonal Genetic Algorithm with Quantization for Global Numerical Optimization[J]. IEEE Transactions on Evolutionary Computation, 2001, 5(1): 41-53.

DOI: 10.16182/j.issn1004731x.joss.23-0385