文章编号:1006-2467(2022)02-0201-13 DOI: 10.16183/j.cnki.jsjtu.2020.435

基于改进候鸟迁徙优化的多目标批量流混合流水车间调度

汤洪涛， 王丹南， 邵益平， 赵文彬， 江伟光， 陈青丰
（浙江工业大学 机械工程学院，杭州 310023）

摘 要：针对 2＋1＋1 型混合流水车间，研究了多目标不相等批量流混合流水车间调度问题，提出一种基于变邻域搜索的自适应候鸟迁徙优化(AMBO)算法，实现了最小化完工时间与最小平均在制品数量的多目标优化.相比原始候鸟迁徙算法，AMBO 算法引入变邻域搜索策略，实现每个算子的权重随迭代次数自适应调整，并提出了时间窗算子，以提升交换算子搜索性能和收敛速度.对随机生成不同规模的订单进行算例研究，结果表明 AMBO 算法比候鸟迁徙优化算法、遗传算法具有更高的求解质量和收敛性能，从而验证了 AMBO 算法的有效性.

关键词：批量流问题；混合流水车间调度问题；变邻域搜索；自适应候鸟迁徙优化；时间窗算子
中图分类号：TH 166 **文献标志码**：A

A Modified Migrating Birds Optimization for Multi-Objective Lot Streaming Hybrid Flowshop Scheduling

TANG Hongtao， *WANG Dannan*， *SHAO Yiping*
ZHAO Wenbin， *JIANG Weiguang*， *CHEN Qingfeng*
(College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou 310023, China)

Abstract: This paper proposes an adaptive migrating birds optimization (AMBO) method based on variable neighborhood search to solve the inequal lot streaming hybrid flowshop scheduling problem (ILS-HFSP) for a 2＋1＋1 hybrid flowshop, which realizes multi-objective optimization of minimizing makespan and minimum average work in process. Compared with the original migrating birds optimization, the AMBO algorithm adopts the variable neighborhood search strategy with an adaptive selection probability of neighborhood operator that is adaptively adjusted with the number of iterations. Besides, a time-window operator is adopted to improve the search performance of exchange operators and convergence rate. Several orders of different scales generated randomly are studied, and the results show that the AMBO algorithm has a higher solution quality and a better convergence performance than the migrating birds optimization algorithm and the genetic algorithm, thereby verifying the effectiveness of the AMBO algorithm.

Key words: lot streaming problem; hybrid flowshop scheduling problem; variable neighborhood search; adaptive migrating birds optimization (AMBO); time window operation

收稿日期:2020-12-28
基金项目:国家重点研发计划(2018YFB1308100)，浙江省自然科学基金(LY19G020010)资助项目
作者简介:汤洪涛(1976-)，男，湖北省十堰市人，副教授，研究方向为生产与物流系统建模与优化、智能工厂规划.
通信作者:邵益平，男，讲师，电话(Tel.):13262619136；E-mail:syp123gh@zjut.edu.cn.

批量流混合流水车间调度问题(LS-HFSP)是批量流问题与混合流水车间调度问题(HFSP)的结合,广泛存在于炼钢、食品加工、制药等工业领域,近年来得到了学者的广泛关注[1-4]. 根据给定拆分子批大小是否相同,可以将批量流问题分为相等批量流问题与不相等批量流问题. 相等批量流混合流水车间调度问题是目前研究较多的 LS-HFSP 问题,相等批量拆分是解决批量流问题的最简单、最直接的方法,一旦确定了子批数量,就可以确定每个子批的大小. 但相等批量拆分只能优化子批的数量,无法优化每个子批的大小,并且在生产实际中大多为不相等批量流问题. 对于不相等批量流混合流水车间调度问题,不仅要优化加工顺序,还要优化子批的数量和大小,从而大大增加了计算的复杂度.

近年来,国内外已经有学者在 ILS-HFSP 领域做了大量研究,但这些工作具有一定的局限性,现有研究以单目标优化[5-10]为主. 文献[7]以最小化最大完工时间为目标,研究了柔性作业车间不相等批量拆分调度问题. 文献[8]以最小化总延迟时间为目标,研究了 K 阶段不可混流的 ILS-HFSP 问题. 文献[9]以最小化最大完工时间为目标,研究了只在最后阶段具有多台并行机的 K 阶段不可混流的 ILS-HFSP 问题. 文献[10]以最小化最大完工时间为目标,研究了考虑批量流与换模时间的柔性生产线调度问题.

目前关于多目标 ILS-HFSP 问题研究较少. 文献[11]针对存在随机扰动的相等批量流混合流水车间调度问题,提出了多目标候鸟迁徙算法,解决了最小化总加工时间与子批开工时间偏差的多目标优化问题. 文献[12]以减少平均停滞时间、能源消耗及提前和延迟造成损失为目标,研究了可变批次的批量流混合流水车间调度问题. 文献[13]以最小化最大完工时间与最小延期为目标,研究了多目标批量流水车间问题. 关于在制品的流水车间调度问题,研究也较少. 文献[14]提出使用过滤算法,研究了卷烟生产中的在制品堆积及有限产能的动态调度问题. 文献[15]以最优缓存区容量为目标优化生产排程,提高了瓶颈资源的利用率. 文献[16]研究生产机组的订单序列,建立了数学模型优化生产速率、在制品数量和生产周期 3 个目标.

ILS-HFSP 是个 NP 难问题,通常采用启发式算法来解决. 候鸟迁徙优化(MBO)算法是一种最新发展的启发式算法,由文献[17]在 2012 年首次提出. 该算法模拟以 V 形编队飞行的鸟类迁徙行为,每一只鸟都代表一种可行解. 在该算法中,鸟群中个体的数量、要考虑的邻域解数、跟随解共享的邻域解数以及领导解更换后鸟群的迭代次数是影响其性能的 4 个重要参数,具有较强的局部搜索能力和简单的结构,已成功应用于二次分配、流水车间调度、信用卡欺诈等不同的研究领域[6-7,18-20].

上述研究表明,对于以最小化在制品数量为目标的多目标 ILS-HFSP 的研究尚处空白,而减少在制品数量对于降低企业成本、提高企业的管理能力有着重要意义,也是实现精益生产的必由之路. 此外,目前研究以批量流混合流水车间调度理论研究为主,其机器阶段数和各阶段机器数量不固定,结合实际研究背景及固定形式的混合流水车间研究较少. 因此,本文以最小化完工时间与最小平均在制品数量为目标,结合工程实际研究了三阶段混合流水车间的多目标 ILS-HFSP,针对 MBO 算法各算子在算法运行不同阶段改进概率具有差异的特点,提出了一种基于变邻域搜索的自适应候鸟迁徙优化(AMBO)算法,并提出时间窗算子以加快算法的收敛速度. 本文研究可为以最小化在制品数量与最大化生产速率的多目标 ILS-HFSP 问题提供理论依据.

1 问题描述

1.1 问题背景

某铝锅制造企业 A 现有一条混合流水线,如图 1 所示. 其中:该生产线由拉伸段、内喷涂段与包装段组成,M_{11}、M_{12} 为拉伸段内的两台相同拉伸机,M_{21} 为内喷涂段内的一台内喷涂机,M_{31} 为包装段内的一台包装机,B_1、B_2 为缓存区. 两台拉伸机的产能均为 450 个/h,内喷涂段的产能为 900 个/h,包装段的产能为 800 个/h. 由于铝锅的型号规格众多,机器切换生产不同型号规格的产品时需要换模,每种型号规格的产品只配备一套模具以降低模具成本. 此外,产品的订单数量变化较大,排程方案需考虑将大订单拆分成若干个小订单以防止产线堵塞. 订单经排程后形成包含产线可执行信息的工单,产线依据工单顺序依次生产.

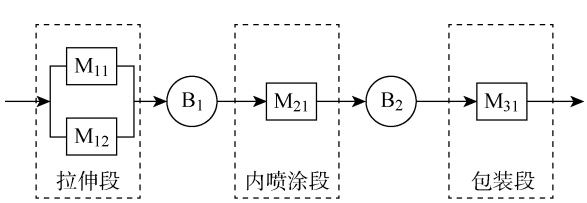

图 1 A 公司铝锅产线构型图

Fig. 1 Configuration diagram of the aluminum pot production line of A Company

该混合流水线的每台拉伸机与内喷涂机同一时刻只能生产一种产品，拉伸机与内喷涂机需加工完当前工单后才能切换生产下一个工单. 内喷涂机 M_{21} 一次只能生产一种产品，拉伸机 M_{11} 与拉伸机 M_{12} 生产的不同产品需要根据加工顺序依次进入内喷涂机 M_{21}，尚未进行内喷涂的产品需在缓存区 B_1 中等待. 内喷涂段生产速率大于包装段，缓存区 B_2 用以暂存未能及时包装的产品.

通过分析各机器生产速率可知，产线的生产瓶颈在于包装段，其最大生产速率总计为 800 个/h，而产线其他机器的产能有剩余. 此外订单的产品数量差别较大，生产小批量订单加上换模时间将使产线的生产速率达不到包装线的速率引起产能损失；若一直生产大批量订单又可能导致在制品堆积问题. 如何在最小化完工时间与在制品数量的两个目标下得到最优调度结果是本研究的主要内容. 本研究的基本假设如下：

(1) 原材料不存在缺料情况；

(2) 每台机器的处理时间为固定值，机器不会出现故障；

(3) 数量较大的订单会拆分成若干个数量不等的工单；

(4) 拉伸机与内喷涂机的换模时间为固定值；

(5) 包装线的设置时间非常短，可认定为无换模时间；

(6) 任何两个订单的产品规格型号都不相同，无法共用一套模具；

(7) 缓存区为无限容量缓存区.

1.2　问题建模

本文研究一个有 4 台机器的三阶段不相等批量流混合流水车间调度问题，该三阶段不相等批量流混合流水车间调度问题具体描述如下：

(1) 本文的研究对象是一个 2+1+1 型三阶段混合流水车间，由第 1 阶段的两台并行(相同)机器和第 2、3 阶段各一台机器组成. 在此混合流水车间中，机器换线生产不同订单的产品时需换模. 第 1 阶段与第 2 阶段、第 2 阶段与第 3 阶段之间各设有一个缓存区. 流水线形式及加工路线均满足 1.1 节所提出的背景要求.

(2) 在以 4 台机器组成的三阶段 2+1+1 型混合流水车间，根据加工顺序处理一个有 n 个订单的任务 J. 订单 $i(i=1,2,\cdots,n)$ 有 C_i^J 个同一规格的产品，C_i^J 大于订单的最小批量 $Q_{\min}$；每个订单将被拆成 N_i^S 个工单(子批)，每个工单的产品数量不同，记 S_{ij} 为订单 i 的第 $j(j=1,2,\cdots,N_i^S)$ 个工单，C_{ij}^S 为订单 i 的第 j 个工单的产品数量. 在此，C_{ij}^S 是必须优化的决策变量.

由问题背景可知，除包装段外其余机器换线时需要进行换模设置，产线的拉伸段生产速率与内喷涂段的生产速率相同且都大于包装段的生产速率. 为使两者产能匹配，存在一个最佳批量 Q，每个产品以这个批量进行加工能使拉伸机、内喷涂段加工时间加上换模时间与包装段的加工时间相匹配.

最佳批量 Q 只有等于瓶颈工序的最佳批量才能达到最大生产速率，记 k 为机器编号，$k=1,2,3,4$，分别为拉伸机 M_{11}、拉伸机 M_{12}、内喷涂机 M_{21}、包装机 M_{31}；t_k 为机器 k 的节拍；t_{c_k} 为机器 k 的换模时间. Q_k 为机器 k(包装机除外，$k=1,2,3$) 生产速率要达到包装机生产速率的最小加工批量，内喷涂机 M_{21} 的最低批量：$Q_3=\dfrac{t_{c_3}}{t_4-t_3}$，拉伸机 M_{11} 与拉伸机 M_{12} 机器完全相同，其加工节拍完全相等，因此拉伸机 M_{11}、M_{12} 的最低批量：$Q_1=Q_2=\dfrac{t_{c_1}}{t_4-\dfrac{t_1}{2}}$. 整条产线的最佳加工批量 $Q=\max\{Q_3,Q_1\}$.

根据上述问题描述以及定义的数学符号，建立如下数学模型：

$$\left.\begin{aligned} &t_{ij3}^{SA}>t_{ij1}^{SA}+C_{ij}^S(t_1-t_3),\\ &\qquad \text{当工单在拉伸机 1 加工时}\\ &t_{ij3}^{SA}>t_{ij2}^{SA}+C_{ij}^S(t_2-t_3),\\ &\qquad \text{当工单在拉伸机 2 加工时} \end{aligned}\right\} \tag{1}$$

$$t_{ijk}^{SA}\geqslant 0 \tag{2}$$

$$t_{ijk}^{F}=t_{ijk}^{SA}+C_{ij}^S t_k \tag{3}$$

$$t_{ij1}^{SA}\geqslant t_{i'j'1}^{F}+t_{c_1} \tag{4}$$

$$t_{ij2}^{SA}\geqslant t_{i'j'2}^{F}+t_{c_2} \tag{5}$$

$$t_{ij3}^{SA}\geqslant t_{i'j'3}^{F}+t_{c_3} \tag{6}$$

$$t_{ij4}^{SA}\geqslant t_{i'j'4}^{F} \tag{7}$$

$$D_{ijk}+D_{i'j''k}\leqslant 1,\quad k=1,2,3 \tag{8}$$

$$C_{ij}^S\geqslant Q_{\min} \tag{9}$$

目标函数：

$$f_1=\min(\max t_{ijk}^{F}) \tag{10}$$

$$f_2=\min N_a=\min\left(\frac{\sum_{t=1}^{t=\max t_{ijk}^{F}} N_W^t}{\max t_{ijk}^{F}}\right) \tag{11}$$

式中：t_{ijk}^{S} 为工单 S_{ij} 在机器 k 上的开始加工时间；t_{ijk}^{F} 为工单 S_{ij} 在机器 k 上的完工时间；记 $S_{i'j'}$ 为根据排程结果在相同机器上工单 S_{ij} 前一个工单；$t_{i'j'k}^{S}$ 为工单 $S_{i'j'}$ 在机器 k 上的开始加工时间；$t_{i'j'k}^{F}$ 为工单 $S_{i'j'}$

在机器 k 上的结束加工时间；N_{W}^{t} 为产线在 t 时刻产线中在制品数量；N_{a} 为从产线开始加工到结束加工期间内的平均在制品数量；D_{ijk} 为二进制变量，当工单 S_{ij} 分配到机器 k 上时为 1，否则为 0；$D_{i'j''k}$ 为二进制变量，当与工单 S_{ij} 不属同一订单的任一子批 $S_{i'j''}$ 分配到机器 k 上时为 1，否则为 0.

式(1)表示同一工单在内喷涂机上的开始加工时间与该工单在拉伸机上开始加工时间的约束；式(2)表示每个工单的开始加工时间必须大于等于 0；式(3)为机器 k 上的工单 S_{ij} 的开始加工时间与完工时间之间的关系；式(4)～(7)分别表示在拉伸机 1、拉伸机 2、内喷涂机、包装线上，下一个工单的开始时间与上一个工单完工时间的约束；式(8)表示不同订单子批次(工单)在拉伸机 1、拉伸机 2 和内喷涂机上是不可混流生产的；式(9)表示拆分后的每个工单的最小产品数；式(10)表示完成所有订单生产的最小化完工时间；式(11)表示产线的平均在制品数量最小化.

2 自适应候鸟迁徙优化算法

基本的 MBO 算法是针对二次分配这种连续函数优化问题提出的，不能直接用于处理 HFSP 这类离散的组合优化问题，需要转换成离散形式. 本文基于候鸟迁徙优化算法，结合 ILS-HFSP 问题，提出一种 AMBO 算法，包括批量拆分规则、编码与解码机制、邻域搜索策略、种群初始化、领导解优化、跟随解优化、领导解替换与算子权重自适应调整等，可有效求解 ILS-HFSP 问题.

2.1 算法具体定义

2.1.1 批量拆分规则 对于批量流问题，需要明确订单的拆分规则. 常见的订单拆分方式为给定一个随机数，对需要拆分的订单随机拆分成该数量个子订单[5]，随机数的选取范围需根据订单大小与问题特征具体问题具体分析. 本研究的批量大小不完全相同，拆分批量的上下限应根据每个订单的大小进行调整.

产品数量大的订单是引起在制品堆积的原因，由于本文订单存在一个最小批量 $Q_{\min}$，规定只有当订单产品数量 $C_i^J > 2Q_{\min}$ 时才能被拆分，所以大订单至少需要拆分成 $\left\lfloor \dfrac{C_i^J}{2Q_{\min}} \right\rfloor$ 个工单，最多能拆分成 $\left\lfloor \dfrac{C_i^J}{Q_{\min}} \right\rfloor$ 个工单.

因此，订单的拆分规则为：① 订单数量 $C_i^J < 2Q_{\min}$ 的订单不进行拆分，其工单数为 $N_i^{\mathrm{S}} = 1$；② 订单数量 $C_i^J > 2Q_{\min}$ 的订单从 $\left[\left\lfloor \dfrac{C_i^J}{2Q_{\min}} \right\rfloor, \left\lfloor \dfrac{C_i^J}{Q_{\min}} \right\rfloor\right]$ 中随机选取一个整数作为订单的工单数 N_i^{S}，并将其随机拆分成 N_i^{S} 个产品数量不相等工单.

2.1.2 编码与解码机制 ILS-HFSP 问题需同时优化订单拆分问题和作业调度问题，因此编码应同时考虑这两个方面，本文采用两阶段编码方案. 用自然数序列代表所有可能的工单排列，编码分为两段：第 1 段表示加工顺序；第 2 段表示待分配的工单编码，每个工单编码对应一个加工顺序编号. 每个工单编码由一个工单编号与其对应工单的数量组成，工单编号数为 4 位整数，工单数量为 5 位整数. 例如，工单编号为 101，数量为 3 000 的工单的工单编码为 010103000. 编码示例如图 2 所示.

1	2	3	4	5	010105400	010202400	010305140	010402800	010501300

图 2 编码示例

Fig. 2 Code sample

每个编码都代表订单拆分后的工单在内喷涂机上的排程结果，为了进一步得到每个工单在各机器上的加工时间序列，计算每个解的适应度值，需对编码进行解码. 由于拉伸机 1 与拉伸机 2 并行工作，每个工单只能在其中一台机器上加工，解码时需要考虑两个问题：① 所有工单在机器上的开始加工时间与完工时间；② 系统平均在制品数量. 其具体过程如下.

假设有 b 个工单，其在内喷涂机上的加工序列为 $\partial = (\partial_1, \partial_2, \cdots, \partial_b)$，即依次将工单 ∂_q 从 ∂ 中取出并执行如下步骤.

步骤 1 选取具有最先空闲时间的拉伸机，将工单 ∂_q 分配到该机器上，确定其在该拉伸机上的开始加工时间与完工时间；

步骤 2 重复步骤 1，直到完成所有工单的拉伸机分配；

步骤 3 根据加工序列与各工单在拉伸机上完工时间依次计算每个工单在内喷涂机上的开始加工时间与完工时间；

步骤 4 根据每个工单在内喷涂机上的完工时间，计算每个工单在包装线的开始加工时间与完工时间，得到加工完所有工单的最终完工时间；

步骤 5 根据解码所得到的各工单在其对应加工机器上的开始加工时间、结束加工时间、各机器的节拍与换模时间，计算流水车间开始加工到结束加

工的平均在制品数量.

2.1.3 适应度函数值　在得到解在各机器上的加工时间后，需要评估解的适应度值来完成领导解、跟随解的排序选择. 本文采用权重和法处理多目标优化问题，由于研究的两个目标具有不同量纲，需先进行目标归一化处理，具体过程如下所示：

$$\widetilde{f}_y = \frac{f_y - \min f_y}{\max f_y - \min f_y}, \quad y = 1,2 \tag{12}$$

式中：$y = 1,2$ 分别表示目标函数(10)与目标函数(11)；$\widetilde{f}_y$ 为目标 f_y 归一化值；$\max f_y$ 和 $\min f_y$ 分别为目标 f_y 的上界和下界. 通过这种方式，每种目标都可以归化为区间[0，1]之间的值. 目标的上界和下界是随着算法的运行实时更新，即每次产生新解，利用其目标值来更新上界和下界. 归一化后的适应度函数为

$$F_{\min} = \lambda_1 \widetilde{f}_1 + \lambda_2 \widetilde{f}_2 \tag{13}$$

式中：λ_1、λ_2 为 2 个目标函数的权重.

最小化总加时间与最小化平均在制品数量是本研究的两个目标，产线优化的首要目标是最大化生产速率使得企业效益最大化，而最小化平均在制品数量是为减少成本，属于次要目标，取权重 $\lambda_1 = 0.6$，$\lambda_2 = 0.4$.

2.2 改进策略

2.2.1 邻域搜索算子　MBO 算法是一种基于邻域搜索的元启发式算法，因此有必要为其确定高效的邻域搜索算子. 基于本文所采用的两阶段编码机制，从文献[21]中引入 4 种常见的邻域算子，分别为插入算子、贪婪插入算子、交换算子、贪婪交换算子. 交换算子计算量小但局部搜索能力有限；贪婪算子虽然能提高局部搜索的结果，但会极大增加计算量. 为平衡计算量与局部搜索结果，基于最佳批量原则本文提出一种时间窗算子，使其以较短的时间得到较优的解. 以上算子都是针对调度子问题，针对批量拆分子问题，本文采用批量调整算子，使各订单的工单数与工单的产品数随算法迭代调整.

(1) 时间窗算子.

由于工单产品数量各不相同，首先定义一个推进时间窗 $W = [t^{es}, t^{lf}]$，t^{es} 为在内喷涂机处任务的开始时间，t^{lf} 为在内喷涂机处任务的结束时间，其是预测时域内包含一定数量工单的时间窗口，而其平均宽度取 $2Qt_3$. 时间窗为 $2Q$ 的匹配度较小，因此增加一个宽放系数 γ，使时间窗的宽度取为

$$(2-2\gamma)Qt_3 \leqslant t^{lf} - t^{es} \leqslant (2+2\gamma)Qt_3 \tag{14}$$

式中：γ 为时间窗的宽放系数，常见的宽放系数为 0.05、0.1、0.2 等，为尽可能匹配较多的订单，取宽放系数为 0.2.

每次搜索的目标是从当前解中随机寻找拉伸机编号不相同的两个未被匹配时间窗的工单，对其进行时间窗匹配，形成一个新的邻域解，且满足如下订单关系式：

$$(2-2\gamma)Qt_3 \leqslant C^S_{\overline{ij}}t_3 + C^S_{\overline{i'j'}}t_3 + 2t_{c_3} \leqslant (2+2\gamma)Qt_3 \tag{15}$$

式中：$C^S_{\overline{ij}}$、$C^S_{\overline{i'j'}}$ 分别为工单 $S_{\overline{ij}}$ 与 $S_{\overline{i'j'}}$ 的产品数量. 时间窗算子的流程图如图 3 所示，其具体实施过程如下.

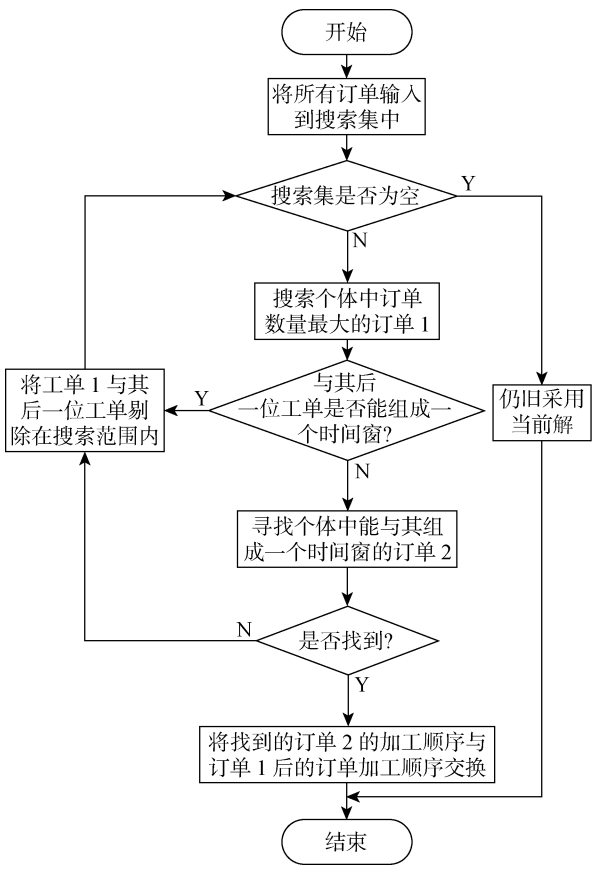

图 3　时间窗算子

Fig. 3　Time window operation

步骤 1　将当前鸟中的所有工单输入到一个搜索集中；

步骤 2　判断搜索集是否为空，若为空则采用随机交换操作形成邻域解并结束邻域搜索；

步骤 3　寻找搜索集中产品数量最大的工单；

步骤 4　判断该工单与其后一工单是否能满足时间窗约束，若满足时间窗约束则将找到的订单 2 加工序号与订单 1 后的加工序号交换，若不能则跳到步骤 5；

步骤 5　寻找个体中能与其满足时间窗约束的

工单，若无法找到，则将该工单从搜索集中剔除并重复步骤 1～5，若找到则将找到的工单与其后一工单位置进行交换形成新的解.

(2) 批量调整算子.

在工单编码段任选一个具有两个以上工单的订单，选择其中两个相邻的工单对其重新分配：随机选取其中一工单中随机数量的产品分配到另一工单中. 若重新分配后的工单数量低于最小批量 $Q_{\min}$，则以 50%概率将该工单全部分配到另一工单中，并将该工单所属订单的工单数减 1；以 50%的概率使该工单产品数量等于最小批量 $Q_{\min}$，并将其余数量的产品分配给另一工单. 例如，一个 1 300 与 4 400 的工单组合，随机调整成 1 600 与 4 100 的组合. 批量调整算子示意图如图 4 所示.

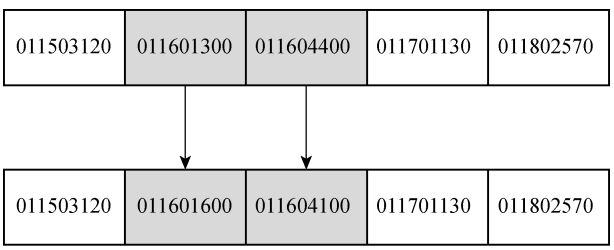

图 4 批量调整算子示意图

Fig. 4 Diagram of batch adjustment operation

上述邻域搜索算子分别针对不同的子问题，如果只采用其中一种很难找到全局最优解. 为了有效地联合两个子问题，提出了以下混合结构.

(1) 混合结构 1：首先执行插入算子，然后执行批量调整算子.

(2) 混合结构 2：首先执行贪婪插入算子，然后执行批量调整算子.

(3) 混合结构 3：首先执行交换算子，然后执行批量调整算子.

(4) 混合结构 4：首先执行贪婪交换算子，然后执行批量调整算子.

(5) 混合结构 5：首先执行时间窗算子，然后执行批量调整算子.

2.2.2 邻域算子权重自适应调整 本文采用了 4 种常见的邻域算子、时间窗算子、批量调整算子以及 5 种混合结构共计 11 种算子. 为有效利用所采用的 11 种不同邻域算子，采用变邻域搜索策略使各算子的邻域得到充分探索. 本文采用轮盘赌的方法，给每个算子以一个权重，在每次进行邻域搜索操作时根据每个算子权重随机选择邻域搜索算子得到邻域解.

邻域搜索的每个算子在算法的不同阶段对解的改进效果各不相同，例如在算法求解的初始阶段应优先选择全局搜索能力强的算子，在后期调整阶段应优先选择局部搜索能力强的算子. 因此本文引入自适应调整策略[22]，使每个邻域算子权重随算法迭代自动调整，其具体计算如下.

给每个算子赋值一个权重 w_O，其范围为[0,1]选择算子使用轮盘赌，每个算子被选择的概率 $P=\frac{w_O}{\sum_{O=1}^{11} w_O}$，在每经过一个权重调整周期 m 次后更新权重：

$$w_O=(1-r)w_O+\frac{r\pi_O}{\theta_O} \tag{16}$$

式中：π_O 为算子 O 经过 m 次迭代后的得分累计情况，其初始值为 0，若表现比最优解好，则取 $\pi_O=\pi_O+1$，对其余算子取原值；θ_O 为算子 O 经过一个权重调整周期时算子 O 使用的次数；r 为反应因子，控制权重调整规则对启发式方法的效率变化作出反应的速度与权重调整幅度. 如果 r 为 0，那么将完全不使用分数，而是保持初始权重. 如果 r 设置为 1，则在最后一代中获得的分数决定权重. 本文取 r 为 0.5，在反应速度与调整幅度之间保持平衡. 设每个算子的初始权重为 1，当算法完成一个迭代周期 m 后更新每个算子被选择的概率 P，并重新将算子的权重设为 1.

2.2.3 扰动机制 MBO 算法的邻域搜索机制过度关注了算法的局部搜索能力，为平衡算法的全局搜索和局部搜索能力，将扰动机制引入到 MBO 算法中. 记 g 为扰动机制触发周期，如果领导解连续 g 次迭代没有更新，则扰动机制触发. 在基本的扰动机制中，防止解陷入局部搜索的方法是用随机产生的新解将其替换. 完全随机的方法会影响到算法的执行效率，为提高算法的执行效率，在连续第 g 次领导解没有更新时，以领导解为种子，引入 Glover 操作产生两个解分别替换领导解的左右跟随解. Glover 操作将解序列拆分成 z 个子解序列，每个子序列以 z 为步长依次从原解序列中挑选组成，然后将 z 个子序列重新组合成扰动后的解序列. Glover 操作的优势在于可以从种子解进行发散产生新解，并且新解继承旧解的序列信息，有利于提高算法的执行效率，已被文献[5,23-24]等用于防止算法陷入局部最优解中，取得了良好的效果，并根据文献[5]采用 $z=3$ 作为 Glover 操作的参数.

2.3 AMBO 算法实现

AMBO 算法包含以下几个步骤：种群初始化、

领导解改进、跟随解改进、领导解替换与算子权重更新、循环迭代与扰动机制. 具体流程如图 5 所示.

步骤 1　初始化种群. 首先，设置 AMBO 算法的 6 个控制参数，包括鸟(解)群中个体的数量(α)，要考虑的邻域解数(β)，跟随解共享的邻域解数(χ)，领导解更换后鸟群的迭代次数(ω)，算子权重自适应调整周期(m)、扰动机制触发周期(g). 然后根据批量拆分规则对所需排程的订单任务拆分后对每个工单赋以随机加工顺序生成一个可行解，重复此过程生成 α 个解，选取适应度最优的解作为领导解，并将其余解随机安置在 V 形种群结构上形成初始种群.

步骤 2　领导解改进. 根据各算子的初始权重得到相应被选择概率，采用轮盘赌方法围绕领导解生成 β 个邻域解. 首先，对所得到的邻域解根据解码结果评估其适应度值，如果邻域解中最优解的适应度值优于领导解，那么领导解将被替换；否则，领导解保持不变. 然后，将未使用的邻域根据其适应度值选取最优的 2χ 个解，分别形成包含 χ 个邻域解的左、右共享邻域集.

步骤 3　跟随解改进. 改进过程是沿着尾巴进行的. 对于左(右)队列中的跟随解 (例如 X)，根据

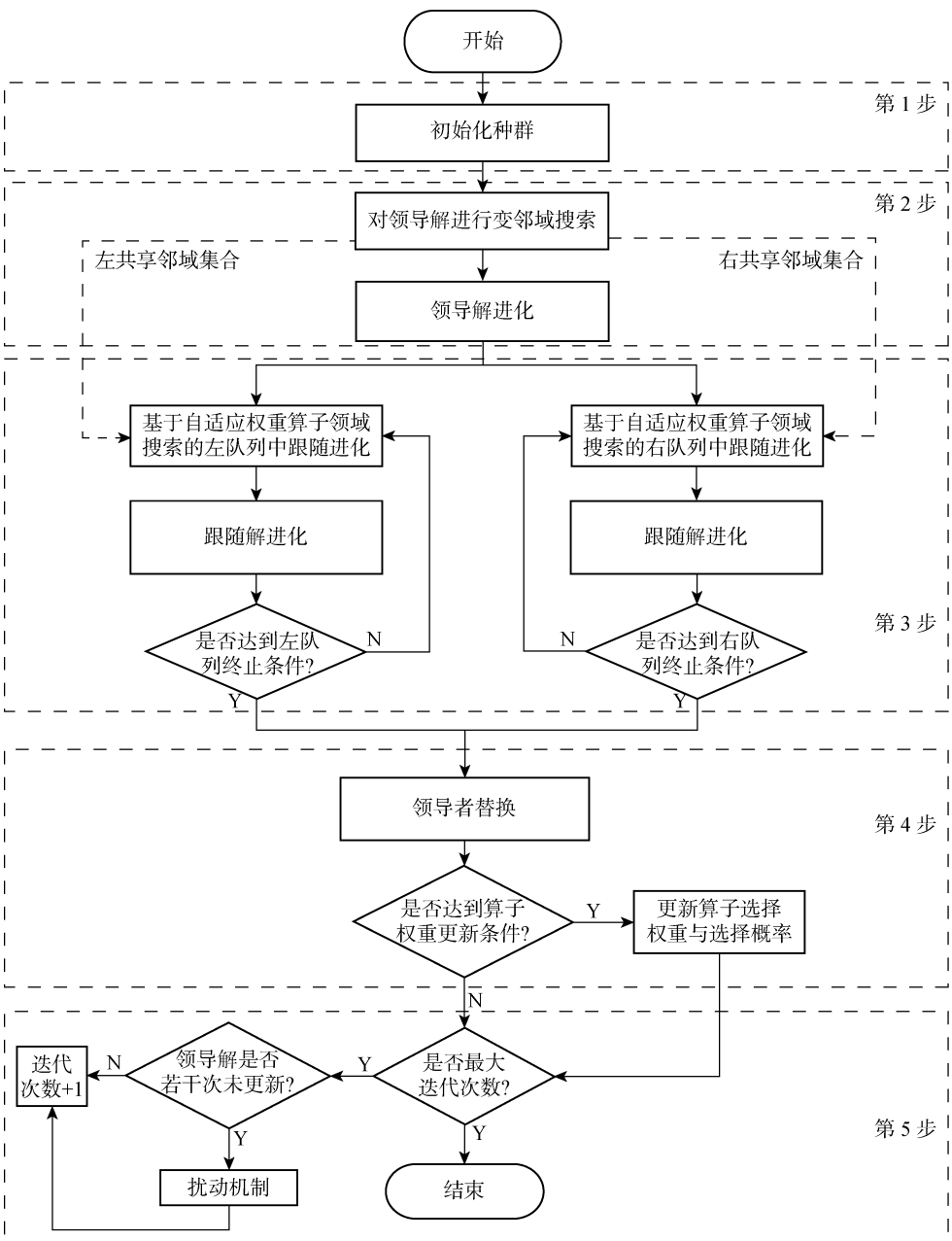

图 5　AMBO 算法流程图

Fig. 5　Flowchart of AMBO algorithm

各邻域算子权重以轮盘赌方式产生 β 个邻域解. 所产生的 β 个邻域解和左(右)共享邻域集中的解被视为 X 的邻域集,对所得的邻域集根据解码结果评估其适应度值. 如果最佳邻域解比 X 具有更好的适应性,则替换 X;否则将保持跟随解不变. 随后,在未使用的邻域中选择最优的 χ 个解,用来重建左(右)共享邻域集. 重复上述过程,直到遍历完左队列和右队列中的所有跟随解为止.

步骤 4 领导解替换与算子权重更新. 在执行第 3 步 ω 次后,若追随解中的最优解比领导解适应值更优,则将最优追随解与当前最优解相互交换以更新领导解,否则解将保持不变. 根据不同算子对解的改进情况更新其算子权重,并判断是否达到邻域搜索算子权重更新周期,若达到邻域搜索算子权重更新周期,依据邻域算子权重自适应调整式(16)分别更新每个算子的权重,并依据权重更新每个算子被选择的概率.

步骤 5 循环迭代与扰动机制. 为了防止算法陷入局部最优,采用 Glover 操作对 g 次迭代后未更新的领导解进行扰动,取两个新解分别替换领导解后的左、右跟随解. 算法在若干次迭代后如果达到最大迭代次数,则算法终止;否则,重复步骤 2~5 直到满足算法终止条件.

3 仿真实验研究

3.1 算例产生规则

A 企业的订单统计分析后具有以下特征,产品数量在 3×10^3 以下、$3\times10^3\sim1\times10^4$ 与 $1\times10^4\sim2\times10^4$ 的订单分别占据其所有订单数的 80%、15% 与 5%,产品总数量分别占所有订单的产品数总和的 50%、30%与 20%. 出于 A 企业接单能力、成本、效率等因素的考虑,最大订单数量不超过 2×10^4,最小订单数量不小于 10^3,即 $Q_{\min}=10^3$. 设定所有订单的产品数量都能被 10 整除,订单的数量特征如表 1 所示.

表 1 订单产品的数量特征

Tab. 1 Quantity characteristics of product orders

产品数量范围	订单数比例/%	产品总数比例/%
$1\times10^3\sim3\times10^3$	80	50
$3\times10^3\sim1\times10^4$	15	30
$1\times10^4\sim2\times10^4$	5	20

文献[5]指出,目前关于批量流水车间调度问题没有基准算例,为此本文根据表 1 的订单产品数量特征,随机产生不同规模的订单任务作为算例来评估算法性能. 各机器的生产速率以及问题背景保持一致:拉伸机 1 与拉伸机 2 的生产速率为 4.5×10^2 个/h,内喷涂机的生产速率为 9×10^2 个/h,包装段的生产速率为 8×10^2 个/h. 拉伸机 1、拉伸机 2 与内喷涂机的换模时间都为 30 min,可以得到最佳加工批量 $Q=3.6\times10^3$.

3.2 参数设置

参数设置通常在元启发式算法的有效性中起到重要作用. 为了得到 AMBO 算法的最优参数,采用实验设计(DOE)方法设计了田口实验. AMBO 算法中有 6 个控制因素,包括了 α、β、χ、ω、m、g. 由于 $\beta\geqslant2\chi+1$,所以参数 β、χ 不满足实验设计要求. 本文采用文献[17]建议的参数 $\beta=3$、$\chi=1$,将 α、ω、m、g 这 4 个参数作为实验因子,以一组订单数量为 20 的 30 个随机案例的平均值来测试在算法运行 100 s 时的平均最优结果值,通过 Minitab17 设计并运行分析最佳参数配置.

参考文献[5-6]中的因子水平,设计了 4 因子 3 水平的 L9 型正交田口实验,其各因子的 3 个水平取值分别为 $\alpha\{\alpha1=25,\alpha2=51,\alpha3=81\}$、$\omega\{\omega1=5,\omega2=10,\omega3=15\}$、$m\{m1=5,m2=10,m3=15\}$、$g\{g1=10,g2=20,g3=30\}$,表 2 为所设计的 L9 正交实验及相应的实验平均值. 在 Minitab17 中分析后得到均值的主效应图如图 6 所示,其中:μ 为每种参数的平均响应值. 均值响应表如表 3 所示,其中:Δ 为每个因子最大平均值减去最小平均值. 从图 6 中可以得到,$\alpha=51$、$\omega=10$、$m=10$、$g=10$ 的参数组合下,AMBO 算法将具有更好的性能. 因此,将 $\alpha=51$、$\beta=3$、$\chi=1$、$\omega=10$、$m=10$、$g=10$ 作为 AMBO 算法的初始化参数.

表 2 正交矩阵及响应值

Tab. 2 Orthogonal matrices and response values

次序	α	ω	m	g	平均值
1	25	5	5	10	0.080 1
2	25	10	10	20	0.080 3
3	25	15	15	30	0.088 1
4	51	5	10	30	0.082 1
5	51	10	15	10	0.076 7
6	51	15	5	20	0.086 1
7	81	5	15	20	0.089 6
8	81	10	5	30	0.086 9
9	81	15	10	10	0.088 6

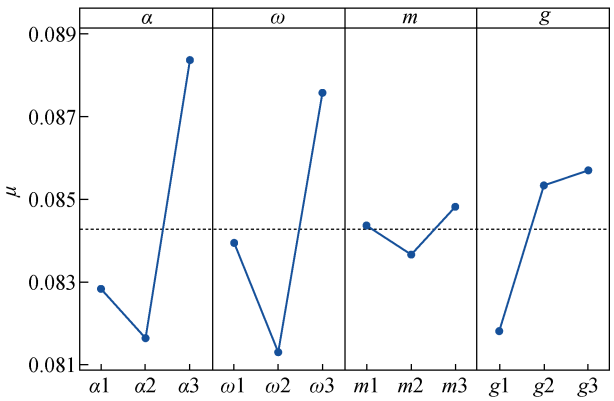

图 6　均值主效应图

Fig. 6　Mean main effects plot

表 3　均值响应表

Tab. 3　Mean response table

水平	α	ω	m	g
1	0.082 83	0.083 93	0.084 37	0.081 8
2	0.081 63	0.081 3	0.083 67	0.085 33
3	0.088 37	0.087 6	0.084 8	0.085 7
Δ	0.006 73	0.006 3	0.001 13	0.003 9
排序	1	2	4	3

3.3　算法测试

为验证 AMBO 算法求解问题的有效性，将其与基本 MBO、遗传(GA)算法对比. 因为问题的 NP 难特性，不能保证得到问题的最优解. 因此，以多次重复实验所得到的最优解的平均值作为衡量算法求解精度，采用最优解的标准偏差评价算法的稳定性. 因为订单随机性的影响，为了能够得到更多可靠的数据，算法在不同订单规模下随机生成 30 个订单任务，再对每个订单任务重复运行 30 次，得到每个任务的最优解平均值与标准偏差，分别对所得到的 30 个订单任务的最优解平均值与标准偏差取平均值，即得到所评价算法在该规模下的最优解平均值与标准偏差.

3 种算法编码规则完全一致，其中遗传算法参数为种群规模为 500，交叉概率 $P_c = 0.8$，交叉算子选用顺序交叉，变异概率 $P_m = 0.1$，变异算子选用插入算子与批量调整算子；AMBO 算法参数为鸟群中个体的数量 $\alpha = 51$，要考虑的邻域解数 $\beta = 3$，跟随鸟共享的邻域解数 $\chi = 1$，领导鸟更换后鸟群的迭代次数 $\omega = 10$，算子权重自适应调整周期 $m = 10$，扰动机制触发周期 $g = 10$；基本 MBO 算法的参数引自文献[17]：鸟群中个体的数量 $\alpha = 51$，要考虑的邻域解($\beta = 3$，跟随鸟共享的邻域解数 $\chi = 1$，领导鸟更换后鸟群的迭代次数 $\omega = 10$，邻域算子采用插入算子、交换算子与批量调整算子. 将算法终止条件统一设置为连续迭代 100 次最优值不变；采用 MATLAB 2014b 编程，测试环境为 Windows 10 系统，CPU 为 i7-7700k、内存为 16 GB. 随机生成 30 个订单总数为 20 的任务，每个任务进行 30 次重复实验，得到的平均值与其标准偏差(SD)结果如表 4 所示.

3 种算法实验所得的最优解均值、解的标准偏差、平均收敛迭代次数及算法平均运行时间的结果如表 5 所示，并根据算法迭代过程绘制了 3 种算法的平均收敛过程图如图 7 所示. 其中：ξ 为迭代次数；κ 为每次迭代中 30 次重复的多目标最优值.

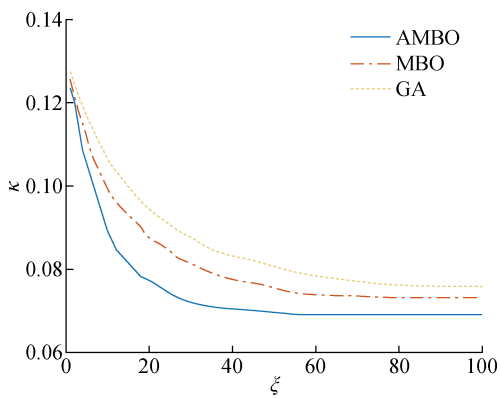

图 7　3 种算法收敛效果图

Fig. 7　Convergence effect diagram of three algorithms

由表 4、表 5 和图 7 可知，GA 算法收敛较快，但易陷入局部最优，求解精度差；而基本 MBO 算法虽然有一定的局部搜索能力，但其搜索能力有限，收敛速度慢且容易陷入局部最优，无法使候鸟集中在最优值附近；而本文提出的 AMBO 算法，采用了自适应算子权重与变邻域搜索，加快收敛速度且增强了局部搜索能力，较其他两种算法其求解精度更高，稳定性更好，求解时间与 MBO 算法较为接近，说明了 AMBO 算法的有效性.

进一步验证算法的有效性与稳定性，用 3 种算法对不同任务量求解. 表 6 分别为订单总数为 30、40、50 的 3 种不同任务规模随机重复平均实验结果.

由表 6 可以看出，在小规模算例中，随着任务规模的增大，最优解均值会增加，算法运行时间将明显增加，算法平均收敛次数增加也十分明显. GA 算法在收敛速度上有一定的优势，但其缺点在于容易陷入局部最优. MBO 算法在运行时间上比 GA 慢，但其全局搜索结果比 GA 更好. AMBO 算法的求解时

表 4 算法结果对比

Tab. 4 Comparison of algorithm results

序号	AMBO		MBO		GA	
	最优解均值	标准偏差	最优解均值	标准偏差	最优解均值	标准偏差
1	0.064 9	0.003 84	0.065 3	0.004 04	0.066 4	0.006 84
2	0.062 5	0.003 35	0.062 6	0.003 34	0.064 2	0.007 59
3	0.068 3	0.003 54	0.072 6	0.003 69	0.076 4	0.007 95
4	0.068 6	0.002 84	0.078 4	0.003 65	0.081 4	0.007 18
5	0.074 8	0.003 21	0.076 5	0.003 64	0.078 7	0.007 68
6	0.064 6	0.003 22	0.079 5	0.003 42	0.083 4	0.007 47
7	0.070 6	0.003 17	0.079 6	0.003 91	0.077 7	0.007 34
8	0.071 5	0.003 36	0.077 1	0.003 87	0.080 4	0.008 47
9	0.067 3	0.003 41	0.071 5	0.004 05	0.076 1	0.007 57
10	0.071 1	0.003 71	0.082 6	0.003 67	0.081 7	0.007 54
11	0.062 1	0.003 15	0.064 5	0.003 78	0.079 4	0.007 81
12	0.073 3	0.003 44	0.079 9	0.003 63	0.084 4	0.007 59
13	0.087 5	0.003 34	0.088 4	0.004 13	0.091 9	0.007 63
15	0.086 5	0.003 25	0.088 6	0.003 57	0.088 4	0.008 04
14	0.068 6	0.003 11	0.072 6	0.003 98	0.075 7	0.007 25
16	0.063 9	0.003 24	0.070 9	0.003 38	0.070 4	0.007 37
17	0.071 6	0.003 15	0.071 9	0.004 01	0.071 7	0.007 94
18	0.063 9	0.003 41	0.065 6	0.003 87	0.073 2	0.007 81
19	0.072 3	0.003 31	0.069 5	0.003 48	0.077 4	0.007 65
20	0.063 3	0.003 45	0.064 4	0.003 67	0.070 6	0.007 42
21	0.064 3	0.003 44	0.068 4	0.003 79	0.070 4	0.008 06
22	0.065 9	0.003 39	0.072 4	0.003 44	0.072 2	0.007 74
23	0.066 5	0.003 38	0.072 6	0.004 04	0.071 9	0.007 35
24	0.064 9	0.003 26	0.068 6	0.003 67	0.070 4	0.007 84
25	0.068 5	0.003 28	0.069 9	0.004 21	0.071 7	0.007 85
26	0.069 6	0.002 85	0.069 9	0.003 68	0.070 7	0.007 52
27	0.072 4	0.003 24	0.072 6	0.003 54	0.073 1	0.007 28
28	0.062 4	0.003 37	0.067 9	0.003 64	0.069 5	0.008 38
29	0.068 8	0.003 15	0.069 5	0.003 92	0.071 4	0.007 63
30	0.069 1	0.003 08	0.071 9	0.003 90	0.073 7	0.007 34
均值	0.069 0	0.003 30	0.073 1	0.003 75	0.075 8	0.007 64

表 5 3 种算法的实验结果

Tab. 5 Experimental results of three algorithms

算法	最优解均值	解的标准偏差	平均收敛迭代次数	算法平均运行时间/s
GA	0.075 8	0.007 64	88	263.13
MBO	0.073 1	0.003 75	76	335.24
AMBO	0.069 0	0.003 30	56	358.12

间相比于 GA 算法较长，与 MBO 算法较接近，但其求解精度与解的标准偏差均优于 MBO 算法与 GA 算法.

上述测试算例规模较小，为进一步测试算例规模对算法的影响，选取任务规模为 60、80、100 的大规模案例，以最大 CPU 运行时间为终止条件，每个任务规模随机重复 30 次，以相对百分比增加率[6](RPI)值与其标准差测试各算法寻优能力. 最大

CPU 运行时间正比于量规模，因此设定 CPU 运行时间为 $n\times\rho$，n 为订单总数，即排产任务规模，ρ 为设置的常数，随着算法规模的增加，算法的测试时间也将增加，为了综合比较算法的性能，设置三种不同的 ρ 分别为 10、20、30 s，在 $\rho=30$ 时所对比算法大部分已达到收敛状态. RPI 值计算公式如下：

$$\mathrm{RPI}(R_v)=\frac{R_v-R_{\min}}{R_{\min}} \tag{17}$$

式中：R_v 为所给算法的第 v 次实验所得目标值；$R_{\min}$ 为该算例下参与比较的几个算法中所得最小目标值，RPI 值越小，算法性能越优. 表 7 为 $n=60$，80，100 这 3 个大规模算例 30 个随机重复实验的 RPI 平均值 $\bar{x}$ 与其标准差 s. 此外，通过 T 检验将 AMBO 算法与其他算法进行比较如表 8 所示，其值为 1 或 −1 表示 AMBO 算法在 95%的置信度上优于或劣于该算法，值为 0 表明两种算法所求得的值没有显著差别.

表 6　3 种算法在不同规模下的实验结果

Tab. 6　Experimental results of three algorithms at different scales

任务规模	算法	最优解均值	解的标准偏差	平均收敛迭代次数/次	算法平均收敛时间/s
30	GA	0.077 1	0.007 98	128	362.24
	MBO	0.074 8	0.003 84	112	466.25
	AMBO	0.070 6	0.003 76	87	493.46
40	GA	0.077 8	0.007 84	167	560.32
	MBO	0.075 2	0.003 91	154	650.21
	AMBO	0.071 4	0.003 64	125	708.72
50	GA	0.080 6	0.008 14	205	780.32
	MBO	0.076 6	0.004 14	179	920.31
	AMBO	0.072 1	0.003 85	162	994.63

表 7　当 $\rho=10$，20，30 时，GA、MBO 与 AMBO 算法 RPI 值比较

Tab. 7　Comparison of RPI among GA, MBO, and AMBO algorithms at $\rho=10$, 20, and 30

任务规模	算法	$\rho=10$		$\rho=20$		$\rho=30$	
		$\bar{x}$/%	s	$\bar{x}$/%	s	$\bar{x}$/%	s
60	GA	5.78	4.78	8.15	4.32	13.01	3.81
	MBO	12.21	3.79	11.06	3.64	8.70	3.43
	AMBO	3.37	2.78	2.84	2.57	2.61	2.44
80	GA	3.73	5.24	7.27	4.71	11.42	4.29
	MBO	11.93	4.13	10.38	3.81	7.94	3.72
	AMBO	3.54	3.11	3.13	2.79	2.89	2.58
100	GA	2.41	5.67	5.61	5.12	7.56	4.52
	MBO	12.71	4.85	10.47	4.45	7.46	4.23
	AMBO	5.74	3.43	3.74	3.12	3.31	2.88

表 8　当 $\rho=10$，20，30 时，AMBO 与对比算法的 T 检验结果比较

Tab. 8　T-test results for AMBO and other algorithms at $\rho=10$, 20, and 30

任务规模	$\rho=10$		$\rho=20$		$\rho=30$	
	GA	MBO	GA	MBO	GA	MBO
60	1	1	1	1	1	1
80	0	1	1	1	1	1
100	−1	1	1	1	1	1

由表 7 和 8 可知，当 $\rho=10$，任务规模为 60 的测试实例中，AMBO 算法均优于 MBO 算法与 GA 算法；任务规模为 80 的测试实例中，AMBO 算法 GA 算法无显著差异，但优于 MBO 算法；任务规模为 100 的测试实例中，AMBO 算法劣于 GA 算法但优于 MBO 算法. 当 $\rho=20$，30 时，AMBO 算法能在所有测试实例中优于 MBO 算法与 GA 算法. AMBO 算法为了保证精度而牺牲了求解速度，在大规模算例中，尽管 AMBO 算法在分配较少的时间（$\rho=10$）时劣于 GA，但在分配更多 CPU 时间（$\rho=20$，30）后，AMBO 仍优于 GA 算法. 因此，在大规模算例下，AMBO 算法总体上优于 MBO 算法与 GA 算法.

从上述测试结果可以看出，AMBO 算法在任务规模为 50 以下的小规模算例中，其求解精度与稳定性均优于 MBO 算法与 GA 算法；在大规模算例中，AMBO 算法的寻优能力优于 MBO 算法，虽然在较少 CPU 时间内会劣于 GA 算法，但总体优于 GA 算法，故能说明 AMBO 算法的有效性.

4 结语

针对 2+1+1 型三阶段混合流水车间的 ILS-HFSP,提出了基于变邻域搜索的 AMBO 算法,解决了最小化完工时间与最小化系统平均在制品数量的多目标优化问题.所设计的算法借鉴了自适应大邻域搜索算法,在邻域算子处设计了时间窗算子与算子权重随算法的迭代自适应调整策略,提高了算法的全局搜索能力与求解精度.通过对不同规模任务的 30 次随机重复实验,表明所建立的调度数学模型及改进的候鸟迁徙优化算法是可行有效的,AMBO 算法相比于 MBO 算法与 GA 算法,具有求解质量更高、最优解标准偏差更小的优点.本研究可以降低混合流水车间的平均在制品数量,填补了 ILS-HFSP 问题关于最小化完工时间与最小化在制品数量多目标优化研究的空白,也更符合工厂实际生产.未来将结合缓存区配置优化与动态排程等方面进行进一步研究.

参考文献:

[1] 周炳海,刘文龙.考虑能耗和准时的混合流水线多目标调度[J].**上海交通大学学报**,2019,53(7):773-779.
ZHOU Binghai, LIU Wenlong. Multi-objective hybrid flow-shop scheduling problem considering energy consumption and on-time delivery[J]. **Journal of Shanghai Jiao Tong University**, 2019, 53(7): 773-779.

[2] ZHOU X J, YU M Q. Semi-dynamic maintenance scheduling for multi-station series systems in multi-specification and small-batch production[J]. **Reliability Engineering & System Safety**, 2020, 195: 106753.

[3] 陶辛阳,夏唐斌,奚立峰.基于健康指数的预防性维护与多目标生产调度联合优化建模[J].**上海交通大学学报**,2014,48(8):1170-1174.
TAO Xinyang, XIA Tangbin, XI Lifeng. Health-index-based joint optimization of preventive maintenance and multi-attribute production scheduling[J]. **Journal of Shanghai Jiao Tong University**, 2014, 48(8): 1170-1174.

[4] 李颖俐,李新宇,高亮.混合流水车间调度问题研究综述[J].**中国机械工程**,2020,31(23):2798-2813.
LI Yingli, LI Xinyu, GAO Liang. Review on hybrid flow shop scheduling problems[J]. **China Mechanical Engineering**, 2020, 31(23): 2798-2813.

[5] ZHANG B, PAN Q K, GAO L, *et al*. An effective modified migrating birds optimization for hybrid flow-shop scheduling problem with lot streaming[J]. **Applied Soft Computing**, 2017, 52: 14-27.

[6] MENG T, PAN Q K, LI J Q, *et al*. An improved migrating birds optimization for an integrated lot-streaming flow shop scheduling problem[J]. **Swarm and Evolutionary Computation**, 2018, 38: 64-78.

[7] ZHANG M, TAN Y T, ZHU J H, *et al*. A competitive and cooperative migrating birds optimization algorithm for vary-sized batch splitting scheduling problem of flexible job-shop with setup time[J]. **Simulation Modelling Practice and Theory**, 2020, 100: 102065.

[8] NADERI B, YAZDANI M. A model and imperialist competitive algorithm for hybrid flow shops with sublots and setup times[J]. **Journal of Manufacturing Systems**, 2014, 33(4): 647-653.

[9] LALITHA J L, MOHAN N R, PILLAI V M. Lot streaming in [N-1](1)+N(m) hybrid flow shop[J]. **Journal of Manufacturing Systems**, 2017, 44: 12-21.

[10] 李航,章旸,叶鸿庆,等.考虑批量流与换模时间的柔性生产线调度方法研究[J].**工业工程与管理**,2020,25(3):179-187.
LI Hang, ZHANG Yang, YE Hongqing, *et al*. Research on flexible production line scheduling with lot streaming and setup times[J]. **Industrial Engineering and Management**, 2020, 25(3): 179-187.

[11] ZHANG B, PAN Q K, GAO L, *et al*. A multi-objective migrating birds optimization algorithm for the hybrid flowshop rescheduling problem[J]. **Soft Computing**, 2019, 23(17): 8101-8129.

[12] LI J Q, TAO X R, JIA B X, *et al*. Efficient multi-objective algorithm for the lot-streaming hybrid flow-shop with variable sub-lots[J]. **Swarm and Evolutionary Computation**, 2020, 52: 100600.

[13] GONG D W, HAN Y Y, SUN J Y. A novel hybrid multi-objective artificial bee colony algorithm for blocking lot-streaming flow shop scheduling problems[J]. **Knowledge-Based Systems**, 2018, 148: 115-130.

[14] 沈倩,管在林,张正敏,等.面向卷烟生产调度的集成产能过滤算法与仿真技术的优化框架[DB/OL].(2020-10-26)[2021-02-07]. https://kns.cnki.net/kcms/detail/11.5946.TP.20201026.1016.012.html.
SHEN Qian, GUAN Zailin, ZHANG Zhengmin, *et al*. An optimization framework based on simulation integrated capacity filtering algorithm for cigarette production scheduling[DB/OL]. (2020-10-26)[2021-02-07]. https://kns.cnki.net/kcms/detail/11.5946.TP.20201026.1016.012.html.

[15] 吕洁,郭婷芳,韩文民.虚拟制造单元瓶颈缓冲区容

量优化[J]. **组合机床与自动化加工技术**，2016(12)：121-124.
LV Jie，GUO Tingfang，HAN Wenmin. Bottleneck buffer allocation optimization of the virtual manufacturing[J]. **Modular Machine Tool & Automatic Manufacturing Technique**，2016(12)：121-124.

[16] ENGEHAUSEN F，LÖDDING H. Managing sequence-dependent setup times—The target conflict between output rate，WIP and fluctuating throughput times for setup cycles[J]. **Production Planning & Control**，2020：1-17.

[17] DUMAN E，UYSAL M，ALKAYA A F. Migrating birds optimization：A new metaheuristic approach and its performance on quadratic assignment problem[J]. **Information Sciences**，2012，217：65-77.

[18] DINDAR O Z. An improvement on the migrating birds optimization with a problem-specific neighboring function for the multi-objective task allocation problem[J]. **Expert Systems With Applications**，2017，67：304-311.

[19] EXPOSITO IZQUIERDO C，DE ARMAS J，LALLA RUIZ E. Multi-leader migrating birds optimization：A novel nature-inspired metaheuristic for combinatorial problems[J]. **International Journal of Bio-Inspired Computation**，2017，10(4)：1.

[20] ALMONACID B，SOTO R，CRAWFORD B. Comparing three simple ways of generating neighboring solutions when solving the cell formation problem using two versions of migrating birds optimization[C]// **2017 17th International Conference on Computational Science and Its Applications (ICCSA)**. Piscataway，NJ，USA：IEEE，2017：1-9.

[21] TONGUR V，ÜLKER E. The analysis of migrating birds optimization algorithm with neighborhood operator on traveling salesman problem[C]// **The 19th Asia Pacific Symposium**. Bangkok，Thailand：Springer，2015：1-11.

[22] ROPKE S，PISINGER D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows[J]. **Transportation Science**，2006，40(4)：455-472.

[23] 姚妮. 混合候鸟迁徙优化算法求解柔性作业车间调度问题[J]. **华中师范大学学报(自然科学版)**，2016，50(1)：38-42.
YAO Ni. Hybrid migrating brids optimization algorithm for the flexible job shop scheduling problem[J]. **Journal of Central China Normal University (Natural Sciences)**，2016，50(1)：38-42.

[24] 唐立力. 求解低碳调度问题的改进型候鸟优化算法[J]. **计算机工程与应用**，2016，52(17)：166-171.
TANG Lili. Improved migrating birds optimization algorithm to solve low-carbon scheduling problem[J]. **Computer Engineering and Applications**，2016，52(17)：166-171.

（本文编辑：石易文）