求解阻塞混合流水车间调度的双层变异迭代贪婪算法

秦浩翔[1], 韩玉艳[1†], 陈庆达[2], 李俊青[3], 桑红燕[1]

(1. 聊城大学 计算机学院，山东 聊城 252000；2. 东北大学 流程工业综合自动化国家重点实验室，沈阳 110004；3. 山东师范大学 信息科学与工程学院，济南 250014)

摘 要: 混合流水车间调度是制造业领域的前沿方向,而研究带有阻塞约束的问题更具有现实意义. 针对阻塞混合流水车间调度问题(BHFSP),以最小化最大完工时间为优化目标建立BHFSP的数学模型并详细阐述其计算过程,在零缓冲区特性的基础上设计一种双层变异策略的迭代贪婪(IGDLM)算法求解BHFSP. 分析传统迭代贪婪(IG)算法中的优势和不足,针对阻塞特性提出双层变异策略来提高解的多样性,进一步平衡所提算法的全局探索和局部搜索能力. 通过100个测试算例的数值仿真以及与5种代表算法的统计比较,验证所提出的双层变异策略与IG融合的算法能够得到更好的目标值,并为中大规模的BHFSP提供更优的调度方案.

关键词: 阻塞；混合流水车间调度；迭代贪婪算法；双层变异；最大完工时间

中图分类号: TP301 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2021.0607 **开放科学(资源服务)标识码(OSID):**

引用格式: 秦浩翔,韩玉艳,陈庆达,等. 求解阻塞混合流水车间调度的双层变异迭代贪婪算法[J]. 控制与决策, 2022, 37(9): 2323-2332.

A double level mutation iterated greedy algorithm for blocking hybrid flow shop scheduling

QIN Hao-xiang[1], *HAN Yu-yan*[1†], *CHEN Qing-da*[2], *LI Jun-qing*[3], *SANG Hong-yan*[1]

(1. School of Computer，Liaocheng University，Liaocheng 252000，China；2. The State Key Laboratory of Synthetical Automation for Process Industries，Northeastern University，Shenyang 110004，China；3. School of Information Science and Engineering，Shandong Normal University，Jinan 250014，China)

Abstract: Hybrid flow shop scheduling (HFSP) is a frontier direction in the manufacture field, and it is of practical significance to study the problem with blocking constraints. Aiming at the blocking hybrid flow shop scheduling problem (BHFSP), this paper establishes the mathematical model of the BHFSP with the objective of minimizing the makespan and describes its calculation process in detail. Based on the zero buffer characteristic, an iterative greedy algorithm based on a double level mutation (IGDLM) algorithm is designed to solve the BHFSP. This paper analyzes the advantages and disadvantages of the traditional IG algorithm, and a double level mutation strategy based on the blocking characteristics is proposed to improve the diversity of the solution, balancing the global exploration and exploitation abilities of the proposed algorithm. The 100 test instances numerical simulations and statistical comparison with five representative algorithms show that the proposed algorithm integrating the double level into the IG algorithm is able to get a better objective value, and provides a better scheduling scheme for the medium and large-scale BHFSP than the compared algorithms.

Keywords: blocking; hybrid flow-shop scheduling problem; iterated greedy algorithm; double-level mutation; maximum completion time

0 引 言

调度问题是影响制造业生产效率的关键,建立合理的计算模型并设计高效的调度优化方法是提高制造业生产效率、降低生产成本的重要途径[1]. 流水车间调度问题(flow-shop scheduling problem, FSP)是众多制造业普遍存在的一类优化问题[2],其包含着复杂的约束条件,既要考虑工件加工的先后顺序,又要协调工件的完工时间,因此,FSP是一类复杂的组合优

收稿日期: 2021-04-11；**录用日期:** 2021-06-17.

基金项目: 国家自然科学基金项目(61803192, 61973203, 61966012, 61773192, 61603169, 61773246, 71533001, 62173216)；山东省高校青年创新人才引进与教育项目.

责任编委: 王凌.

†通讯作者. E-mail: hanyuyan@lcu-cs.com.

化问题.

混合流水车间调度问题(hybrid flow-shop scheduling problem, HFSP)是一类比FSP更为复杂的组合优化问题,其涵盖FSP的所有特征,常见于炼钢-连铸[3-4]、化工[5]、微电子[6]等生产过程中. 与FSP不同,HFSP突破了加工机器的唯一性约束,即在任意加工阶段,每个工件都可以在给定机器集中的某台机器上进行加工. 因此,研究HFSP可以减少工件序列的完工时间,进而提高企业的生产效率.

自1973年Salvador[7]提出HFSP以来,流水车间调度与并行机调度相结合的问题开始受到广大学者的关注和研究. 为求解HFSP,学者们提出了不同的求解算法,主要包括精确算法、启发式算法和元启发式算法3类[8]. 精确算法如分支定界法[9]、动态规划算法[10]可以在小规模且简单的问题上求得精确解,但由于HFSP属于非确定性多项式难题,最优解的搜索过程涉及解空间的组合爆炸,所以其难以求解大规模复杂问题. 启发式算法如NEH(Nawaz, Enscore and Ham)算法[11]、MinMax-NEH算法[12]在求解复杂问题时优化效果不如元启发式算法,但它们常被嵌入到元启发式算法中产生初始解.

在元启发式算法中,例如遗传算法(Genetic algorithm, GA),是求解大规模非线性问题的有效算法[13-14]. 为解决HFSP,文献[15]考虑低碳指标, 提出了一种新型蛙跳(shuffled frog leaping algorithm, SFLA)算法优化总延迟时间和总能耗;文献[16]建立了描述空间问题的概率模型,并基于该模型通过采样产生新个体,更新概率模型的参数;文献[17]针对HFSP的NP-hard特性,提出了一种变邻域改进遗传(improved genetic algorithm-variable neighborhood search, IGA-VNS)算法;文献[18]研究了以总流经时间为目标的批量流HFSP,提出了两种竞争机制并分别用于提高在群体中更好解的概率和增强两条线之间的相互作用;文献[19]以最小化最大完工时间为目标,提出了一种新的前向译码与后向译码相结合的混合表示法,并引入一个新的控制参数来平衡全局探索与局部开发;文献[20]通过引入调度规则和构造性启发式算法对PSO(particle swarm optimization)算法的初始解进行改进,将变邻域搜索算法与PSO算法相结合,以减少计算量.

考虑到不同阶段机器之间存在有限或者无限的缓冲区,上述有关HFSP的研究在某一阶段加工完工件后,将其放入缓冲区等待下游的空闲机器,但未考虑实际生产成本以及机器之间无缓冲区的约束限制,导致加工完的工件或产品因无缓冲区限制被阻塞在当前机器上,延长了工件的等待时间及整个工件序列的完工时间,降低了整个工件序列的加工效率. 由此可知,对于不同规模的工件序列,工件排序上的差异会造成不同程度的阻塞,找到一个最优的工件排序序列来减少无缓冲区所造成的阻塞问题,可以降低生产成本,提高制造业的生产效率. 鉴于此,本文研究带阻塞约束的HFSP,这是研究问题的动机.

针对带阻塞约束的HFSP,即BHFSP(blocking hybrid flow-shop scheduling problem),调整序列中部分工件的加工顺序,有可能缩短工件被阻塞在机器上的时间,从而降低完工时间. 如图1所示的两个不同的调度序列,加工顺序分别为$1 \to 2 \to 3 \to 4 \to 5$, $1 \to 2 \to 4 \to 3 \to 5$,完工时间分别为24和22,局部工件3与4的排列顺序不同,所造成的完工时间也不同. 故在较好解的基础上加强可行域的局部搜索能力有助于找到更好的解,进而减少因阻塞延长的完工时间,提高工件的加工效率. 迭代贪婪(iterated greedy, IG)算法通常使用启发式算法获得较好解,在此基础上,通过破坏和重构策略加强对较好解所在局部邻域的搜索能力,已被成功用于求解HFSP优化问题[21].

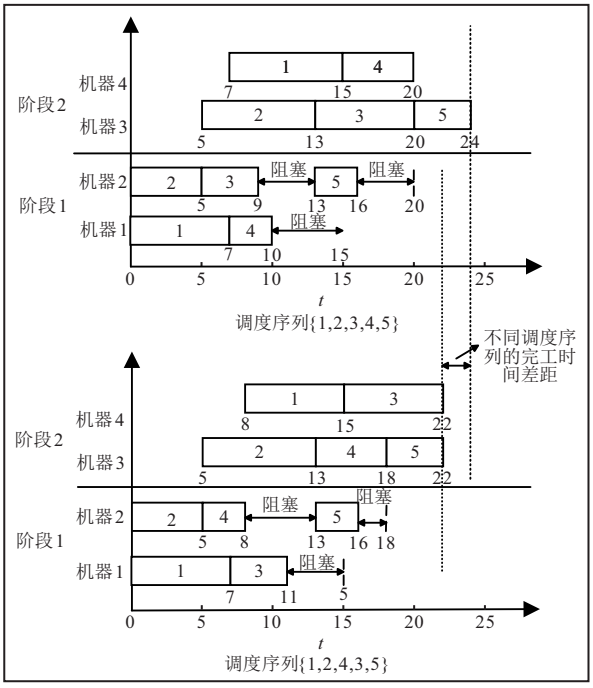

图1 不同调度序列比较

在求解FSP上,IG表现出了较优的性能,文献[22]为解决无等待的FSP,提出了一种基于禁忌的重构策略来增强IG的搜索能力;文献[23]以最小化总加权完工时间为目标,使用了IG算法求解不相关平行机的大规模FSP;文献[24]将传统IG算法分为两

个阶段,解决了柔性作业车间调度(flexible job shop scheduling problem, FJSP)的排序和路由子问题.

以上研究均使用了IG算法求解流水车间的相关调度问题,其在局部邻域探索上展现了很好的效果. 然而,仅仅采用局部邻域探索难以使解在迭代过程中跳出搜索范围,降低了解的多样性. 传统IG算法使用模拟退火准则提高解的多样性,但实验效果并不明显[25]. 针对要解决的BHFSP, IG算法的LocalSearch策略在前期搜索中展现了较强的局部搜索性能,然而,当迭代一定次数后,继续使用局部搜索策略已经难以改善工件序列的阻塞状况,因此,需要设计一种新策略对后期解进行更大范围的探索. 文献[26]表明,遗传算法的变异策略在全局解的探索上具有较强的寻优能力,可以使解跳出局部搜索范围. 鉴于此,本文引入一种新策略替换IG算法中的LocalSearch策略,即基于双层变异策略的迭代贪婪算法(iterated greedy-double level mutation, IGDLM),通过该算法提高IG算法的全局探索能力,这是研究算法设计的动机.

从以上问题和算法设计两个角度出发,为求解无缓冲区的BHFSP,本文基于文献[27]建立优化问题的计算模型,紧接着结合BHFSP的特点设计一种基于交换操作的双层变异策略,所设计的策略可弥补IG算法在迭代后期全局探索上的不足,提高解的多样性,进一步减少阻塞约束所延长的完工时间.

1　阻塞混合流水车间调度问题

BHFSP的优化目标与制造业生产效率、产品的产量和品质要求以及加工机器的生产能力息息相关. 根据制造业实际生产优化问题的原则: "在规定的时间范围内获得的调度序列能够满足生产要求的原则、阻塞情况尽可能少的原则和完工时间尽可能短的原则,合理安排调度序列,优化控制BHFSP的加工先后次序,确保生产的顺利进行",总结为以下两点要求: 阻塞约束影响小和完工时间最小化. 本节首先从符号定义、决策变量、约束条件、优化目标4个方面描述BHFSP的计算模型,紧接着给出一个具体实例详细说明阻塞约束对工件完工时间造成的影响.

BHFSP可以描述为: n个工件要经过S个加工阶段,每一个加工阶段有并行但数目不一定等同的M_i台加工机器,机器数为$M_i \geqslant 1$且至少有一个加工阶段机器的数量满足$M_i > 1$. 在已知条件下,问题输入是待调度的工件序列,问题输出是工件序列的最大完工时间的最小值. 针对该问题,工件在加工时需要满足以下假设: 1) 机器和工件在0时刻均是可用的状态; 2) 每个工件有s个加工阶段,并且每个工件必须依次在每个阶段的某一台机器上进行加工; 3) 所有工件在所有机器上的加工时间是已知的; 4) 每台机器在同一时刻只能加工一个工件; 5) 每个工件在同一时刻只能被一台机器加工; 6) 机器之间不存在缓冲区,若某一工件加工完后下一阶段的机器全部都处于忙的状态,则该工件被阻塞在当前阶段的机器上,直至下个阶段某台机器忙状态解除; 7) 工件一旦被加工,其加工过程不能被中断或者抢占. 为了方便描述该问题,定义如下数学符号:

J: 工件集合$\{1, 2, \ldots, n\}$, n为总的工件数;

j: 工件序号,且$j \in J$;

S: S为阶段集合$\{1, 2, \ldots, L\}$, L为总加工阶段数;

s: 加工阶段编号, $s \in S$;

m: 机器编号;

M: 一个极大正数;

t_{jm}: 工件j在机器m上的加工时间;

M_s: 阶段的并行机数;

K_s: 阶段的并行机集合, $K_s = \{1, \ldots, m, \ldots, M_s\}$.

决策变量定义如下:

B_{sj}: 工件j在阶段s的开始时间;

E_{sj}: 工件j在阶段s的完工时间;

D_{sj}: 工件j在阶段s的释放时间;

$C_{\max}$: 工件序列的最大完工时间;

X_{jm}: 若工件j安排在机器m上加工,则等于1,否则等于0.

$Y_{jj'm}$: 若在机器m上,工件j先于工件j'加工,则$j < j'$,否则等于0.

优化目标为

$$\min\ C_{\max}. \tag{1}$$

约束条件为

$$\sum_{m \in K_s} X_{jm} = 1,\ \forall j \in J,\ s \in S; \tag{2}$$

$$E_{sj} = B_{sj} + \sum_{m \in K_s} (t_{jm} X_{jm}),\ \forall j \in J,\ s \in S; \tag{3}$$

$$D_{sj} = B_{s+1j},\ \forall j \in J,\ s \in \{1, 2, \ldots, S-1\}; \tag{4}$$

$$D_{sj} \leqslant B_{sj'} + M(3 - Y_{jj'm} - X_{jm} - X_{j'm}),\ \forall j \in J,\ j' \in J,\ j < j',\ s \in S,\ m \in K_s; \tag{5}$$

$$D_{sj'} \leqslant B_{sj} + M(2 + Y_{jj'm} - X_{jm} - X_{j'm}),\ \forall j \in J,\ j' \in J,\ j < j',\ s \in S,\ m \in K_s; \tag{6}$$

$$E_{sj} \leqslant D_{sj},\ \forall j \in J,\ s \in S; \tag{7}$$

$$D_{Lj} \leqslant C_{\max},\ \forall j \in J; \tag{8}$$

$$B_{sj} \geqslant 0,\ \forall j \in J,\ s \in S. \tag{9}$$

其中:式(1)表示优化的最大完工时间;约束(2)表示工件在任意加工阶段只能在一台机器上加工;约束(3)表示工件开始时间与完工时间的关系,即工件完工时间等于工件开始时间与其加工时间之和;约束(4)～(6)表示阻塞约束,约束(4)表示工件在下一阶段开始时间等于前一阶段的释放时间,成对约束(5)和(6)保证同一机器加工的非重叠性,即在同一机器上后一个加工工件的开始时间不少于前一个加工工件的释放时间;约束(7)保证工件只有在加工完成后才可释放;约束(8)表示最大完工时间的约束;约束(9)表示工件开始时间必须大于等于0.

本文采用基于整数工件排序的编码方式,为了更简略和直观地描述该数学模型,下面给出一个简单实例的甘特图(图2 BHFSP),以便理解.

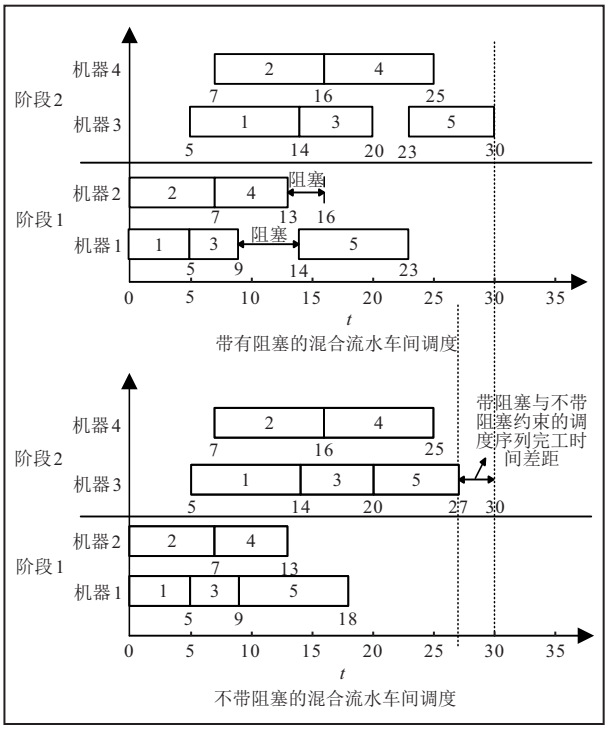

图 2 带阻塞与不带阻塞的混合流水车间调度对比

图2展示了带阻塞约束与不带阻塞约束的HFSP的甘特图. 在带阻塞约束的加工工厂里,工件3和工件4发生了不同程度的阻塞,其中工件3在时刻9发生了阻塞,在时刻14阻塞状态解除,阻塞时长为5,随后被送到下一个阶段的机器3上进行加工. 由于最大完工时间是由最后一个加工完毕的工件决定的,对比图2中不带阻塞约束的HFSP加工流程,工件3因在阶段1的9～14时间段里发生了阻塞,导致工件5在时刻23才开始第2阶段的加工,开始加工时间延长了$23-20=3$.

由以上描述可知,同一个调度序列在不同约束条件下的完工时间不同,带阻塞约束的HFSP完工时间明显大于不带阻塞约束的HFSP完工时间,阻塞约束的存在确实延长了工件的完工时间,并且随着工件规模的不断扩大,其阻塞对完工时间的影响也会随之加大. 因此已有的调度序列不再满足实际生产需求,此时就需要通过某些策略和方法对序列进行调整来减少阻塞约束对序列完工时间造成的影响,这是本文研究问题的基本特点和难点所在.

2 基于双层变异策略的迭代贪婪算法

由图2所示的实例可知,阻塞会导致后续工件的开始加工时间延后,拖慢了整体工件序列的完工时间,从而降低了工厂的生产效率. 对于传统IG算法而言,局部搜索策略和破坏重构策略都是基于插入操作实现的,每次操作只改变了调度序列中较少工件的相对位置,降低了解的多样性. 为此,本文采用基于交换的变异策略弥补IG算法在迭代后期搜索不充分这一缺陷,从而降低阻塞对完工时间的影响. 考虑到局部最优解和最优解的位置随工件序列的变化发生无规则运动,本文结合阻塞约束对工件调度带来的影响提出一种双层变异策略,从而改善IG算法在迭代后期对解探索上的不足.

2.1 初始化

对于BHFSP中工件序列的优化问题而言,工件序列可能因排列顺序不同和阻塞程度不同而导致序列完工时间的延长. 为了尽可能减少因排序和阻塞对完工时间造成的影响,采用合理的初始化策略来生成好的工件序列就变得十分重要. NEH算法在传统IG算法中通常被用于产生流水车间调度相关问题的初始解且已被验证是非常有效的启发式方法[11]. 因此,将NEH引入本文所提算法的初始化中来调整工件的排列顺序,进而减少阻塞对完工时间造成的影响. NEH启发式方法的步骤如下.

step 1:计算所有工件总加工时间$\mathrm{TP}_j=\sum_{s=1}^{S}p_{sj}$,$j\in\{1,2,\ldots,J\}$;按照$\mathrm{TP}_j$对各工件进行降序排列,得到初始序列$\pi^{\mathrm{origin}}=\{\pi_1,\pi_2,\ldots,\pi_J\}$.

step 2:取π^{origin}中前两个工件π_1和π_2,对其排序可得到序列$\{\pi_1,\pi_2\}$和$\{\pi_2,\pi_1\}$,将最大完工时间较小的序列作为当前调度序列,记为π^{new},并令$j=3$.

step 3:取π^{origin}中第j个工件π_j将其插入到π^{new}的所有可能位置,可得到j个不同的新序列,用最大完

工时间最小的新序列替换π^{new}.

step 4: 令$j = j + 1$. 如果$j \leqslant J$,则转到step 3; 否则,返回当前调度π^{new},初始化过程结束.

2.2　破坏和重构策略

对于BHFSP而言,对工件序列进行小范围的调整可以减少工件被阻塞在机器上的时间,进而缩短工件序列的完工时间. 该操作需要使用局部搜索能力强的策略对当前解的局部邻域进行探索,而破坏和重构策略通过改变部分工件的相对位置对解进行局部探索,具有较强的局部搜索能力. 因此,本文引入破坏和重构策略来调整部分工件的排列顺序,进而减少阻塞对完工时间造成的影响,达到提升解性能的目的. 破坏和重构策略操作步骤如下.

step 1: 对于初始化阶段产生的序列$\pi^{\text{new}} = \{\pi_1, \pi_2, \ldots, \pi_J\}$,从$\pi^{\text{new}}$中随机选取$d$个不同的工件组成序列$\pi^{\text{remove}}$.

step 2: 令$j = 1$,将π_1^{remove}插入到π^{new}所有可能的位置,共有$J - d + 1$种不同的插入方式. 将最大完工时间最小的序列作为当前调度序列$\pi^{\text{new}'}$,并令$j = j + 1$.

step 3: 取出π_j^{remove},将其插入到$\pi^{\text{new}'}$所有可能的位置,共有$J - d + j$种不同的插入方式. 将最大完工时间最小的序列作为当前调度序列$\pi^{\text{new}'}$.

step 4: 令$j = j + 1$. 如果$j \leqslant d$,则转到step 3; 否则,返回当前调度$\pi^{\text{new}'}$,破坏和重构策略结束.

2.3　双层变异策略

在BHFSP中,n个工件有$n!$种不同的排列方式,随着工件个数的不断增多,解空间的广阔性也会急剧扩大. 此外,序列的阻塞时间会随着工件排序的改变而发生无规律变化,而序列阻塞时间发生改变则会引起阻塞位置和全局最优解的位置发生无规律变化,从而使工件序列的最大完工时间发生无规律的改变. 然而,工件排列及阻塞位置与完工时间存在复杂的非线性关系,导致很难找到全局最优解所在的准确位置. 因此,在全局解的探索上需要设计跳跃性能更强的策略,通过改变更多不同工件的位置来减少序列的阻塞时间,扩大解的搜索邻域,进而缩短因阻塞延长的完工时间. 在传统IG算法中重构策略使用大量插入操作来改变工件的排列顺序,如果在重构策略后通过LocalSearch策略对工件序列执行插入操作,则改变的依旧是部分工件的相对位置,解的多样性难以得到提升. 因此设计基于交换操作的双层变异策略替换传统IG算法中的局部搜索策略. 一方面可以提高解的多样性,避免算法陷入局部最优; 另一方面,选取时间复杂度低且搜索效率高的变异策略可以增加迭代次数,使进化过程更加充分地进行. 此外,由于插入操作的时间复杂度为$O(n^3)$,如果算法的终止时间固定,则会耗费过多的时间,算法整体迭代的次数必然减少,从而降低了搜索到更优解的可能性. 为了弥补IG算法在迭代后期对全局探索的不足,提高解的多样性,搜索策略需要具有较强的全局搜索能力. 由文献[26]可知,GA算法的变异策略具有较强的全局搜索能力且使用交换操作产生解的时间复杂度更低($O(n^2)$). 因此,本文从以下两方面出发,设计基于交换操作的双层变异策略替换传统IG算法中的局部搜索策略. 一方面可以提高解的多样性,避免算法陷入局部最优; 另一方面,选取时间复杂度低且搜索效率高的变异策略可以增加迭代次数,使进化过程更加充分地进行. 具体操作如下.

设置变异概率P_m、随机数r_1和r_2,r_1、$r_2 \in (0, 1)$. 当$r_1 < P_m$时,采用简单的交换策略实施第1层变异过程; 当$r_2 < P_m$时,采用折半交换策略实施第2层变异过程.

第1层变异过程: 采用简单的交换策略,对于破坏重构操作产生的序列$\pi^{\text{new}'}$,随机选择两个工件a和b,且$a \neq b$,交换这两个工件,如果得到的新工件序列$\pi^{\text{new}''}$最大完工时间小于$\pi^{\text{new}'}$的最大完工时间,则替换$\pi^{\text{new}'}$作为新的工件序列进行同样的交换操作,直到交换的次数达到J,第1层变异过程结束.

第2层变异过程: 采用折半交换变异策略,对于由第1层变异策略得到的序列,将$\pi^{\text{new}''}$赋值给π,并将π的工件序列记为$\pi = \{\pi_1, \pi_2, \ldots, \pi_{J-1}, \pi_J\}$,有以下3个步骤.

step 1: 令下标p^{Front}指向序列π第1个位置的工件π_1,下标p^{Back}指向序列的最后一个位置的工件π_J,交换π_1和π_J,得到$\pi' = \{\pi_J, \pi_2, \ldots, \pi_{J-1}, \pi_1\}$,如果$\pi'$优于$\pi$,则$\pi = \pi'$,否则保持不变. 假设$\pi'$优于$\pi$,此时的序列变为$\pi = \{\pi_J, \pi_2, \ldots, \pi_{J-1}, \pi_1\}$.

step 2: p^{Front}后移一个位置,p^{Back}前移一个位置,交换$\pi = \{\pi_J, \pi_2, \ldots, \pi_{J-1}, \pi_1\}$中两个下标所指向的工件$\pi_2$和$\pi_{J-1}$,得到$\pi' = \{\pi_J, \pi_{J-1}, \ldots, \pi_2, \pi_1\}$,如果$\pi'$优于$\pi$,则$\pi = \pi'$,否则$\pi$保持不变.

step 3: 重复step 2的操作,直到满足$p^{\text{Front}} \geqslant p^{\text{Back}}$,第2层变异策略结束.

2.4　接受准则

执行以上过程后,需要判断是否将当前解作为下一次参与迭代的解,一般而言,最简单的接受准则是: 若当前解比初始解的完工时间更短,则用当前解替

换初始解,否则舍弃当前解. 此操作虽然简单易行,但在多样性的保持上效果并不明显. 因此,本文使用模拟退火(simulated annealing, SA)准则提升解的多样性[28],其思想为: 如果新生成解的性能不如当前解,则本文依然选择以一定概率$\Big($即random $\leqslant$ $\exp\{-(C_{\max}(\pi^{\text{current}}) - C_{\max}(\pi^{\text{origin}}))/\text{Temperature}\}$, $\text{Temperature} = T\cdot\Big(\sum\limits_{s=1}^{S}\sum\limits_{j=1}^{J}p_{s,j}\Big)\Big/10\cdot J\cdot S$, random $\in(0,1)$, $T\in(0,1)\Big)$接受当前解,而非直接用新解替换掉当前解.

3 仿真实验

本文所提算法基于Visual Studio 2019, C++实现,仿真在Windows10.0系统,处理器Intel Core i7, 2.60 GHz, 16 GB RAM的PC机上运行,待测试问题的工件数量取值范围是$J\in\{20,40,60,80,100\}$,工件加工阶段取值范围是$S\in\{5,10\}$,每个阶段中机器数的个数为$M_s\in\{1,5\}$,加工时间$p_{s,j}\in\{1,99\}$. 共有10种不同的组合方式($J\times S$),每种组合集包含10个不同的算例,共$5\times2\times10=100$个算例,经过多次预实验测试,将参数P_m、T设置为$P_m=0.1, T=0.5$.

为了分析本文所提的IGDLM算法的性能,验证其有效性,将该算法与传统IG算法[21]以及同样用于解决HFSP的GA算法[14]、有效的候鸟优化(effective migrating birds optimization, EMBO)算法[18]、离散人工蜂群(discrete artificial bee colony, DABC)算法[19]和改进离散粒子群优化(discrete particle swarm optimization, DPSO)算法[20]进行比较,其比较算法的参数设置如表1所示.

表1 算法的参数设置

算法	种群大小 P size	破坏工件数 d	交叉率 P_c	变异率 P_m	恒定因子 a	降温因子 T
IGDLM	1	4	/	0.3	/	0.5
IGA	1	3	/	/	/	0.5
GA	100	/	0.7	0.1	4	0.85
DABC	20	/	/	/	30	/
EMBO	25	/	/	/	10	/
DPSO	100	/	/	/	200	/

同时,为了公平起见,设置终止条件Timelimit $=J\times S\times$ CPU, CPU $\in\{5,10\}$,且一旦到达Timelimit规定的时间点,算法将终止运行. 仿真实验共有100个算例,每个算例重复运行5次.

3.1 参数的敏感度测试

破坏策略中参数d是影响算法整体性能非常关键的一个参数,其提取出来的工件数目d对平衡算法的局部搜索和全局探索以及协调算法的强化性和多样性起着至关重要的作用. 对于本文破坏重构策略而言,过高的d值会使算法的性能下降,耗费时间增多,但d的值如果过小,则会造成搜索范围过小且对解的开发力度不够. 为此,本文对参数d进行了敏感度测试,测试集为$d\in\{2,3,4,5,6\}$,用所有算例的最大完工时间的平均值作为测试结果,结果如图3所示.

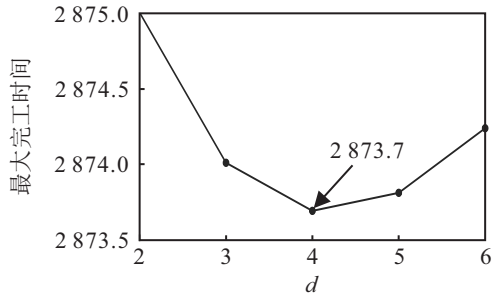

图3 不同d值的效应

由图3可知,当$d=4$时,最大完工时间的平均值最小,当$d<4$时,虽然算法可能具有较强的局部搜索能力,但对所有工件进行探索需要较长的时间,进而降低了算法的收敛速度. 当$d>4$时,虽然算法具有较快的收敛速度,但是局部搜索能力较差. 因此,本文选择$d=4$可以平衡算法的收敛速度和局部搜索能力.

在双层变异策略中,不同的变异概率P_m对解的性能的影响程度也不同,为了找到使目标值达到最优的调度序列,本文对P_m进行了敏感度测试,测试集设为$P_m\in\{0.1,0.2,0.3,0.4,0.5\}$,用所有算例的最大完工时间的平均值作为测试结果,结果如图4所示. 当$P_m=0.3$时,工件序列的最大完工时间值最小;当$P_m<0.3$时, 变异概率较低,工件序列改变的次数较少,所提策略对多样性的改进并不明显;当$P_m>0.3$时,变异概率较高,工件序列改变的次数过多,导致算法多样性和强化性的不平衡. 因此,本文选择$P_m=0.3$作为变异概率.

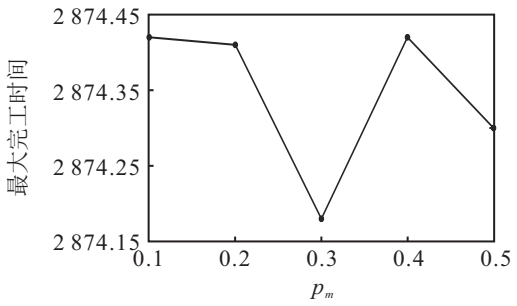

图4 不同P_m的效应

3.2 实验结果分析

为了验证所提双层变异策略的性能,表2(CPU $=5$)给出了不带双层变异策略(IGDLMN_DLM)和

带双层变异策略(IGDLM)所有算例得到的最大完工时间的均值(AVG)以及最小值(min). 此外,表2还给出了显著性水平为0.05的Wilcoxon Two-Sided Rank Sum检验,“†”和“‡”分别表示IGDLM的AVG值和min值优于IGDLMN_DLM,两者差异性明显. 在表2中:“$J \times S$”表示工件数和加工阶段数,如“20×5”表示20个工件,每个工件要经过5个加工阶段. 表2中加粗字体表示求解同一规模问题得到的最好值.

表 2　CPU = 5有无双层变异策略的最大完工时间

$J \times S$	IGDLMN_DLM		IGDLM	
	AVG	min	AVG	min
20×5	794.5	546	**794.1**	**545**
20×10	**1 350.2**	**1 218**	**1 350.2**	**1 218**
40×5	1 371.1†	904‡	**1 367.3**	**889**
40×10	**2 217.1**	**1 565**	2 217.7	**1 565**
60×5	**2 761.1**	**1 586**	**2 761.1**	**1 586**
60×10	3 184.2†	2 101‡	**3 180.3**	**2 076**
80×5	**3 404.5**	**1 484**	**3 404.5**	1 487
80×10	4 273.9	**3 758**	**4 272.9**	**3 758**
100×5	3 908.6†	1 532‡	**3 906.1**	**1 506**
100×10	5 502.5†	5 193‡	**5 487.6**	**5 163**
Total_AVG	2 876.77	1 988.7	**2 874.18**	**1 979.3**

由表2可知:针对10组的所有算例的min值,IGDLMN_DLM得到了5个最好的min值,而IGDLM得到9个最好的min值,因此,除在80×5规模的问题上IGDLM略大于IGDLMN_DLM的min值外,在其他规模的问题上均得到了最好的min值. 对于AVG值,IGDLM除在40×10规模的问题上比IGDLMN_DLM的AVG值仅仅大0.6之外,在其他规模的问题上均得到了最好的AVG值,而IGDLMN_DLM仅仅得到了4个最好的AVG值. 总之,由表2可知,IGDLM算法分别在所有规模上的最大完工时间的均值和最小值的平均值上均优于IGDLMN_DLM,其原因是带双层变异策略的IG算法在平衡解的多样性和强化性的同时也提升了在迭代后期的全局搜索能力,使解的性能得到进一步的提升.

表3(CPU = 5)和表4(CPU = 10)中的每一行数据(除最后一行)均为IGDLM算法与被比较算法在求解同一规模中10个算例所得到最大完工时间的均值(AVG)以及最小值(min). 此外,表3和表4给出了显著性水平为0.05的Wilcoxon Two-Sided Rank Sum检验,“†”和“‡”分别表示IGDLM的AVG值和min值优于被比较算法,两者差异性明显.

由表3和表4可知,IGDLM算法在求解所有问题中得到最大完工时间的AVG值均优于其他算法. 通过分析表3可知,IGDLM在规模为20×10的问题中求解最大完工时间得到的min值比DABC得到的min值多9,但在其他规模上均得到了最小值,此外,IGA可以得到4个规模分别为20×5、40×10、60×5和80×10问题的最小值,DABC可以在规模为20×10的问题上求得最小的最大完工时间值,其他被比较算法(GA、EMBO和DPSO)均没有得到最好的AVG值和min值. 由表4可知,IGDLM除在求解规模为60×10的问题中得到的min值不如EMBO外,在其他所有规模的问题上均能得到最好的min值. 此外,IGA可以在规模分别为20×5、40×10、60×5和80×10的问题上得到最好的min值,DABC可以在规模为20×10、60×5的问题上得到最好的min值,EMBO可以在规模为60×10的问题上得到1个最好的min值,而被比较算法GA和DPSO无法得到最好的min值. 最后,从表3和表4的计算结果来看,IGDLM算法在表3和表4中所有规模的Total_AVG都好于其他算法.

表 3　CPU = 5时所有算法所获得最大完工时间的AVG和min

$J \times S$	IGDLM		IGA		GA		DABC		EMBO		DPSO	
	AVG	min	AVG	min	AVG	min	AVG	min	AVG	min	AVG	min
20×5	**794.1**	**545**	794.2	**545**	799.3†	559‡	796.4	554‡	795.7	546	832.8†	589‡
20×10	1 350.2	**1 218**	**1 349.9**	1 218	1 360.1†	**1 218**	1 356.6†	**1 209**	1 358.6	**1 218**	1 477.7†	1 297‡
40×5	**1 367.3**	**889**	1 368.6	894	1 387.3†	923‡	1 399.4†	931‡	1 372.8†	903‡	1 482†	1 017‡
40×10	**2 217.7**	**1 565**	2 219.5	**1 565**	2 241.5†	1 585‡	2 263.7†	1 582‡	2 232.1†	1 581‡	2 403.7†	1 657‡
60×5	**2 761.1**	**1 586**	**2 761.1**	**1 586**	2 762.5	1 600‡	2 763.9	1 599‡	2 762.1	1 730‡	2 806.2†	2 340‡
60×10	**3 180.3**	**2 076**	3 186.5†	2 094‡	3 202.8†	2 114‡	3 244.7†	2 190‡	3 189.5†	2 071‡	3 331.7†	2 340‡
80×5	**3 404.5**	**1 487**	3 405.9†	1 492‡	3 417.6†	1 560‡	3 442.9†	1 627‡	3 407.9†	1 500‡	3 471.2†	1 564‡
80×10	**4 272.9**	**3 758**	4 277.2†	**3 758**	4 293†	3 767‡	4 380.8†	3 767‡	4 280.5†	3 767‡	4 414.7†	3 819‡
100×5	**3 906.1**	**1 506**	3 909.1†	1 527‡	3 919.2†	1 599‡	3 939.3†	1 660‡	3 913†	1 529‡	3 953.3†	1 666‡
100×10	**5 487.6**	**5 163**	5 507.6†	5 188‡	5 523.7†	5 201‡	5 760.7†	5 194‡	5 507.3†	5 194‡	5 800.1†	5 245‡
Total_AVG	**2 874.18**	**1 979.3**	2 877.96	1 986.7	2 890.7	2 012.6	2 934.84	2 031.3	2 881.95	2 003.9	2 997.34	2 153.4

表 4 CPU = 10时所有算法获得最大完工时间的AVG和min

$J \times S$	IGDLM		IGA		GA		DABC		EMBO		DPSO	
	AVG	min	AVG	min	AVG	min	AVG	min	AVG	min	AVG	min
20×5	794.1	**545**	**794**	**545**	799.3	559‡	795	551‡	795.7	546	825.6†	589‡
20×10	**1 349.9**	**1 218**	**1 349.9**	**1 218**	1 360.1†	1 218	1 354†	**1 209**	1 358.6†	**1 218**	1 448.9†	1 297‡
40×5	**1 365.5**	**888**	1367.3	893	1 387.3†	923‡	1 391.8†	923‡	1 372.8†	903‡	1 475.9†	1 017‡
40×10	**2 216.6**	**1 565**	2 217.9	**1 565**	2 241.5†	1 585‡	2 251.9†	1 582‡	2 232.1†	1 581‡	2 392.8†	1 610‡
60×5	**2 761.1**	**1 586**	**2 761.1**	**1 586**	2 762.5	1 600‡	**2 761.1**	**1 586**	2 762.1	1 593‡	2 801.3†	1 657‡
60×10	**3 178**	**2 074**	3 184.4†	2 094‡	3 202.8†	2 114‡	3 228.7†	2 168‡	3 189†	**2 067**	3 331.7†	2 340‡
80×5	**3 402.6**	**1 483**	3 404.2†	1 491‡	3 416.7†	1 560‡	3 433†	1 614‡	3 407†	1 492‡	3 471.2†	1 564‡
80×10	**4 272.7**	**3 758**	4 275.5†	**3 758**	4 291.4†	3 767‡	4 357.2†	3 764‡	4 280.5†	3 767‡	4 414.7†	3 819‡
100×5	**3 905.7**	**1 503**	3 908.3†	1 525‡	3 917.6†	1 584‡	3 934.2†	1 631‡	3 911.4†	1 514‡	3 952†	1 666‡
100×10	**5 485**	**5 163**	5 503.3†	5 188‡	5 519.4†	5 194‡	5 722.3†	5 194‡	5 502.4†	5 194‡	5 800.1†	5 245‡
Total_AVG	**2 873.12**	**1 978.3**	2 876.59	1 986.3	2 889.86	2 010.4	2 922.92	2 022.2	2 881.16	1 987.5	2 991.42	2 080.4

由以上实验结果可知,所提出的基于阻塞约束设计的双层变异策略在求解BHFSP优化问题上是有效的. 所提双层变异策略有效的原因可能是因约束引起的加工工件被阻塞在当前机器上延长了工件序列的最大完工时间,所提策略可以及时调整工件的位置,进而探索更广阔的解空间.

为了更直观地观测最优调度序列加工及阻塞状况,本文从100个测试算例中各选取20 × 5规模(TI_02)和40 × 5规模(TI_22)的1个算例,列出其最优调度序列表,如表5所示,当CPU等于10时,TI_02和TI_22算例的最大完工时间分别为768和1 211. IGDLM求解上述算例所得调度序列的甘特图如图5和图6所示.

甘特图中横坐标刻度表示最大完工时间,纵坐标刻度表示不同阶段上的各个机器编号,此外,以“工件编号-阶段”来标注每一个工件.

表 5 IGDLM求解TI_02和TI_22所得的最优调度序列及最大完工时间

算例	最优调度序列	最大完工时间
TI_02	20-12-7-1-13-15-5-9-10-3-2-4-6-11-17-19-8-14-18-16	768
TI_22	40-5-15-7-12-9-21-33-28-22-26-3-30-23-36-39-24-1-27-10-19-11-35-34-37-20-14-32-16-8-29-4-18-13-38-6-25-31-17-2	1 211

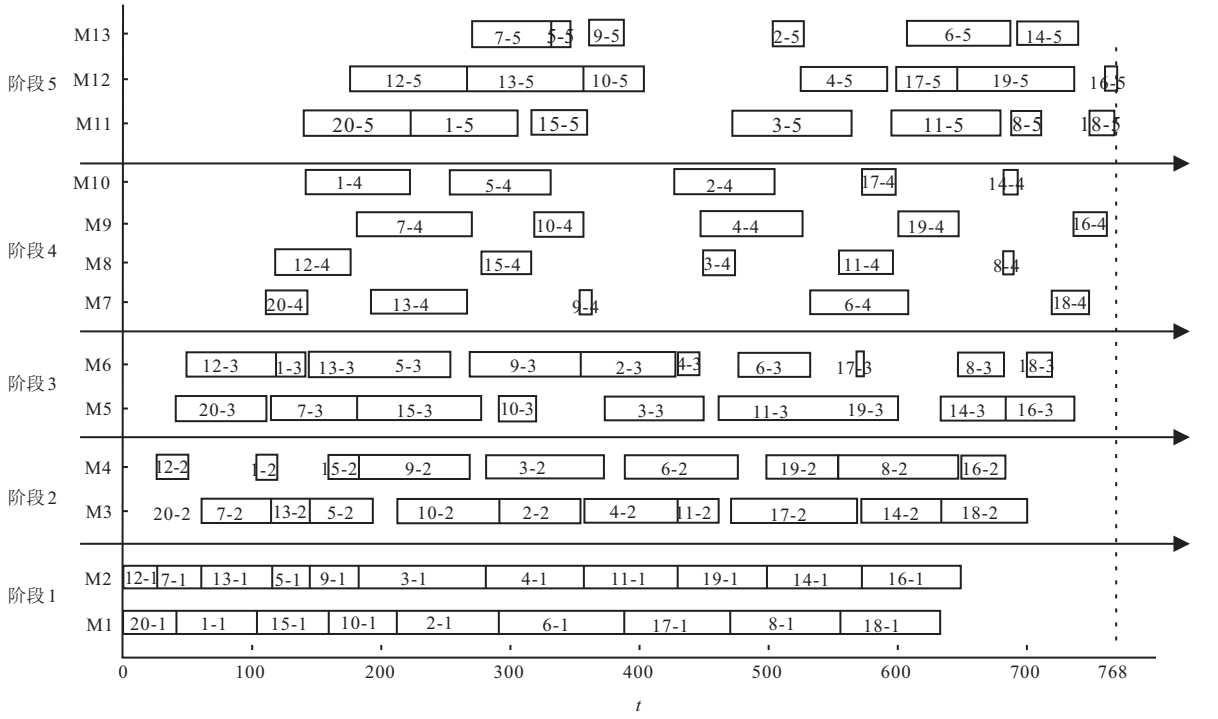

图 5 TI_02最优调度序列甘特图

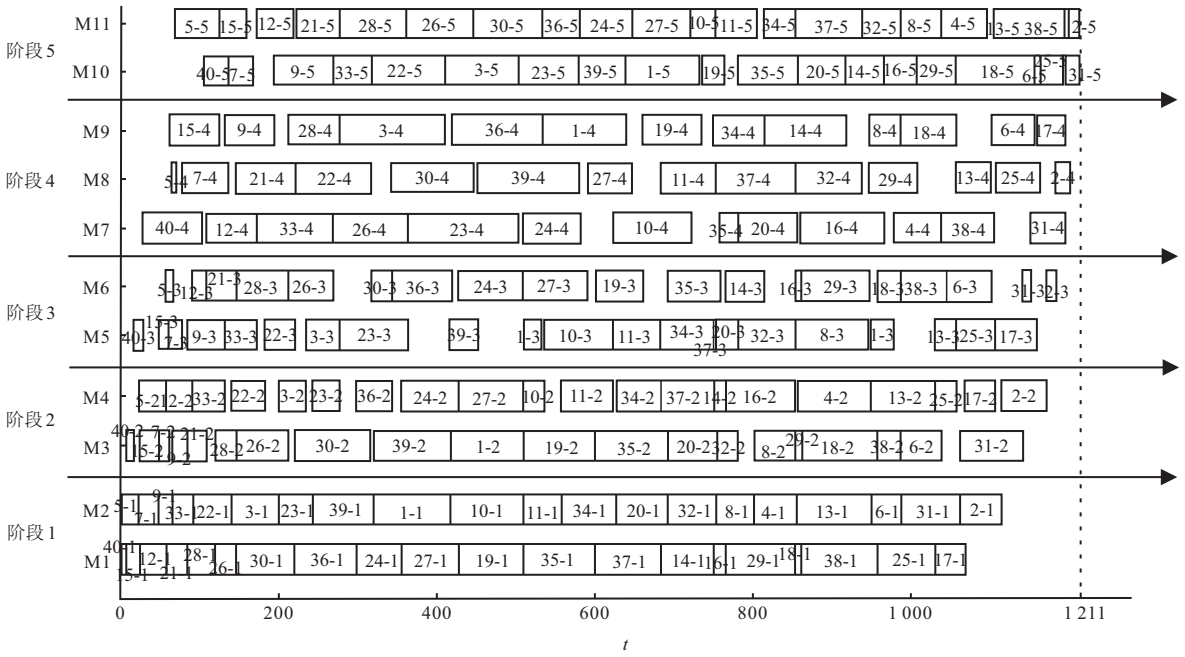

图 6　TI_22最优调度序列甘特图

4　结　论

带阻塞约束的HFSP是制造业亟需解决的重要问题,本文对带阻塞约束的HFSP(即BHFSP)展开了研究. 首先,对BHFSP进行了描述,构建了以最小化最大完工时间为优化目标的计算模型;随后,针对BHFSP特点设计了基于双层变异策略的迭代贪婪算法进行问题求解;最后,通过与其他解决相似问题的经典算法进行对比分析,验证所提算法在求解方面均有较强的性能. 未来将考虑阻塞约束对分布式流水车间调度和不确定性调度问题造成的影响,考虑制造业生产中的能耗对现实生活造成的影响,结合问题特点进行建模并设计新策略.

参考文献(References)

[1] 黄敏, 付亚平, 王洪峰, 等. 设备带有恶化特性的作业车间调度模型与算法[J]. 自动化学报, 2015, 41(3): 551-558.
(Huang M, Fu Y P, Wang H F, et al. Job-shop scheduling model and algorithm with machine deterioration[J]. Acta Automatica Sinica, 2015, 41(3): 551-558.)

[2] Reza Hejazi S, Saghafian S. Flowshop-scheduling problems with makespan criterion: A review[J]. International Journal of Production Research, 2005, 43(14): 2895-2929.

[3] 俞胜平, 柴天佑, 郑秉霖. 炼钢连铸混合智能优化调度方法及应用[J]. 系统工程学报, 2010, 25(3): 379-386.
(Yu S P, Chai T Y, Zheng B L. Hybrid intelligent optimal scheduling method for steelmaking and continuous casting and its application[J]. Journal of Systems Engineering, 2010, 25(3): 379-386.)

[4] 俞胜平, 柴天佑. 开工时间延迟下的炼钢-连铸生产重调度方法[J]. 自动化学报, 2016, 42(3): 358-374.
(Yu S P, Chai T Y. Rescheduling method for starting time delay in steelmaking and continuous casting production processes[J]. Acta Automatica Sinica, 2016, 42(3): 358-374.)

[5] 韩彪, 江永亨, 王凌, 等. 基于即时交货的离散时间模型及其在炼油过程调度优化中的应用[J]. 控制与决策, 2020, 35(6): 1361-1368.
(Han B, Jiang Y H, Wang L, et al. Instant delivery based discrete-time model and its application in refinery process scheduling optimization[J]. Control and Decision, 2020, 35(6): 1361-1368.)

[6] 张龙, 许川佩, 刘民, 等. 微电子生产过程调度问题基于指标快速预报的分解算法[J]. 控制与决策, 2020, 35(1): 139-146.
(Zhang L, Xu C P, Liu M, et al. An indexes fast prediction based decomposition method for scheduling problem in microelectronic production process[J]. Control and Decision, 2020, 35(1): 139-146.)

[7] Salvador M S. A solution to a special class of flow shop scheduling problems, symposium on the theory of scheduling and its applications[J]. Symposium on the Theory of Scheduling and Its Applications, 1973, 86: 83-91.

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

[9] Kis T, Pesch E. A review of exact solution methods for the non-preemptive multiprocessor flowshop problem[J]. European Journal of Operational Research, 2005, 164(3): 592-608.

[10] Xuan H, Tang L X. Scheduling a hybrid flowshop with batch production at the last stage[J]. Computers & Operations Research, 2007, 34(9): 2718-2733.

[11] Nawaz M, Enscore E E Jr, Ham I Jr. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem[J]. Omega, 1983, 11(1): 91-95.

[12] Ronconi D P. A note on constructive heuristics for the flowshop problem with blocking[J]. International Journal of Production Economics, 2004, 87(1): 39-48.

[13] 唐立新, 吴亚萍. 混合流水车间调度的遗传下降算法[J]. 自动化学报, 2002, 28(4): 637-641.
(Tang L X, Wu Y P. A genetic descent algorithm for hybrid flow shop scheduling[J]. Acta Automatica Sinica, 2002, 28(4): 637-641.)

[14] Nejati M, Mahdavi I, Hassanzadeh R, et al. Multi-job lot streaming to minimize the weighted completion time in a hybrid flow shop scheduling problem with work shift constraint[J]. The International Journal of Advanced Manufacturing Technology, 2014, 70(1/2/3/4): 501-514.

[15] 雷德明, 杨冬婧. 基于新型蛙跳算法的低碳混合流水车间调度[J]. 控制与决策, 2020, 35(6): 1329-1337.
(Lei D M, Yang D J. A novel shuffled frog-leaping algorithm for low carbon hybrid flow shop scheduling[J]. Control and Decision, 2020, 35(6): 1329-1337.)

[16] 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.)

[17] 崔琪, 吴秀丽, 余建军. 变邻域改进遗传算法求解混合流水车间调度问题[J]. 计算机集成制造系统, 2017, 23(9): 1917-1927.
(Cui Q, Wu X L, Yu J J. Improved genetic algorithm variable neighborhood search for solving hybrid flow shop scheduling problem[J]. Computer Integrated Manufacturing Systems, 2017, 23(9): 1917-1927.)

[18] Zhang B, Pan Q K, Gao L, et al. An effective modified migrating birds optimization for hybrid flowshop scheduling problem with lot streaming[J]. Applied Soft Computing, 2017, 52: 14-27.

[19] Pan Q K, Wang L, Li J Q, et al. A novel discrete artificial bee colony algorithm for the hybrid flowshop scheduling problem with makespan minimisation[J]. Omega, 2014, 45: 42-56.

[20] Marichelvam M K, Geetha M, Tosun Ö. An improved particle swarm optimization algorithm to solve hybrid flowshop scheduling problems with the effect of human factors — A case study[J]. Computers & Operations Research, 2020, 114: 1-9.

[21] Ruiz R, Stützle T. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem[J]. European Journal of Operational Research, 2007, 177(3): 2033-2049.

[22] Ding J Y, Song S J, Gupta J N D, et al. An improved iterated greedy algorithm with a Tabu-based reconstruction strategy for the no-wait flowshop scheduling problem[J]. Applied Soft Computing, 2015, 30: 604-613.

[23] Rodriguez F J, Lozano M, Blum C, et al. An iterated greedy algorithm for the large-scale unrelated parallel machines scheduling problem[J]. Computers & Operations Research, 2013, 40(7): 1829-1841.

[24] Al Aqel G, Li X Y, Gao L. A modified iterated greedy algorithm for flexible job shop scheduling problem[J]. Chinese Journal of Mechanical Engineering, 2019, 32(1): 1-11.

[25] Huang J P, Pan Q K, Gao L. An effective iterated greedy method for the distributed permutation flowshop scheduling problem with sequence-dependent setup times[J]. Swarm and Evolutionary Computation, 2020, 59: 1-18.

[26] 宋存利. 求解混合流水车间调度的改进贪婪遗传算法[J]. 系统工程与电子技术, 2019, 41(5): 1079-1086.
(Song C L. Improved greedy genetic algorithm for solving the hybrid flow-shop scheduling problem[J]. Systems Engineering and Electronics, 2019, 41(5): 1079-1086.)

[27] 孟磊磊, 张超勇, 任彩乐, 等. 求解带有阻塞限制的HFSP的MILP模型与改进回溯搜索算法[J]. 中国机械工程, 2018, 29(22): 2647-2658.
(Meng L L, Zhang C Y, Ren C L, et al. MILP models and an improved BSA for hybrid flow shop scheduling problems with blocking[J]. China Mechanical Engineering, 2018, 29(22): 2647-2658.)

[28] Osman I, Potts C. Simulated annealing for permutation flow-shop scheduling[J]. Omega, 1989, 17(6): 551-557.

作者简介

秦浩翔 (1997−), 男, 硕士生, 从事智能优化方法与调度的研究, E-mail: 987352978@qq.com;

韩玉艳 (1985−), 女, 副教授, 博士, 从事智能优化方法与调度、多目标优化、动态调度等研究, E-mail: hanyuyan@lcu-cs.com;

陈庆达 (1988−), 男, 讲师, 博士, 从事计算智能和工艺流程、 计算智能及其在工业领域的应用的研究, E-mail: cqd0309@126.com;

李俊青 (1976−), 男, 教授, 博士生导师, 从事智能优化方法与调度、多目标优化、智能建筑优化等研究, E-mail: lijunqing@lcu-cs.com;

桑红燕 (1981−), 女, 教授, 博士, 从事智能优化方法与调度、多目标优化等研究, E-mail: sanghongyan@lcu-cs.com.

(责任编辑：闫 妍)