# 考虑运输资源限制的动态柔性车间调度问题及自适应竞争重构算法

王 聪[1,2] 魏立新[1,2] 孙 浩[1,2*] 呼子宇[1,2] 崔慧慧[3]
1.燕山大学工业计算机控制工程河北省重点实验室,秦皇岛,066004
2.燕山大学电气工程学院,秦皇岛,066004
3.北京航天新立科技有限公司,北京,100039

**摘要**:针对具有运输资源的动态柔性车间调度问题设计了两阶段自适应竞争重构算法(TACRA)。初始阶段,TACRA在静态环境中运行。一旦机器故障发生,TACRA进入重调度阶段。TACRA包含可增强算法探索与开发能力的删除算子和重构算子,以及基于算子历史性能的自适应选择机制。15个测试实例中,TACRA分别在反向世代距离、超体积和适应度上取得11、15和15个最优结果。

**关键词**:柔性车间调度;多目标优化;运输资源;动态事件;自适应选择
**中图分类号**:TP182
DOI:10.3969/j.issn.1004-132X.2026.07.020 **开放科学(资源服务)标识码(OSID)**:

# Dynamic Flexible Job Shop Scheduling Problem Considering Transportation Resource Constraints and Adaptive Competitive Reconfiguration Algorithm

WANG Cong[1,2] WEI Lixin[1,2] SUN Hao[1,2*] HU Ziyu[1,2] CUI Huihui[3]
1.Key Laboratory of Industrial Computer Control Engineering of Hebei Province, Yanshan University, Qinhuangdao, Hebei, 066004
2.School of Electrical Engineering, Yanshan University, Qinhuangdao, Hebei, 066004
3.Beijing Aerospace Xinli Technology Co., Ltd., Beijing, 100039

**Abstract**: This paper proposes a two-stage adaptive competitive reconfiguration algorithm (TACRA) for the dynamic flexible job shop scheduling problem with transportation resources. In the initial phase, TACRA operates in a static environment; once a machine breakdown occurs, it switches to a rescheduling phase. Deletion and reconstruction operators are designed to enhance the algorithm's exploration and exploitation capabilities, and an adaptive selection mechanism is introduced based on the historical performance of these operators. Experimental results on 15 test instances show that TACRA achieves the optimal inverted generational distance in eleven cases, the optimal hypervolume in fifteen cases, and the optimal fitness metric in fifteen cases.

**Key words**: flexible job shop scheduling; multi-objective optimization; transportation resource; dynamic event; adaptive selection

## 0 引言

现代制造业正经历智能制造的快速转型与多品种小批量生产模式的普及,这一趋势对生产调度的灵活性、响应速度和抗干扰能力提出更高要求[1-2]。柔性作业车间调度问题(flexible job shop scheduling problem,FJSP)直接关系生产效率的提高和成本的降低,因此受到广泛关注。一般的FJSP包含两个子问题即机器分配和工序排序[3],但研究通常将优化重点放在子问题上,默认运输资源充足或运输时间可被忽略,即常采用假设自动引导车辆(AGV)的数量无限。实际上,工件需要被AGV运送到指定机器加工;机器没有释放工件,AGV也无法运输[4];AGV的选择直接影响运输路线和运输时间,从而影响最终完工时间。因此机器和AGV的协同调度对最大限度提高设备利用率和生产效率非常关键,具有运输资源的柔性作业车间调度问题(flexible job shop scheduling problem with transportation resources,FJSPT)具有重要的研究意义。

实际的调度会遇到很多的突发情况,如机器故障、新工作订单到来、交货期限改变[5],从而打乱最初设计的调度方案。为减小机器故障的不利

**收稿日期**:2025-06-20
**基金项目**:国家重点研发计划(2022YFB3705504);国家自然科学基金(62273295);河北省自然科学基金(F2024203089);河北省重点研发计划(21310301D);河北省重点实验室项目(22567612H)

影响，调度算法引入重调度策略，如基于知识的两阶段模因算法框架[6]在重调度阶段采用知识驱动的可变邻域搜索策略，以快速找到最优解并深入探索解空间。WEN等[7]设计了一种结合遗传算法和局部搜索算法的混合算法来处理机器故障下的动态调度。TANG等[8]提出自适应调整参数和改进精英选择策略，以增强重调度的稳定性。一些使用集成强化学习的遗传算法可应对机器突发故障带来的生产调度优化难题[3]，即产生高质量的解，但预训练耗时太长，难以满足算法的效率要求，尤其是在处理高维状态空间或大规模车间问题。

在具有运输资源的动态柔性作业车间调度问题（dynamic flexible job shop schedule problem with transportation resources，DFJSPT）中，机器故障影响机器与AGV。机器发生故障时，立即停止工作，受影响的工件必须及时分配给其他可用机器，空闲的AGV被迅速调用。因此，需要同步调度机器和AGV，以确保生产的顺利进行。

为提高调度系统的灵活性和应急响应能力，应对日益复杂和不断变化的挑战，本文针对DFJSPT设计了两阶段自适应竞争重构算法（two-stage adaptive competitive reconfiguration algorithm，TACRA）。

## 1　DFJSPT的描述与建模

### 1.1　问题描述

DFJSPT需要解决耦合的4个子问题：①确定工序的顺序；②为每个工序选择加工机器；③为每个工序分配一台AGV完成运输任务；④机器故障发生后重新调度。

假设有$n$个待加工工件、$m$台机器、$a$个AGV。工件$J_i$（$i$=1，2，…，$n$）的第$j$个工序$O_{ij}$由可选机器集$M_{ij}$中的一台机器完成。DFJSPT从数量有限的AGV（可运输任何工件）中选择一台分配给工序，并划分相应的运输时间。每个被分配的AGV在特定时间窗内只服务于特定工序[9]。开始加工时，AGV将工件从装卸LU区运送到加工机器。工序$O_{ij}$在机器$M_k$∈$M_{ij}$上完成后，AGV运输$J_i$到下一台机器，最终完成的工件由AGV运送回装卸LU区。

DFJSPT由初始阶段、重调度阶段组成。初始阶段内，没有机器发生故障，所有资源都可用。机器发生故障时，重调度被触发。故障修复完成前，故障机器不能工作。机器故障包含3个关键因素：故障机器的索引$k_{\mathrm{B}}$、机器故障的发生时间$B_{\mathrm{S}}$和修复时间$B_{\mathrm{T}}$。所有未处理和受故障影响的工序将根据重调度策略重新计划。DFJSPT模型包含如下约束和假设：①机器在同一时刻只能加工一个工件，AGV在同一时刻只能运输一个工件；②工序的顺序约束只对同一工件有效；③所选机器是工序的可选机器之一；④除机器故障外，加工不能中断；⑤故障发生后，故障机器不再工作，直到修复完成；⑥AGV沿最短路径行驶，运输过程中没有拥挤和中断；⑦AGV在机器之间的运输时间已知。DFJSPT数学模型中涉及的符号及其含义如表1所示。

**表1　模型符号及含义**

**Tab.1　Notations and meanings of DFJSPT**

| 符号 | 定义 |
|---|---|
| $i$ | 工件索引，$i$∈{1，2，…，$n$} |
| $j$ | 工序索引，$j$∈{1，2，…，$n_i$} |
| $k$ | 机器索引，$k$∈{1，2，…，$m$} |
| $v$ | AGV索引，$v$∈{1，2，…，$a$} |
| $n$ | 工件数量 |
| $m$ | 机器数量 |
| $a$ | AGV数量 |
| $O_{ij}$ | 工件$i$的第$j$个工序 |
| $J$ | 工件集合，以$i$为索引 |
| $J_i$ | 工件$i$的工序集合，以$j$为索引 |
| $M$ | 机器集合，以$k$为索引 |
| $M_{ij}$ | $O_{ij}$的可选机器集合 |
| $V$ | AGV集合，以$v$为索引 |
| $t_{ijk}$ | $O_{ij}$在机器$M_k$上的处理时间 |
| $C_{\max}$ | 最大完成时间 |
| $t_{\mathrm{AM}}$ | 当前调度方案的机器平均运行时间 |
| $t_{\mathrm{AV}}$ | 当前调度方案的AGV平均负载时间 |
| $t_{\mathrm{S},ij}$ | 工件$i$的第$j$个工序的开始时间 |
| $t_{\mathrm{E},ij}$ | 工件$i$的第$j$个工序的结束时间 |
| $k_{\mathrm{B}}$ | 故障机器 |
| $B_{\mathrm{S}}$ | 故障的发生时间 |
| $B_{\mathrm{T}}$ | 故障的修复时间 |
| $G$ | 一个无穷大的数字（或限制） |
| $L$ | 工序数的总和 |
| $t_{ijqkv}$ | $O_{ij}$在机器$M_q$加工完成后，$V_v$将$J_i$运输到$M_k$的时间 |
| $x_{ijk}$ | 0-1变量，若$O_{ij}$在$M_k$上完成，则$x_{ijk}$=1，否则$x_{ijk}$=0 |
| $z_{ijhgk}$ | 0-1变量，若$M_k$在$O_{ij}$之前完成$O_{hg}$，则$z_{ijhgk}$=1，否则$z_{ijhgk}$=0 |
| $\alpha_{ijqkv}$ | 0-1变量，若$O_{ij}$在机器$M_q$加工完成后，$V_v$将$J_i$运输到$M_k$，则$\alpha_{ijqkv}$=1，否则$\alpha_{ijqkv}$=0 |
| $X_{\mathrm{os}}$ | 工序排序编码，表示所有工序的加工顺序 |
| $X_{\mathrm{ms}}$ | 机器选择编码，包含工序排序编码中各工序对应的加工机器 |
| $X_{\mathrm{vs}}$ | AGV调度编码，表示AGV调度顺序 |

### 1.2　数学模型

考虑生产对效率的要求，将最小化最大完工时间作为目标$f_1$。为防止设备长期过载、增加故

障风险，将最小化负载不均衡度即每台机器和AGV的负载偏离平均负载的程度（比率）[10]作为目标$f_2$。$f_1$与$f_2$的目标函数如下：

$$f_1:\min C_{\max}=\min\left(\max\sum_{i\in J,j\in J_i}t_{E,ij}\right) \tag{1}$$

$$f_2:\min\left\{\frac{1}{t_{AM}}\left|\sum_{k\in M}\left(\sum_{i\in J}\sum_{j\in J_i}t_{ijk}x_{ijk}-t_{AM}\right)\right|+\frac{1}{t_{AV}}\sum_{v\in V}\left|\sum_{i\in J}\sum_{j\in J_i}\sum_{q\in M}\sum_{k\in M_{ij}}t_{ijqkv}\alpha_{ijqkv}-t_{AV}\right|\right\} \tag{2}$$

约束：

$$t_{E,hg}\leqslant t_{S,ij}+(1-z_{ijhgk})G$$
$$\forall i\in J,j\in J_i,h\in J\backslash\{i\},g\in J_h,k\in M_{ij}\cap M_{hg} \tag{3}$$

$$t_{E,ij}-t_{S,ij}=\begin{cases}G & t_{S,ij}\leqslant B_S\leqslant t_{E,ij}\text{且}k=k_B\\ t_{ijk}x_{ijk} & \text{其他}\end{cases} \tag{4}$$

$$B_S+B_T\leqslant t_{S,ij}+(1-x_{ijk})G$$
$$B_S\leqslant t_{S,ij}\leqslant B_S+B_T\text{且}k=k_B \tag{5}$$

$$t_{S,ij}\geqslant t_{E,i(j-1)}\quad\forall i\in J,j\in J_i \tag{6}$$

$$C_{\max}\geqslant t_{E,ij}\quad\forall i\in J,j\in J_i \tag{7}$$

$$\sum_{k\in M_{ij}}x_{ijk}=1\quad\forall i\in J,j\in J_i \tag{8}$$

$$t_{AM}=\frac{1}{m}\sum_{k\in M}\sum_{i\in J}\sum_{j\in J_i}t_{ijk}x_{ijk} \tag{9}$$

$$t_{AV}=\frac{1}{a}\sum_{v\in V}\sum_{i\in J}\sum_{j\in J_i}\sum_{q\in M}\sum_{k\in M_{ij}}t_{ijqkv}\alpha_{ijqkv} \tag{10}$$

$$\sum_{v\in V}\sum_{q\in M_{ij-1}}\sum_{k\in M_{ij}}\alpha_{ijqkv}=1\quad\forall i\in J,j\in J_i \tag{11}$$

$$\sum_{v\in V}\sum_{q\in M_{i(j-1)}}\sum_{k\in M_{ij}}x_{ijk}\alpha_{ijqkv}=1\quad\forall i\in J,j\in J_i \tag{12}$$

$$t_{E,ij},t_{S,ij},t_{ijk},t_{ijqkv}\geqslant 0,$$
$$\forall i\in J,j\in J_i,q\in M_{ij-1},k\in M_{ij},v\in V \tag{13}$$

$$z_{ijhgk}\in\{0,1\},$$
$$\forall i\in J,j\in J_i,h\in J\backslash i,g\in J_h,k\in M_{ij}\cap M_{hg} \tag{14}$$

式(3)表示一台机器同时只能处理一个工序，其中，$J$为工件集合，$J\backslash\{i\}$为剔除工件$i$的工件集合。式(4)表示机器$M_{k_B}$发生故障，导致工序$O_{ij}$中断，工序结束时间$t_{E,ij}$被无限延迟，其他机器则不受影响，满足加工不中断的常规约束。式(5)表示在故障机器$M_{k_B}$正在加工且开始时间位于故障修复窗口内的工序$O_{ij}$，开始加工时间$t_{S,ij}$必须被延迟到修复完成时间$B_S+B_T$之后。由式(6)可知，同一工件的工序存在先后约束。式(7)表示任何工序的完成时间都不能超过最大完工时间。式(8)表示一个工序只能在一台机器上完成。式(11)表示运输操作仅由一个AGV完成。式(12)表示机器和AGV的耦合关系。

### 1.3 确定重调度阶段的资源可用性

重调度阶段需要更新所有受故障影响和未处理工序的状态，而确定这些工序是重调度面临的第一个问题。由图1所示的初始调度可知，$M_3$的故障发生时间$T_b$=5.5 h，$O_{12}$是正在处理的工序。由于$O_{22}$、$O_{32}$的AGV运输过程$M_1\rightarrow M_2$和$M_3\rightarrow M_1$已经开始，因此将$O_{22}$和$O_{32}$归类为处理中的工序。值得注意的是，即使$O_{22}$和$O_{32}$尚未完成，也会被取消，并在重调度时被新的工序取代，以保证生产过程尽快恢复正常。$O_{13}$、$O_{33}$是未开始的工序，重新安排未开始的工序时，相应的运输工序也会发生变化。

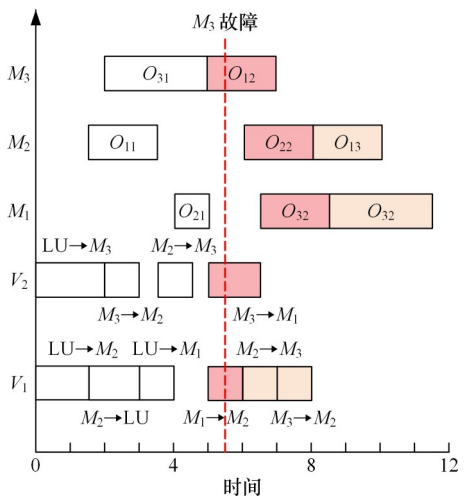



**图1 DFJSPT的初始调度示例**

**Fig.1 An example of initial scheduling for DFJSPT**

## 2 两阶段自适应竞争重构算法

TACRA在初始阶段和重调度阶段的处理方法不同。初始阶段时，TACRA利用NSGA-Ⅱ[11]优化种群。当检测到机器故障时，TACRA进入重调度阶段。为增强算法的探索和开发能力，设计出竞争的删除算子和重构算子，以及依赖算子历史性能的自适应选择机制。删除算子从解中删除性能差的工序、机器和AGV。重构算子利用精英解的知识生成新的解。因此，TACRA能探索更大的解空间，找到更好的解决方案。图2为TACRA流程图。

### 2.1 编码和解码

对解进行编码涉及3个子问题：工序排序(OS)、机器选择(MS)和AGV调度(VS)，因此使用三层编码方法。如图3所示，首先生成OS向量，然后根据OS和相应的约束确定MS向量和VS向量。OS、MS、VS组合的三层编码构成问题的一个解决方案，每层的长度$L$为所有工件包含的加工工序总数。OS、MS、VS三层编码的具体含义：

(1)OS层编码。OS层编码中的元素为工件编号，第一次出现$i$表示为工序$O_{i1}$，第二次出现$i$表示为工序$O_{i2}$，依此类推。

(2)MS层编码。MS层编码中的元素为可选

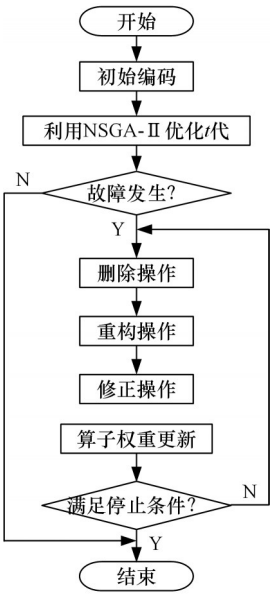



**图2　TACRA的流程图**

**Fig.2　Flowchart of TACRA**

机器集合中的机器索引。如图3所示，MS层编码中的第一个元素2表示工序$O_{11}$选择可选机器集$\{M_1, M_3\}$中的机器$M_3$。

(3)VS层编码。VS层编码的元素为被选AGV的索引。如图3所示，VS层编码第二个元素2表示工序$O_{21}$对应的工件$J_2$由$V_2$运输。

| | $O_{11}$ | $O_{21}$ | $O_{22}$ | $O_{31}$ | $O_{32}$ | $O_{12}$ |
|---|---|---|---|---|---|---|
| OS | 3 | 1 | 1 | 2 | 2 | 3 |
| MS | 2 | 2 | 2 | 4 | 2 | 1 |
| VS | 1 | 2 | 1 | 1 | 2 | 2 |

**图3　编码图示**

**Fig.3　Illustration of the coding method**

## 2.2　删除操作

删除操作从当前解$X\in P_t$的编码中删除指定数量的元素，并返回$X$剩余的元素作为编码$X_d$。删除时暂时忽略工序之间的约束关系。第$t$代删除元素的数量计算公式为

$$N_{d,t}=L(Q_1-(Q_1-Q_0)t/T) \tag{15}$$

式中：$L$为$X$的长度；$Q_0$为初始删除率；$Q_1$为最终删除率；$T$为进化总代数。

删除操作包括4个删除算子：

(1)随机删除算子(RD)。从OS层中随机删除$N_{d,t}$个元素后，MS和VS相应位置的元素也被删除，将删除的元素保存在集合$\varphi$中，以用于随后的重构操作。

(2)基于完成时间的删除算子(CD)。对于OS，按照完成时间

$$t_{E,ij}=t_{S,ij}+t_{ijk} \tag{16}$$
$$ij=X_{os}[l], k=X_{ms}[l], l\in\{1,2,\cdots,L\}$$

式中：$ij$表示工序索引；$l$表示编码位置索引。

从晚到早依次删除$N_{d,t}$个元素，同时删除MS和VS中相应位置的元素。

(3)基于机器运行时间和AGV负载时间的删除算子(RL)。RL用于平衡机器和AGV的工作压力，减小故障概率。对MS和VS删除工作时间较长的$N_{d,t}$个元素。机器运行时间$t_{run,k}$和AGV负载时间$t_{laod,v}$分别为

$$t_{run,k}=t_{ijk} \tag{17}$$
$$t_{load,v}=t_{cl,ijv}-t_{sl,ijv}v=X_{vs}[l] \tag{18}$$

式中：$t_{sl,ijv}$、$t_{cl,ijv}$分别为AGV运输$O_{ij}$的开始和结束时间。

(4)基于机器空闲时间和AGV空载时间的删除算子(IN)。从MS和VS中删除$N_{d,t}$个空闲时间和空载时间较长的元素。MS每个元素的空闲时间为

$$t_{idle,k}=t_{S,ij}-\max(t_{E,ij-1}, t_{E,hg}z_{ijhgk}) \tag{19}$$
$$hg=X_{os}[l'], k=X_{ms}[l], l, l'\in\{1,2,\cdots,L\}$$

式中：$t_{idle,k}$为机器$M_k$的空闲时间；$t_{E,ij-1}$为$O_{ij-1}$的结束时间；$hg$表示$X_{os}$中的一个元素；$l$和$l'$表示编码位置索引。

对于VS每个元素的空载时间为

$$t_{noload,v}=t_{ce,ijv}-t_{se,ijv} \tag{20}$$

式中：$t_{se,ijv}$、$t_{ce,ijv}$分别为AGV运输$O_{ij}$前的空载运输的开始和结束时间；$v$为机器索引。

## 2.3　重构操作

应用删除算子后，一些表现不佳的元素被删除，因此须设计重构算子来重建被破坏的解$X_d$，生成新解$X_r$。重构生成新解$X_r$的首要目标是增强解的探索能力，其次是提高种群的多样性。为实现这些目标，迭代过程中将保留的非支配解集$P_e$作为精英解集来指导重构过程。删除算子对应的重构算子如下。

(1)随机插入重构算子(RI)。RI针对删除算子RD设计。对于OS层编码，删除元素随机插入$X_d$，然后同样，按照OS插入元素的位置向MS层编码和VS层编码插入对应元素。因此RI可加解的多样性。

(2)基于完成时间的重构算子(CR)。CR针对删除算子CD设计。由于最小化最大完工时间是目标$f_1$，因此选择$P_e$中最小化最大完工时间最小的精英解$X_e$重构$X_d$，即将$X_e$中$X_d$的空位置对应的元素插入$X_d$的空位，填充被删除的元素，得到新解$X_r$。

(3)基于负载不均衡度的重构算子(LI)。LI针对删除算子RL设计。最小化负载不均衡度是目标$f_2$，选择$P_e$中最小化负载不均衡度最小的精英解$X_e$来重构$X_d$。与CR相同，$X_e$对应$X_d$的空位置对应的元素被插入$X_d$的空位置。

(4)基于加权目标的重构算子(WR)。WR针对删除算子IN设计。为进一步挖掘平衡目标$f_1$

和$f_2$的高质量解，使用加权目标值最小的精英解$X_e$来重构$X_d$，目标$f_1$、$f_2$的加权目标权重均为0.5。

图4给出了重构操作的两个例子。由于选择的重构算子不同，图4a、图4b所示的重构过程略有不同。图4a中，原始解$X$首先执行RD，随机移除元素$O_{12}$、$O_{22}$，获得$X_d$。RI将被移除元素随机插入$X_d$、得到新解$X_r$。在图4b中，$X$使用RL移除较差的$O_{11}$、$O_{22}$。图4b中的LI利用精英解$X_e$将$O_{12}$、$O_{22}$放在$X_d$的空位、得到新解$X_{r,new}$。

### 2.4 修正操作

重构算子生成新解$X_r$后，部分元素的分配可能违反约束，导致$X_r$不可行，须修正不可行的解。OS不可行解含有重复、缺失或错误排序的元素。如图4a所示，解$X$经过RD和RI处理后，新解$X_r$出现$O_{12}$先于$O_{11}$的错误。修正操作调整$O_{12}$、$O_{11}$的顺序，使新解$X_r$成为可行的解决方案。如图4b例子所示，利用精英解$X_e$重构后，新解$X_{r,new}$的OS中重复出现$O_{12}$，但缺少$O_{11}$。修正操作保留第一次出现的$O_{12}$，删除多余的重复元素，并用缺少的元素$O_{11}$替换重复元素。MS不可行解的某些元素即机器不在工序对应的可选机器集$M_{ij}$中。如果工序对应的机器不属于$M_{ij}$，则随机选择一台属于$M_{ij}$的机器来替换不属于$M_{ij}$的机器。由于工件可由任一AGV运输，因此不需要修正VS。经过修正操作逐一检查和纠正编码后，新解$X_r$为可行解。

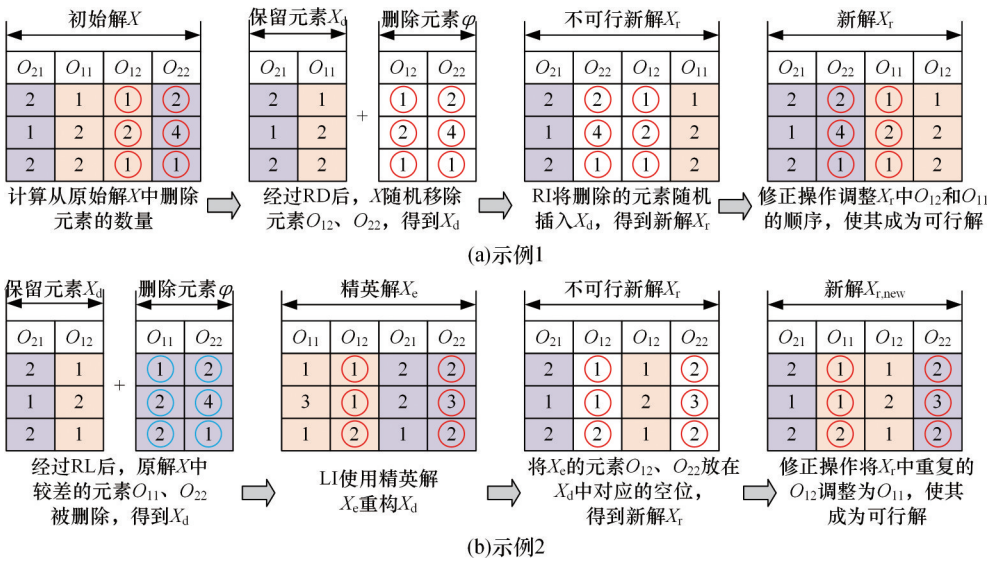



**图4 自适应竞争重构算法的两个示例**

**Fig.4 Two examples of adaptive competitive reconfiguration algorithm**

### 2.5 算子权重更新

重构算子与删除算子匹配，因此删除算子的权重就是删除算子下次被选择的概率：

$$w_{i,t+1}=(1-\lambda)w_{i,t}+\lambda S_i/N_{de,i} \tag{21}$$

式中：系数$\lambda$控制对$w_{i,t}$变化的灵敏度，$0\leqslant\lambda\leqslant1$；$w_{i,t}$为$t$代时第$i$个删除算子的权重，$i=1,2,3,4$；$S_i$为第$i$个删除算子的得分；$N_{de,i}$为第$i$个删除算子被选择的次数。

新解$X_r$支配原始解$X$时，$S_i=3$；$X_r$与$X$互不支配时，$S_i=2$；$X_r$被$X$支配时，$S_i=1$。这种更新方法使较差的解也有机会被保留，并在一定程度上增加种群的多样性。

### 2.6 算法流程

算法在重调度阶段的具体流程如下：

输入：$t$代种群$P_t$，种群大小$N$，最大迭代次数$T$。

输出：$t+1$代新种群$P_{t+1}$。

Step1 初始化删除算子权重$\boldsymbol{W}=(w_1, w_2, w_3, w_4)$和算子的使用次数$N_u=(0,0,0,0)$，设置$P_t$解的索引$i=1$。

Step2 根据$N_{d,t}=L(Q_1-(Q_1-Q_0)t/T)$计算删除元素的数量。

Step3 根据算子的权重$w_t$，使用轮盘赌方法选择删除算子$d_j$，其中，$j$为删除算子的索引，$j=1,2,3,4$。

Step4 使用$d_j$删除$X_i\in P_t$中较差的元素得到$X_d$，设置$N_{de,i}\leftarrow N_{de,i}+1$。

Step5 通过$d_j$对应的重构算子的重建$X_d$，得到$X_r$。

Step6 修正$X_r$得到可行的新解。

Step7 基于$X_r$和$X_i$的支配关系，利用$w_{i,t+1}=(1-\lambda)w_{i,t}+\lambda S_i/N_{de,i}$更新$d_j$的权重$w_j$。

Step8 $i\leftarrow i+1$，对下一个解优化，若$i>N$，执行Step9，否则返回Step2。

Step9 若算法达到终止条件$t>T$，结束并返

回新一代种群$P_{t+1}$，否则返回Step1，进行下一代优化，设置$t \leftarrow t+1$。

本文算法的计算复杂度。初始阶段的计算复杂度主要由NSGA-Ⅱ决定，NSGA-Ⅱ的计算复杂度是$O(TN^2)$。重调度阶段的复杂度主要来自Step1～9，Step1～9步骤与$X$的长度$L$线性相关。由于T代中每代处理$N$个解，因此重调度阶段的总复杂度为$O(TNL)$。

## 3　仿真结果及分析

为全面评估TACRA的竞争力，在MB测试集上比较TACRA与4种常用算法。所有算法都使用Python 3.8.0编程，测试在配备i5-10400 CPU和8GB RAM的计算机上进行。

### 3.1　测试函数

MB测试集模拟了AGV运输的车间发生机器故障的场景。每个测试算例均预先定义故障机器的索引$k_B$、机器故障的发生时间$B_S$和修复时间$B_T$。设计3种生产规模的车间，其中，小规模(MB1-5)车间生产5～15个工件，中等规模(MB6-10)车间生产20～35个工件，大规模(MB11-15)车间生产40～50个工件。更具体的参数设置如表2所示，其中，$n_o$为工件包含工序数量；$n_m$为每道工序的可选机器数量；$t_p$为每道工序的加工时间。

**表2　MB测试集的设置**

**Tab.2　Setting of MB test instances**

| 测试集 | $n$ | $m$ | $a$ | $n_o$ | $n_m$ | $t_p$ | $k_B$ | $B_S$ | $B_T$ |
|---|---|---|---|---|---|---|---|---|---|
| MB1 | 5 | 3 | 2 | 3-6 | 1-3 | 5-20 | 1 | 10 | 10 |
| MB2 | 5 | 4 | 3 | 3-6 | 1-3 | 5-20 | 2 | 12 | 10 |
| MB3 | 10 | 5 | 3 | 3-6 | 2-4 | 5-20 | 3 | 14 | 10 |
| MB4 | 15 | 6 | 4 | 3-6 | 2-4 | 5-20 | 4 | 16 | 10 |
| MB5 | 15 | 8 | 4 | 3-6 | 2-5 | 5-20 | 5 | 18 | 10 |
| MB6 | 20 | 10 | 5 | 3-6 | 3-6 | 5-20 | 7 | 20 | 10 |
| MB7 | 20 | 11 | 6 | 3-6 | 3-6 | 5-20 | 9 | 22 | 10 |
| MB8 | 25 | 12 | 6 | 3-6 | 3-7 | 5-20 | 10 | 24 | 10 |
| MB9 | 30 | 13 | 7 | 3-6 | 4-7 | 5-20 | 11 | 26 | 10 |
| MB10 | 35 | 15 | 7 | 3-6 | 4-8 | 5-20 | 12 | 28 | 10 |
| MB11 | 40 | 16 | 8 | 3-6 | 3-8 | 5-20 | 13 | 30 | 10 |
| MB12 | 40 | 17 | 8 | 3-6 | 3-7 | 5-20 | 14 | 32 | 10 |
| MB13 | 45 | 18 | 9 | 3-6 | 3-6 | 5-20 | 15 | 34 | 10 |
| MB14 | 45 | 19 | 9 | 3-6 | 3-6 | 5-20 | 16 | 38 | 10 |
| MB15 | 50 | 20 | 10 | 3-6 | 3-5 | 5-20 | 18 | 40 | 10 |

### 3.2　性能评价指标

为避免目标$f_1$和$f_2$的量纲与数量级的差异影响优化方向，先归一化处理目标值，再计算指标。此外，20次重复实验后，将不同算法获得的非支配解集合作为真实Pareto前沿。三个评价指标具体含义如下：

(1)反向世代距离(inverse generation distance，IGD)[12]是衡量解集收敛性和分布性的综合度量，IGD越小，算法得到解集的综合性能越好。

(2)超体积(hypervolume，HV)[13]衡量计算解集收敛性的算法性能，HV越大，算法得到解集的收敛性越好。

(3)适应度$f_c$直观表示算法的目标值结果，计算的公式为

$$f_c = 0.5(f_1' + f_2') \tag{22}$$

$$f_1' = \frac{f_1 - \min f_1}{\max f_1 - \min f_1} \tag{23}$$

$$f_2' = \frac{f_2 - \min f_2}{\max f_2 - \min f_2} \tag{24}$$

$f_c$越小，算法得到解集的性能越好。

### 3.3　算法参数设置

为评估TACRA的竞争力，选择双种群协作遗传算法DCGA[14]、采用精英保留策略和邻域搜索的方法EGANS[15]、基于博弈论的候鸟优化算法GT-MBO[5]、经典的非支配排序算法NSGA-Ⅱ[11]进行对比。对比算法根据DFJSPT的模型和目标函数调整相关操作。算法的基本参数设置如表3所示，对比算法的参数设置按照对应文献。

**表3　算法参数设置**

**Tab.3　Algorithms parameter setting**

| 参数 | 设置值 |
|---|---|
| 种群大小$N$ | 100 |
| 最大迭代次数$T$ | 100 |
| 全局选择概率$P_1$ | 0.3 |
| 局部选择概率$P_2$ | 0.7 |
| 最初删除率$Q_0$ | 0.1 |
| 最终删除率$Q_1$ | 0.5 |
| 删除算子的初始权重$w_0$ | 0.6 |
| 权重更新系数$\lambda$ | 0.3 |

### 3.4　实验结果及分析

表4～表6列出了5种算法的IGD、HV和$f_c$的均值与标准差。每种算法在MB实例重复20次实验，最佳值以粗体突出显示。分析IGD、HV和$f_c$的结果可以发现：

(1)IGD可评估算法的收敛性和多样性。如表4所示，TACRA在15个测试实例中获得11个最优值。由于删除、重构、修正操作相互弥补，TACRA在大多数实例都得到最佳性能。GT-MBO在MB1上的性能明显优于TACRA的原因是，GT-MBO采用混合驱动重调度策略，利用领导者引导进化，更适合小规模问题。EGANS在MB3、MB8上的性能更好，这是由于EGANS包

表4 五种算法的IGD

Tab.4 IGD of five algorithms

| 问题 | DCGA | | EGANS | | GT-MBO | | NSGA-II | | TACRA | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 平均值 | 标准差 | 平均值 | 标准差 | 平均值 | 标准差 | 平均值 | 标准差 | 平均值 | 标准差 |
| MB1 | 0.0892 | 0.0229 | 0.0752 | 0.0336 | **0.0354** | 0.0311 | 0.1041 | 0.0957 | 0.0680 | 0.0256 |
| MB2 | 0.0776 | 0.0410 | 0.0600 | 0.0892 | 0.0555 | 0.0529 | 0.1823 | 0.0825 | **0.0476** | 0.0176 |
| MB3 | 0.0754 | 0.0628 | **0.0489** | 0.0419 | 0.0618 | 0.0888 | 0.1318 | 0.0482 | 0.0574 | 0.0166 |
| MB4 | 0.0876 | 0.0673 | 0.0628 | 0.0978 | 0.0588 | 0.0671 | 0.1158 | 0.1040 | **0.0491** | 0.0252 |
| MB5 | 0.0913 | 0.0537 | 0.0511 | 0.0398 | 0.0571 | 0.0395 | 0.1073 | 0.0506 | **0.0489** | 0.0175 |
| MB6 | **0.0448** | 0.0203 | 0.0450 | 0.0129 | 0.0665 | 0.0418 | 0.0807 | 0.0475 | 0.0476 | 0.0136 |
| MB7 | 0.0455 | 0.0173 | 0.0619 | 0.0297 | 0.0600 | 0.0369 | 0.1225 | 0.0794 | **0.0429** | 0.0131 |
| MB8 | 0.0548 | 0.0281 | **0.0438** | 0.0071 | 0.0552 | 0.0442 | 0.1159 | 0.0633 | 0.0474 | 0.0106 |
| MB9 | 0.0758 | 0.0626 | 0.0680 | 0.0766 | 0.0637 | 0.0165 | 0.0938 | 0.0571 | **0.0552** | 0.0166 |
| MB10 | 0.0502 | 0.0288 | 0.0477 | 0.0122 | 0.0505 | 0.0086 | 0.0942 | 0.0467 | **0.0439** | 0.0141 |
| MB11 | 0.0565 | 0.0284 | 0.0531 | 0.0173 | 0.0513 | 0.0200 | 0.0919 | 0.0369 | **0.0510** | 0.0133 |
| MB12 | 0.0722 | 0.0586 | 0.0586 | 0.0253 | 0.0549 | 0.0225 | 0.0635 | 0.0092 | **0.0448** | 0.0172 |
| MB13 | 0.0866 | 0.0847 | 0.0814 | 0.0797 | 0.0691 | 0.0278 | 0.0772 | 0.0579 | **0.0554** | 0.0152 |
| MB14 | 0.0523 | 0.0274 | 0.0526 | 0.0164 | 0.0521 | 0.0145 | 0.0890 | 0.0144 | **0.0509** | 0.0109 |
| MB15 | 0.0600 | 0.0183 | 0.0604 | 0.0187 | 0.0663 | 0.0595 | 0.1037 | 0.0427 | **0.0529** | 0.0177 |

表5 五种算法的HV

Tab.5 HV of five algorithms

| 问题 | DCGA | | EGANS | | GT-MBO | | NSGA-II | | TACRA | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 平均值 | 标准差 | 平均值 | 标准差 | 平均值 | 标准差 | 平均值 | 标准差 | 平均值 | 标准差 |
| MB1 | 74.837 | 18.695 | 62.777 | 10.461 | 60.835 | 9.0499 | 61.044 | 12.115 | **85.095** | 2.9685 |
| MB2 | 71.529 | 6.6379 | 70.964 | 7.4243 | 63.500 | 4.8936 | 63.337 | 5.3362 | **83.823** | 3.5108 |
| MB3 | 68.849 | 6.8794 | 73.311 | 11.457 | 77.163 | 5.0284 | 62.593 | 6.5477 | **89.201** | 4.1179 |
| MB4 | 69.269 | 4.3718 | 63.151 | 5.8314 | 78.821 | 9.6351 | 64.984 | 3.0140 | **85.793** | 6.0228 |
| MB5 | 67.344 | 12.055 | 68.542 | 6.7448 | 73.873 | 13.809 | 64.831 | 6.0616 | **85.771** | 3.6485 |
| MB6 | 62.328 | 8.0136 | 63.959 | 3.9824 | 73.139 | 10.389 | 64.764 | 7.5026 | **86.526** | 5.9479 |
| MB7 | 59.846 | 11.224 | 75.001 | 8.9823 | 76.198 | 8.4659 | 66.543 | 7.2482 | **82.307** | 5.6637 |
| MB8 | 57.548 | 9.5850 | 59.419 | 8.6541 | 78.007 | 10.277 | 58.692 | 16.176 | **84.840** | 3.8781 |
| MB9 | 62.168 | 5.8404 | 66.568 | 10.331 | 77.602 | 10.924 | 54.571 | 4.7941 | **87.033** | 5.7585 |
| MB10 | 58.071 | 14.220 | 61.998 | 7.9076 | 76.484 | 4.5330 | 49.694 | 14.047 | **85.082** | 5.7691 |
| MB11 | 66.919 | 10.233 | 68.266 | 9.4580 | 75.417 | 6.1675 | 52.634 | 7.9778 | **88.563** | 4.3833 |
| MB12 | 57.512 | 6.6036 | 63.577 | 9.1754 | 75.419 | 7.1283 | 45.786 | 8.9290 | **87.865** | 3.6188 |
| MB13 | 55.772 | 7.9861 | 64.285 | 9.3998 | 74.980 | 14.833 | 54.082 | 15.933 | **84.980** | 6.9572 |
| MB14 | 58.949 | 10.321 | 77.199 | 6.0653 | 78.589 | 7.1954 | 52.140 | 7.5359 | **86.705** | 4.7324 |
| MB15 | 56.028 | 6.4603 | 65.509 | 8.6427 | 69.842 | 11.814 | 53.220 | 7.6134 | **85.829** | 7.5220 |

含精英保留策略，得到良好的多样性解。TACRA在MB11-15的整体性能优于其他算法。

(2)大规模车间调度和机器故障对算法提出挑战。如表6所示，TACRA对所有问题都具有最好的适应度。NSGA-Ⅱ和DCGA在$f_c$上的性能较差是由于它们缺少处理动态事件的策略。DCGA是一种群体协作的方法，更适合小规模、低复杂度的问题。大规模问题上，TACRA的$f_c$明显优于其他算法，说明它能有效解决DFJSPT。

结合IGD、HV和$f_c$可知，TACRA具有较好的收敛性，但是种群多样性还有提升的空间，须给出解决这一问题。

图5为8个测试实例的适应度演化曲线。GT-MBO优化较慢是因为它没有利用历史精英解提高优化效率。TACRA采用精英解和算子的历史性能指导进化方向，在大多数问题上收敛更快且适应度更小，能很好处理大规模实例MB13和MB15。

图6所示为6个MB测试实例的Pareto前沿，可以看出，TACRA获得的大多数解位于目标空间中目标值较小区域，优于其他算法。从Pareto解集中，工厂可根据自己的偏好选择满足需求的解，如偏好最短完工时间时，可选择$f_1$最小的解，而偏好最小负载不均衡度时，可选择$f_2$最小的解。

**表 6　五种算法的适应度**

**Tab.6　fitness of five algorithms**

| 问题 | DCGA | | EGANS | | GT-MBO | | NSGA-II | | TACRA | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 平均值 | 标准差 | 平均值 | 标准差 | 平均值 | 标准差 | 平均值 | 标准差 | 平均值 | 标准差 |
| MB1 | 0.1104 | 0.0913 | 0.1447 | 0.0295 | 0.0478 | 0.0485 | 0.1682 | 0.0402 | **0.0354** | 0.0078 |
| MB2 | 0.1157 | 0.0270 | 0.1000 | 0.0294 | 0.0522 | 0.0211 | 0.1191 | 0.0261 | **0.0388** | 0.0076 |
| MB3 | 0.1100 | 0.0222 | 0.1012 | 0.0575 | 0.0768 | 0.0248 | 0.1127 | 0.0394 | **0.0244** | 0.0131 |
| MB4 | 0.0917 | 0.0278 | 0.0819 | 0.0302 | 0.0833 | 0.0382 | 0.0845 | 0.0184 | **0.0221** | 0.0080 |
| MB5 | 0.0832 | 0.0584 | 0.1006 | 0.0420 | 0.0846 | 0.0555 | 0.0985 | 0.0292 | **0.0319** | 0.0096 |
| MB6 | 0.1076 | 0.0217 | 0.1007 | 0.0220 | 0.0871 | 0.0511 | 0.0932 | 0.0436 | **0.0219** | 0.0070 |
| MB7 | 0.1162 | 0.0584 | 0.0892 | 0.0370 | 0.0883 | 0.0351 | 0.0830 | 0.0393 | **0.0367** | 0.0152 |
| MB8 | 0.1191 | 0.0453 | 0.0727 | 0.0362 | 0.0782 | 0.0442 | 0.0990 | 0.0478 | **0.0284** | 0.0112 |
| MB9 | 0.1137 | 0.0333 | 0.0783 | 0.0439 | 0.0886 | 0.0525 | 0.0958 | 0.0249 | **0.0260** | 0.0137 |
| MB10 | 0.0912 | 0.0550 | 0.0787 | 0.0306 | 0.1080 | 0.0268 | 0.1019 | 0.0675 | **0.0331** | 0.0159 |
| MB11 | 0.1285 | 0.0478 | 0.0875 | 0.0458 | 0.0865 | 0.0295 | 0.1225 | 0.0237 | **0.0197** | 0.0091 |
| MB12 | 0.1236 | 0.0365 | 0.0893 | 0.0417 | 0.0965 | 0.0397 | 0.1243 | 0.0370 | **0.0466** | 0.0086 |
| MB13 | 0.1386 | 0.0661 | 0.0638 | 0.0290 | 0.1151 | 0.0778 | 0.1151 | 0.0927 | **0.0328** | 0.0202 |
| MB14 | 0.1031 | 0.0528 | 0.0729 | 0.0236 | 0.0710 | 0.0288 | 0.1006 | 0.0366 | **0.0255** | 0.0106 |
| MB15 | 0.1400 | 0.0323 | 0.0910 | 0.0478 | 0.0901 | 0.0455 | 0.1023 | 0.0294 | **0.0258** | 0.0148 |

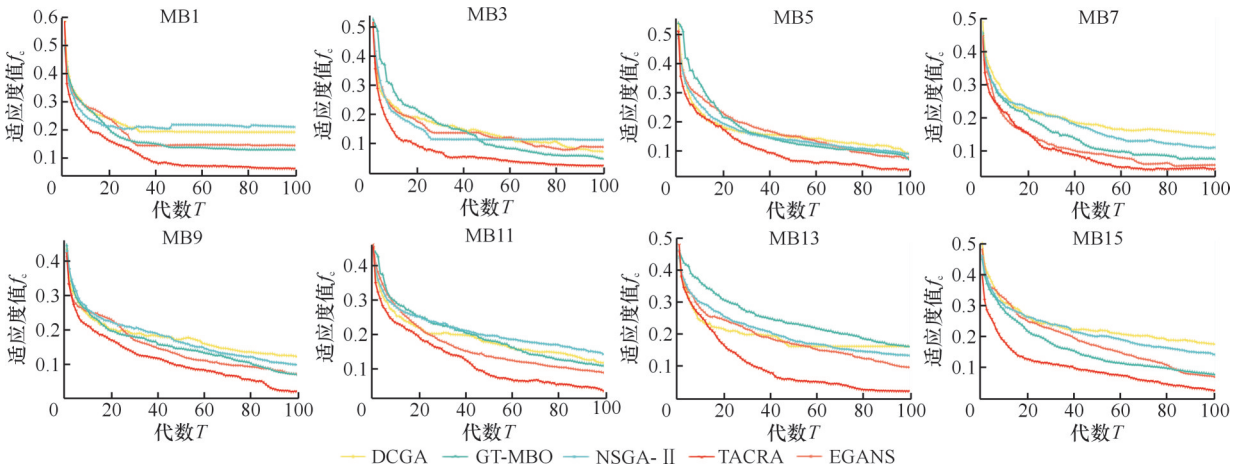



**图 5　进化过程中各算法适应度值变化曲线**

**Fig.5　Variation curves of fitness values of each algorithm during the evolution process**

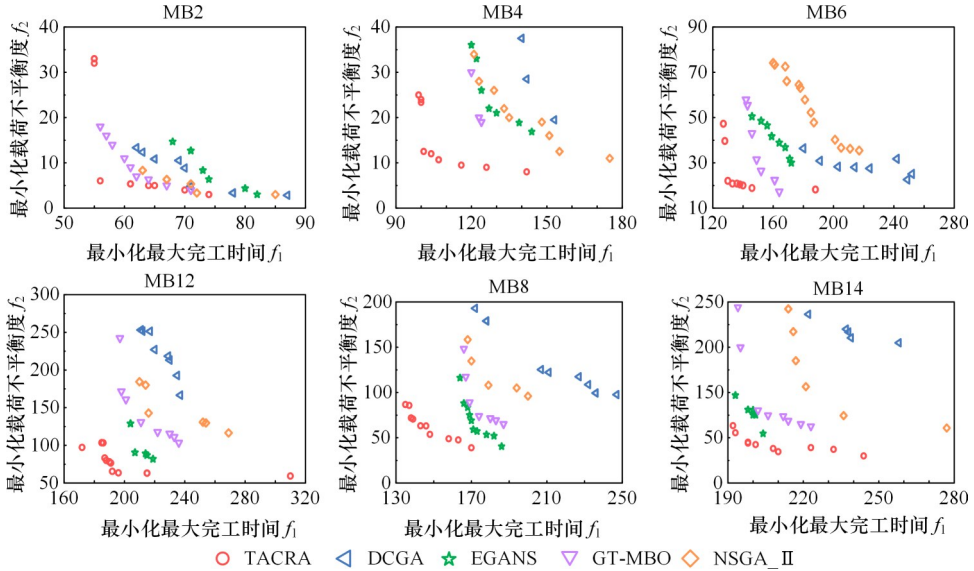



**图 6　算法在 MB 测试问题的 Pareto 前沿对比**

**Fig.6　Comparison results of the Pareto frontier in the MB test problem**

## 4 结论

针对具有运输资源约束的动态车间问题，提出一种两阶段自适应竞争重构算法。为增强算法的探索和开发能力，设计了包含多个删除算子和重构算子的算法，并提出自适应选择算子的机制。与对比算法相比，TACRA在大多数测试实例上表现出更好的优化能力和计算效率。未来将在DFJSPT的基础上考虑分布式车间运输资源约束的调度问题，并将TACRA用于实际的车间生产。

**参考文献：**


[1] 张国辉，蔡翌豪，李志霄，等. 考虑有限AGV运输资源的柔性作业车间调度研究[J]. 中国机械工程，2025，36(8)：1811-1823.
ZHANG Guohui，CAI Yihao，LI Zhixiao，et al. Research on Flexible Job Shop Scheduling Problems Considering Limited AGV Transportation Resources[J]. China Mechanical Engineering，2025，36(8)：1811-1823.

[2] 呼子宇，李紫晗，孙浩，等. 基于决策变量关系的动态多目标优化算法[J]. 控制与决策，2024，39(1)：78-86.
HU Ziyu，LI Zihan，SUN Hao，et al. A Dynamic Multi-objective Optimization Algorithm Based on the Relationship of Decision Variables[J]. Control and Decision，2024，39(1)：78-86.

[3] 徐新胜，吴松泽，杜文，等. 考虑低碳的柔性作业车间分批调度方法[EB/OL]. 中国机械工程，2025，1-16.（2025-03-25）[2026-06-12]. https://link.cnki.net/urlid/42.1294.TH.20250325.1330.015.
XUXinsheng，WUSongze，DUWen，et al. Consideration of Low-Carbon Flexible Job Shop Batch Scheduling Methods [EB/OL]. China Mechanical Engineering，2025，1-16.（2025-03-25）[2026-06-12]. https://link.cnki.net/urlid/42.1294.TH.20250325.1330.015.

[4] XU Gongjie，BAO Qiang，ZHANG Hongliang. Multi-objective Green Scheduling of Integrated Flexible Job Shop and Automated Guided Vehicles[J]. Engineering Applications of Artificial Intelligence，2023，126：106864.

[5] WEI Lixin，HE Jinxian，GUO Zeyin，et al. A Multi-objective Migrating Birds Optimization Algorithm Based on Game Theory for Dynamic Flexible Job Shop Scheduling Problem[J]. Expert Systems with Applications，2023，227：120268.

[6] LUO Cong，GONG Wenyin，LU Chao. Knowledge-driven Two-stage Memetic Algorithm for Energy-efficient Flexible Job Shop Scheduling with Machine Breakdowns[J]. Expert Systems with Applications，2024，235：121149.

[7] 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.

[8] TANG Hongtao，XIAO Yu，ZHANG Wei，et al. A DQL-NSGA-III Algorithm for Solving the Flexible Job Shop Dynamic Scheduling Problem[J]. Expert Systems with Applications，2024，237：121723.

[9] 方遒，宋豪杰，卢弘，等. 集成工人和AGV的多要素柔性作业车间调度方法[J]. 机械工程学报，2025，61(18)：330-343.
FANG Qiu，SONG Haojie，LU Hong，et al. Research on Multi-factors Flexible Job Shop Scheduling Problem with Workers and AGVs[J]. Journal of Mechanical Engineering，2025，61(18)：330-343.

[10] LIU Jiaojiao，SUN Baofeng，LI Gendao，et al. Multi-objective Adaptive Large Neighbourhood Search Algorithm for Dynamic Flexible Job Shop Schedule Problem with Transportation Resource[J]. Engineering Applications of Artificial Intelligence，2024，132：107917.

[11] 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.

[12] 张国辉，李志霄，张利平，等. 基于强化学习协同进化算法求解柔性作业车间节能调度问题[J]. 计算机应用研究，2025，42(7)：2039-2047.
ZHANG Guohui，LI Zhixiao，ZHANG Liping，et al. Reinforcement Learning Based Co-evolutionary Algorithm for Solving Flexible Job Shop Energy Efficient Scheduling Problem[J]. Application Research of Computers，2025，42(7)：2039-2047.

[13] 马训德，毕利，王俊杰. 基于群体免疫算法的绿色车间调度研究[J]. 系统仿真学报，2024，36(11)：2578-2591.
MA Xunde，BI Li，WANG Junjie. Research on Green Job Shop Scheduling Based on Herd Immunity Optimizer[J]. Journal of System Simulation，2024，36(11)：2578-2591.

[14] HAN Xiaoqing，CHENG Weiyao，MENG Leilei，et al. A Dual Population Collaborative Genetic Algorithm for Solving Flexible Job Shop Scheduling Problem with AGV[J]. Swarm and Evolutionary Computation，2024，86：101538.

[15] CHEN Nanlei，XIE Naiming，WANG Yuquan. An Elite Genetic Algorithm for Flexible Job Shop Scheduling Problem with Extracted Grey Processing Time[J]. Applied Soft Computing，2022，131：109783.


（编辑 张 洋）


**作者简介：王 聪**，女，1998年生，博士研究生。研究方向为柔性车间调度优化、智能优化算法设计。E-mail：congwang@stumail.ysu.edu.cn。**孙 浩***（通信作者），男，1985年生，副教授、博士研究生导师。研究方向为车间调度优化、智能优化算法、深度神经网络。发表论文20余篇。E-mail：sunhao@ysu.edu.cn。



**本文引用格式：**

王聪，魏立新，孙浩，等. 考虑运输资源限制的动态柔性车间调度问题及自适应竞争重构算法[J]. 中国机械工程，2026，37(7)：1708-1716.

WANG Cong，WEI Lixin，SUN Hao，et al. Dynamic Flexible Job Shop Scheduling Problem Considering Transportation Resource Constraints and Adaptive Competitive Reconfiguration Algorithm[J]. China Mechanical Engineering，2026，37(7)：1708-1716.