控制与决策
Control and Decision

基于多班教学优化的多目标分布式混合流水车间调度

雷德明， 苏斌

引用本文:
雷德明, 苏斌. 基于多班教学优化的多目标分布式混合流水车间调度[J]. 控制与决策, 2021, 36(2): 303−313.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2020.0549

基于多班教学优化的多目标分布式混合流水车间调度

雷德明†, 苏 斌

(武汉理工大学 自动化学院，武汉 430070)

摘 要: 单工厂环境下的混合流水车间调度问题已受到广泛关注,而多工厂环境下的分布式混合流水车间调度问题(distributed hybrid flow shop scheduling problem, DHFSP)研究进展则较小. 针对考虑顺序相关准备时间的DHFSP,提出一种多班教学优化(multi-class teaching-learning-based optimization, MTLBO)算法以同时最小化最大完成时间和最大延迟时间. 该算法采用双串编码方式,将种群划分成s个班级,每个班级的进化都由两个教师阶段和一个学生阶段组成;引入一种班级质量评价方式,实现奖惩机制和淘汰过程. 通过大量实验测试MTLBO的性能,计算结果表明,MTLBO对于所求解的DHFSP具有较强优势.

关键词: 混合流水车间调度；分布式调度；多班教学优化；多目标优化

中图分类号: TP18 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2020.0549 **开放科学(资源服务)标识码(OSID):**

引用格式: 雷德明,苏斌. 基于多班教学优化的多目标分布式混合流水车间调度[J]. 控制与决策, 2021, 36(2): 303-313.

Multi-class teaching-learning-based optimization for multi-objective distributed hybrid flow shop scheduling

LEI De-ming†, SU Bin

(School of Automation，Wuhan University of Technology，Wuhan 430070，China)

Abstract: Hybrid flow shop scheduling problems have been extensively addressed in single factory, however, distributed hybrid flow shop scheduling problems (DHFSP) are seldom investigated in multiple factories. In this study, the DHFSP with sequence-dependent setup times is studied and a multi-class teaching-learning-based optimization (MTLBO) algorithm is proposed to minimize makespan and maximum tardiness simultaneously. A two-string representation is adopted. The s classes are formed and the evolution of each class consists of two teacher phases and one learner phase. Class evaluation is introduced, a reward and punishment mechanism is executed on classes, and elimination process is also applied. A number of experiments are conducted and the computational results demonstrate that the MTLBO is very competitive for the DHFSP.

Keywords: hybrid flow shop scheduling；distributed scheduling；multi-class teaching-learning-based optimization；multi-objective optimization

0 引 言

作为一种经典车间调度问题,混合流水车间调度问题(hybrid flow shop scheduling problem, HFSP)已广泛应用于电子、造纸、纺织、化工、飞机引擎和半导体等制造业[1-2],在过去的10多年里受到了人们的广泛关注. 研究者采用精确算法、启发式方法和智能优化算法等解决考虑各种目标和工艺约束的HFSP[3-9].

随着经济全球化的不断深入,生产制造从传统的单工厂生产转变为多工厂生产,因此,多工厂环境下的分布式调度问题受到研究者的重视[10-27],其中分布式流水车间调度研究取得了较大进展[11-18],而分布式混合流水车间调度问题(distributed hybrid flow shop scheduling problem, DHFSP)的研究成果相对偏少[19-27]. Ying等[19]针对考虑多处理机任务的DHFSP,建立了一种混合整数线性规划模型,并提出了自适应迭代贪婪(IG)算法;Hao等[20]提出了一种混合头脑风暴优化算法以优化最大完成时间;Lei等[21]研究了考虑顺序相关准备时间(sequence-dependent setup

收稿日期: 2020-05-10；**修回日期:** 2020-07-27.
基金项目: 国家自然科学基金项目(61573264).
责任编委: 王凌.
†通讯作者. E-mail: deminglei11@163.com.

times, SDST)的分布式两阶段HFSP,并提出了一种考虑模因组分组的混合蛙跳算法以最小化最大完成时间和拖期工件数; Shao等[22]提出了一种多邻域IG算法以最小化最大完成时间; Zheng等[23]针对模糊DHFSP,提出了一种分布估计算法与IG相结合的协同进化算法以优化总延迟时间和鲁棒性.

在经济全球化背景下,研究DHFSP不仅能有效地完成所有工厂的工件加工,实现产能提升、库存成本下降和制造资源利用率提高等,而且能快速响应市场变化. 与单工厂环境下的HFSP相比,DHFSP的解决能够给企业带来更大的生产性能改善,而且DHFSP广泛存在于实际制造场合,例如多个晶圆制造工厂的晶圆生产. 另外,由于工厂分配子问题的引入,DHFSP比HFSP更复杂,求解难度更大. 因此,非常有必要研究DHFSP.

教学优化(TLBO)算法模拟实际班级中教师的教学过程和学生的学习过程,主要包括教师阶段和学生阶段. 与其他智能算法如遗传算法相比,TLBO具有结构简单、参数少、易理解和易实现等特点,已成功地应用于各种生产调度问题的求解[9,28-34],其在生产调度问题求解方面的优势和性能已得到验证,尤其在调度问题的多目标优化方面取得了一些进展. Lei等[9]运用改进的TLBO算法解决了一种考虑能耗的HFSP,以同时最小化总能耗和总延迟时间; Li等[28]针对实际的流水车间调度问题,提出了一种离散TLBO算法以最小化最大完成时间和不稳定性; Xie等[29]关于置换流水车间调度给出了一种TLBO算法和变邻域搜索算法的混合算法以同时最小化最大完成时间和最大延迟时间; 雷德明[33]针对低碳柔性作业车间调度问题,提出一种基于新型优化机理的TLBO算法以同时最小化总碳排放和平均延迟时间. 这些研究显示TLBO在多目标优化方面具有较强的搜索能力和优势,只是TLBO很少用来实现DHFSP的多目标优化. 由于TLBO区别于其他智能算法的特点及其在生产调度,尤其是在调度问题多目标优化方面的优势,为此,可以应用TLBO求解DHFSP以同时最优化多个目标.

现有的TLBO通常只考虑一个班[9,28-33],多班教学优化(MTLBO)算法的相关研究较少[34-36]. 实际上,多班策略有利于在不同的班级实现多样化搜索,从而有利于保持较高的种群多样性; 同时,现有多班策略[34-36]很少考虑班级质量评价和奖惩机制等以加强对好班的搜索并避免过多计算资源浪费在差班上. 为此,针对考虑SDST的DHFSP,本文提出一种基于新型优化机理的MTLBO以同时最小化最大完成时间和最大延迟时间. 该MTLBO采用双串编码方式,整个种群划分为s个班级,每个班级的进化由两个教师阶段和一个学生阶段组成,在班级质量评价基础上,提出一种奖惩机制,采用淘汰过程以改善差班质量. 通过大量实验测试MTLBO的性能,计算结果表明,MTLBO对于所求解的DHFSP具有较强的搜索优势.

1 多目标分布式混合流水车间调度问题

1.1 符号定义

问题描述和建模中用到的符号定义如下:

F: 工厂数量;

n: 工件数量;

m: 加工阶段数量;

f: 工厂编号, $f = 1, 2, \ldots, F$;

i, j: 工件编号, $i, j = 1, 2, \ldots, n$;

g: 加工阶段编号, $g = 1, 2, \ldots, m$;

S_g: 加工阶段g的并行机数量;

k: 加工阶段g中机器编号, $k = 1, 2, \ldots, S_g$;

$M_{f,g,k}$: 工厂f第g阶段的第k台并行机;

$p_{i,f,g,k}$: 工件i在机器$M_{f,g,k}$上的加工时间;

d_i: 工件i的交货期;

$u_{j,i,f,g,k}$: 工件j, i在机器$M_{f,g,k}$上依次加工时,工件i的准备时间;

$u_{0,i,f,g,k}$: 工件i为机器$M_{f,g,k}$首次加工的工件时,工件i的准备时间;

$B_{i,g}$: 工件i在第g个加工阶段的加工开始时间;

$C_{i,g}$: 工件i在第g个加工阶段的加工结束时间;

C_i: 工件i的完成时间;

H: 一个充分大的正数;

$C_{\max}$: 最大完成时间;

$T_{\max}$: 最大延迟时间.

决策变量如下:

$X_{i,f}$: 若工件i在工厂f中加工,则等于1; 否则,等于0.

$Y_{i,f,g,k}$: 若工件i在机器$M_{f,g,k}$上加工,则等于1; 否则,等于0.

$Z_{j,i,f,g,k}$: 若工件j先于工件i在机器$M_{f,g,k}$上加工,则等于1; 否则,等于0.

1.2 问题描述

考虑SDST的DHFSP描述如下: 存在F个地理位置各异的工厂和n个工件,工厂f包含一个混合流水车间,该车间包含m个阶段,第g个阶段由S_g台不相关并行机$M_{f,g,1}, M_{f,g,2}, \ldots, M_{f,g,S_g}$组成. 每个工

件可在任意一个工厂依次完成m道工序,每道工序可在相应阶段的任意一台机器上加工. 工件的准备时间依赖于工件的加工顺序,如果工件j、i在机器$M_{f,g,k}$上依次加工,则工件i的准备时间为$u_{j,i,f,g,k}$;如果工件i为机器$M_{f,g,k}$首次加工的工件,则工件i的准备时间为$u_{0,i,f,g,k}$. 问题的假设条件如下:

1) 所有工件相互独立并从零时刻起可以加工,且加工过程不能中断;

2) 所有机器在任意时刻都可用,不考虑机器损坏和维修消耗的时间;

3) 一台机器同一时刻只能加工一个工件;

4) 工件被分配至某一工厂后不能再去其他工厂加工,并需经过各个加工阶段,且在每个加工阶段都有一道工序;

5) 一个工件同一时刻只能在一台机器上加工;

6) 同一工件在两个阶段之间的加工不能重叠.

DHFSP包括3个子问题: 1) 工厂分配,将工件分配到合适的工厂; 2) 机器分配,为工厂内的工件分配一合适的机器; 3) 调度这3个子问题间具有强耦合关系. 工厂分配直接决定机器分配和调度的优化方向,当工厂分配变化时,需再一次调整机器分配和调度,如果未将工件分配到合适的工厂,则整个问题的解不可能是最优的.

DHFSP的目标是分配工件到合适的工厂、为工厂内每个工件选择合适的机器并确定各机器上工件的加工顺序以最小化目标$C_{\max}$和$T_{\max}$.

1.3 模型建立

考虑SDST的DHFSP数学模型如下:

$$\min(C_{\max}, T_{\max}). \tag{1}$$

s.t.

$$\sum_{f=1}^{F} X_{i,f} = 1,\ i = 1,2,\dots,n; \tag{2}$$

$$\sum_{k=1}^{S_g} Y_{i,f,g,k} = 1,\ i = 1,2,\dots,n,$$
$$f = 1,2,\dots,F,\ g = 1,2,\dots,m; \tag{3}$$

$$B_{i,g} \geqslant C_{i,g-1},\ i = 1,2,\dots,n,\ g = 2,3,\dots,m; \tag{4}$$

$$C_{i,g} \geqslant C_{j,g} + u_{j,i,f,g,k} + p_{i,f,g,k} - (1 - Z_{j,i,f,g,k}) \cdot H,$$
$$i,j = 1,2,\dots,n, \text{且} i \neq j,\ f = 1,2,\dots,F,$$
$$g = 1,2,\dots,m,\ k = 1,2,\dots,S_g; \tag{5}$$

$$C_i = C_{i,m},\ i = 1,2,\dots,n; \tag{6}$$

$$C_{\max} \geqslant C_i,\ i = 1,2,\dots,n; \tag{7}$$

$$T_{\max} \geqslant \max\{C_i - d_i, 0\},\ i = 1,2,\dots,n; \tag{8}$$

$$X_{i,f} \in \{0,1\},\ i = 1,2,\dots,n,\ f = 1,2,\dots,F; \tag{9}$$

$$Y_{i,f,g,k} \in \{0,1\},\ i = 1,2,\dots,n,\ f = 1,2,\dots,F,$$
$$g = 1,2,\dots,m,\ k = 1,2,\dots,S_g; \tag{10}$$

$$Z_{j,i,f,g,k} \in \{0,1\},\ i,j = 1,2,\dots,n, \text{且} i \neq j,$$
$$f = 1,2,\dots,F,\ g = 1,2,\dots,m,\ k = 1,2,\dots,S_g. \tag{11}$$

其中: 式(1)为目标函数; 式(2)确保每个工件的加工过程只能在一个工厂中完成; 式(3)确保任一阶段每个工件只能在一台机器上加工; 式(4)确保工件后一道工序的开始加工时刻不小于前一道工序加工结束时刻; 式(5)表示在机器$M_{f,g,k}$上若工件j先于工件i加工,则工件i在机器$M_{f,g,k}$上的完工时间应大于等于工件j在机器$M_{f,g,k}$上的完工时间、工件i的准备时间以及工件i在机器$M_{f,g,k}$上的加工时间之和,以确保每台机器同一时刻只能处理一个工件; 式(6)表示工件的完工时间等于此工件的最后一道工序的完工时间; 式(7)、(8)分别定义了最大完成时间和最大延迟时间的表达; 式(9)～(11)定义了决策变量的取值范围.

对于优化目标为$C_{\max}$、$T_{\max}$的DHFSP,如果$C_{\max}(x) \leqslant C_{\max}(y)$, $T_{\max}(x) \leqslant T_{\max}(y)$,并且$C_{\max}(x) < C_{\max}(y)$或者$T_{\max}(x) < T_{\max}(y)$,则$x \succ y$,即$x$支配$y$或者$y$受$x$支配. 对于集合$\Phi$和解$x \in \Phi$,如果$x$不受集合$\Phi$的其他任何解支配,则$x$关于集合$\Phi$是非劣的;如果不受问题解空间内的任何其他解支配,则该解为Pareto最优解.

2 基于MTLBO的DHFSP

TLBO[37]的搜索从种群即一个班开始,其主要步骤包括教师阶段和学生阶段. 在教师阶段,教师x_{teacher}作为算法所获得的最好解,将其知识传递给学生,解x根据下式产生新解x_{new}:

$$x_{\text{new}} = x + \alpha(x_{\text{teacher}} - (T \times x_{\text{mean}})). \tag{12}$$

其中: T表示教学因子,等于1或2; x_{mean}表示所有解的平均值; α表示区间$[0,1]$内均匀分布的随机数.

在学生阶段,学生x_i通过与其同学x_j的交互学习来增加他们的知识. 有

$$x_{\text{new},i} = \begin{cases} x_i + \alpha(x_i - x_j),\ f(x_i) < f(x_j); \\ x_i + \alpha(x_j - x_i),\ \text{otherwise}. \end{cases} \tag{13}$$

以上两阶段中,如果$x_{\text{new}}(x_{\text{new},i})$优于$x(x_i)$,则$x_{\text{new}}(x_{\text{new},i})$将替代$x(x_i)$.

应用TLBO解决调度问题时,利用式(12)和(13)

产生新解的可行性无法得到保证,现有研究[9,28]往往应用离散TLBO求解调度问题. 本文也运用离散TLBO求解DHFSP.

通常TLBO只有一个班级. 目前,MTLBO研究有所进展[34-36]. 不过,这些研究未考虑班级质量评价和基于评价结果的奖惩机制. 根据指定的评价指标对班级质量进行计算并排序,通过奖惩机制区别对待各班级,以加强对好班的搜索并避免过多计算资源浪费在差班上.

2.1 初始化和多班构造

现有文献经常采用两种方式对HFSP进行编码:随机键编码方式[3-5]和基于工件的编码[6-8]. 其中:随机键编码采用实数串描述第1阶段机器分配及各机器上的工件加工顺序,而后续阶段的工件加工顺序及机器分配均采用启发式规则;基于工件的编码则应用工件排列确定工件的加工顺序,而各阶段的机器分配由启发式规则决定. 现有DHFSP文献中,往往对工厂分配和调度两个子问题进行编码[20-27],机器分配也采用启发式规则. 为此,本文也采用双串编码方式描述工厂分配和调度的解并运用启发式规则确定机器分配.

对于具有n个工件和F个工厂的DHFSP,应用工厂分配串$[h_1,h_2,\dots,h_n]$和调度串$[\pi_1,\pi_2,\dots,\pi_n]$描述它的解,其中$\pi_i\in[1,2,\dots,n]$,$h_i\in\{1,2,\dots,F\}$表示工件$J_i$分配给工厂$h_i$.

当根据工厂分配串确定每个工厂所分配的工件后,每个工厂内的调度问题就是HFSP,其调度串为DHFSP的调度串的子串,是该工厂内所有工件的排列,机器分配根据启发式规则确定. 解码过程的详细步骤如下:

1) $g=1$.

2) 从调度串的第1个工件π_1开始依次安排工件的加工,对于每个工件π_i,根据工厂分配串确定其工厂f,然后确定该工件在工厂f阶段g的各台机器上的完成时间,选择完成时间最小的机器为π_i的加工机器. 如果存在多台机器拥有相同的最小完工时间,则选择编号最小的机器.

3) $g=g+1$.

4) 重复步骤2)$\sim$步骤3),直至所有阶段加工完成.

随机生成N个初始解以构成初始化种群P,将种群P划分成s个班,每个班都由一名教师和θ名学生组成. 详细步骤如下:首先对种群P中所有解非支配排序[38],得到解的rank值;假设有β个解的rank为1,如果$\beta>s$,则从这β个解中随机选择s个解作为教师,否则确定rank值最小的s个解作为教师;最后,随机选择一名教师x^i_{teacher}和θ名学生形成班c_i,其中$N=s\times(1+\theta)$.

2.2 班级进化

班级进化过程由EvolutionClass(c_i,w)描述,它包括两个教师阶段和一个学生阶段.

EvolutionClass(c_i,w)的具体过程如下:

1) 教师自学习阶段. 对于每名教师x^i_{teacher},执行该教师与随机选择的教师$x^j_{\text{teacher}}(j\neq i)$间的全局搜索,再对它执行多邻域搜索$\gamma_w$次.

2) 教师教学阶段. 对于每个学生$x\in c_i$,按相同概率选择x^i_{teacher}或者教师$x^e_{\text{teacher}}(e\neq i)$,执行学生与选中的教师之间的全局搜索.

3) 学生阶段. 如果$w=1$,则确定班c_i中最好$\theta\times q_w$名学生的集合Δ,对于每个学生$x\in\Delta$,随机选择学生$y\in\Delta$, $y\neq x$,执行x与y之间的全局搜索.

如果$w=2$,则确定班c_i中最好的$\theta\times q_w$名学生的集合Θ_1. 对于每个学生$x\in\Theta_1$,以相同概率执行以下两操作:x与随机选择的学生$y\in\Theta_1$之间的全局搜索或者x的多邻域搜索2次.

如果$w=3$,则构造c_i中最好的$\theta\times q_w$名学生的集合Θ_2. 对于每个学生$x\in\Theta_2$,以相同概率从集合Θ_2或者Θ_1中随机选一个解y,执行x与y间的全局搜索.

如果$w=4$,则确定c_i中最好的$\theta\times q_w$名学生. 对每名学生x,执行x与随机选择的$y\in\Theta_1$间的全局搜索.

4) 根据解的质量更新教师.

上述过程中:γ_w是一个整数,q_w是一个实数,即

$$\gamma_w=\begin{cases}6,\ w=2;\\ 4,\ \text{otherwise.}\end{cases}\tag{14}$$

$$q_w=\begin{cases}0.6,\ w=2;\\ 0.5,\ w=1,3;\\ 0.4,\ w=4.\end{cases}\tag{15}$$

解x与y之间的全局搜索描述如下:以相同的概率选择工厂分配串或调度串,如果工厂分配串被选中,则对x和y的工厂分配串执行两点交叉操作;否则,对x和y的调度串执行部分映射交叉(PMX)[39]. 当执行两点交叉或者PMX时,两个随机选到的交叉位置k_1、k_2需满足$|k_1-k_2|\leqslant n/2$条件,同时只产生一个子代z. 替换条件如下:如果$z\succ x$,则z替换x并更新外部档案Ω;否则,如果z与x互不支配,

则从x和z中随机选择一个解来替换x,并用z更新外部档案Ω.

外部档案Ω存储MTLBO产生的非劣解并采用如下方式更新:将z加入到Ω中,对Ω内的所有元素根据Pareto支配关系进行比较,剔除受支配解.

本文应用了8个邻域结构,其中4个邻域结构与最大完工时间相关,另外4个邻域结构与最大延迟时间相关.

$\mathcal{N}_1$描述如下:先确定完成时间最大的工厂l,并从中随机选择一个工件$J_i \in A_l$,再随机选择一个工厂$r \neq l$并随机选择工件$J_j \in A_r$,互换工件J_i和J_j,其中A_l表示所有分配到工厂l中的工件集. $\mathcal{N}_2$与$\mathcal{N}_1$相类似,将工厂l中随机选择的工件J_i转移至工厂r中. $\mathcal{N}_1$和$\mathcal{N}_2$能够产生新的工厂分配串.

$\mathcal{N}_3$通过确定完成时间最大的工厂l并在调度串中互换该工厂内随机选择的两个工件来产生新解. $\mathcal{N}_4$的过程如下:确定完成时间最大的工厂l,随机选择两个工件,假设两个工件在调度串中的位置分别为k_1、k_2, $k_1 < k_2$,将位置k_2处的工件插入至位置k_1.

$\mathcal{N}_5$与$\mathcal{N}_1$相类似,只是需要确定具有最大延迟时间的工厂l和延迟时间最大的工件$J_i \in A_l$. $\mathcal{N}_6$与$\mathcal{N}_2$具有相同的步骤,只是需根据最大延迟时间确定工厂l和工件J_i.

邻域结构$\mathcal{N}_7$描述如下:先确定具有最大延迟时间的工厂l和延迟时间最大的工件$J_i \in A_l$,假设J_i在工厂l的工件排列中的位置为k,如果$k > 1$,则随机选择一个工件J_j,该工件在工厂l的工件排列中位置小于k,并互换J_j与J_i;否则执行邻域结构$\mathcal{N}_5$. J_i与J_j的交换操作是在调度串$[\pi_1, \pi_2, \ldots, \pi_n]$中进行的. $\mathcal{N}_8$与$\mathcal{N}_7$的区别之处在于执行插入操作,即将J_i插入至J_j的位置.

多邻域搜索过程描述如下:依次执行$\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_8$,对于每个$\mathcal{N}_t$,产生一个新解$z \in \mathcal{N}_t(x)$,如果$z$满足替换条件,则用$z$替换$x$并更新$\Omega$,其中$\mathcal{N}_t(x)$表示使用邻域结构$\mathcal{N}_t$产生的$x$邻域解集.

首先在班c_i中执行非支配排序和拥挤距离计算[38],解$x \in c_i$的质量η_i^x定义如下:

$$\eta_i^x = \mathrm{rank}_i^x - \frac{\mathrm{dist}_i^x}{\varepsilon + \sum\limits_{y \in H_{\mathrm{rank}_i^x}} \mathrm{dist}_i^y}. \tag{16}$$

其中:rank_i^x和dist_i^x分别表示rank值和拥挤距离,$H_{\mathrm{rank}_i^x}$表示与x具有相同rank_i^x的所有解构成的集合,ε是一个非常小的正数.

显然η_i^x越小,解的质量越好. η_i^x最小的解x为班c_i中的新教师.

2.3　算法描述

MTLBO的具体过程描述如下:

1) 随机生成N个解,初始化种群P,将种群P分成s个班级.

2) 依次执行EvolutionClass$(c_i, 1)\mu$次,$i = 1, 2, \ldots, s$.

3) 首先评价各班质量并根据班级质量λ_i升序排列,其中c_1为好班,中班有$c_2, \ldots, c_{s-1}$,差班为c_s;然后执行EvolutionClass$(c_1, 2)\mu + \delta$次. 对于中班$c_i, i = 2, \ldots, s-1$,依次运行EvolutionClass$(c_i, 3)$ μ次. 执行EvolutionClass$(c_s, 4)$ $\mu - \delta$次.

4) 运行淘汰过程Elimination(c_s).

5) 若满足终止条件,则停止搜索;否则转到3).

上述过程中:μ为整数,λ_i表示班c_i的质量,终止条件为目标向量估计次数max_it.

班级评价过程如下:首先对种群P执行非支配排序和拥挤距离计算[38];然后根据式(16)计算η_i^x,注意此时获取rank^x、dist^x和H_{rank^x}是针对整个种群排序和计算得到的;最后,计算c_i的质量λ_i,即

$$\lambda_i = \sum_{x \in c_i} \eta_i^x. \tag{17}$$

奖惩机制如下:好班迭代$\mu + \delta$次,普通班迭代μ次,差班迭代$\mu - \delta$次,其中δ表示奖励值或者惩罚值. 与普通班相比,好班增加δ次搜索作为奖励,差班减少δ次搜索作为惩罚.

淘汰过程Elimination(c_s)用来改善差班c_s的质量,其详细步骤如下:

1) 确定最差的解$x \in c_s$,随机选一个非劣解$y \in \Omega$,将集合$\mathcal{M}$置空.

2) 重复执行以下过程L次. 对y执行多邻域搜索,得到新解$z \in \mathcal{N}_t(y), t = 1, 2, \ldots, 8$. 如果$z \succ y$或者$z$与$y$间互不支配,则用$y$更新$\mathcal{M}$, z替换y并更新外部档案Ω;否则,用z更新$\mathcal{M}$.

3) 如果y未得到改进,则从集合$\mathcal{M}$中随机选一个解替换x;否则,用y替换x.

上述步骤中:L是整数,根据实验设置成5;$\mathcal{M}$与Ω具有相同的更新方式.

MTLBO具有以下特征:1) 构造多个班级,每个班级的进化由两个教师阶段和一个学生阶段组成,并提出4种方法实现学生阶段;2) 引入一种班级质量评价方式,对好班和差班应用奖惩机制并对差班增加了淘汰过程. 学生阶段的多样化实现、奖惩机制与淘汰过程的应用,使得MTLBO不易陷入局部最优,充分利用了好解并避免了计算资源浪费在差解上,这样有助于

MTLBO取得高质量的计算结果.

3 计算实验

为了测试MTLBO在解决考虑SDST的DHFSP方面的性能,进行了大量的计算实验. 这些实验运用Microsoft Visual C++ 2015 编程实现,并运行于具有8.0 G RAM 2.40 GHz CPU的计算机上.

3.1 测试实例、对比算法以及评价指标

随机生成66个测试实例进行计算实验,表1描述了这些实例的信息. 在这些实例中,加工时间$p_{i,l,g,k} \in [10, 80]$,准备时间$u_{0,i,l,g,k}$、$u_{j,i,l,g,k} \in [5, 10]$,交货期$d_i \in [d_{\min}, d_{\max}]$. 其中: $d_{\max} = (80+10) \times m \times n/(2.5 \times F \times \max\limits_{g,l}\{S_{g,l}\})$, $d_{\min} = (10+5) \times m$.

表1 实例的相关信息

实 例	n
1, 9, 17, 25, 33, 41	20
2, 10, 18, 26, 34, 42	30
3, 11, 19, 27, 35, 43, 49, 55, 61	40
4, 12, 20, 28, 36, 44, 50, 56, 62	50
5, 13, 21, 29, 37, 45, 51, 57, 63	60
6, 14, 22, 30, 38, 46, 52, 58, 64	80
7, 15, 23, 31, 39, 47, 53, 59, 65	100
8, 16, 24, 32, 40, 48, 54, 60, 66	120
实 例	S_g
1 ~ 8, 25 ~ 32, 49 ~ 54	3, 3
9 ~ 16, 33 ~ 41, 55 ~ 60	3, 3, 4, 2
17 ~ 24, 42 ~ 48, 61 ~ 66	3, 3, 4, 2, 4, 3, 3, 2
实 例	F
1 ~ 24	2
25 ~ 48	3
49 ~ 66	4

多目标优化算法结果评价指标较多,本文采用DI_R[40]和$\mathcal{C}$[41]两个指标评价算法的性能[21].

本文选用CCA[7]、CMA [13]和自适应大范围邻域搜索(ALNS)[18]作为对比算法.

CCA用来解决HFSP以最小化最大完成时间和总延迟时间. 添加工厂分配串和一些相关操作后,CCA能求解DHFSP,具体描述如下. 使用4个基本操作: 1) 等概率选取调度串或工厂分配串以执行交换操作; 2) 等概率选取调度串或工厂分配串以执行插入操作; 3) 与CCA原有策略一致; 4) 殖民地与殖民国家间工厂分配串的两点交叉.

CMA采用两个子群优化两个不同的目标,使用了基于多个搜索操作间的竞争策略和基于知识的局部搜索操作,并让两个子群每隔一定代数进行交互以改善两个目标间的平衡性. CMA原本用来求解多目标分布式置换流水车间调度问题,采用MTLBO的编码策略后,该算法可以求解DHFSP.

ALNS针对三目标分布式可重入置换流水车间调度问题而提出. 它包含了许多破坏和修复操作,当应用ALNS处理DHFSP时,只需删除两个与成本相关的破坏操作,并将文献[18]第4.5.1小节总成本最小化调整为最大延迟时间最小化即可.

CCA在HFSP求解方面具有较强的搜索优势,而CMA和ALNS在分布式流水车间调度求解方面性能优良. 如上所述,3种算法仅需较小的修改就能直接求解DHFSP,实现问题的多目标优化. 由于缺乏直接求解本文DHFSP的对比算法,而CCA、CMA和ALNS经过小的修改就能与MTLBO进行对比,本文选择它们作为对比算法.

当MTLBO去掉奖励、惩罚和淘汰过程策略,每个班级的进化过程与EvolutionClass(c_i,1)相同之后,即可得到MTLBO1. 对比MTLBO和MTLBO1用来检验去掉奖惩机制和淘汰过程等的效果.

3.2 参数设置

MTLBO具有5个主要参数: 目标向量估计次数max_it、种群规模N、班级个数s、迭代次数μ和奖惩次数δ. 对这5个参数的2种设置,即max_it的0.8×10^5、10^5, N的30、40, s的3、4, μ的6、8以及δ的3、4共32种组合,应用Taguchi方法[13]对实例37进行参数设置. 最后,确定使MTLBO性能更好的一组参数为max_it $= 10^5$, $N = 30$, $s = 3$, $\mu = 6$, $\delta = 3$.

MTLBO1未使用参数μ和δ,其他参数的设置与MTLBO相同.

除了终止条件外,CCA、CMA和ALNS的所有参数均直接来自文献[7,13,18]. 3个对比算法的终止条件都与MTLBO相同.

3.3 结果与分析

每种算法关于每个实例独立运行20次,5种算法的计算结果如表2和表3所示. 其中: Ω^*由集合$\bigcup\limits_{l=1}^{5}\Omega_l$中的非劣解组成,$\Omega_1$和$\Omega_2$分别表示MTLBO和MTLBO1的外部档案,$\Omega_{3(4)(5)}$表示CMA (CCA) (ALNS)的外部档案. 令CM表示CMA, CC表示CCA, A表示ALNS, M和M1分别表示MTLBO和MTLBO1. 表4给出了MTLBO和对比算法的计算时间. 图1给出了5种算法的非劣解分布图.

根据工件数不同,测试实例包括小规模实例6个、中规模实例33个和大规模实例27个. 如表2和表3所示: MTLBO关于所有小规模实例所得的DI_R优于MTLBO1,关于3个小规模实例$\mathcal{C}$(M1, M)小于

$\mathcal{C}$(M, M1); 关于至少25个中规模实例, MTLBO获得了比MTLBO1更小的DI_R和比$\mathcal{C}$(M, M1)更小的$\mathcal{C}$(M1, M), 其中关于21个实例, $\mathcal{C}$(M, M1)等于1, 这表明MTLBO1的所有解都受MTLBO的非劣解支配; 大规模实例中, 关于26个实例, MTLBO的DI_R小于MTLBO1的相应值, 且关于27个实例, $\mathcal{C}$(M ,M1)优于$\mathcal{C}$(M1, M), 其中23个实例对应的$\mathcal{C}$(M, M1)等于1. 由图1也可看出MTLBO与MTLBO1之间显著的性能差异, 这表明, 奖惩机制和淘汰过程等策略的引入大大改善了MTLBO的性能.

可以发现, 在相似的计算时间内, MTLBO关于大部分实例所得的结果均优于3个对比算法. 关于小规模实例, MTLBO取得了优于CCA、 CMA和ALNS的计算结果. 对于中规模实例, MTLBO关于9个实例的DI_R至少比CMA的相应值小8, 即CMA的解远离MTLBO的非劣解, 且关于其中16个实例, $\mathcal{C}$(M, CM)等于1; MTLBO关于27个实例获得了比CCA更小的DI_R, 关于31个实例所得的$\mathcal{C}$(M, CC)大于$\mathcal{C}$(CC, M); MTLBO关于大多数中规模实例取得了比ALNS更优的计算结果.

表2　5种算法关于指标DI_R的计算结果

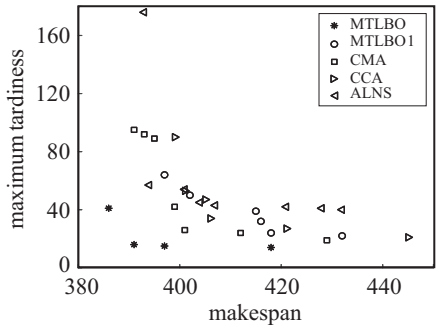
实例	M	M1	CM	CC	A	实例	M	M1	CM	CC	A
1	4.545	13.636	18.182	9.091	0.000	34	2.620	3.354	7.231	9.629	7.838
2	5.592	8.287	10.439	1.773	8.562	35	10.100	5.413	2.448	5.573	5.116
3	0.000	0.000	3.390	11.864	11.887	36	0.000	9.608	10.280	12.764	8.586
4	9.776	19.098	5.493	15.604	10.570	37	1.183	0.000	9.258	16.459	6.029
5	11.237	11.032	5.271	11.924	8.360	38	8.110	10.871	1.999	16.773	8.126
6	0.000	7.387	5.556	10.497	5.606	39	15.175	19.260	8.249	19.004	26.731
7	0.000	10.804	11.437	18.841	14.055	40	5.706	8.610	0.897	14.851	5.094
8	0.962	8.163	4.162	10.327	8.119	41	4.284	5.819	3.982	3.454	8.629
9	16.086	19.753	5.357	10.407	27.399	42	16.596	23.348	12.577	4.636	17.869
10	1.739	2.609	4.348	0.000	9.529	43	0.000	9.759	8.680	10.854	3.892
11	1.556	6.156	5.628	5.195	3.941	44	1.370	4.166	1.370	5.500	3.435
12	8.984	7.697	4.314	10.792	9.233	45	5.402	8.887	3.142	11.163	8.049
13	2.216	1.601	6.520	14.481	9.796	46	1.176	5.294	0.000	15.882	6.471
14	0.693	10.000	10.621	9.516	7.489	47	0.000	26.095	17.889	20.567	11.721
15	0.000	18.721	4.506	21.239	16.524	48	11.111	11.556	0.000	14.835	16.889
16	2.523	6.604	9.694	15.940	2.523	49	6.841	13.417	7.459	11.833	4.950
17	1.901	7.624	4.326	4.170	6.392	50	11.462	12.285	0.875	21.087	10.609
18	2.479	14.817	4.569	14.027	4.097	51	0.000	23.214	7.143	21.429	19.643
19	0.000	13.433	20.896	5.970	22.388	52	0.000	7.609	9.572	10.870	7.609
20	1.480	5.544	3.710	6.642	5.998	53	0.000	17.267	5.679	11.882	6.992
21	2.803	2.802	0.000	16.400	2.810	54	4.400	8.125	3.500	17.082	9.316
22	0.000	4.262	2.690	7.196	3.320	55	1.302	9.016	4.324	10.256	5.966
23	5.327	1.754	0.703	10.105	6.886	56	0.000	14.148	10.755	13.755	16.072
24	0.000	10.213	4.442	31.490	11.342	57	1.289	4.147	13.328	16.302	11.364
25	0.000	15.092	15.941	15.941	10.574	58	15.597	20.652	5.466	19.946	13.287
26	1.032	5.466	4.164	2.712	4.855	59	0.000	13.309	6.707	13.450	9.071
27	0.000	12.629	12.778	21.849	15.598	60	0.000	12.166	9.293	20.108	8.480
28	4.966	4.906	4.038	5.915	3.586	61	10.571	18.477	1.607	6.297	9.613
29	1.149	2.299	10.075	0.000	5.747	62	0.000	23.161	22.251	27.556	31.402
30	0.000	6.458	11.690	11.390	8.310	63	0.000	21.111	4.444	20.000	3.333
31	0.000	10.495	7.823	10.581	10.782	64	9.134	10.397	1.751	8.365	13.800
32	1.008	3.104	4.739	9.827	4.798	65	0.000	15.979	5.755	14.733	12.161
33	3.343	16.693	12.281	16.173	3.880	66	0.000	17.222	11.002	6.611	11.761

表3 5种算法关于指标$\mathcal{C}$的计算结果

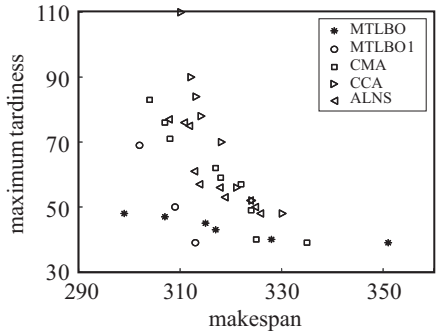
实例	$\mathcal{C}_1$(M, M1)	$\mathcal{C}_1$(M1, M)	$\mathcal{C}_1$(M, CM)	$\mathcal{C}_1$(CM, M)	$\mathcal{C}_1$(M, CC)	$\mathcal{C}_1$(CC, M)	$\mathcal{C}_1$(M, A)	$\mathcal{C}_1$(A, M)
1	1.000	0.000	1.000	0.000	1.000	0.000	0.000	1.000
2	0.800	0.000	0.600	0.000	0.400	0.250	0.500	0.500
3	0.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
4	1.000	0.000	0.857	0.000	0.833	0.000	1.000	0.000
5	1.000	0.000	0.800	0.000	0.857	0.000	0.200	0.333
6	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
7	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
8	1.000	0.000	0.857	0.000	1.000	0.000	1.000	0.000
9	0.000	1.000	0.000	0.500	0.000	1.000	0.000	0.000
10	1.000	0.000	1.000	0.000	0.000	1.000	1.000	0.000
11	0.800	0.000	1.000	0.000	1.000	0.000	0.667	0.000
12	0.500	0.000	0.875	0.000	1.000	0.000	1.000	0.000
13	0.750	0.500	0.727	0.000	1.000	0.000	1.000	0.000
14	0.333	0.286	1.000	0.000	0.000	0.429	0.833	0.000
15	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
16	1.000	0.000	1.000	0.000	1.000	0.000	0.750	0.667
17	0.667	0.250	0.750	0.250	1.000	0.000	1.000	0.000
18	0.500	0.000	0.800	0.000	1.000	0.000	0.333	0.400
19	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
20	0.000	0.000	0.000	0.667	1.000	0.000	1.000	0.000
21	0.500	0.500	0.000	1.000	1.000	0.000	1.000	0.000
22	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
23	0.500	0.000	0.333	0.000	1.000	0.000	1.000	0.000
24	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
25	0.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
26	1.000	0.000	1.000	0.000	0.750	0.000	1.000	0.000
27	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
28	1.000	0.000	1.000	0.000	0.833	0.000	0.667	0.000
29	1.000	0.000	1.000	0.000	0.000	1.000	1.000	0.000
30	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
31	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
32	0.800	0.000	1.000	0.000	1.000	0.000	0.800	0.000
33	0.500	0.750	0.571	0.500	0.000	0.750	0.600	0.000
34	0.400	0.250	0.625	0.250	0.875	0.125	0.625	0.250
35	0.667	0.000	0.500	0.000	1.000	0.000	0.250	0.000
36	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
37	0.000	1.000	1.000	0.000	1.000	0.000	1.000	0.000
38	0.667	0.200	0.375	0.000	1.000	0.000	0.778	0.000
39	1.000	0.000	0.833	0.000	1.000	0.000	1.000	0.000
40	1.000	0.000	0.600	0.000	1.000	0.000	0.857	0.000
41	1.000	0.000	1.000	0.000	0.667	0.000	0.500	0.500
42	1.000	0.000	1.000	0.000	0.750	0.000	1.000	0.000
43	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
44	1.000	0.000	0.500	0.500	1.000	0.000	0.500	0.000
45	1.000	0.000	0.800	0.000	1.000	0.000	1.000	0.000
46	1.000	0.000	0.000	1.000	1.000	0.000	1.000	0.000
47	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
48	1.000	0.000	0.000	1.000	1.000	0.000	1.000	0.000
49	1.000	0.000	0.833	0.000	1.000	0.000	0.750	0.000
50	1.000	0.000	0.333	0.500	1.000	0.000	0.500	0.000
51	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
52	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
53	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
54	1.000	0.000	0.571	0.400	1.000	0.000	1.000	0.000
55	1.000	0.000	0.833	0.000	1.000	0.000	0.818	0.000
56	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
57	0.667	0.667	0.778	0.333	1.000	0.000	1.000	0.000
58	1.000	0.000	0.909	0.000	1.000	0.000	1.000	0.000
59	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
60	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
61	1.000	0.000	0.600	0.000	0.500	0.000	1.000	0.000
62	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
63	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
64	1.000	0.000	0.714	0.000	0.000	1.000	1.000	0.000
65	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
66	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000

表4　MTLBO与对比算法的计算时间　　s

实例	MTLBO	CMA	CCA	ALNS	实例	MTLBO	CMA	CCA	ALNS	实例	MTLBO	CMA	CCA	ALNS
1	1.053	1.052	1.104	0.847	23	7.379	7.430	7.770	8.179	45	4.522	4.697	4.596	4.938
2	1.241	1.192	1.240	1.173	24	8.793	8.914	9.041	9.555	46	5.894	5.983	5.996	6.288
3	1.365	1.360	1.389	1.496	25	1.016	1.061	1.090	0.782	47	7.394	7.681	7.461	7.823
4	1.574	1.598	1.588	1.892	26	1.206	1.232	1.253	1.101	48	8.824	9.188	8.822	9.375
5	1.740	1.789	1.742	2.309	27	1.357	1.356	1.451	1.430	49	1.364	1.375	1.389	1.378
6	2.149	2.157	2.166	2.769	28	1.542	1.485	1.604	1.835	50	1.501	1.534	1.548	1.784
7	2.573	2.530	2.530	3.123	29	1.688	1.652	1.756	2.179	51	1.679	1.705	1.720	2.197
8	3.029	2.915	2.864	3.570	30	2.093	1.990	2.089	2.535	52	2.045	2.102	2.074	2.599
9	1.299	1.332	1.412	1.037	31	2.473	2.327	2.412	2.945	53	2.457	2.429	2.418	2.982
10	1.596	1.621	1.732	1.565	32	2.836	2.697	2.777	3.345	54	2.878	2.804	2.790	3.214
11	1.950	1.907	2.075	2.016	33	1.270	1.254	1.334	1.005	55	1.905	1.930	2.007	1.864
12	2.278	2.232	2.426	2.584	34	1.649	1.569	1.693	1.477	56	2.229	2.276	2.325	2.343
13	2.676	2.552	2.779	3.101	35	1.927	1.872	2.011	1.950	57	2.607	2.618	2.682	2.950
14	3.357	3.197	3.441	3.779	36	2.267	2.187	2.597	2.436	58	3.253	3.233	3.306	3.582
15	4.146	3.951	4.245	4.582	37	2.593	2.509	2.794	3.019	59	3.979	3.922	4.023	4.355
16	4.883	4.637	4.981	5.273	38	3.354	3.195	3.461	3.711	60	4.756	4.649	4.730	5.149
17	1.880	1.846	2.010	1.625	39	4.060	3.868	4.169	4.495	61	3.073	3.156	3.256	3.023
18	2.524	2.571	2.718	2.443	40	4.820	4.596	4.883	5.261	62	3.775	3.879	3.935	3.851
19	3.156	3.147	3.421	3.281	41	1.821	1.798	1.926	1.497	63	4.435	4.570	4.622	4.759
20	3.808	3.765	4.096	4.105	42	2.474	2.488	2.578	2.286	64	5.890	6.035	5.993	6.104
21	4.495	4.464	4.812	5.128	43	3.165	3.219	3.279	3.134	65	7.287	7.540	7.367	7.670
22	6.014	5.936	6.346	6.761	44	3.876	3.993	3.959	4.020	66	8.845	9.004	8.819	9.248



(a) 实例 31



(b) 实例 57

图 1　5种算法关于两个实例的非劣解分布图

在大规模实例求解方面, MTLBO的性能也更为优越. 关于26个实例MTLBO的DI_R小于CCA的相应值, 关于25个实例$\mathcal{C}$(M, CC)大于$\mathcal{C}$(CC, M); 同样, MTLBO取得了比CMA、ALNS更优的计算结果, 例如, MTLBO关于7个实例的DI_R至少比CMA的相应值小8, 而关于17个实例的$\mathcal{C}$(M, CM)等于1. 图1也显示了MTLBO的性能优势.

MTLBO构造多个班级独立进化, 设计4种策略实现学生阶段, 将奖惩机制作用于各班级上以区分好班和差班的计算资源, 这样, 好班的解能得到充分的探索, 避免过多计算资源浪费在差班上; 并且在每个阶段都注重全局搜索与局部搜索间的协调优化. 这些机理维持了探索与利用的良好平衡, 因此, MTLBO对于所求解的DHFSP具有较强的搜索优势.

4　结　论

近年来, 分布式调度已成为调度研究的热点问题, 但是DHFSP很少得到关注. 本文研究了考虑SDST的DHFSP, 并提出了一种新型MTLBO算法以同时最小化最大完成时间和最大延迟时间. 该算法采用双串编码方式, 种群划分为s个班级, 每个班级的进化都由两个教师阶段和一个学生阶段组成, 引入班级质量评价方式, 实现了奖惩机制和淘汰过程. 通过大量实验测试了新策略的有效性和MTLBO的搜索性能.

未来笔者将继续关注分布式混合流水车间调度问题,并尝试使用一些多种群算法解决这一问题,例如人工蜂群算法. 另外,分布式调度问题中的运输时间很常见,在后续的研究中也会考虑它. 同时将会解决与能源相关的DHFSP和具有能耗约束的DHFSP.

参考文献(References)

[1] Ruiz R, Vázquez-Rodríguez J A. The hybrid flow shop scheduling problem[J]. European Journal of Operational Research, 2010, 205(1): 1-18.

[2] Low C, Hsu C J, Su C T. A two-stage hybrid flowshop scheduling with a function constraint and unrelated alternative machines[J]. Computers & Operations Research, 2008, 35(3): 845-853.

[3] Rashidi E, Jahandar M, Zandieh M. An improved hybrid multi-objective parallel genetic algorithm for hybrid flow shop scheduling with unrelated parallel machines[J]. International Journal of Advanced Manufacturing Technology, 2010, 49(9/10/11/12): 1129-1139.

[4] Karimi N, Zandieh M, Karamooz H R. Bi-objective group scheduling in hybrid flexible flowshop: A multiphase approach[J]. Expert Systems with Applications, 2010, 37(6): 4024-4032.

[5] Naderi B, Zadieh M, Khaleghi Ghoshe Balagh A, et al. An improved simulated annealing for hybrid flowshops with sequence-dependent setup and transportation times to minimize total completion time and total tardiness[J]. Expert Systems with Applications, 2009, 36(6): 9625-9633.

[6] 王圣尧, 王凌, 许烨, 等. 求解混合流水车间调度问题的分布估计算法[J]. 自动化学报, 2012, 38(3): 437-443.
(Wang S Y, Wang L, Xu Y, et al. An estimation of distribution algorithm for solving hybrid flow-shop scheduling problem[J]. Acta Automatica Sinica, 2012, 38(3): 437-443.)

[7] Karimi N, Davoupour H. Multi-objective colonial competitive algorithm for hybrid flowshop problem[J]. Applied Soft Computing, 2016, 49: 725-733.

[8] Marichelvam M K, Prabaharan T, Yang X S. A discrete firefly algorithm for the multi-objective hybrid flowshop scheduling problems[J]. IEEE Transactions on Evolutionary Computation, 2014, 18(2): 301-305.

[9] Lei D M, Gao L, Zheng Y L. A novel teaching-learning-based optimization algorithm for energy-efficient scheduling in hybrid flow shop[J]. IEEE Transactions on Engineering Management, 2018, 65(2): 330-340.

[10] 王凌, 邓瑾, 王圣尧. 分布式车间调度优化算法研究综述[J]. 控制与决策, 2016, 31(1): 1-11.
(Wang L, Deng J, Wang S Y. Survey on optimization algorithms for distributed shop scheduling[J]. Control and Decision, 2016, 31(1): 1-11.)

[11] Gao J, Chen R, Deng W. An efficient tabu search algorithm for the distributed permutation flowshop scheduling problem[J]. International Journal of Production Research, 2013, 51(3): 641-651.

[12] Pan Q K, Gao L, Wang L, et al. Effective heuristic and metaheuristics to minimize total flowtime for the distributed permutation flowshop problem[J]. Expert Systems with Applications, 2019, 124: 309-324.

[13] Deng J, Wang L. A competitive memetic algorithm for multi-objective distributed permutation flow shop scheduling problem[J]. Swarm and Evolutionary Computation, 2017, 32: 121-131.

[14] Shao W S, Pi D C, Shao Z S. Optimization of makespan for the distributed no-wait flow shop scheduling problem with iterated greedy algorithms[J]. Knowledge-Based Systems, 2017, 137: 163-181.

[15] Lin J, Wang Z J, Li X D. A backtracking search hyper-heuristic for the distributed assembly flow-shop scheduling problem[J]. Swarm and Evolutionary Computation, 2017, 36: 1-12.

[16] Zhang G H, Xing K Y, Cao F. Scheduling distributed flowshops with flexible assembly and set-up time to minimise makespan[J]. International Journal of Production Research, 2018, 56(9): 3226-3244.

[17] Xiong F L, Xing K Y. Meta-heuristics for the distributed twostage assembly scheduling problem with bi-criteria of makespan and mean completion time[J]. International Journal of Production Research, 2014, 52(9): 2743-2766.

[18] Rifai A P, Nguyen H T, Dawal S Z M. Multi-objective adaptive large neighborhood search for distributed reentrant permutation flow shop scheduling[J]. Applied Soft Computing, 2016, 40: 42-57.

[19] Ying K C, Lin S W. Minimizing makespan for the distributed hybrid flowshop scheduling problem with multi-processor tasks[J]. Expert Systems with Applications, 2018, 92: 132-141.

[20] Hao J H, Li J Q, Du Y, et al. Solving distributed hybrid flowshop scheduling problems by a hybrid brain storm optimization algorithm[J]. IEEE Access, 2019, 7: 66879-66894.

[21] Lei D M, Wang T. Solving distributed two-stage hybrid flowshop scheduling using a shuffled frog-leaping algorithm with memeplex grouping[J]. Engineering Optimization, 2020, 52(9): 1461-1474.

[22] Shao W S, Shao Z S, Pi D C. Modeling and multi-neighborhood iterated greedy algorithm for distributed hybrid flow shop scheduling problem[J]. Knowledge-Based Systems, 2020, 194: 105527.

[23] Zheng J, Wang L, Wang J J. A cooperative coevolution algorithm for multi-objective fuzzy distributed hybrid flow shop[J]. Knowledge-Based Systems, 2020, 194: 105536.

[24] 蔡劲草, 雷德明. 考虑准备时间的分布式两阶段混合流水车间调度[J]. 计算机集成制造系统, 2020, 26(8):

2170-2179.
(Cai J C, Lei D M. Distributed two-stage hybrid flow shop scheduling with setup times[J]. Computer Integrated Manufacturing Systems, 2020, 26(8): 2170-2179.)

[25] 雷德明, 王甜. 基于改进蛙跳算法的分布式两阶段混合流水车间调度[J]. 控制与决策, DOI: 10.13195/j.kzyjc.2019.0472.
(Lei D M, Wang T. An improved shuffled frog leaping algorithm for the distributed two-stage hybrid flow shop scheduling[J]. Control and Decision, DOI: 10.13195/j.kzyjc.2019.0472.)

[26] Cai J C, Zhou R, Lei D M. Dynamic shuffled frog-leaping algorithm for distributed hybrid flow shop scheduling with multiprocessor tasks[J]. Engineering Applications of Artificial Intelligence, 2020, 90: 103540.

[27] Cai J C, Zhou R, Lei D M. Fuzzy distributed two-stage hybrid flow shop scheduling problem with setup time: Collaborative variable search[J]. Journal of Intelligent & Fuzzy Systems, 2020, 38(3): 3189-3199.

[28] Li J Q, Pan Q K, Mao K. A discrete teaching-learning-based optimisation algorithm for realistic flow shop scheduling problem[J]. Engineering Applications of Artificial Intelligence, 2015, 37: 279-292.

[29] Xie Z P, Zhang C Y, Shao X Y, et al. An effective hybrid teaching-learning-based optimization algorithm for permutation flow shop scheduling problem[J]. Advances in Engineering Software, 2014, 77: 35-47.

[30] Xu Y, Wang L, Wang S Y, et al. An effective teaching-learning-based optimization algorithm for the flexible jobshop scheduling problem with fuzzy processing time[J]. Neurocomputing, 2015, 148: 260-268.

[31] Buddala R, Mahapatra S S. Two-stage teaching-learning-based optimization method for flexible job-shop scheduling under machine breakdown[J]. The International Journal of Advanced Manufacturing Technology, 2019, 100(5/6/7/8): 1419-1432.

[32] Shao W S, Pi D C, Shao Z S. A hybrid discrete teaching-learning-based meta-heuristic for solving no-idle flow shop scheduling problem with total tardiness criterion[J]. Computers & Operations Research, 2018, 94: 89-105.

[33] 雷德明. 基于新型教学优化算法的低碳柔性作业车间调度[J]. 控制与决策, 2017, 32(9): 1621-1627.
(Lei D M. Novel teaching-learning-based optimization algorithm for low carbon scheduling of flexible job shop[J]. Control and Decision, 2017, 32(9): 1621-1627.)

[34] 张梅, 杨晟轩, 朱金辉. 基于多小组协同学习教学算法的车间作业调度问题[J]. 控制与决策, 2018, 33(8): 1354-1362.
(Zhang M, Yang S X, Zhu J H. Teaching-learning-based optimization algorithm with group collaboration for job shop scheduling problem[J]. Control and Decision, 2018, 33(8): 1354-1362.)

[35] Farshchin M, Camp C V, Maniat M. Multi-class teaching-learning-based optimization for truss design with frequency constraints[J]. Engineering Structures, 2016, 106: 355-369.

[36] Zou F, Wang L, Hei X H, et al. Teaching-learning-based optimization with dynamic group strategy for global optimization[J]. Information Sciences, 2014, 273: 112-131.

[37] Rao R V, Savsani V J, Vakharia D P. Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems[J]. Computer-Aided Design, 2011, 43(3): 303-315.

[38] 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.

[39] Goldberg D E, Lingle R. Alleles, loci, and the traveling salesman problem[C]. Proceedings of the 1st International Conference on Genetic Algorithms and Their Applications. Los Angeles, 1985: 154-159.

[40] Knowles J D, Corne D W. On metrics for comparing nondominated sets[C]. Proceedings of the 2002 Congress on Evolutionary Computation. Honolulu: IEEE, 2002: 711-716.

[41] 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.

作者简介

雷德明 (1968−), 男, 教授, 博士生导师, 从事智能系统优化与控制等研究, E-mail: deminglei11@163.com;

苏斌 (1996−), 男, 硕士生, 从事制造系统智能优化与调度的研究, E-mail: binwhut@163.com.

(责任编辑：李君玲)