智能系统学报
CAAI TRANSACTIONS ON INTELLIGENT SYSTEMS

含忽略工序和不相关机的混合流水车间调度

轩华, 樊银格, 李冰

引用本文:
轩华,樊银格,李冰. 含忽略工序和不相关机的混合流水车间调度[J]. 智能系统学报, 2022, 17(3): 459–470.
XUAN Hua,FAN Yinge,LI Bing. Hybrid flowshop scheduling with missing operations and unrelated machines[J]. *CAAI Transactions on Intelligent Systems*, 2022, 17(3): 459–470.

在线阅读 View online: https://dx.doi.org/10.11992/tis.202103006

DOI: 10.11992/tis.202103006
网络出版地址: https://kns.cnki.net/kcms/detail/23.1538.TP.20211210.2314.004.html

含忽略工序和不相关机的混合流水车间调度

轩华, 樊银格, 李冰
(郑州大学 管理工程学院, 河南 郑州 450001)

摘　要: 研究从炼钢等生产过程提炼出的含忽略工序和不相关并行机的混合流水车间调度问题, 以最小化最大完工时间为目标, 建立整数规划模型, 并提出结合全局搜索、自适应遗传算法和候鸟优化的遗传候鸟优化算法以求解该模型。在算法中采用与处理时间相关的全局搜索和随机程序以获得初始种群, 提出自适应交叉和变异操作改进遗传算法解, 在迭代进程中, 引入基于工件、机器和工序位 3 种邻域搜索结构的候鸟优化算法更新最佳解。仿真实验中将遗传候鸟优化算法的实验结果与几种启发式算法进行对比, 证明了模型和算法的有效性。

关键词: 忽略工序; 不相关并行机; 混合流水车间; 全局搜索; 自适应遗传算法; 领域搜索; 最大完工时间; 遗传候鸟优化算法
中图分类号: TP39;TB49　　**文献标志码:** A　　**文章编号:** 1673−4785(2022)03−0459−12

中文引用格式: 轩华, 樊银格, 李冰. 含忽略工序和不相关机的混合流水车间调度 [J]. 智能系统学报, 2022, 17(3): 459–470.
英文引用格式: XUAN Hua, FAN Yinge, LI Bing. Hybrid flowshop scheduling with missing operations and unrelated machines[J]. CAAI transactions on intelligent systems, 2022, 17(3): 459–470.

Hybrid flowshop scheduling with missing operations and unrelated machines

XUAN Hua, FAN Yinge, LI Bing
(School of Management Engineering, Zhengzhou University, Zhengzhou 450001, China)

Abstract: The hybrid flowshop scheduling problem with missing processes and unrelated parallel machines extracted from steelmaking and other production processes is studied. An integer programming model is formulated to minimize the maximum completion time, and a genetic migrating birds optimization algorithm based on the global search, self-adaptive genetic algorithm, and migrating birds optimization is proposed to solve the model. The initial population is obtained by the global search considering the machine processing time and random procedure. Then, self-adaptive crossover and mutation operators are developed to improve the solutions of the genetic algorithm. In the iterative process, migrating birds optimization combined with three neighborhood search structures based on the job, machine, and operation position is introduced to update the best solutions. Finally, the experimental results of the genetic migrating birds optimization algorithm are compared with several heuristic algorithms to prove the effectiveness of the proposed model and algorithm.

Keywords: missing operations; unrelated parallel machines; hybrid flowshop; global search; self-adaptive genetic algorithm; neighborhood search; maximum completion time; genetic migrating birds optimization algorithm

含忽略工序的混合流水车间调度(hybrid flowshop scheduling with missing operation, HFSMO)在炼钢、不锈钢、塑料等制造工业较为常见。在经典混合流水车间调度问题中, 通常假定工件经所有工序处理, 但在实际生产中, 工件会忽略某些工序, 即一些工件可能不经某些工序处理, 如炼钢-连铸生产中, 由电弧炉或转炉生产的钢水要在高温下经精炼和连铸工序进行加工, 不同的钢种对生产路线的要求会有所不同, 像 Q235

收稿日期: 2021−03−03.　**网络出版日期:** 2021−12−13.
基金项目: 国家自然科学基金项目(U1804151); 河南省科技攻关计划项目(202102310310).
通信作者: 轩华. E-mail: hxuan@zzu.edu.cn.

普通碳钢要略过 RH(Ruhrstahl-Hausen)真空精炼炉阶段[1]。考虑到并行机的新旧程度或异构性，由此衍生出本文所研究的带不相关并行机的 HFSMO。由于 HFS (hybrid flowshop scheduling) 问题是 NP-hard[2]，带不相关并行机的 HFSMO 是 HFS 问题的扩展且其更为复杂，它不仅要确定工件的处理序列，还需确定工件在每道未忽略工序的机器分配，所以本文研究的更复杂的带不相关并行机的 HFSMO 也是 NP-hard。

目前，已有不少学者研究 HFSMO。就两道工序的 HFSMO 而言，Tseng 等[3] 研究了工序 1 有一台机器且工序 2 有两台同构机的情况，假定工序 1 可忽略，提出了一种启发式算法以最小化最大完工时间。就含同构并行机的多工序 HFSMO 而言，为最小化最大完工时间，Saravanan 等[4-5] 提出了遗传算法、模拟退火算法和粒子群算法，Marichelvam 等[6] 基于遗传算法和分散搜索算法提出了一种改进的混合遗传分散搜索算法，Dios 等[7-8] 提出了一些分派规则和改进启发式来求解该问题；为最小化平均拖期，Saravanan 等[9] 提出了遗传算法与模拟退火算法；为最小化平均滞留时间、总提前、总拖期和关键机器跳过率的加权和，Li 等[10] 针对铁水系统提出了一种改进离散人工蜂群算法；为了最小化最大完工时间、总等待时间以及处理时间与标准处理时间的偏差之和，Long 等[1] 针对炼钢-连铸生产系统提出了一种改进遗传算法。

目前，已有不少学者在不相关并行机环境下研究 HFS 问题。针对单目标问题，Meng 等[11] 研究了节能 HFS，并提出改进的遗传算法以最小化机器闲置消耗。为最小化最大完工时间，Qin 等[12] 研究了批量调度 HFS，提出两阶段蚁群算法；罗函明等[13] 针对多工序 HFS，提出离散布谷鸟算法；轩华等[14] 针对带有限缓冲 HFS，提出基于遗传算法和禁忌搜索的混合启发式算法。针对多目标问题，Zhou 等[15] 研究了带模糊处理时间的 HFS，设计差分进化算法以最小化总加权交付损失和总能耗。Yu 等[16] 针对带机器能力约束和依赖加工序列设置时间的 HFS，提出带多重解码框架的进化算法以最小化总拖期和总设置时间。Zhou 等[17] 研究了带节能区间的 HFS，提出新的帝国竞争算法以解决以总能耗和最大完工时间为目标的双目标问题。

就目前查阅的文献而言，HFSMO 已引起众多学者的关注，既有的研究聚焦于同构机环境。现有的带不相关并行机的 HFS 不考虑带忽略工序特性，关于不相关并行机环境下 HFSMO 问题的研究尚缺乏，还有待进一步探讨。就优化算法而言，文献 [1, 3-10] 的研究说明了进化算法求解 HFSMO 问题有较大潜力，如文献 [4, 9] 利用单独的遗传算法和模拟退火算法求解含同构机的最大完工时间问题。混合算法通常能扩大搜索的范围，提高解的质量，因此关于混合算法求解这类问题的研究还有待进一步探讨。候鸟优化(migrating birds optimization, MBO)算法作为一种新兴的群智能优化算法，已广泛应用于生产调度领域，如流水车间调度[18-21]、混合流水车间调度[22-23] 和柔性作业车间调度[24-26]，它最早是由 Duman 等[27] 提出并应用于二次分配问题。因此，本文结合全局搜索、自适应遗传算法和候鸟优化提出一种遗传候鸟优化算法(genetic migrating birds optimization algorithm, GMBOA)解决含不相关并行机的 HFSMO，通过仿真实验验证所提算法的有效性和可行性。

1　问题描述与建模

1.1　问题描述

含不相关并行机的 HFSMO 问题包括来自集合 $J=\{1,2,\cdots,n\}$ 且将在 h 道工序上处理的 n 个工件，每道工序 s 有 m_s 台不相关并行机，即对于每道工序，工件在 m_s 台并行机上的处理时间相互独立，仅取决于工件与机器的匹配程度。每个工件可能会忽略某些工序。每台机器一次只能处理一个工件，而每个工件一次至多在一台机器进行处理。调度目标是确定工件处理序列和机器分配，以最小化最大完工时间。所研究问题的其他假设如下：

1)工件的开工应在其释放时间之后；

2)机器准备时间与工件顺序无关，且包含在处理时间中；

3)所有机器在整个计划时间段内连续可用；

4)工件一旦在某台机器上开始处理后，不允许中断，直至该工序完成；

5)工件处理无优先级要求；

6)工序之间的转移时间忽略不计；

7)相邻工序之间的缓冲区容量无限。

1.2　模型构建

由前述可知，每个工件 j 实际访问的总工序数 $O_j \leqslant h$，虽然工件需经 h 道工序完成其处理任务，但部分工件 j 的处理略过了 $h-O_j$ 道工序，即这些工件未经 $h-O_j$ 道工序处理而直接进入后续工序。含不相关并行机的 HFSMO 模型如下：

所研究问题的目标是满足所有约束条件下最小化最大完工时间 $C_{\max}$，即

$$\min z = C_{\max} \tag{1}$$

$$\text{s.t.}\quad C_{\max} \geqslant C_{jh}, \forall j \tag{2}$$

式中: C_{jh}表示工件j($j=1,2,\cdots,n$)在工序h的完工时间。式 (2) 通过检查工件在最终工序h的完工时间, 确定最大完工时间。

$$C_{js} = (1-W_{js})\cdot C_{j,s-1} + W_{js}\cdot\left(B_{js} + \sum_{k=1}^{m_s} X_{jks}\cdot P_{jks}\right), \forall j,s \tag{3}$$

式中: m_s为工序s($s=1,2,\cdots,h$)可利用的机器数; F为一个足够大的数; P_{jks}为工件j在工序s的机器k($k=1,2,\cdots,m_s$)上的处理时间; W_{js}为二元参数, 若工件j在工序s上处理, 其值为 1, 否则为 0; B_{js}表示工件j在工序s的开工时间; X_{jks}为二元变量, 若工件j在工序s的机器k上处理, 其值为 1, 否则为 0。式 (3) 定义了工件在每道工序的完工时间, 若工件j未在工序s处理, 则它在该工序的完工时间为它在紧前未忽略工序的完工时间。

$$\sum_{k=1}^{m_s} X_{jks} = 1, \forall j,s$$

该约束确保每个工件在任一工序只能分派到一台机器进行处理。

$$C_{js} \leqslant B_{j,s+1}, s \in \{1,2,\cdots,h-1\}, \forall j$$

该约束说明了每个工件只完完成前一工序的处理任务后方可开始下一道工序的处理。

$$B_{j1} \geqslant R_j, \forall j$$

式中R_j为工件j的释放时间。该约束描述了工件只有到达生产系统才可开始处理。

$$B_{js} \geqslant (C_{is} - Z_{jiks}\cdot\Omega)\cdot W_{js}W_{is}, \forall j,i,k,s, j\neq i \tag{4}$$

$$B_{is} \geqslant (C_{js} - (1-Z_{jiks})\cdot\Omega)\cdot W_{js}W_{is}, \forall j,i,k,s, j\neq i \tag{5}$$

式中: Z_{jiks}为二元变量, 若工件i和j在工序s的机器k上处理且工件j早于工件i, 其值为 1, 否则为 0。这两个约束表示: 同一道工序分派在同一台机器处理的两个工件之间的优先级关系, 若$Z_{jiks}=0$且$W_{js}=W_{is}=1$, 约束式 (4) 描述了工件j在工序s的机器k上的开工时间必须在工件i的完工时间之后, 若$Z_{jiks}=1$且$W_{js}=W_{is}=1$, 约束式 (5) 描述了工件i在工序s的机器k上的开工时间必须在工件j的完工时间之后。

$$B_{js} \geqslant 0, C_{js} \geqslant 0, \forall j,s \tag{6}$$

$$X_{jks} \in \{0,1\}, Z_{jiks} \in \{0,1\}, \forall j,s \tag{7}$$

约束式 (6)、(7) 定义了变量取值范围。

2　遗传候鸟优化算法

遗传算法已广泛用于求解 HFSMO 问题, 为使遗传算法有效求解含不相关并行机的 HFSMO 问题, 设计基于机器号的编码方案, 采用考虑机器处理时间的全局搜索和随机程序生成初始种群以提高初始解的质量, 设计自适应更新策略以计算交叉概率ξ及变异概率ψ, 以此执行交叉和变异操作。最后, 引入结合邻域搜索的候鸟优化算法以扩大遗传算法解的邻域搜索范围, 从而获得较好的近优解。

2.1　编码、解码和适应度选择

含不相关并行机的 HFSMO 需确定工件在每道工序的机器分配, 因此设计基于机器号的整数编码方案以表述机器分配序列σ。考虑到 HFS 的多工序处理需求, 令每个工件所忽略的工序数不超过 $h-2$, 根据忽略工序比例p, 随机产生每个工件的未忽略工序信息序列τ, 如图 1($n=5, h=5$和$p=0.6$), 其中U_j^s表示工件j在阶段s的工序; 然后基于τ生成相应工件的机器号M_j^s, 为平衡并行机器的负荷, 令其值满足[1,m_s]的均匀分布, 从而形成长度为$n\cdot h\cdot(1-p)$的一个染色体, 其中每个元素(即工序位上的数值)为对应工件所在工序分配的机器号。

U_1^3	U_1^4	U_2^2	U_2^3	U_3^4	U_3^5	U_4^4	U_4^5	U_5^2	U_5^5

图 1　未忽略工序信息序列

Fig. 1　Information sequence of unmissing operations

初始化种群时, 由于每道工序所含的机器为不相关并行机, 考虑不同的机器处理同一工件的时间不同, 而最大完工时间与工件的处理时间相关。因此, 基于张国辉等[28] 的研究提出考虑处理时间的全局搜索以产生一个个体, 而其他个体则由随机程序生成。令数组$\theta=\{p_{s11},p_{s21},\cdots,p_{sm_{11}},p_{s12},\cdots,p_{sm_{21}},p_{s1h},p_{s2h},\cdots,p_{sm_hh}\}$为记录从工序 1 到工序$h$的每台机器的累计处理时间的一维数组, 其初始值均设为 0, 全局搜索从任一工件j开始, 对它的第一道未忽略工序u, 将可利用并行机的工件处理时间分别与数组θ内该机器位置的数值相加, 即对$k=1,2,\cdots,m_u$, 计算$\{P_{jku}+p_{sku}\}$, 从中选择累计处理时间最短的机器k'作为工件j在工序u所分配的机器, 将该机器号填入个体内工序位$(h(1-p)(j-1)+1)$, 令$p_{suk'}=\min\{P_{jku}+p_{sku}\}$, 更新数组$\theta$; 然后, 对工件$j$的第$2,3,\cdots,h(1-p)$道工序重复上述过程, 从而完成该工件所有工序的机器分配。接着, 再从剩余工件集内随机选择一个工件执行上述过程, 直至完成所有工件未忽略工序的机器分配。结合随机程序产生的$(e-1)$个个体共同构成了如图 2 所示的种群规模为e的初始种群。

为计算最大完工时间, 需为分配至同一台机器的工件进行排序, 为此, 设计了基于最短处理时间的解码方案, 即对于机器分配序列σ, 当

$s=1$时，将分派到在同一台机器处理的工件按照最短处理时间规则进行排序，当$s>1$时，对同一台机器上处理的工件采用先到先加工规则生成处理序列，然后基于最早空闲机器原则依次将各工件安排在所分配机器。对于工件的忽略工序，则按照约束式 (3) 确定其在忽略工序的完工时间。

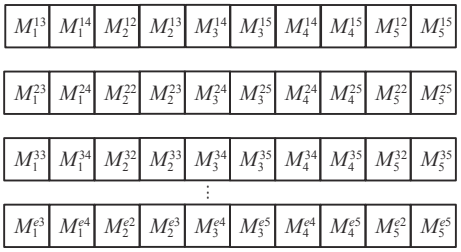

图 2　初始种群

Fig. 2　Initial population

本文的目标是最小化最大完工时间，故将适应度函数表示为目标函数的倒数，即个体g的适应度函数为$F(g)=1/C_{\max}(g)$。采用轮盘赌选择法，个体适应度值越大，被选择的概率也越大。

2.2　交叉操作

采用单点交叉和均匀两点交叉两种方式执行交叉操作。在 0 和 1 中随机生成一个整数v，当$v=0$时，执行单点交叉，即随机选择两个父代个体E_1和E_2，随机生成一个工序位$x(1\leqslant x\leqslant n\cdot h\cdot(1-p))$，交换$E_1$和$E_2$中所选工序位$x$的机器号，保持其他工序位的机器号不变，从而生成子代个体D_1和D_2，如图 3。当$v=1$时，执行均匀两点交叉，首先，随机选择两个父代个体E_1和E_2，随机生成两个工序位x和$y(1\leqslant x<y\leqslant n\cdot h\cdot(1-p))$，将$E_1$和$E_2$分为 3 段，然后，从{0, 1, 2}中随机产生一个数q，当$q=0$时，将E_1和E_2中处于工序位x之前的区段交换，作为子代个体D_1和D_2的第一段；$q=1$时，交换E_1和E_2中工序位x和y之间的区段，作为D_1和D_2的中间段；$q=2$时，对E_1和E_2中处于工序位y之后的区段进行交换，作为D_1和D_2的第三段；保留子代个体中其他段与父代个体相同，以产生子代个体D_1和D_2，如图 4。

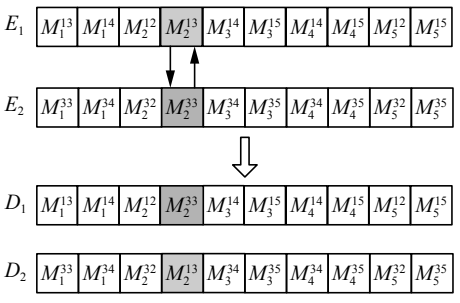

图 3　单点交叉

Fig. 3　Single-point crossover

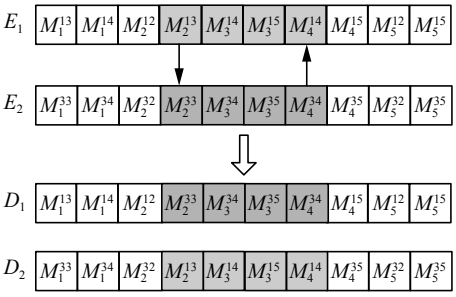

图 4　均匀两点交叉（q=1）

Fig. 4　Uniform two-point crossover（q=1）

采用自适应更新策略确定交叉概率ξ，用以确定是否执行上述交叉操作，计算如下[29]：

$$\xi=\begin{cases}\xi_{\max}-(\xi_{\max}-\xi_{\min})\left(\dfrac{\lambda}{\beta}+\dfrac{F(g)}{F_{\max}-F_{\text{avg}}}\right), & F(g)\geqslant F_{\text{avg}}\\ \xi_{\min}, & F(g)<F_{\text{avg}}\end{cases}$$

式中：λ为当前迭代数；β为最大迭代数；F_{avg}为当前种群的平均适应度值。

2.3　变异操作

变异操作是根据变异概率通过改变父代个体的机器号以产生子代个体的过程。本文提出了基于随机机器选择的单点变异和基于机器最短处理时间的多点变异两种方式。

1）基于随机机器选择的单点变异

从父代个体E_1中随机选择一个工序位$x(1\leqslant x\leqslant n\cdot h\cdot(1-p))$，将该工序位的机器号重新在$[1,m_s]$之间随机生成，如图 5。

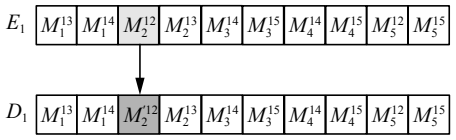

图 5　基于随机机器选择的单点变异

Fig. 5　Single point mutation based on random machine selection

2）基于机器最短处理时间的多点变异

推广 Chang 等[29]的研究，提出基于机器最短处理时间的多点变异操作。从父代个体E_1依次取出工序位x的机器号，比较从区间 [0, 1] 生成的随机数w与变异概率ψ，若$w<\psi$，则从工序位x可利用的并行机中选择处理时间最短的机器替换原有机器号，否则，保持原有机器号不变，对个体内所有工序位完成上述操作后，生成新个体D_1，如图 6。

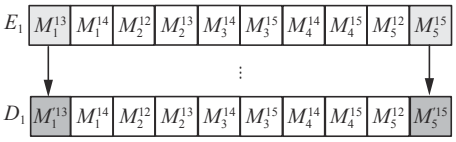

图 6　基于机器最短处理时间的多点变异

Fig. 6　Multi-point mutation based on the shortest processing time of the machine

自适应变异概率 ψ 由式 (8) 确定:

$$\psi=\begin{cases}\psi_{\max}-(\psi_{\max}-\psi_{\min})\left(\dfrac{\lambda}{\beta}+\dfrac{F(g)}{F_{\max}-F_{\text{avg}}}\right), F(g)\geqslant F_{\text{avg}}\\ \psi_{\min}, F(g)<F_{\text{avg}}\end{cases} \tag{8}$$

2.4　基于 3 种邻域搜索结构的候鸟优化

为提高算法的搜索能力, 进一步改善遗传算法解的质量, 设计引入基于工件、机器和工序位的 3 种邻域搜索结构的候鸟优化算法。

1)邻域搜索结构N_1

随机选择一个工序u, 从在该工序处理的工件集中随机生成两个工件π_1和π_2, 将它们在该工序处理的机器号进行交换, 重复该过程直至达到最大循环次数$L_{\max}$。

2)邻域搜索结构N_2

从机器集中寻找处理工件数超过两个的机器, 分别计算每台机器上工件处理时间之和, 从中获取具有最大总处理时间的机器k', 进而得到它对应的工序u, 从该机器处理的工件集中选取处理时间最长的工件π, 将$\min\{P_{\pi ku}\}(k=1,2,\cdots,m_u)$对应的机器作为该工件分配的机器。

3)邻域搜索结构N_3

随机生成一个工序位x, 确定它所对应的工件$\pi=\lceil x/[h\cdot(1-p)]\rceil$和工序$u=x-\pi\cdot h\cdot(1-p)$, 将该工序位的机器号依次设置为$1,2,\cdots,m_u$, 分别计算相应个体的适应度值, 从中选取适应度值最大的机器号填入该工序位, 重复该过程直至达到最大循环次数$\eta_{\max}$。

引入上述邻域搜索结构执行候鸟优化算法, 具体步骤如下:

1)参数初始化。设置$L_{\max}$、$\eta_{\max}$以及候鸟优化最大迭代数γ, 巡回数$G_{\max}$, 令$\bar{\omega}=1$, $G_1=1$, $\alpha=1$, 候鸟种群规模为δ。

2)初始种群生成。设计初始种群由 3 部分构成: ①对每个工件的未忽略工序, 从可利用的并行机中选择工件处理时间最短的机器作为该工件分配的机器, 从而产生一个个体; ②应用前述全局搜索方法产生50% $(\delta-1)$个个体; ③从 GA 解中选取最好的50% $(\delta-1)$个个体。从δ个个体中选取最大完工时间最小的个体作为领飞鸟, 其余均为跟飞鸟, 若跟飞鸟与领飞鸟相同, 则将领飞鸟通过基于机器最短处理时间的多点变异或邻域搜索结构$N_1(L_{\max}=2)$产生新跟飞鸟; 若新跟飞鸟与其余跟飞鸟相同, 则对新跟飞鸟继续应用邻域搜索结构N_1进行更新, 直至产生与其余跟飞鸟不同的新跟飞鸟。

3)领飞鸟进化。随机产生 [1, 6] 之间的一个整数r, 若$r=1$, 令$L_{\max}=2$, 对领飞鸟执行N_1; 若$r=2$, 则执行N_2; 若$r=3$, 令$\eta_{\max}=1$, 执行N_3; 若$r=4$, 令$L_{\max}=4$, 执行N_1; 若$r=5$, 令$\eta_{\max}=2$, 执行N_3; 若$r=6$, 令$L_{\max}=6$, 执行N_1。该过程往复o次产生领飞鸟的o个邻域解, 若其中的最佳解优于当前领飞鸟, 则更新领飞鸟, 取其余$(o-1)$个邻域解中的最佳解加入共享邻域解集X_L和X_R。

4)右侧跟飞鸟进化。对右侧队列中每个个体g, 执行与 3)相同的邻域搜索过程产生$(o-1)$个邻域解, 构成集合D_R, 找到$X_R\cup D_R$内的最佳解, 若其优于当前解, 则更新跟飞鸟, 清空X_R, 从$X_R\cup D_R$未用的解集中选择最好的邻域解加入X_R。

5)左侧跟飞鸟进化。对左侧队列中每个个体g, 仍采用与 3)相同的邻域搜索过程产生$(o-1)$个邻域解, 构成集合D_L, 若$X_L\cup D_L$内最佳解优于当前解, 则更新跟飞鸟, 清空X_L, 将$X_L\cup D_L$未用的解集中最好的邻域解加入X_L。

6)$G_1=G_1+1$, 若$G_1<G_{\max}$, 返回 3), 否则, G_1=1, 执行 7)。

7)领飞鸟更新。若$\alpha=1$, 将左侧队列的第一个跟飞鸟作为新领飞鸟, 将原领飞鸟移至左侧队列末端, 令$\alpha=0$; 否则, 将右侧队列的第一个跟飞鸟作为新领飞鸟, 把原领飞鸟移至右侧队列末端, 令$\alpha=1$。

8)$\bar{\omega}=\bar{\omega}+1$, 若$\bar{\omega}>\gamma$, 算法停止, 输出最佳解; 否则, 返回 3)。

2.5　遗传候鸟优化算法流程

首先, 考虑本文研究的问题是含不相关机的 HFSMO, 其不仅与工件的处理序列有关, 还与工件在每道工序的机器分配相关, 因此本文设计基于机器号的编码方案, 在解码时采用最短处理时间和先进先出规则确定工件处理序列; 采用考虑机器处理时间的全局搜索产生一个个体, 用随机程序生成剩余$(e-1)$个个体以生成种群规模为e的初始种群; 计算适应度, 根据轮盘赌法选择个体生成新种群; 对新种群根据自适应更新策略计算交叉概率ξ及变异概率ψ, 以此执行交叉和变异操作, 得到遗传进化后的最佳解并记录历史最佳解。最后, 当最佳解T次没变时, 按照基于 3 种邻域搜索结构的候鸟优化产生新解, 若新解优于历史最佳解, 则更新历史最佳解, 否则保持原解不变, 重复上述过程直至满足算法的停止标准。

综上所述, 所研究的 GMBOA 执行如下:

1)设置种群规模e, 最大迭代数β和累计数T, 令$Z^*=F$, $\lambda=1$, $t=1$;

2)若$\lambda>\beta$或 CPU 达到最大运行时间, 程序停止, 输出最佳解; 否则, 执行 3);

3)结合随机程序和全局搜索产生初始种群;

4)计算每个个体的适应度值,使用轮盘赌选择法获取适应度值高的个体;

5)根据交叉概率ξ,从单点交叉和均匀两点交叉 2 种方式随机选择 1 种生成新个体;

6)根据变异概率ψ,随机执行基于随机机器选择的单点变异或基于机器最短处理时间的多点变异;

7)若当前迭代得到的最佳解$Z^{\lambda} \neq Z^{*}$,$Z^{*}=\min\{Z^{\lambda},Z^{*}\}$,$t=1$,$\lambda=\lambda+1$,转至 2);否则,$t=t+1$,执行 8);

8)若$t=T$,执行 9),否则,$\lambda=\lambda+1$,转至 2);

9)调用候鸟优化算法,若得到的最佳解优于Z^{*},更新Z^{*},否则,保留Z^{*}不变;$t=1$,$\lambda=\lambda+1$,转至 2)。

3　仿真实验

本文所提的遗传候鸟优化算法将全局搜索、自适应遗传算法和候鸟优化相结合,为了验证所提出的算法性能,从传统算法、与其他算法结合的混合算法、解码规则、已发表的文献的算法 4 个角度选择传统遗传算法 (traditional genetic algorithm, TGA),结合 3 种领域搜索结构候鸟优化算法 (migrating birds optimization & neighborhood search, MBO&NS)、基于最长处理时间规则解码的遗传候鸟优化算法 (genetic migrating birds optimization algorithm L, GMBOAL)、结合局域搜索的自适应遗传算法 (adaptive genetic algorithm & local search, AGA&LS) 与文献 [30] 的改进人工蜂群算法 (improved artificial bee colony, IDABC) 进行对比。采用 Matlab R2014b 进行编程,在 CPU 为 Inter Core i5-5200U,内存为 4 GB,主频 2.2 GHz 的微机上运行。

3.1　参数设置

为公平比较 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 这 6 种算法,TGA 中的ξ和ψ的取值通过仿真实验得到最佳的一组设置,即$\xi=0.8$和$\psi=0.2$;GMBOAL、AGA&LS 和 GMBOA 中的$\xi_{\max}$、$\xi_{\min}$、$\psi_{\max}$和$\psi_{\min}$的取值是通过仿真实验得到最佳的一组设置,即$\xi_{\max}=0.9$、$\xi_{\min}=0.5$、$\psi_{\max}=0.2$和$\psi_{\min}=0.02$;参考文献 [22],并经过仿真实验测试 GMBOA 中的$\delta=31$、$\gamma=10$、$G_{\max}=10$、$o=3$;基于 GA 的 4 种算法和 IDABC 的种群规模$e=100$;由于 MBO&NS 的种群规模必须为奇数,所以其种群规模$e=101$;IDABC 的参数与文献 [30] 的设置相同,最大迭代数$\beta=100$,最大 CPU 时间为 720 s;考虑到 GA 的运行时间较短,将 MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 的停止时间设为 TGA 运行 100 次所需时间。

问题产生如下:$n=\{20,30,40,50,80,100,120,150\}$,$h=\{5,10,15,20\}$,$m_s=5$。借鉴 Dios 等[7-8] 关于忽略工序比例的设定,将本文的忽略工序比例p分别取为 20%、40% 和 60%。每个工件在同一道工序不同机器上的处理时间P_{jks}不同且满足 [1, 99] 之间的均匀分布。

3.2　数据实验与结果分析

参数$\{n,h,p\}$的不同组合产生 96 组问题规模,每种规模随机运行 10 个实例,取 10 次测试结果的平均值作为对应规模问题的测试结果。

定义相对偏差为

$$R_1=\frac{C_{\mathrm{TGA}}-C_{\mathrm{GMBOA}}}{C_{\mathrm{GMBOA}}}\times 100\%$$

$$R_2=\frac{C_{\mathrm{MBO\&\,NS}}-C_{\mathrm{GMBOA}}}{C_{\mathrm{GMBOA}}}\times 100\%$$

$$R_3=\frac{C_{\mathrm{GMBOA^L}}-C_{\mathrm{GMBOA}}}{C_{\mathrm{GMBOA}}}\times 100\%$$

$$R_4=\frac{C_{\mathrm{AGA\&\,LS}}-C_{\mathrm{GMBOA}}}{C_{\mathrm{GMBOA}}}\times 100\%$$

$$R_5=\frac{C_{\mathrm{IDABC}}-C_{\mathrm{GMBOA}}}{C_{\mathrm{GMBOA}}}\times 100\%$$

由于本文所提的算法迭代 100 次的 CPU 时间略长,为在较短的运行时间内测试所提算法的有效性,兼顾算法对比的公平性,将 TGA 迭代 100 次所需的时间作为对比算法 MBO&NS、GMBOAL、AGA&LS、IDABC 和所提出算法 GMBOA 的停止条件,以此对比算法的性能,所以表 1~6 列出的 CPU 时间为 TGA 迭代 100 次的时间。表 1~6 列出了 5 种算法求解不同规模问题的实验结果。

表 1　忽略工序比例为 20% 时中小规模问题的测试结果

Table 1　Testing results for small and medium scale problems with the proportion 20% of missing operations

$n\times h$	TGA	MBO&NS	GMBOAL	AGA&LS	IDABC	GMBOA	R_1/%	R_2/%	R_3/%	R_4/%	R_5/%	CPU时间/s
20×5	209.9	208.2	184.1	176.5	188.9	172.3	21.82	20.84	6.85	2.44	9.63	22.01
30×5	280.1	260.3	224.2	218.1	228.4	203.7	37.51	27.79	10.06	7.07	12.13	25.19
40×5	318.6	284.9	261.5	242.4	261.7	232.0	37.33	22.80	12.72	4.48	12.80	28.84
50×5	335.7	335.0	285.0	271.8	296.0	263.2	27.55	27.28	8.28	3.27	12.46	31.44
20×10	337.3	355.8	299.0	285.8	300.9	277.1	21.73	28.40	6.85	3.14	8.59	44.14

续表 1

$n\times h$	TGA	MBO&NS	GMBOAL	AGA&LS	IDABC	GMBOA	R_1/%	R_2/%	R_3/%	R_4/%	R_5/%	CPU时间/s
30×10	387.5	409.7	358.6	338.2	364.1	327.1	18.47	25.25	9.63	3.39	11.31	52.65
40×10	475.7	438.4	390.3	378.7	395.0	362.9	31.08	20.80	7.55	4.35	8.85	60.41
50×10	513.3	501.8	431.8	405.4	425.2	399.2	28.58	25.70	8.17	1.55	6.51	67.85
20×15	408.1	431.7	373.8	376.8	405.6	358.2	13.93	20.52	4.36	5.19	13.23	70.29
30×15	506.5	530.0	441.6	421.2	441.6	409.8	23.60	29.33	7.76	2.78	7.76	84.45
40×15	542.2	582.1	478.4	468.4	484.5	451.4	20.12	28.95	5.98	3.77	7.33	98.44
50×15	664.6	633.4	530.2	522.6	535.4	518.2	28.25	22.23	2.32	0.85	3.32	111.03
20×20	562.7	572.3	463.0	453.8	509.5	447.2	25.83	27.97	3.53	1.48	13.93	100.58
30×20	578.7	635.0	526.5	506.2	538.9	495.6	16.77	28.13	6.23	2.14	8.74	119.15
40×20	752.3	693.1	568.8	565.3	580.3	552.3	36.21	25.49	2.99	2.35	5.07	136.65
50×20	792.7	782.2	635.1	637.5	640.0	634.5	24.93	23.28	0.09	0.47	0.87	156.92
平均	479.1	478.4	403.2	391.8	412.3	381.5	25.86	25.30	6.46	3.05	8.91	75.63

表 2　忽略工序比例为 20% 时大规模问题测试结果

Table 2　Testing results for large scale problems with the proportion 20% of missing operations

$n\times h$	TGA	MBO&NS	GMBOAL	AGA&LS	IDABC	GMBOA	R_1/%	R_2/%	R_3/%	R_4/%	R_5/%	CPU时间/s
80×5	475.1	468.6	414.5	414.5	408.8	375.0	26.69	24.96	10.53	1.75	9.01	40.47
100×5	563.0	555.2	499.5	461.1	495.6	443.4	26.97	25.21	12.65	3.99	11.77	49.34
120×5	574.9	639.4	564.6	520.4	556.1	508.8	12.99	25.21	10.97	2.28	9.30	58.26
150×5	715.7	742.2	640.7	607.3	644.3	587.6	21.80	26.31	9.04	3.35	9.65	68.37
80×10	679.6	641.9	551.3	538.6	535.8	527.2	28.91	21.76	4.57	2.16	1.63	96.02
100×10	748.5	743.6	635.8	614.0	620.9	605.2	23.68	22.87	5.06	1.45	2.59	112.20
120×10	893.1	844.7	748.7	706.3	707.7	690.6	29.32	22.31	8.41	2.27	2.48	125.84
150×10	1 041.7	975.5	841.4	788.7	810.0	769.4	35.39	26.79	9.36	2.51	5.28	150.04
80×15	870.7	838.0	700.7	680.3	673.6	667.4	30.46	25.56	4.99	1.93	0.93	150.26
100×15	1 037.4	933.4	783.6	780.8	765.2	764.8	35.64	22.04	2.46	2.09	0.05	181.06
120×15	1 100.4	1 029.9	889.2	878.1	867.8	867.2	26.89	18.76	2.54	1.26	0.07	213.00
150×15	1 234.9	1 091.1	996.5	972.4	961.5	960.9	28.51	13.55	3.70	1.20	0.06	259.82
80×20	989.2	981.7	825.9	803.0	796.5	795.0	24.43	23.48	3.89	1.01	0.19	218.39
100×20	1 117.7	1 095.5	922.4	914.8	895.5	894.5	24.95	22.47	3.12	2.27	0.11	262.73
120×20	1 326.5	1 211.5	1 020.4	1 003.6	991.2	989.7	34.03	22.41	3.10	1.40	0.15	303.08
150×20	1 305.6	1 368.0	1 150.1	1 135.1	1 124.5	1 123.3	16.23	21.78	2.39	1.05	0.11	380.04
平均	917.1	885.0	761.6	738.7	740.9	723.1	26.68	22.84	6.05	2.00	3.34	166.81

表 3　忽略工序比例为 40% 时中小规模问题测试结果

Table 3　Testing results for small and medium scale problems with the proportion 40% of missing operations

$n\times h$	TGA	MBO&NS	GMBOAL	AGA&LS	IDABC	GMBOA	R_1/%	R_2/%	R_3/%	R_4/%	R_5/%	CPU时间/s
20×5	163.7	183.3	149.9	145.4	151.5	143.6	14.00	27.65	4.39	1.25	5.50	20.95
30×5	212.2	209.5	175.1	172.7	183.3	163.1	30.10	28.45	7.36	5.89	12.39	24.42
40×5	253.3	209.8	189.5	184.0	197.8	179.4	41.19	16.95	5.63	2.56	10.26	27.81

续表 3

$n\times h$	TGA	MBO&NS	GMBOAL	AGA&LS	IDABC	GMBOA	R_1/%	R_2/%	R_3/%	R_4/%	R_5/%	CPU时间/s
50×5	271.0	233.6	218.8	218.7	221.7	204.3	32.65	14.34	7.10	7.05	8.52	30.58
20×10	286.5	276.2	241.8	232.1	250.0	226.7	26.38	21.84	6.66	2.38	10.28	41.27
30×10	343.5	304.7	280.0	267.8	283.2	260.2	32.01	17.10	7.61	2.92	8.12	51.89
40×10	322.9	339.7	292.1	285.8	298.0	272.5	18.50	24.66	7.19	4.88	9.36	58.25
50×10	364.0	380.0	319.9	319.7	329.6	306.0	18.95	24.18	4.54	4.48	7.71	65.19
20×15	370.0	353.0	297.5	298.2	327.4	283.7	30.42	24.43	4.86	5.11	15.40	63.90
30×15	395.7	394.5	341.9	342.9	350.4	323.8	22.21	21.83	5.59	5.90	8.21	77.76
40×15	428.9	429.6	349.1	356.5	362.0	339.6	26.30	26.50	2.80	4.98	6.60	91.36
50×15	548.7	470.1	393.0	394.8	397.5	385.5	42.33	21.95	1.95	2.41	3.11	102.45
20×20	402.0	425.0	344.8	346.6	391.3	335.4	19.86	26.71	2.80	3.34	16.67	88.09
30×20	488.9	484.3	401.0	404.3	435.3	393.0	24.40	23.23	2.04	2.88	10.76	107.85
40×20	465.8	530.7	425.6	423.3	450.3	412.4	12.95	28.69	3.20	2.64	9.19	126.48
50×20	575.9	586.1	474.0	469.0	491.8	458.0	25.74	27.97	3.49	2.40	7.38	144.08
平均	368.3	363.1	305.9	303.9	320.1	293.0	26.12	23.53	4.83	3.82	9.34	70.15

表 4　忽略工序比例为 40% 时大规模问题测试结果

Table 4　Testing results for large scale problems with the proportion 40% of missing operations

$n\times h$	TGA	MBO&NS	GMBOAL	AGA&LS	IDABC	GMBOA	R_1/%	R_2/%	R_3/%	R_4/%	R_5/%	CPU时间/s
80×5	331.6	353.6	293.3	291.7	307.3	277.2	19.62	27.56	5.81	5.23	10.86	40.12
100×5	413.2	416.8	364.7	348.7	368.1	332.2	24.38	25.47	9.78	4.97	9.75	46.86
120×5	466.1	475.7	412.9	410.4	423.5	383.3	21.60	24.11	7.72	7.07	10.49	53.21
150×5	560.1	529.9	473.7	452.7	476.4	422.5	32.57	25.42	12.12	7.15	12.76	62.78
80×10	463.4	494.2	411.2	404.2	410.2	390.5	18.67	26.56	5.30	3.51	5.04	89.57
100×10	571.2	555.8	475.9	462.7	480.4	456.2	25.21	21.83	4.32	1.42	5.30	103.36
120×10	687.2	618.9	537.1	528.6	537.1	513.4	33.85	20.55	4.62	2.96	4.62	118.47
150×10	688.3	650.9	598.8	590.5	610.0	575.2	19.66	13.16	4.10	2.66	6.05	143.00
80×15	650.9	626.3	521.2	509.0	510.7	500.0	30.18	25.26	4.24	1.80	2.14	141.67
100×15	704.0	686.8	588.1	577.3	568.1	564.6	24.69	21.64	4.16	2.25	0.62	168.76
120×15	742.8	767.4	659.5	648.0	635.2	629.5	18.00	21.91	4.77	2.94	0.91	196.93
150×15	892.9	889.7	741.8	735.1	724.7	719.7	24.07	23.62	3.07	2.14	0.69	225.99
80×20	783.6	716.8	599.4	595.7	585.4	581.6	34.73	23.25	3.06	2.42	0.65	198.27
100×20	911.1	811.5	685.0	671.6	665.8	663.8	37.26	22.25	3.19	1.18	0.30	233.16
120×20	987.4	923.9	778.0	761.2	755.5	754.8	30.82	22.40	3.07	0.85	0.09	269.46
150×20	1 106.6	1 013.6	846.0	840.6	835.1	834.9	32.54	21.40	1.33	0.68	0.02	325.50
平均	685.0	658.2	561.7	551.8	555.8	537.5	26.74	22.90	5.04	3.08	4.39	151.07

表 5　忽略工序比例为 60% 时中小规模问题测试结果

Table 5　Testing results for small and medium scale problems with the proportion 60% of missing operations

$n\times h$	TGA	MBO&NS	GMBOAL	AGA&LS	IDABC	GMBOA	R_1/%	R_2/%	R_3/%	R_4/%	R_5/%	CPU时间/s
20×5	108.0	103.9	88.0	93.4	112.4	87.7	23.15	18.47	0.34	6.50	28.16	19.03

续表 5

$n\times h$	TGA	MBO&NS	GMBOAL	AGA&LS	IDABC	GMBOA	R_1/%	R_2/%	R_3/%	R_4/%	R_5/%	CPU时间/s
30×5	131.4	128.9	110.9	121.2	119.1	108.2	21.44	19.13	2.50	12.01	10.07	22.81
40×5	144.6	123.9	116.6	121.6	126.9	113.0	27.96	9.65	3.19	7.61	12.30	26.38
50×5	184.9	159.8	135.8	150.3	155.2	130.8	41.36	22.17	3.82	14.91	18.65	28.89
20×10	179.0	178.4	161.9	159.3	196.9	161.1	11.11	10.74	0.50	-1.12	22.22	37.10
30×10	243.3	214.0	193.3	189.3	196.9	184.8	31.66	15.80	4.60	2.44	6.55	46.70
40×10	253.4	245.9	202.6	202.6	210.2	193.6	30.89	27.01	4.65	4.65	8.57	53.31
50×10	289.1	249.6	218.0	217.0	230.1	205.6	40.61	21.40	6.03	5.54	11.92	59.73
20×15	271.9	262.7	230.5	222.7	265.3	224.8	20.95	16.86	2.54	-0.93	18.02	56.62
30×15	299.2	273.6	236.6	241.6	258.3	229.7	30.26	19.11	3.00	5.18	12.45	71.15
40×15	307.0	315.5	257.0	262.8	277.1	252.5	21.58	24.95	1.78	4.08	9.74	83.09
50×15	345.8	335.5	271.7	280.4	286.0	265.6	30.20	26.32	2.30	5.57	7.68	94.93
20×20	296.0	303.1	260.1	269.4	313.3	263.4	12.38	15.07	-1.25	2.28	18.94	78.94
30×20	336.9	352.3	291.0	292.9	318.6	284.2	18.54	23.96	2.39	3.06	12.10	98.34
40×20	401.7	367.5	304.5	316.8	331.3	300.7	33.59	22.21	1.26	5.35	10.18	114.12
50×20	434.9	415.1	316.5	328.1	337.8	314.6	38.24	31.95	0.60	4.29	7.37	130.60
平均	264.2	251.9	212.2	216.8	233.5	207.5	27.12	20.30	2.39	5.09	13.43	63.86

表 6　忽略工序比例为 60% 时大规模问题测试结果

Table 6　Testing results for large scale problems with the proportion 60% of missing operations

$n\times h$	TGA	MBO&NS	GMBOAL	AGA&LS	IDABC	GMBOA	R_1/%	R_2/%	R_3/%	R_4/%	R_5/%	CPU时间/s
80×5	255.9	204.0	186.5	194.0	189.7	179.0	42.96	13.97	4.19	8.38	5.98	38.14
100×5	281.2	241.9	220.3	232.4	235.2	212.4	32.39	13.89	3.72	9.42	10.73	43.26
120×5	318.6	289.1	261.0	265.3	284.7	252.4	26.23	14.54	3.41	5.11	12.80	51.75
150×5	371.3	349.8	312.7	308.0	342.0	307.5	20.75	13.76	1.69	0.16	11.22	58.90
80×10	359.3	328.0	277.3	274.5	278.8	262.2	37.03	25.10	5.76	4.69	6.33	79.09
100×10	407.8	388.5	332.1	336.7	322.3	316.1	29.01	22.90	5.06	6.52	1.96	92.60
120×10	394.9	424.3	367.6	364.2	364.9	342.4	15.33	23.92	7.36	6.37	6.57	106.71
150×10	507.5	477.6	401.4	399.1	407.1	387.6	30.93	23.22	3.56	2.97	5.03	126.37
80×15	414.2	445.1	347.4	362.0	362.8	346.4	19.57	28.49	0.29	4.50	4.73	127.90
100×15	480.3	465.7	381.5	389.6	385.8	373.0	28.77	24.85	2.28	4.45	3.43	148.57
120×15	588.4	534.1	438.9	437.9	430.5	428.1	37.44	24.76	2.52	2.29	0.56	170.06
150×15	598.2	564.3	479.1	491.8	483.6	465.9	28.40	21.12	2.83	5.56	3.80	205.08
80×20	527.7	495.6	411.4	412.8	423.4	401.3	31.50	23.50	2.52	2.87	5.51	178.11
100×20	579.6	548.5	452.9	455.5	450.6	447.5	29.52	22.57	1.21	1.79	0.69	208.59
120×20	641.0	594.3	505.5	501.3	496.7	494.1	29.73	20.28	2.31	1.46	0.05	241.16
150×20	766.8	675.5	566.7	568.4	557.3	556.6	37.77	21.36	1.81	2.12	0.13	288.17
平均	468.3	439.1	371.4	374.6	376.0	360.8	29.83	21.14	3.16	4.29	4.97	135.28

从表 1~6 可知：

1）当p = 20%时，对于中小规模问题，在平均 CPU 时间 75.63 s 内，由 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 得到的平均目标值分别为 479.1、478.4、403.2、391.8、412.3、381.5。GMBOA 得到的目标值较 TGA 改进 25.86%，较 MBO&NS 改进 25.30%，较 GMBOAL 改进 6.46%，较 AGA&LS 改进 3.05%，较 IDABC 改进 8.91%。

对于大规模问题，在平均 CPU 时间 166.81 s 内，由 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 得到的平均目标值分别为 917.1、885.0、761.6、738.7、740.9、723.1。GMBOA 的目标值较 TGA 改进 26.68%，较 MBO&NS 改进 22.84%，较 GMBOAL 改进 6.05%，较 AGA&LS 改进 2.00%，较 IDABC 改进 3.34%。

从平均性能来看，对于不同规模问题，在总平均时间 121.22 s 内，TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 的平均目标值分别为 698.1、681.7、582.4、565.3、576.6、552.3。GMBOA 的目标值较 TGA 改进 26.27%，较 MBO&NS 改进 24.07%，较 GMBOAL 改进 6.26%，较 AGA&LS 改进 2.53%，较 IDABC 改进 6.13%。整体来看，在相同 CPU 时间内，GMBOA 的表现明显优于 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC。

2）当p = 40%时，对于中小规模问题，在平均 CPU 时间 70.15 s 内，由 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 得到的平均目标值分别为 368.3、363.1、305.9、303.9、320.1、293.0。GMBOA 的目标值较 TGA 改进 26.12%，较 MBO&NS 改进 23.53%，较 GMBOAL 改进 4.83%，较 AGA&LS 改进 3.82%，较 IDABC 改进 9.34%。

对于大规模问题，在平均 CPU 时间 151.07 s 内，由 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 得到的平均目标值分别为 685.0、658.2、561.7、551.8、555.8、537.5。GMBOA 的目标值较 TGA 改进 26.74%，较 MBO&NS 改进 22.90%，较 GMBOAL 改进 5.04%，较 AGA&LS 改进 3.08%、较 IDABC 改进 4.39%。

从平均性能来看，对于不同规模问题，在总平均时间 110.61 s 内，TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 的平均目标值分别为 526.7、510.7、433.8、427.9、438.0、415.3。GMBOA 的目标值较 TGA 改进 26.43%，较 MBO&NS 改进 23.22%，较 GMBOAL 改进 4.94%，较 AGA&LS 改进 3.45%，较 IDABC 改进 6.87%。因此，GMBOA 比 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 产生了较好的解。

3）当p = 60%时，对于中小规模问题，在平均 CPU 时间 63.86 s 内，由 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 得到的平均目标值分别为 264.2、251.9、212.2、216.8、233.5、207.5。GMBOA 的目标值较 TGA 改进 27.12%，较 MBO&NS 改进 20.30%，较 GMBOAL 改进 2.39%，较 AGA&LS 改进 5.09%，较 IDABC 改进 13.43%。

对于大规模问题，在平均 CPU 时间 135.28 s 内，由 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 得到的平均目标值分别为 468.3、439.1、371.4、374.6、376.0、360.8。GMBOA 的目标值较 TGA 改进 29.83%，较 MBO&NS 改进 21.14%，较 GMBOAL 改进 3.16%，较 AGA&LS 改进 4.29%、较 IDABC 改进 4.97%。

虽然小规模问题 20×10、20×15 和 20×20 的 3 个实例中 GMBOA 的目标值略差于 AGA&LS 或 GMBOAL，但随着问题规模的增大，GMBOA 得到的解的质量一致优于其他 5 种算法。

从平均性能来看，对于不同规模问题，在总平均时间 99.57s 内，TGA、MBO&NS、GMBOAL、AGA&LS、IDABC 和 GMBOA 的平均目标值分别为 366.3、345.5、291.8、295.7、304.8、284.2。GMBOA 的目标值较 TGA 改进 28.48%，较 MBO&NS 改进 20.72%，较 GMBOAL 改进 2.78%，较 AGA&LS 改进 4.69%，较 IDABC 改进 9.20%。因此，GMBOA 表现最好，尤其是对于大规模问题。

4）综上可知，在相同 CPU 时间内，虽然当忽略工序比例较高时 GMBOA 求解小规模问题的一些实例的表现略差于其他算法，但平均性能均优于 TGA、MBO&NS、GMBOAL、AGA&LS、IDABC。

4　结束语

本文研究了含忽略工序和不相关并行机的混合流水车间调度问题，以最小化最大完工时间为目标建立了整数规划模型，进而结合全局搜索、自适应遗传算法和候鸟优化提出一种遗传候鸟优化算法以获取近优解。首先，设计基于机器号的编码方案，采用全局搜索和随机程序生成初始种群，然后执行随迭代进化过程而调整的自适应交叉和变异操作，从而得到改进的 GA 解，引入基于 3 种邻域结构的候鸟优化算法扩大解的搜索空间以更新 GA 解。大量随机数据的仿真实验证明所提遗传候鸟优化算法能够在相同的 CPU 时间

内得到满意的近优解。未来研究可将所提算法推广到多目标 HFSMO 问题或尝试其他近似算法(如禁忌搜索、人工蜂群等)求解含不相关并行机的 HFSMO 问题。

参考文献:

[1] LONG Jianyu, ZHENG Zhong, GAO Xiaoqiang, et al. Scheduling a realistic hybrid flow shop with stage skipping and adjustable processing time in steel plants[J]. Applied soft computing, 2018, 64: 536–549.

[2] RUIZ R, VÁZQUEZ-RODRÍGUEZ J A. The hybrid flow shop scheduling problem[J]. European journal of operational research, 2010, 205(1): 1–18.

[3] TSENG C T, LIAO C J, LIAO Taixiang. A note on two-stage hybrid flowshop scheduling with missing operations[J]. Computers and industrial engineering, 2008, 54(3): 695–704.

[4] SARAVANAN M, SRIDHAR S, HARIKANNAN N. Application of meta-heuristics for minimization of makespan in multi-stage hybrid flow shop scheduling problems with missing operations[J]. International review of mechanical engineering, 2014, 8(5): 916–923.

[5] SARAVANAN M, SRIDHAR S, HARIKANNAN N. Optimization of realistic multi-stage hybrid flow shop scheduling problems with missing operations using meta-heuristics[J]. International journal of engineering and technology, 2014, 6(1): 484–496.

[6] MARICHELVAM M K, PRABAHARAN T. Performance evaluation of an improved hybrid genetic scatter search (IHGSS) algorithm for multistage hybrid flow shop scheduling problems with missing operations[J]. International journal of industrial and systems engineering, 2014, 16(1): 120–141.

[7] DIOS M, FERNANDEZ-VIAGAS V, FRAMINAN J M. Efficient heuristics for the hybrid flow shop scheduling problem with missing operations[J]. Computers and industrial engineering, 2018, 115: 88–99.

[8] DIOS M, FRAMINAN J M. Constructive heuristics comparison in hybrid flow shop scheduling environments with missing operations[C]//Proceedings of 2015 International Conference on Industrial Engineering and Systems Management. Seville: IEEE, 2015: 858−868.

[9] SARAVANAN M, SRIDHAR S, HARIKANNAN N. Minimization of mean tardiness in hybrid flow shop with missing operations using genetic algorithm[J]. Journal of advanced manufacturing systems, 2016, 15(2): 43–55.

[10] LI Junqing, PAN Quanke, DUAN Peiyong. An improved artificial bee colony algorithm for solving hybrid flexible flowshop with dynamic operation skipping[J]. IEEE transactions on cybernetics, 2016, 46(6): 1311–1324.

[11] MENG Leilei, ZHANG Chaoyong, SHAO Xinyu, et al. Mathematical modelling and optimisation of energy-conscious hybrid flow shop scheduling problem with unrelated parallel machines[J]. International journal of production research, 2019, 57(4): 1119–1145.

[12] QIN Wei, ZHUANG Zilong, LIU Yang, et al. A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly[J]. Computers and industrial engineering, 2019, 138: 106115.

[13] 罗函明, 罗天洪, 吴晓东, 等. 求解混合流水车间调度问题的离散布谷鸟算法 [J]. 计算机工程与应用, 2020, 56(22): 264–271.
LUO Hanming, LUO Tianhong, WU Xiaodong, et al. Discrete cuckoo search algorithm for solving hybrid flow-shop scheduling problem[J]. Computer engineering and applications, 2020, 56(22): 264–271.

[14] 轩华, 郑倩倩, 李冰. 带不相关并行机和有限缓冲 MHFS 调度的混合启发式算法 [J]. 控制与决策, 2021, 36(3): 565–576.
XUAN Hua, ZHENG Qianqian, LI Bing. Hybrid heuristic algorithm for multi-stage hybrid flow shop scheduling with unrelated parallel machines and finite buffers [J]. Control and decision, 2021, 36(3): 565–576.

[15] ZHOU Binghai, LIU Wenlong. Energy-efficient multi-objective scheduling algorithm for hybrid flow shop with fuzzy processing time[J]. Proceedings of the institution of mechanical engineers, part I:journal of systems and control engineering, 2019, 233(10): 1282–1297.

[16] YU Chunlong, ANDREOTTI P, SEMERARO Q. Multi-objective scheduling in hybrid flow shop: evolutionary algorithms using multi-decoding framework[J]. Computers & industrial engineering, 2020, 147: 106570.

[17] ZHOU Rui, LEI Deming, ZHOU Xinmin. Multi-objective energy-efficient interval scheduling in hybrid flow shop using imperialist competitive algorithm[J]. IEEE access, 2019, 7: 85029–85041.

[18] BENKALAI I, REBAINE D, GAGNÉ C, et al. The migrating birds optimization metaheuristic for the permutation flow shop with sequence dependent setup times[J]. IFAC-Papers on line, 2016, 49(12): 408–413.

[19] TONGUR V, ÜLKER E. Migrating birds optimization for flow shop sequencing problem[J]. Journal of computer and communications, 2014, 2(4): 142–147.

[20] 张素君, 顾幸生. 基于离散候鸟迁徙优化算法的置换流水车间调度问题 [J]. 华东理工大学学报(自然科学版), 2016, 42(3): 412–419.

ZHANG Sujun, GU Xingsheng. A discrete migrating birds optimization algorithm for permutation flow shop scheduling problem[J]. Journal of East China University of Science and Technology (natural science edition), 2016, 42(3): 412–419.

[21] ZHANG Sujun, GU Xingsheng, ZHOU Funa. An improved discrete migrating birds optimization algorithm for the no-wait flow shop scheduling problem[J]. IEEE access, 2020, 8: 99380–99392.

[22] 任彩乐, 张超勇, 孟磊磊, 等. 基于改进候鸟优化算法的混合流水车间调度问题 [J]. 计算机集成制造系统, 2019, 25(3): 643–653.
REN Caile, ZHANG Chaoyong, MENG Leilei, et al. Hybrid flow-shop scheduling problems based on improved migrating birds optimization algorithm[J]. Computer integrated manufacturing systems, 2019, 25(3): 643–653.

[23] ZHANG Biao, PAN Quanke, GAO Liang, et al. A multi-objective migrating birds optimization algorithm for the hybrid flowshop rescheduling problem[J]. Soft computing, 2019, 23(17): 8101–8129.

[24] LI Hongchan, CAO Bangqin, ZHU Haodong. A variable neighborhood migrating birds optimization algorithm for flexible job shop scheduling[J]. International journal of performability engineering, 2017, 13(7): 1020–1029.

[25] LI Hongchan, ZHU Haodong, JIANG Tianhua. Modified migrating birds optimization for energy-aware flexible job shop scheduling problem[J]. Algorithms, 2020, 13(2): 44.

[26] ZHANG Mei, TAN Yingtong, ZHU Jinhui, 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.

[27] 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.

[28] 张国辉, 高亮, 李培根, 等. 改进遗传算法求解柔性作业车间调度问题 [J]. 机械工程学报, 2009, 45(7): 145–151.
ZHANG Guohui, GAO Liang, LI Peigeng, et al. Improved genetic algorithm for the flexible job-shop scheduling problem[J]. Journal of mechanical engineering, 2009, 45(7): 145–151.

[29] CHANG H C, LIU T K. Optimisation of distributed manufacturing flexible job shop scheduling by using hybrid genetic algorithms[J]. Journal of intelligent manufacturing, 2017, 28(8): 1973–1986.

[30] XUAN Hua, ZHENG Huixian, LI Bing. An improved discrete artificial bee colony algorithm for flexible flow-shop scheduling with step deteriorating jobs and sequence-dependent setup times[J]. Mathematical problems in engineering, 2019, 2019: 8520503.

作者简介：

轩华，教授，主要研究方向为生产计划与调度、物流优化与控制。主持国家自然科学基金项目、教育部人文社科项目、博士后基金特殊资助项目、博士后基金面上项目等科研项目 8 项。发表学术论文 42 篇。

樊银格，硕士研究生，主要研究方向为物流优化与控制。

李冰，教授，博士生导师，主要研究方向为物流优化与控制。河南省省级青年骨干教师。发表学术论文 36 篇。