

## 求解模糊分布式调度的竞争人工蜂群算法

郑友莲

引用本文

郑友莲. 求解模糊分布式调度的竞争人工蜂群算法[J]. 计算机科学, 2026, 53(7): 178-185.

ZHENG Youlian. [Competitive Artificial Bee Colony Algorithm for Fuzzy Distributed Scheduling](#)[J].

Computer Science, 2026, 53(7): 178-185.

---

## 相似文章推荐 (请使用火狐或 IE 浏览器查看文章)

**Similar articles recommended (Please use Firefox or IE to view the article)**

[FFiT:基于FasterViT的快联合帧插值去模糊方法](#)

FFiT:Faster Frame Interpolation Transformer Based on FasterViT

计算机科学, 2026, 53(7): 34-44. <https://doi.org/10.11896/jsjcx.250400046>

[基于Lasso优化的AHP-FEC气象设备健康度评估技术研究](#)

Research on Health Evaluation Technology of AHP-FEC Meteorological Equipment Based onLasso Optimization

计算机科学, 2026, 53(6A): 250400123-16. <https://doi.org/10.11896/jsjcx.250400123>

[基于大模型的模糊测试驱动生成](#)

Fuzzing Driver Generation Based on Large Language Models

计算机科学, 2026, 53(6A): 250400113-8. <https://doi.org/10.11896/jsjcx.250400113>

[面向物联网的基于模糊聚类的DTN路由算法](#)

Fuzzy Clustering-based DTN Routing Algorithm for IoT

计算机科学, 2026, 53(6): 358-366. <https://doi.org/10.11896/jsjcx.250400094>

[基于均值漂移的模糊三支聚类](#)

Fuzzy Three-way Clustering Based on Mean Shift

计算机科学, 2026, 53(6): 332-338. <https://doi.org/10.11896/jsjcx.250600056>

# 求解模糊分布式调度的竞争人工蜂群算法

郑友莲

湖北大学计算机学院 武汉 430062

**摘要** 分布式调度是分布式制造的重要环节,也是目前调度研究的焦点,而不确定性是分布式制造的本质特性。针对考虑工厂适用性和附加资源的模糊分布式两阶段混合流水车间调度问题,提出一种竞争人工蜂群(CABC)算法以最小化最大完成时间和总一致指标。为了获得高质量优化结果,将种群划分为两个蜂群,构建了两个蜂群间的竞争过程。该过程定义取胜蜂群为雇佣蜂群,失败蜂群为跟随蜂群,整个过程由雇佣蜂阶段和额外搜索、自适应跟随蜂阶段组成。给出了5种搜索策略并实现了蜂群间的差异化搜索,设计了一种自适应种群重构方法以避免蜂群间的过度竞争。进行了大量计算实验,实验结果验证了CABC新策略的有效性和它在求解模糊分布式调度方面的较强搜索优势。

**关键词:** 模糊;分布式调度;混合流水车间;人工蜂群算法;智能优化

**中图分类号** TP273

## Competitive Artificial Bee Colony Algorithm for Fuzzy Distributed Scheduling

ZHENG Youlian

School of Computer Science, Hubei University, Wuhan 430062, China

**Abstract** Distributed scheduling is important part of distributed manufacturing and the focus of the scheduling research. Uncertainty is the basic feature of distributed manufacturing. To solve fuzzy distributed two-stage hybrid flow shop scheduling problem with factory eligibility and additional resources, a competition artificial bee colony(CABC) algorithm is proposed to minimize makespan and total agreement index. To obtain high quality solutions, the whole population is divided into two bee swarms and competition process between them is constructed. In the competition process, the winning swarm is used as employed bee swarm and another one acts as onlooker bee swarm. The competition process is composed of employed bee phase and extra search, and adaptive onlooker bee phase. Five search strategies are given and the diversified search is implemented. An adaptive population reconstruction is applied to avoid excessive competition between swarms. Experiments are conducted and the computational results reveal that new strategies of CABC are effective and CABC has the promising search advantages on solving the considered problem.

**Keywords** Fuzzy, Distributed scheduling, Hybrid flow shop, Artificial bee colony, Intelligent optimization

## 1 引言

混合流水车间调度问题由 $n$ 个工件和至少两个阶段组成。两阶段混合流水车间调度问题(Two-Stage Hybrid Flow Shop Scheduling Problem, THFSP)仅包含两个阶段,其中至少一个阶段由并行机组成。通常包括3种情形:阶段1(2)是并行机阶段,阶段2(1)是单机阶段,两阶段都是并行机阶段。THFSP研究受到研究者的广泛关注<sup>[1-10]</sup>。

近年来,多工厂生产或分布式制造替代传统的单工厂集中式生产,成为主流生产和制造模式。分布式调度是在多工厂分布式环境下的调度问题, $F$ 个工厂共同完成所有工件加工。目前,该问题受到研究者的广泛关注<sup>[11-21]</sup>。作为THFSP在分布式制造环境下的扩展,分布式两阶段混合流水车间调度问题(Distributed Two-stage Hybrid Flow Shop Schedu-

ling Problem, DTHFSP)研究取得了一些进展。Wang等<sup>[11]</sup>运用反馈人工蜂群(Artificial Bee Colony, ABC)算法求解考虑工厂适用性和附加资源的问题。针对模糊DTHFSP, Cai等<sup>[12]</sup>设计了一种协作变量搜索算法, Xi等<sup>[13]</sup>提出了一种由Q-学习算法动态调整结构的新型教学优化算法。Dong等<sup>[14]</sup>应用混合非劣排序遗传算法-III求解绿色问题。由于引入了反馈、协作、学习等机制和混合等手段,相应的智能算法搜索能力显著增强,但也不可避免地带来了一些问题,如算法结构比原算法更复杂,实现难度加大。

如上所述,DTHFSP研究主要考虑了准备时间和模糊加工时间,较少关注工厂适用性和附加资源。其中,工厂适用性指至少存在一个工件不能由某个工厂加工<sup>[22]</sup>,而附加资源则是完成工件加工所需的机械手、自动导引小车、托盘等<sup>[23]</sup>。由于实际分布式工厂往往同时存在模糊加工时间、工厂适用

性和附加资源约束(任意时刻所有机器所使用的每一种附加资源总和不能超过给定阈值),因此本研究考虑工厂适用性、附加资源的模糊 DTHFSP,并充分利用 ABC 算法在求解 DTHFSP 和分布式 HFSP 方面的搜索优势和能力<sup>[11,24-26]</sup>,以及竞争机制对算法性能的改进能力,设计新型 ABC 算法对所研究的模糊 DTHFSP 进行优化求解。

本文的主要工作内容如下:

1)描述考虑工厂适用性和附加资源的模糊 DTHFSP,问题的目标是最大完成时间和总一致指标。

2)针对上述问题,引入蜂群间的竞争过程,设计一种新型竞争人工蜂群(CABC)算法。竞争过程定义取胜蜂群为雇佣蜂群,失败蜂群为跟随蜂群,整个过程由雇佣蜂阶段和额外搜索、自适应跟随蜂阶段组成。此外,还实现了蜂群间的差异化搜索。

3)为了测试 CABC 在求解模糊 DTHFSP 中的搜索优势,运用 60 个测试实例,通过与常规 ABC 和所选择的 3 种对比算法进行比较,验证了 CABC 新策略的有效性和 CABC 的搜索优势。

## 2 问题描述

考虑工厂适用性和附加资源的模糊 DTHFSP 描述如下:存在  $n$  个工件,以及属于同一家制造企业且地理位置各异的  $F$  个工厂,每个工厂  $f$  都由一个两阶段混合流水车间组成,其中工厂  $f$  的阶段 1 由  $m_f$  台不相关的并行机组成,阶段 2 仅包含一台机器  $Ma_f$ 。 $M_{fk}$  表示工厂  $f$  中阶段 1 的第  $k$  台机器, $p_{ifk}$  和  $pt_{if}$  分别为工件  $J_i$  在  $M_{fk}$  和  $Ma_f$  上的加工时间, $r_{ifk}$  表示  $J_i$  在  $M_{fk}$  上加工时所需的附加资源量,每个工厂只考虑一种附加资源。 $d_i$  和  $\Theta_i$  分别为  $J_i$  的交货期和能加工  $J_i$  的机器集合。

考虑工厂适用性,它指不是所有工厂都能加工每个工件,即至少存在一个工件  $J_i$ ,其机器集  $\Theta_i \subset \{1, 2, \dots, F\}$ 。在每个工厂,考虑附加资源约束,它指在每个工厂  $f$ ,任意时刻阶段 1 各机器所占用的附加资源之和不能超过给定阈值  $R_{\max}^f$ 。除了上述约束,其他约束见文献[11]。

在模糊环境下加工时间为三角模糊数(TFN), $p_{ifk} = (p_{ifk}^1, p_{ifk}^2, p_{ifk}^3)$ , $pt_{if} = (pt_{if}^1, pt_{if}^2, pt_{if}^3)$ ,交货期描述为  $d_i = (d_i^1, d_i^2)$ ,工件延迟交货的程度用一致指标  $AI_i$  表示。

与 THFSP 相比,DTHFSP 多了一个工厂分配子问题,用来确定工件的加工工厂。

问题的目的是在满足约束条件并求解子问题的情况下,最小化以下两个目标:

$$C_{\max} = \max_{i=1,2,\dots,n} \{C_i\} \quad (1)$$

$$T_{AI} = \sum_{i=1}^n AI_i \quad (2)$$

其中, $C_i$  为工件  $J_i$  的完成时间;目标  $C_{\max}$  是所有工件的最大完成时间; $T_{AI}$  为总一致指标,即所有工件的一致指标之和; $AI_i$  是工件  $J_i$  的一致指标<sup>[12]</sup>, $AI_i = \text{area}(C_i \cap d_i) / \text{area}(C_i)$ , $\text{area}(C_i)$  和  $\text{area}(C_i \cap d_i)$  分别表示  $C_i$  和  $C_i \cap d_i$  所在区域的面积。

由于加工时间用 TFN 表示,为了计算问题的目标函数,

需要用到 TFN 的求和、比较和取大 3 种操作。

对于 TFN  $s = (s_1, s_2, s_3)$  和  $t = (t_1, t_2, t_3)$ ,求和操作定义如下:

$$s + t = (s_1 + t_1, s_2 + t_2, s_3 + t_3) \quad (3)$$

比较操作根据  $c_1(s) = (s_1 + 2s_2 + s_3)/4$ ,  $c_2(s) = s_2$ ,  $c_3(s) = s_3 - s_1$  确定两个 TFN 之间的大小关系,具体描述如下:如果  $c_1(s) > c_1(t)$ ,则  $s > t$ ;如果  $c_1(s) = c_1(t)$ , $c_2(s) > c_2(t)$ ,则  $s > t$ ;如果  $c_1(s) = c_1(t)$ , $c_2(s) = c_2(t)$  且  $c_3(s) > c_3(t)$ ,则  $s > t$ 。

$s \vee t$  表示  $s$  和  $t$  的取大操作,由于  $s \vee t$  往往不再是 TFN,Lei 等<sup>[27]</sup>提出一种近似取大操作:若  $s > t$ ,则  $s \vee t = s$ ,否则  $s \vee t = t$ 。

## 3 竞争人工蜂群算法

### 3.1 初始化与种群划分

采用双串表示方法,对于具有  $n$  个工件、 $F$  个工厂的模糊 DTHFSP,将其解表示为机器分配串  $[M_{l_1 h_1}, M_{l_2 h_2}, \dots, M_{l_n h_n}]$  和调度串  $[\pi_1, \pi_2, \dots, \pi_n]$ ,其中  $1 \leq l_i \leq F$ ,  $h_i \in \{1, 2, \dots, m_{l_i}\}$ ,  $M_{l_i h_i}$  表示工件  $J_i$  所分配的机器, $\pi_i \in \{1, 2, \dots, n\}$ 。

解码过程描述如下:

1)对于每台机器  $M_{fk}$ ,首先根据机器分配串确定分配到该机器上的所有工件,然后根据调度串,得到这些工件的一个排列。从排列的第一个工件开始,依次针对每个工件,在满足附加资源约束条件下,安排工件在  $M_{fk}$  上的加工。

2)在每个工厂  $f$ ,根据机器分配串,确定分配到该工厂的所有工件,按照这些工件在第一阶段的完成时间,从短到长,依次安排工件在  $Ma_f$  上的加工。

安排工件  $J_i$  在  $M_{fk}$  上加工时,在满足附加资源约束和加工时间情况下,如果  $J_i$  能插入到  $M_{fk}$  的空闲时间段,则选择开始时间最小的空闲时间段。

随机产生具有  $N$  个初始解的种群  $P$ 。

引入了种群划分,采用如下过程将种群  $P$  划分为蜂群  $S_1, S_2$ :首先对种群  $P$  中的所有解进行分支排序<sup>[28]</sup>,假设  $rank_{x_1} \leq rank_{x_2} \leq \dots \leq rank_{x_N}$ ;然后将解蜂群  $x_1, x_3, x_5 \dots$  加入到蜂群  $S_1$ ,剩余解加入到蜂群  $S_2$  中。其中, $rank_x$  为解  $x$  的秩。

上述基于解排序的划分方法将使得两个蜂群的解的质量相近。

### 3.2 蜂群间的竞争过程

利用两种全局搜索算子  $GS_1$  和  $GS_2$  以及 8 种邻域结构  $\mathcal{N}_1 - \mathcal{N}_8$  构建了 5 种搜索策略。 $GS_1$  和  $GS_2$  分别为机器分配串的两点交叉和调度串的顺序交叉算子。 $\mathcal{N}_1$  的描述如下:从调度串中随机选择  $\pi_k (k > 1)$ ,如果  $AI_{\pi_k} < 1$ ,则将  $\pi_k$  插入到调度串中随机选择的位置  $l < k$ 。 $\mathcal{N}_2$  则通过选择具有最小一致指标的  $\pi_k (k > 1)$ ,并将其插入到调度串中随机确定的位置  $l < k$  来产生新解。 $\mathcal{N}_3$  的步骤如下:确定完成时间最大的工厂  $f$ ,从工厂  $f$  内完成时间最大的机器  $M_{fk}$  上随机选择一个工件  $J_i$ ,将其转移到完成时间最小的机器  $M_{gl}$  上。 $\mathcal{N}_4$  与  $\mathcal{N}_3$  步骤类似,两者的区别在于将所选择的  $J_i$  转移到最小完成时间的机器  $M_{gl} (g \in \Theta_i)$  上。当执行  $\mathcal{N}_5$  时,采用  $\mathcal{N}_3$  的

原则选择  $J_i$ , 运用  $\mathcal{N}_4$  的方法确定机器  $M_{g_i}$  ( $g \in \Theta_i$ ), 然后从  $M_{g_i}$  上随机选择工件  $J_j$ , 互换机器分配串上的  $M_{l_{h_i}} (=M_{j_k})$  与  $M_{l_{h_j}} (=M_{g_i})$ 。  $\mathcal{N}_6$  的描述如下: 从调度串上随机选择两个片段, 然后互换所选中的片段, 要求这两个片段无交集。假设调度串为  $[10, 9, 8, 1, 6, 4, 5, 3, 2, 7]$ , 片段  $[9, 8, 1]$  和  $[3, 2]$  被选择并互换, 则调度串变成  $[10, 3, 2, 1, 6, 4, 5, 9, 8, 1, 7]$ 。  $\mathcal{N}_7$  的过程如下: 随机选择工件  $J_i$  和工厂  $f \in \Theta_i$ , 从该工厂随机选择一台机器替代  $M_{l_{h_i}}$ 。  $\mathcal{N}_8$  为调度串的插入操作。

提出了 5 种搜索策略  $SO_1-SO_5$ 。  $SO_1$  的描述如下: 对于解  $x$ , 先随机选择解  $y$ , 执行  $GS_1$ , 产生新解  $z$ , 如果  $z \succ x$  或  $z$  和  $x$  彼此非受支配, 则用  $z$  替代  $x$ , 并用  $x$  更新外部档案  $\Omega$ ; 接着采用同样的方式对  $x, y$  执行  $GS_2$ , 并根据上述条件判断  $z$  能否替代  $x$  和是否更新  $\Omega$ 。  $SO_i$  ( $i \geq 2$ ) 的步骤如下: 对于解  $x$ , 依次执行邻域结构  $N_{2(i-2)+1}, N_{2(i-2)+2}$ ; 当执行  $N_i$  时, 产生新解  $z \in N_i(x)$ , 根据上述条件判断  $z$  能否替代  $x$  和是否更新  $\Omega$ 。 其中,  $N_i(x)$  表示由  $N_i$  产生的  $x$  的邻域解的集合。

在上述 5 种搜索策略基础上, 建立了蜂群  $S_1$  和  $S_2$  间的竞争过程, 该过程首先动态确定哪个蜂群为雇佣蜂群  $EB$ , 哪个蜂群为跟随蜂群  $OB$ , 然后依次执行雇佣蜂阶段和跟随蜂阶段。

第  $gen$  代蜂群  $S_1$  和  $S_2$  间的竞争过程如下:

1) 如果  $gen > 1$ , 则满足  $Evo_l^{gen-1} > Evo_{3-l}^{gen-1}$  的蜂群  $S_l$  为取胜蜂群,  $\beta_l = \beta_l + 1$ ; 而蜂群  $S_{3-l}$  则为失败蜂群,  $\beta_{3-l} = 0$ 。 如果  $gen = 1$  或  $Evo_l^{gen-1} = Evo_{3-l}^{gen-1}$ , 则随机选择  $S_l$  为取胜蜂群,  $\beta_l = \beta_l + 1$ ; 而定义  $S_{3-l}$  为失败蜂群,  $\beta_{3-l} = 0$ 。

2) 对于每个解  $x_i \in S_l$ , 随机选择解  $y \in S_l$ , 对解  $x_i, y$  执行  $SO_1$ 。

3) 对蜂群  $S_l$  中的所有解执行非支配排序, 对于具有最小  $rank_{x_i}$  的每个  $x_i \in S_l$ , 随机选择解  $y \in S_l$ , 对解  $x_i, y$  执行  $SO_2$ 。

4) 若  $rand < Evo_l^{gen-1} / (Evo_l^{gen-1} + Evo_{3-l}^{gen-1})$ , 则对  $S_{3-l}$  中所有解进行非支配排序, 对每个解  $x_i \in S_{3-l}$ , 随机选择具有最小  $rank_{x \in S_{3-l}}$  的解  $y \in S_{3-l}$ , 对选中的解执行从  $SO_2$  和  $SO_3$  中随机选择的策略; 否则, 对每个解  $x_i \in S_{3-l}$ , 执行从  $SO_4$  和  $SO_5$  中随机选择的策略。

其中,  $rand$  为服从区间  $[0, 1]$  上均匀分布的随机数,  $\beta_l$  是蜂群  $S_l$  在竞争过程中取胜的次数,  $Evo_l^{gen-1}$  表示蜂群  $S_l$  在第  $gen-1$  代的进化质量。

$$Evo_l^{gen-1} = \sum_{x \in S_l} \lambda_x^{gen-1} / \eta_l^{gen-1}$$

(4)

其中,  $\lambda_x^{gen-1}$  表示在第  $gen-1$  代  $x$  得到更新的次数,  $\eta_l^{gen-1}$  为蜂群  $S_l$  在第  $gen-1$  代的总搜索次数。

步骤 1) 中, 进化质量占优的蜂群  $S_l$  在竞争中取胜, 为雇佣蜂群, 而失败的蜂群则为跟随蜂群。 步骤 2) 为雇佣蜂阶段, 是针对  $S_l$  的搜索过程, 自适应跟随蜂阶段如步骤 4) 所示, 当该步骤中的条件满足时,  $S_{3-l}$  的所有解跟随该蜂群内的最优解。 步骤 3) 是对取胜蜂群进行的额外搜索。 步骤 2)、步骤 3)、步骤 4) 的搜索策略各异, 实现了差异化搜索, 且两个蜂群采取何种搜索策略, 取决于它们之间的竞争结果, 这使得每个蜂群的策略动态确定。

3.3 算法描述

CABC 的主要步骤描述如下:

1) 随机产生初始种群  $P$ ,  $P$  中的非劣解构成初始  $\Omega$ , 对所有解  $x_i \in P$ ,  $trail_i = 0$ , 令  $gen = 1$ 。

2) 将种群  $P$  划分为蜂群  $S_1, S_2$ , 对  $S_i$ , 令  $\beta_i = 0$ 。

3) 执行蜂群  $S_1$  和  $S_2$  间的竞争过程。

4) 运行侦查蜂阶段。

5) 执行种群重构。

6) 计算蜂群  $S_1$  和  $S_2$  的  $Evo_1^{gen}$  和  $Evo_2^{gen}$ 。  $gen = gen + 1$ , 如果终止条件成立, 则停止搜索; 否则转到步骤 2)。

侦查蜂阶段的过程如下: 对于  $x_i \in S_l$  ( $l = 1, 2$ ), 如果  $trail_i \geq Limit$ , 则对解  $x_i$  执行搜索策略  $SO_5$ 。 其中,  $trail_i$  为解  $x_i$  连续未更新次数, 如果  $x_i$  在搜索过程中得到更新, 则  $trail_i = 0$ , 否则  $trail_i = trail_i + 1$ 。

种群重构过程如下: 当某个蜂群  $S_i$  对应的  $\beta_i \geq \alpha$ , 则将蜂群  $S_1$  和  $S_2$  合并为新种群  $P$ , 然后采取步骤 1) 的方法将种群  $P$  划分为蜂群  $S_1$  和  $S_2$ 。

上述过程中,  $\beta_i \geq \alpha$  意味着蜂群  $S_i$  连续取胜了至少  $\alpha$  代, 这说明在这连续的  $\alpha$  代,  $S_i$  固定为雇佣蜂群, 搜索策略多样化和动态选择将无法实现, 因此进行种群重构。

图 1 描述了 CABC 的流程。 与常规 ABC 不同, CABC 具有新的结构, 引入了种群划分、两个蜂群间的竞争和重构。

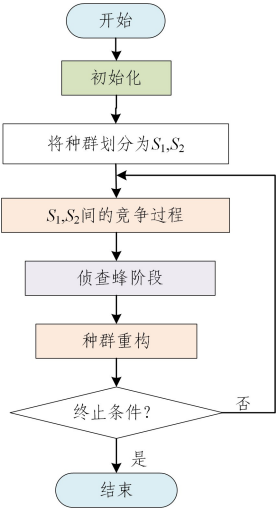


图 1 算法流程图  
Fig. 1 Flow chart of CABC

与现有 ABC 算法相比, CABC 具有如下新特征: 具有全新算法结构, 由蜂群  $S_1$  和  $S_2$  间的竞争过程、侦查蜂阶段和种群重构组成, 雇佣蜂群、跟随蜂群以及相应雇佣蜂阶段和跟随蜂阶段在竞争过程中动态确定, 同时动态地选择搜索策略, 竞争、重构和多种策略的使用有助于增强算法的全局搜索能力, 降低陷入局部最优解的可能。

4 计算实验

本章通过大量计算实验来测试 CABC 的搜索性能, 所有实验利用 Microsoft Visual C++ 2019 编程实现, 并运用于 8.0GB RAM 2.40GHz CPU 的 PC 上。

4.1 测试实例、对比算法与参数实验

文献[12]提供了本文所研究问题的测试实例。为了使用这些实例,去掉这些实例原有的准备时间,而增加附加资源和工件的加工机器集合,其中附加资源和加工机器集合的数据来自于文献[11]。本文选择文献[12]中的前 60 个实例做计算实验。

选择了 3 种对比算法:竞争 memetic 算法(Competitive Memetic Algorithm,CMA<sup>[29]</sup>)、协作变搜索(Collaborative Variable Search,CVS<sup>[12]</sup>)和基于 Pareto 的分布估计算法(Pareto-based Estimation of Distributed Algorithm,PEDA<sup>[30]</sup>)。当在解码过程中加入对附加资源和机器适用性的处理后,这些算法均能直接用来求解 DTHFSP。

构建 ABC 算法以揭示 CABC 新策略的有效性。从 CABC 中剔除竞争过程、重构,雇佣蜂阶段按照竞争过程的步骤 2)执行,跟随蜂阶段则用常规 ABC 的方法进行,可得到要构建的 ABC。跟随蜂阶段,对于所选择的蜜源,从 8 个邻域结构中随机选择一个邻域结构。

选择常见的 3 个性能指标:IGD<sup>[31]</sup>,C<sup>[32]</sup>和  $\rho$ <sup>[11]</sup>。参考集  $\Omega^*$  由所有算法的非支配解集并集中的非劣解组成,IGD 和  $\rho$  的计算需要用到参考集。

CABC 的参数包括种群规模  $N$ 、Limit、 $\alpha$  和终止条件。实验结果显示:当 0.2n 秒 CPU 时间达到时,CABC 关于所有实例都能充分收敛;当上述 CPU 时间达到时,ABC 和 3 个对比算法关于所有实例也能充分收敛。因此,5 种算法的终止条

件均设置为 0.2n 秒 CPU 时间。

通过实验发现,当  $N=50,\alpha=3,Limit=5$  时,CABC 关于所有实例所获得的计算结果均优于具有其他参数组合的 CABC。因此,直接使用上述参数设置。除终止条件外,3 种对比算法的参数设置直接来自于文献[12,29,30]。实验结果表明,在上述终止条件下,使用相关文献的参数设置,对比算法所获得的计算结果最好。

4.2 结果与分析

对 CABC,ABC,CVS,CMA,PEDA 这 5 种算法进行了测试和比较。每种算法对每个实例随机运行 10 次,各算法对应的结果如表 1—表 4 所列,其中 A,C,M,V,P 分别表示 CABC,ABC,CMA,CVS,PEDA。表 5 列出了算法之间两两配对样本  $t$ -检验的结果。图 2 给出了 5 种算法在 95%置信区间下的均值。图 3 给出了所有算法关于 2 个实例的非劣解分布。如表 1—表 4 所列,CABC 关于 56 个实例所得到的 IGD 值小于 ABC 的相应结果,CABC 关于 59 个实例取得了优于 ABC 的  $\rho$  值,且 CABC 关于大多数实例所得到的  $\rho$  值大于 0.6。这说明,参考集内的大多数解由 CABC 提供。此外,CABC 关于 30 个实例所得 C(C,A)的值等于 1,表明 ABC 所产生的所有解均受 CABC 的非劣解支配。从图 2 和图 3 也可看出,CABC 的性能显著优于 ABC。表 5 的结果表明 CABC 优于 ABC 在统计意义上成立。因此,CABC 的新策略(如竞争过程)显著改善了其性能,新策略合理有效。

表 1 所有算法的 IGD 值  
Table 1 IGD values of all algorithms

| No. | CABC  | ABC   | CVS   | CMA   | PEDA  | No. | CABC  | ABC   | CVS   | CMA   | PEDA  |
|-----|-------|-------|-------|-------|-------|-----|-------|-------|-------|-------|-------|
| 1   | 0     | 0.183 | 0.583 | 0.502 | 0.631 | 31  | 0.156 | 0.242 | 0.221 | 0.292 | 0.298 |
| 2   | 0.138 | 0.125 | 0.580 | 0.433 | 0.397 | 32  | 0     | 0.308 | 0.534 | 0.311 | 0.314 |
| 3   | 0.139 | 0.124 | 0.261 | 0.224 | 0.403 | 33  | 0.089 | 0.209 | 0.263 | 0.344 | 0.207 |
| 4   | 0     | 0.214 | 0.619 | 0.461 | 0.591 | 34  | 0.093 | 0.325 | 0.296 | 0.425 | 0.288 |
| 5   | 0.103 | 0.245 | 0.530 | 0.506 | 0.508 | 35  | 0     | 0.133 | 0.330 | 0.455 | 0.355 |
| 6   | 0.082 | 0.277 | 0.350 | 0.398 | 0.448 | 36  | 0.051 | 0.104 | 0.190 | 0.238 | 0.299 |
| 7   | 0.071 | 0.354 | 0.531 | 0.567 | 0.604 | 37  | 0     | 0.171 | 0.224 | 0.284 | 0.259 |
| 8   | 0     | 0.413 | 0.473 | 0.583 | 0.594 | 38  | 0.067 | 0.023 | 0.147 | 0.147 | 0.182 |
| 9   | 0.107 | 0.283 | 0.373 | 0.526 | 0.393 | 39  | 0.046 | 0.203 | 0.458 | 0.518 | 0.441 |
| 10  | 0.114 | 0.267 | 0.328 | 0.564 | 0.446 | 40  | 0.070 | 0.237 | 0.215 | 0.401 | 0.343 |
| 11  | 0     | 0.384 | 0.485 | 0.495 | 0.412 | 41  | 0.155 | 0.167 | 0.124 | 0.233 | 0.085 |
| 12  | 0     | 0.299 | 0.463 | 0.489 | 0.428 | 42  | 0     | 0.158 | 0.190 | 0.206 | 0.164 |
| 13  | 0.105 | 0.255 | 0.304 | 0.275 | 0.322 | 43  | 0     | 0.146 | 0.187 | 0.236 | 0.163 |
| 14  | 0.193 | 0.253 | 0.281 | 0.353 | 0.339 | 44  | 0     | 0.201 | 0.380 | 0.285 | 0.326 |
| 15  | 0.035 | 0.200 | 0.283 | 0.365 | 0.318 | 45  | 0.081 | 0.159 | 0.188 | 0.324 | 0.212 |
| 16  | 0.030 | 0.212 | 0.278 | 0.420 | 0.394 | 46  | 0.006 | 0.123 | 0.104 | 0.195 | 0.210 |
| 17  | 0.062 | 0.162 | 0.333 | 0.345 | 0.457 | 47  | 0     | 0.286 | 0.285 | 0.392 | 0.351 |
| 18  | 0.089 | 0.233 | 0.497 | 0.618 | 0.565 | 48  | 0     | 0.203 | 0.189 | 0.399 | 0.371 |
| 19  | 0.033 | 0.200 | 0.226 | 0.280 | 0.280 | 49  | 0.022 | 0.137 | 0.176 | 0.289 | 0.138 |
| 20  | 0.046 | 0.298 | 0.306 | 0.555 | 0.407 | 50  | 0.017 | 0.173 | 0.153 | 0.210 | 0.231 |
| 21  | 0.004 | 0.092 | 0.274 | 0.251 | 0.293 | 51  | 0.310 | 0.305 | 0.036 | 0.352 | 0.131 |
| 22  | 0     | 0.375 | 0.631 | 0.583 | 0.510 | 52  | 0     | 0.122 | 0.100 | 0.155 | 0.140 |
| 23  | 0.104 | 0.289 | 0.356 | 0.389 | 0.329 | 53  | 0     | 0.126 | 0.545 | 0.453 | 0.505 |
| 24  | 0     | 0.277 | 0.229 | 0.319 | 0.176 | 54  | 0.046 | 0.097 | 0.116 | 0.140 | 0.118 |
| 25  | 0.021 | 0.105 | 0.067 | 0.351 | 0.184 | 55  | 0.037 | 0.229 | 0.111 | 0.438 | 0.155 |
| 26  | 0.108 | 0.355 | 0.606 | 0.522 | 0.464 | 56  | 0.019 | 0.088 | 0.076 | 0.096 | 0.091 |
| 27  | 0     | 0.260 | 0.278 | 0.464 | 0.343 | 57  | 0     | 0.158 | 0.175 | 0.289 | 0.224 |
| 28  | 0.241 | 0.202 | 0.221 | 0.366 | 0.326 | 58  | 0.068 | 0.210 | 0.268 | 0.299 | 0.343 |
| 29  | 0     | 0.275 | 0.338 | 0.426 | 0.417 | 59  | 0     | 0.214 | 0.231 | 0.242 | 0.225 |
| 30  | 0     | 0.257 | 0.153 | 0.294 | 0.333 | 60  | 0.171 | 0.232 | 0.114 | 0.261 | 0.145 |

表 2 所有算法的  $\rho$  值  
Table 2  $\rho$  values of all algorithms

| No. | CABC  | ABC   | CVS   | CMA   | PEDA | No. | CABC  | ABC   | CVS   | CMA | PEDA  |
|-----|-------|-------|-------|-------|------|-----|-------|-------|-------|-----|-------|
| 1   | 1.000 | 0     | 0     | 0     | 0    | 31  | 0.800 | 0     | 0.200 | 0   | 0     |
| 2   | 0.800 | 0.200 | 0     | 0.200 | 0    | 32  | 1.000 | 0     | 0     | 0   | 0     |
| 3   | 0.250 | 0.500 | 0.250 | 0     | 0    | 33  | 0.750 | 0     | 0     | 0   | 0.250 |
| 4   | 1.000 | 0     | 0     | 0     | 0    | 34  | 0.800 | 0     | 0.200 | 0   | 0     |
| 5   | 0.625 | 0.250 | 0.125 | 0     | 0    | 35  | 1.000 | 0     | 0     | 0   | 0     |
| 6   | 0.857 | 0     | 0.143 | 0     | 0    | 36  | 0.667 | 0.333 | 0     | 0   | 0     |
| 7   | 0.857 | 0.143 | 0     | 0     | 0    | 37  | 1.000 | 0     | 0     | 0   | 0     |
| 8   | 1.000 | 0     | 0     | 0     | 0    | 38  | 0.600 | 0.400 | 0     | 0   | 0     |
| 9   | 0.750 | 0     | 0.250 | 0     | 0    | 39  | 0.875 | 0     | 0.125 | 0   | 0     |
| 10  | 0.750 | 0     | 0.250 | 0     | 0    | 40  | 0.636 | 0     | 0.364 | 0   | 0     |
| 11  | 1.000 | 0     | 0     | 0     | 0    | 41  | 0.800 | 0     | 0     | 0   | 0.200 |
| 12  | 1.000 | 0     | 0     | 0     | 0    | 42  | 1.000 | 0     | 0     | 0   | 0     |
| 13  | 0.571 | 0.143 | 0.286 | 0     | 0    | 43  | 1.000 | 0     | 0     | 0   | 0     |
| 14  | 0.667 | 0     | 0.333 | 0     | 0    | 44  | 1.000 | 0     | 0     | 0   | 0     |
| 15  | 0.833 | 0.167 | 0     | 0     | 0    | 45  | 0.714 | 0     | 0.286 | 0   | 0     |
| 16  | 0.667 | 0     | 0.333 | 0     | 0    | 46  | 0.818 | 0     | 0.182 | 0   | 0     |
| 17  | 0.750 | 0.250 | 0.125 | 0     | 0    | 47  | 1.000 | 0     | 0     | 0   | 0     |
| 18  | 0.875 | 0.125 | 0     | 0     | 0    | 48  | 1.000 | 0     | 0     | 0   | 0     |
| 19  | 0.857 | 0     | 0.143 | 0     | 0    | 49  | 0.889 | 0     | 0.111 | 0   | 0     |
| 20  | 0.875 | 0     | 0.125 | 0     | 0    | 50  | 0.818 | 0     | 0.182 | 0   | 0     |
| 21  | 0.800 | 0.200 | 0     | 0     | 0    | 51  | 0.600 | 0     | 0.400 | 0   | 0     |
| 22  | 1.000 | 0     | 0     | 0     | 0    | 52  | 1.000 | 0     | 0     | 0   | 0     |
| 23  | 0.800 | 0     | 0.200 | 0     | 0    | 53  | 1.000 | 0     | 0     | 0   | 0     |
| 24  | 1.000 | 0     | 0     | 0     | 0    | 54  | 0.875 | 0     | 0.125 | 0   | 0     |
| 25  | 0.500 | 0.167 | 0.333 | 0     | 0    | 55  | 0.750 | 0     | 0.250 | 0   | 0     |
| 26  | 0.833 | 0     | 0.167 | 0     | 0    | 56  | 0.889 | 0     | 0.111 | 0   | 0     |
| 27  | 1.000 | 0     | 0     | 0     | 0    | 57  | 1.000 | 0     | 0     | 0   | 0     |
| 28  | 0.600 | 0     | 0.400 | 0     | 0    | 58  | 0.800 | 0     | 0.200 | 0   | 0     |
| 29  | 1.000 | 0     | 0     | 0     | 0    | 59  | 1.000 | 0     | 0     | 0   | 0     |
| 30  | 1.000 | 0     | 0     | 0     | 0    | 60  | 1.000 | 0     | 0     | 0   | 0     |

表 3 所有算法关于C的结果(1)  
Table 3 Results of all algorithms on  $C$  (1)

| No. | $C(C,A)$ | $C(A,C)$ | $C(C,V)$ | $C(V,C)$ | $C(C,M)$ | $C(M,C)$ | $C(C,P)$ | $C(P,C)$ |
|-----|----------|----------|----------|----------|----------|----------|----------|----------|
| 1   | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 2   | 0.750    | 0        | 1.000    | 0        | 0.750    | 0        | 1.000    | 0        |
| 4   | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 5   | 0.500    | 0.167    | 0.800    | 0        | 1.000    | 0        | 1.000    | 0        |
| 6   | 1.000    | 0        | 0.833    | 0        | 1.000    | 0        | 1.000    | 0        |
| 7   | 0.875    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 8   | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 9   | 1.000    | 0        | 0.500    | 0        | 1.000    | 0        | 1.000    | 0        |
| 10  | 0.875    | 0        | 0.600    | 0        | 1.000    | 0        | 1.000    | 0        |
| 11  | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 12  | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 13  | 0.500    | 0.167    | 0.600    | 0.167    | 1.000    | 0        | 1.000    | 0        |
| 14  | 1.000    | 0        | 0.500    | 0        | 1.000    | 0        | 1.000    | 0        |
| 15  | 0.857    | 0        | 0.800    | 0        | 1.000    | 0        | 1.000    | 0        |
| 16  | 1.000    | 0        | 0.667    | 0.200    | 1.000    | 0        | 1.000    | 0        |
| 17  | 0.625    | 0        | 0.500    | 0        | 0.857    | 0        | 1.000    | 0        |
| 18  | 0.909    | 0        | 0.857    | 0        | 1.000    | 0        | 1.000    | 0        |
| 19  | 0.929    | 0        | 0.800    | 0        | 1.000    | 0        | 1.000    | 0        |
| 20  | 1.000    | 0        | 0.857    | 0        | 1.000    | 0        | 1.000    | 0        |
| 21  | 0.800    | 0.200    | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 22  | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 23  | 1.000    | 0        | 0.667    | 0        | 1.000    | 0        | 1.000    | 0        |
| 24  | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 25  | 0.750    | 0.167    | 0.667    | 0.333    | 1.000    | 0        | 1.000    | 0        |
| 26  | 1.000    | 0        | 0.800    | 0        | 1.000    | 0        | 1.000    | 0        |
| 27  | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 28  | 1.000    | 0        | 0.818    | 0        | 1.000    | 0        | 1.000    | 0        |
| 29  | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |
| 30  | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        | 1.000    | 0        |

表 4 所有算法关于C的结果(2)  
Table 4 Results of all algorithms on C(2)

| No. | C(C,A) | C(A,C) | C(C,V) | C(V,C) | C(C,M) | C(M,C) | C(C,P) | C(P,C) |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|
| 31  | 1.000  | 0      | 0.667  | 0      | 1.000  | 0      | 1.000  | 0      |
| 32  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 33  | 1.000  | 0      | 0.800  | 0      | 1.000  | 0      | 0.833  | 0      |
| 34  | 1.000  | 0      | 0.800  | 0      | 1.000  | 0      | 1.000  | 0      |
| 35  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 36  | 0.667  | 0.333  | 0.500  | 0.167  | 1.000  | 0      | 1.000  | 0      |
| 37  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 38  | 0.600  | 0.250  | 0.333  | 0.250  | 1.000  | 0      | 1.000  | 0      |
| 39  | 0.833  | 0      | 0.500  | 0      | 1.000  | 0      | 1.000  | 0      |
| 40  | 1.000  | 0      | 0.429  | 0      | 1.000  | 0      | 1.000  | 0      |
| 41  | 1.000  | 0      | 0.800  | 0      | 1.000  | 0      | 0.800  | 0      |
| 42  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 43  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 44  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 45  | 1.000  | 0      | 0.500  | 0      | 1.000  | 0      | 1.000  | 0      |
| 46  | 1.000  | 0      | 0.667  | 0.182  | 1.000  | 0      | 1.000  | 0      |
| 47  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 48  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 49  | 1.000  | 0      | 0.833  | 0      | 1.000  | 0      | 1.000  | 0      |
| 50  | 1.000  | 0      | 0.800  | 0.100  | 1.000  | 0      | 1.000  | 0      |
| 51  | 1.000  | 0      | 0.600  | 0      | 1.000  | 0      | 0.800  | 0      |
| 52  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 53  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 54  | 1.000  | 0      | 0.800  | 0      | 1.000  | 0      | 0.857  | 0      |
| 55  | 1.000  | 0      | 0.750  | 0.250  | 1.000  | 0      | 1.000  | 0      |
| 56  | 1.000  | 0      | 0.800  | 0      | 1.000  | 0      | 1.000  | 0      |
| 57  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 58  | 1.000  | 0      | 0.800  | 0      | 1.000  | 0      | 1.000  | 0      |
| 59  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |
| 60  | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      | 1.000  | 0      |

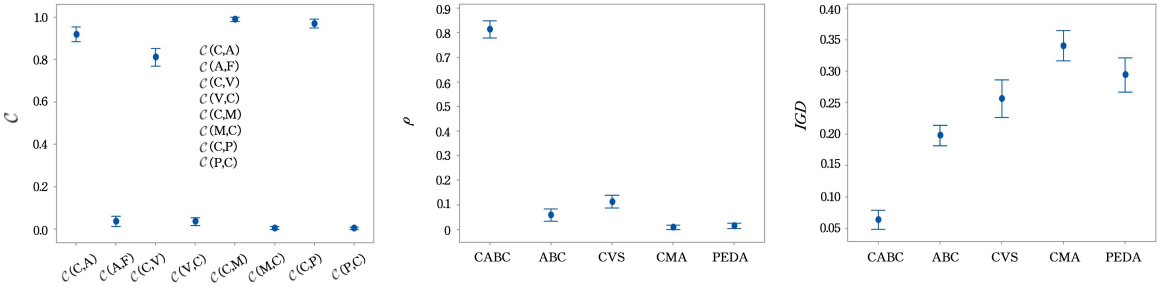


图 2 5 种算法在 95%置信区间下的均值图  
Fig.2 Mean plot of 5 algorithms at 95% confidence interval

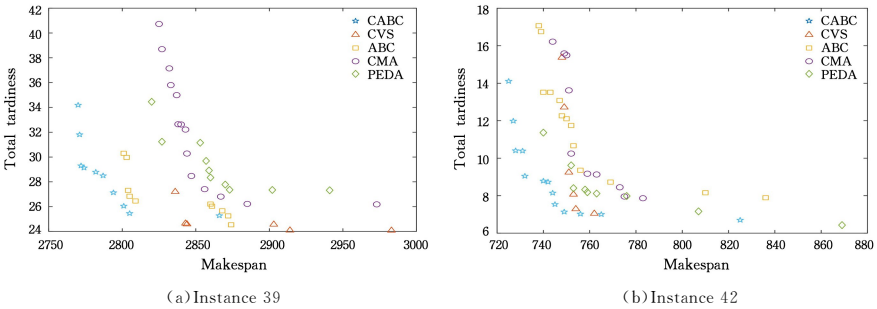


图 3 所有算法关于 2 个实例的非劣解分布图  
Fig.3 Non-dominated solution distribution graph for all algorithms about 2 instances

表 5 配对样本 t-检验的结果  
Table 5 Results of paired sample t-test

| t-检验             | IGD | $\rho$ | C |
|------------------|-----|--------|---|
| t-检验 (CABC,ABC)  | 0   | 0.001  | 0 |
| t-检验 (CABC,CVS)  | 0   | 0      | 0 |
| t-检验 (CABC,CMA)  | 0   | 0      | 0 |
| t-检验 (CABC,PEDA) | 0   | 0      | 0 |

表 1—表 4 还列出了 CABC 与 3 种对比算法之间的显著性能差异。CABC 关于 22 个实例所得的 IGD 值为 0;几乎关于所有实例都取得了对比算法更大的  $\rho$  值;关于大多数实例,  $C(C,V), C(C,M), C(C,P)$  均等于 1,这说明对比算法所产生的解均受 CABC 的非劣解支配。图 2 和图 3 也揭示了 CABC 与其对比算法之间的性能差异是显著的,而表 5 表明

上述显著性具有统计意义。总之,CABC 在求解模糊 DTHF-SP 方面具有较强的搜索优势。

**结束语** 本文针对考虑工厂适用性和附加资源的模糊 DTHFSP,通过引入蜂群间的竞争过程,设计了一种新型竞争人工蜂群算法以最小化最大完成时间和总一致指标。该算法将种群划分为两个蜂群,在两个蜂群间的竞争过程中,定义取胜蜂群为雇佣蜂群,失败蜂群为跟随蜂群,整个过程由雇佣蜂阶段和额外搜索、自适应跟随蜂阶段组成。给出了 5 种搜索策略并实现了蜂群间的差异化搜索。此外,还运用自适应种群重建避免蜂群间的过度竞争。运用 60 个实例进行仿真实验,结果表明,算法的新策略合理有效,算法在求解模糊 DTHFSP 方面具有较强的搜索优势。

在下一步工作中,将进一步探讨分布式调度,尤其是实际制造过程的分布式调度。由于智能优化算法在生产调度方面具有广泛的应用,积累了大量成果,未来将进一步探索如何将强化学习、机器学习等融合到智能算法中,以进一步提升智能算法的搜索性能。

参 考 文 献

[1] YAN Q,WU N Q,HE Y F,et al. Adaptive genetic algorithm for two-stage hybrid flow-shop scheduling with sequence-dependent setup time and no-interruption requirement[J]. Expert Systems with Applications,2022,208:118068.

[2] WANG M,ZHANG J,ZHANG P,et al. Independent double DQN-based multi-agent reinforcement learning approach for on-line two-stage hybrid flow shop scheduling with batch machines [J]. Journal of Manufacturing Systems,2022,65:694-708.

[3] HIDRI L,ELSHERBEENY A M. Optimal solution to the two-stage hybrid flow shop scheduling problem with removal and transportation times[J]. Symmetry,2022,14(7):1424.

[4] LIU H,ZHAO F Q,WANG L,et al. An estimation of distribution algorithm with multiple intensification strategies for two-stage hybrid flow-shop scheduling problem with sequence-dependent setup time [J]. Applied Intelligence, 2022, 53: 5160-5178.

[5] ZHANG S,HE Z W,YANG Z,et al. Effective upper and lower bounds for a two-stage reentrant flexible flow shop scheduling problem[J]. Computers and Operations Research, 2023, 153: 106183.

[6] KOMAKI G M,TEYMOURIAN E,KAYVANFAR V. Minimising makespan in the two-stage assembly hybrid flow shop scheduling problem using artificial immune systems[J]. International Journal of Production Research,2016,54:963-983.

[7] LANG S,REGGELIN T,SCHMIDT J,et al. NeuroEvolution of augmenting topologies for solving a two-stage hybrid flow shop scheduling problem;a comparison of different solution strategies [J]. Expert Systems with Applications,2021,172:114666.

[8] FIGIELSKA E. A heuristic for scheduling in a two-stage hybrid flow shop with renewable resources shared among the stages [J]. European Journal of Operational Research,2014,236(2): 433-444.

[9] WANG S,LIU M. Two-stage hybrid flow shop scheduling with

preventive maintenance using multi-objective tabu search method [J]. International Journal of Production Research,2014,52(5): 1495-1508.

[10] WANG S J,WANG X D,CHU F,et al. An energy-efficient two-stage hybrid flow shop scheduling problem in a glass production [J]. International Journal of Production Research,2020,58(8): 2283-2314.

[11] WANG Y M,LEI D M. A feedback-based artificial bee colony algorithm for solving distributed two-stage hybrid flow shop scheduling[J]. Control Theory and Applications,2024,41(6): 1137-1146.

[12] 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 and Fuzzy Systems, 2020,38(3):3189-3199.

[13] XI B J,LEI D M. Q-learning teaching-learning optimization for distributed two-stage hybrid flow shop scheduling with fuzzy processing time[J]. Complex System Modeling and Simulation, 2022,2(2):113-129.

[14] DONG J,YE C M. Green scheduling of distributed two-stage reentrant hybrid flow shop considering distributed energy resources and energy storage system[J]. Computers and Industrial Engineering,2022,169:108146.

[15] LEI D M,WANG T. An improved shuffled frog leaping algorithm for the distributed two-stage hybrid flow shop scheduling [J]. Control and Decision,2021,36(1):241-248.

[16] CEI JC,LEI D M. Distributed two-stage flow shop scheduling with setup time[J] Computers Integrated Manufacturing Systems,2020,25(8):2170- 2179.

[17] XU T P,ZHAO F Q,ZHANG J L,et al. Distributed flexible flow-shop scheduling problem with time-of-use electricity tariffs constraint and its solving algorithm[J]. Computers Integrated Manufacturing Systems,2025,31(4):1299-1313.

[18] MRAIHI T,DRISS Q B,EL-HAOUZI H B. Distributed permutation flow shop scheduling problem with worker flexibility; review,trends and model proposition[J]. Expert Systems with Applications,2024,238:121947.

[19] ZHANG Z P,FU Y P,GAO K Z,et al. A learning-driven multi-objective cooperative artificial bee colony algorithm for distributed flexible job shop scheduling problems with preventive maintenance and transportation operations[J]. Computers and Industrial Engineering,2024,196:110484.

[20] LI R.WANG L,GONG W Y,et al. An evolutionary multitasking memetic algorithm for multi-objective distributed heterogeneous welding flow shop scheduling[J]. IEEE Transactions on Evolutionary Computation,2025,29(6):2287-2298.

[21] SHAO Z S,SHAO W S,CHEN J R,et al. A feedback learning-based selection hyper-heuristic for distributed heterogeneous hybrid blocking flow-shop flexible scheduling problem with flexible assembly and setup time[J]. Engineering Applications of Artificial Intelligence,2024,131:107818.

[22] QIN H,LI T,TENG Y,et al. Integrated production and distribution scheduling in distributed hybrid flow shops[J]. Memetic

Computing,2021,13(2):185-202.

[23] EDIS E B,OGUZ C,OZKARAHAN I. Parallel machine scheduling with additional resources: Notation, classification, models and solution methods[J]. European Journal of Operational Research,2013,230:449-463.

[24] GENG K,LIU L,WU Z. Energy-efficient distributed heterogeneous re-entrant hybrid flow shop scheduling problem with sequence dependent setup times considering factory eligibility constraints[J]. Scientific Reports,2022,12(1):18741.

[25] LI Y L,LI X Y,GAO L,et al. A discrete artificial bee colony algorithm for distributed hybrid flow shop scheduling problem with sequence-dependent setup Times[J]. International Journal of Production Research,2020,59(13):3880-3899.

[26] LI Y L,LI X Y,GAO L,et al. An improved artificial bee colony algorithm for distributed heterogeneous hybrid flow shop scheduling problem with sequence-dependent setup times [J]. Computers & Industrial Engineering,2020,147:106638.

[27] LEI D M. Solving fuzzy job shop scheduling problems using random key genetic algorithm [J]. International Journal of Advanced Manufacturing Technology,2010,49:253-262.

[28] 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.

[29] 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.

[30] SHAO W S,SHAO Z S,PI D C. A Pareto-Based Estimation of Distribution Algorithm for Solving Multiobjective Distributed No-Wait Flow-Shop Scheduling Problem With Sequence-Dependent Setup Time[J]. IEEE Transactions on Automation Science and Engineering,2019,16(3):1344-1360.

[31] BOSMAN P, THIERENS D. The balance between proximity and diversity in multiobjective evolutionary algorithms [J]. IEEE Transactions on Evolutionary Computation,2003,7(2):174-188.

[32] ZITZLER E, THIELE L. Multi-objective evolutionary algorithms: a comparative case study and the strength Pareto approach[J]. IEEE Transactions on Evolutionary Computation,1999,3(4):257-271.



**ZHENG Youlian**, born in 1972, Ph.D, associate professor. Her main research interest is intelligent optimization and scheduling.

(责任编辑:何杨)