控制与决策
Control and Decision

混合离散人工蜂群算法求解含不相关并行机的分布式柔性流水线调度

轩华, 李文婷, 李冰

引用本文:
轩华,李文婷,李冰. 混合离散人工蜂群算法求解含不相关并行机的分布式柔性流水线调度[J]. 控制与决策, 2023, 38(3): 779–789.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2021.1438

混合离散人工蜂群算法求解含不相关并行机的分布式柔性流水线调度

轩 华†, 李文婷, 李 冰

(郑州大学 管理学院，郑州 450001)

摘 要: 研究每阶段含不相关并行机的分布式柔性流水线调度问题. 考虑顺序相关准备时间和工件动态到达时间,以最小化总加权提前/拖期惩罚为目标建立整数规划模型,提出一种融合离散差分进化算法、变邻域下降算法和局域搜索的混合离散人工蜂群算法以获取近优解. 该算法采用基于工厂-工件号的编码以及基于机器最早空闲时间的动态解码机制,通过随机规则和均衡分派策略生成初始工厂-工件序列群,在引领蜂阶段引入离散差分进化算法产生优质工厂-工件序列,在跟随蜂阶段利用变邻域下降算法在被选择序列附近继续搜索以得到邻域序列,在侦察蜂阶段设计基于关键/非关键工厂间插入的局域搜索提高算法搜索能力. 通过仿真实验测试不同规模的算例,实验结果表明,所提出的混合离散人工蜂群算法表现出较好的求解性能.

关键词: 分布式柔性流水线调度；不相关并行机；混合离散人工蜂群算法；离散差分进化算法

中图分类号: N945 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2021.1438

引用格式: 轩华,李文婷,李冰. 混合离散人工蜂群算法求解含不相关并行机的分布式柔性流水线调度[J]. 控制与决策, 2023, 38(3): 779-789.

Hybrid discrete artificial bee colony algorithm for distributed flexible flowline scheduling with unrelated parallel machines

XUAN Hua†, LI Wen-ting, LI Bing

(School of Management，Zhengzhou University，Zhengzhou 450001，China)

Abstract: A distributed flexible flowline scheduling problem with unrelated parallel machines at each stage is studied. Considering sequence-dependent setup times and job dynamic arrival times, an integer programming model is established with the objective of minimizing total weighted earliness and tardiness penalty. A hybrid discrete artificial bee colony algorithm is proposed combined with the discrete differential evolution algorithm, the variable neighborhood descent algorithm and local search so as to obtain near optimal solutions. In this algorithm, factory-job number based encoding is applied and a dynamic decoding mechanism with the earliest machine idle time is designed. The initial factory-job sequence group is then generated by using a random rule and an average assignment strategy. In the leading bee phase, the discrete differential evolution algorithm is introduced to yield factory-job sequences with high quality. In the following bee phase, a variable neighborhood descent algorithm is used to search around the selected sequences in order to gain neighborhood sequences. Local search based on insertion between critical/non-critical factories is designed to enhance algorithm search ability in the scout bee phase. Simulation experiments are performed on different scale problems, and testing results demonstrate the proposed hybrid discrete artificial bee colony algorithm has a better resolution performance.

Keywords: distributed flexible flowline scheduling；unrelated parallel machines；hybrid discrete artificial bee colony algorithm；discrete differential evolution algorithm

0 引 言

在经济全球化、顾客需求多样化和市场竞争不确定性增大的趋势下,传统制造业正从集中式单工厂生产到分布式多工厂柔性生产模式转变[1]. 传统的车间调度主要解决单个工厂的资源配置与优化问题,而分布式车间调度需同时协调多个工厂的生产资源,已经成为生产调度领域的研究热点. 在实际生产环境中,并行机的磨损程度不同会影响工件的处理

收稿日期: 2021-08-16；**录用日期:** 2021-12-30.
基金项目: 国家自然科学基金项目(U1804151, U1604150).
责任编委: 刘民.
†通讯作者. E-mail: hxuan@zzu.edu.cn.

时间,而且同一台机器进行工件转换常需要进行机器清理或更换部件等操作,因此合理安排工件的机器分配和加工顺序具有重要的现实意义. 本文以提前/拖期惩罚的加权和最小化为生产目标,考虑顺序相关准备时间和工件动态到达时间,研究含不相关并行机的分布式柔性流水线调度问题(distributed flexible flowline scheduling problem with unrelated parallel machines, DFFSP-UPM).

近年来,很多学者对含不相关并行机的柔性流水线调度问题(flexible flowline scheduling problem with unrelated parallel machines, FFSP-UPM)进行了研究. 为了解决最大完工时间(makespan)问题,秦浩翔等[2]提出双层变异迭代贪婪算法,Sun等[3]提出混合分布估计算法. 为最小化总加权完工时间,Xuan等[4]考虑恶化工件和顺序相关准备时间,提出了改进离散人工蜂群算法.

在分布式车间调度领域,各分布式工厂包含的机器环境有流水线、柔性流水线、柔性作业车间等. 对于分布式阻塞流水线调度,Shao等[5]和Zhang等[6]分别提出了混合增强离散果蝇优化算法和混合离散差分进化算法以解决makespan问题. 针对分布式置换流水线调度,Huang等[7]考虑顺序相关准备时间,提出了构造式启发式算法和离散人工蜂群(discrete artificial bee colony, DABC)算法以最小化makespan; Khare等[8]提出了混合离散Harris鹰算法和迭代贪婪算法以最小化总拖期.

FFSP与分布式调度的集成则形成DFFSP. 在等同并行机假设下,以最小化makespan为目标,Shao等[9]提出基于分解策略的构造式启发式算法和多邻域迭代贪婪算法. 在均匀并行机假设下,Zheng等[10]针对模糊DFFSP提出了结合分布估计算法和迭代贪婪搜索的协同进化算法以同时优化模糊总拖期和鲁棒性. 在不相关并行机假设下,Shao等[11]考虑无等待约束,应用多种构造式启发式算法以解决最大完成时间问题;雷德明等[12]考虑顺序相关准备时间,提出多班教学优化算法以同时最小化makespan和最大延迟时间. Li等[13-14]构建了基于机器位置的数学模型,并提出DABC算法求解了含等同/不相关并行机的DFFSP.

对于分布式柔性作业车间调度,吴锐等[15]设计了改进人工蜂群算法以最小化makespan; Li等[16]提出基于帕累托的混合禁忌搜索算法以同时实现最小化makespan、最大负荷、总负荷和提前/拖期惩罚4个目标.

综上所述,关于DFFSP的研究多以makespan为优化目标,机器环境多为等同机,缺少总加权提前/拖期惩罚问题以及不相关并行机环境的综合考量. 上述分布式车间调度主要针对所有工件在零时刻可用的静态确定性问题,由于车间地理位置偏差和订单释放的不确定性等,待加工工件进入生产车间之前实际需要消耗一定的时间用于运输等,且已有的关于工件动态到达时间的研究多出现在单机、并行机、流水线、单件车间和开放车间调度[17-18],故考虑工件动态到达时间的DFFSP-UPM会使所研究问题更贴近实际生产. 因此,本文提出融合离散差分进化算法(discrete differential evolution, DDE)、变邻域下降(variable neighborhood descent, VND)算法和局域搜索(local search, LS)的混合离散人工蜂群(hybrid discrete artificial bee colony, HDABC)算法,以解决含顺序相关准备时间和动态到达时间的DFFSP-UPM的总加权提前/拖期惩罚问题.

1 问题建模

1.1 问题描述

有F个位于不同地理位置的相同工厂,每个工厂包含一个由G个生产阶段组成的柔性流水线,工厂f内的阶段t由U_t台不相关并行机构成,且至少存在一个阶段$U_t > 1$. 假设有N个相互独立的工件,每个工件j在时刻$r_j(r_j > 0)$到达生产车间的始端,且有一个交货期d_j. 每个工件在任一工厂依次访问G个生产阶段,在同一台机器上加工的两个相邻工件之间需要一个准备时间,其大小取决于工件的加工顺序. 一台机器一次至多加工一个工件,且一个工件只能在一台机器上加工. 根据三域表示法,该问题可表述为

$$\mathrm{DFFG}(U_1, U_2, \ldots, U_G)|P_{jftm} = P_{jtm}|E/T.$$

其中: $\mathrm{DFFG}(U_1, U_2, \ldots, U_G)$表示每阶段有$U_t$台不相关并行机的分布式柔性流水线,$P_{jftm} = P_{jtm}$表示每个工件在任一工厂内阶段$t$机器$m$上的处理时间都相同,$E/T$表示优化目标是总加权提前/拖期惩罚达到最小.

1.2 符号定义

1) 参数.

f: 工厂标号,$f \in \{1, \ldots, F\}$,F为工厂数;

j: 工件标号,$j \in \{1, \ldots, N\}$,N为工件数;

t: 生产阶段标号,$t \in \{1, \ldots, G\}$,G为生产阶段数;

m: 机器标号,$m \in \{1, \ldots, U_t\}$,U_t为阶段t的不相关并行机数;

P_{jtm}: 工件j在阶段t的机器m上的处理时间;

$S_{j'jtm}$: 在阶段t的机器m上从工件j'到工件j的准备时间,若在机器m上工件j紧随工件j'之后加工,则$S_{j'jtm}>0$,否则$S_{j'jtm}=0$;

r_j: 工件j的动态到达时间;

d_j: 工件j的交货期;

L: 一个足够大的正数;

α_j: 工件j的提前惩罚系数;

β_j: 工件j的延迟惩罚系数;

E_j: 工件j的提前期;

T_j: 工件j的拖期.

2) 决策变量.

B_{jt}: 工件j在生产阶段t的开工时间;

C_{jt}: 工件j在生产阶段t的完工时间;

CT_j: 工件j的完工时间;

X_{jf}: 0-1二元变量,若工件j分配至工厂f加工,则$X_{jf}=1$,否则$X_{jf}=0$;

Y_{jftm}: 0-1二元变量,若工件j在工厂f内阶段t的机器m上加工,则$Y_{jftm}=1$,否则$Y_{jftm}=0$;

$Z_{j'jftm}$: 0-1二元变量,若工件j'和j均分配至工厂f内阶段t的机器m上加工且工件j'在工件j的紧前面,则$Z_{j'jftm}=1$,否则$Z_{j'jftm}=0$.

1.3 模型构建

$$\min\ \mathrm{TWET}=\sum_{j=1}^{N}(\alpha_jE_j+\beta_jT_j). \tag{1}$$

$$\text{s.t. } C_j+E_j-T_j=d_j,\ \forall\, j; \tag{2}$$

$$E_j\geqslant d_j-B_{jG}-\sum_{f=1}^{F}\sum_{m=1}^{U_G}Y_{jfGm}(P_{jGm}+S_{j'jGm}),\ \forall\, j',j,\ j'\neq j; \tag{3}$$

$$T_j\geqslant B_{jG}+\sum_{f=1}^{F}\sum_{m=1}^{U_G}Y_{jfGm}(P_{jGm}+S_{j'jGm})-d_j,\ \forall\, j',j,j'\neq j; \tag{4}$$

$$\sum_{f=1}^{F}X_{jf}=1,\ \forall\, j; \tag{5}$$

$$X_{jf}=\sum_{m=1}^{U_t}Y_{jftm},\ \forall\, j,f,t; \tag{6}$$

$$B_{j1}\geqslant r_j,\ \forall\, j; \tag{7}$$

$$C_{jt}-C_{j,t-1}\geqslant\sum_{f=1}^{F}\sum_{m=1}^{U_t}Y_{jftm}(P_{jtm}+S_{j'jtm}),\ \forall\, j',j,t,j'\neq j,t\neq 1; \tag{8}$$

$$Z_{j'jftm}+Z_{jj'ftm}\leqslant 1,\ \forall\, j',j,f,t,m,j'\neq j; \tag{9}$$

$$C_{jt}\geqslant C_{j't}+\sum_{f=1}^{F}\sum_{m=1}^{U_t}Y_{jftm}(P_{jtm}+S_{j'jtm})-L\times(3-Y_{j'ftm}-Y_{jftm}-Z_{j'jftm}),\ \forall\, j',j,t,j'\neq j; \tag{10}$$

$$\mathrm{CT}_j=C_{jG},\ \forall\, j; \tag{11}$$

$$E_j,T_j\geqslant 0,\ \forall\, j; \tag{12}$$

$$B_{jt},C_{jt}\geqslant 0,\ \forall\, j,t; \tag{13}$$

$$X_{jf},Y_{jftm},Z_{j'jftm}\in\{0,1\},\ \forall\, j',j,f,t,m,j'\neq j. \tag{14}$$

式(1)为所研究问题的目标,即最小化总加权提前/拖期惩罚;约束(2)描述提前期和拖期与交货期之间的对应关系;约束(3)、(4)和(12)定义提前期和拖期;约束(5)说明每个工件只能在一个工厂加工;约束(6)确保每个工件只能在一台机器上加工;约束(7)说明工件在第一个生产阶段的开工时间需在它到达生产车间之后;约束(8)表示同一工厂内一个工件的工序优先级关系;约束(9)和(10)确保一台机器在任一时刻至多处理一个工件;约束(11)定义工件的完工时间;约束(13)和(14)说明决策变量的取值范围.

2 混合离散人工蜂群算法

ABC算法包含引领蜂、跟随蜂和侦察蜂3个搜索阶段. 鉴于混合算法可以平衡搜索的广度和深度,离散差分进化算法具有较强的全局搜索能力,基于问题特征的变邻域下降算法能对优质区域进行二次搜索,局域搜索可替换无法改进的序列以避免无效搜索,因此对每个阶段离散化改进后提出HDABC算法.

2.1 编码和解码

2.1.1 基于工厂-工件号的编码机制

考虑到分布式柔性流水线各个工厂之间的并行与独立性,提出基于工厂-工件号的整数编码机制,将调度解表述为长度为N的工厂-工件序列,它由工厂序列$\tau=\{\tau_f|f=1,2,\cdots,F\}$和工件子序列$\tau_f=[J_{f-1},J_{f-2},\cdots,J_{f-n^f}]$组成,其中$\tau_f$描述了在工厂$f$内第一个生产阶段加工的工件加工顺序,即它包含了分配至该工厂的工件号以及这些工件的加工顺序信息,且有$\sum_f n^f=N$.

2.1.2 基于机器最早空闲时间的动态解码机制

在阶段1按照已知工件子序列依次安排工件,而后续阶段则根据前一阶段各工件的完工时间和该阶段上机器的最早空闲时间来确定,故提出一种动态解码机制,具体过程如下:

阶段1: 对于τ_f中的第一个工件J_{f-1},从可选不相关并行机器集$\{UM_{1-1}^f,\dots,UM_{1-U_1}^f\}$选择处理时间最短的机器$q$作为该工件的加工机器,并从可选机器集内移除,继续按照最短处理时间规则依次为$J_{f-2},\dots,J_{f-U_1}$分派机器,并更新可选机器集直至完成前U_1个工件的机器分配,令$B_{J_{f-j}1}=r_{J_{f-j}}$,记录各机器的最早空闲时间idl_{1m}为分配至该机器的工件完工时间. 对于工件$J_{f-j}(j=U_1+1,\dots,n^f)$,选择具有最小$\mathrm{idl}_{1m}$值的机器$q$作为加工该工件的机器,令$B_{J_{f-j}1}=\min\{r_{J_{f-j}},\mathrm{idl}_{1q}+S_{J_{f-(j-1)}J_{f-j}1q}\}$,更新该机器的最早空闲时间$\mathrm{idl}_{1q}=C_{J_{f-j}1}$.

阶段$t(t>1)$: 将τ_f包含的工件按照阶段$t-1$的完工时间升序排列,对于前U_t个工件,按照最短处理时间规则依次分配至U_t台机器,令$B_{J_{f-j}t}=C_{J_{f-j},t-1}$,记录$\mathrm{idl}_{tm}$;对于其余工件$J_{f-j}(j=U_t+1,\dots,n^f)$,将其分配至具有最小$\mathrm{idl}_{tm}$值的机器$q$,令$B_{J_{f-j}t}=\max\{C_{J_{f-j},t-1},\mathrm{idl}_{tq}+S_{J_{f-(j-1)}J_{f-j}1q}\}$,更新$\mathrm{idl}_{tq}=C_{J_{f-j}t}$.

给定$\tau=[4,12,11,3,9,1;2,5,7,6,8,10]$;假设$G=U_t=3$;$S_{j'jtm}$仅与工件相关,即可表示为$S_{j'j}$;$r_j$设置为$\{3,2,2,1,2,3,2,3,2,3,2,1\}$;工件处理时间和准备时间如表1和表2所示.

表1 处理时间

t	P_{1tm}	P_{2tm}	P_{3tm}	P_{4tm}	P_{5tm}	P_{6tm}	P_{7tm}	P_{8tm}	P_{9tm}	P_{10tm}	P_{11tm}	P_{12tm}
1	2	5	9	9	3	5	8	2	7	9	8	4
	4	6	5	6	2	3	6	3	5	9	4	9
	7	8	2	3	5	9	4	5	5	5	5	9
2	4	9	5	3	4	2	6	9	8	8	9	6
	5	7	4	2	5	3	5	2	5	9	9	9
	6	5	3	5	8	4	7	6	5	5	7	8
3	6	5	3	3	4	5	4	5	3	5	2	7
	9	3	5	4	3	6	6	4	4	6	9	6
	4	6	8	2	2	4	8	9	5	9	9	7

表2 准备时间

j'	$S_{j'1}$	$S_{j'2}$	$S_{j'3}$	$S_{j'4}$	$S_{j'5}$	$S_{j'6}$	$S_{j'7}$	$S_{j'8}$	$S_{j'9}$	$S_{j'10}$	$S_{j'11}$	$S_{j'12}$
1	0	3	2	3	4	4	5	4	2	5	1	5
2	3	0	1	5	2	5	1	3	3	1	4	1
3	2	2	0	2	1	3	4	1	4	5	1	2
4	3	1	4	0	5	4	1	2	5	3	2	4
5	3	2	2	4	0	3	5	1	3	2	4	3
6	2	4	3	1	4	0	5	2	2	2	2	3
7	4	5	2	5	4	3	0	2	2	1	4	2
8	5	3	4	5	3	3	5	0	3	2	3	5
9	1	5	5	5	2	5	3	5	0	2	5	1
10	4	2	1	5	2	1	3	3	4	0	5	2
11	1	3	2	3	2	4	5	3	2	5	0	3
12	4	3	5	2	2	1	5	3	2	5	2	0

以工厂1为例说明具体解码过程(工厂2同理).

阶段1: $\tau_1=[4,12,11,3,9,1]$.

$B_{41}=r_4=1,$

$C_{41}=B_{41}+\min\{P_{411},P_{412},P_{413}\}=1+3=4,$

$Y_{4113}=1,\mathrm{idl}_{13}=C_{41}=4;$

$B_{12,1}=r_{12}=1,$

$C_{12,1}=B_{12,1}+\min\{P_{12,1,1},P_{12,1,2}\}=1+4=5,$

$Y_{12,1,1,1}=1,\mathrm{idl}_{11}=C_{12,1}=5;$

$B_{11,1}=r_{11}=2,$

$C_{11,1}=B_{11,1}+\min\{P_{11,1},2\}=2+4=6,$

$Y_{11,1,1,2}=1,\mathrm{idl}_{12}=C_{11,1}=6;$

$\min\{\mathrm{idl}_{11},\mathrm{idl}_{12},\mathrm{idl}_{13}\}=4\Rightarrow Y_{3113}=1,$

$B_{31}=\min\{r_3,\mathrm{idl}_{13}+S_{43}\}=7,$

$C_{31}=B_{31}+P_{313}=7+2=9,$

$\mathrm{idl}_{13}=C_{31}=9;$

$\min\{\mathrm{idl}_{11},\mathrm{idl}_{12},\mathrm{idl}_{13}\}=5\Rightarrow Y_{9111}=1,$

$B_{91}=\min\{r_9,\mathrm{idl}_{11}+S_{12,9}\}=7,$

$C_{91}=B_{91}+P_{911}=7+7=14,$

$\mathrm{idl}_{11}=C_{91}=14;$

$\min\{\mathrm{idl}_{11},\mathrm{idl}_{12},\mathrm{idl}_{13}\}=6\Rightarrow Y_{1112}=1,$

$B_{11}=\min\{r_1,\mathrm{idl}_{12}+S_{11,1}\}=7,$

$C_{11}=B_{11}+P_{112}=7+4=11,$

$\mathrm{idl}_{12}=C_{11}=11.$

阶段2: $4\rightarrow 12\rightarrow 11\rightarrow 3\rightarrow 1\rightarrow 9.$

$B_{42}=C_{41}=4,$

$C_{42}=B_{42}+\min\{P_{421},P_{422},P_{423}\}=$
$4+2=6,$

$Y_{4122}=1,\mathrm{idl}_{22}=C_{42}=6;$

$B_{12,2}=C_{12,1}=5,$

$C_{12,2}=B_{12,2}+\min\{P_{12,2,1},P_{12,2,3}\}=$
$5+6=11,$

$Y_{12,1,2,1}=1,\mathrm{idl}_{21}=C_{12,2}=11;$

$B_{11,2}=C_{11,1}=6,$

$C_{11,2}=B_{11,2}+\min\{P_{11,2,3}\}=6+7=13,$

$Y_{11,1,2,3}=1,\mathrm{idl}_{23}=C_{11,2}=13;$

$\min\{\mathrm{idl}_{21},\mathrm{idl}_{22},\mathrm{idl}_{23}\}=6\Rightarrow Y_{3122}=1,$

$B_{32}=\max\{C_{31},\mathrm{idl}_{22}+S_{43}\}=10,$

$C_{32}=B_{32}+P_{322}=10+4=14,$

$\mathrm{idl}_{22}=C_{32}=14;$

$\min\{\mathrm{idl}_{21},\mathrm{idl}_{22},\mathrm{idl}_{23}\}=11\Rightarrow Y_{1121}=1,$

$B_{12}=\max\{C_{11},\mathrm{idl}_{21}+S_{12,1}\}=15,$

$C_{12}=B_{12}+P_{121}=15+4=19,$

$\mathrm{idl}_{21}=C_{12}=19;$

$\min\{\mathrm{idl}_{21},\mathrm{idl}_{22},\mathrm{idl}_{23}\}=13\Rightarrow Y_{9123}=1,$

$B_{92}=\max\{C_{91},\mathrm{idl}_{23}+S_{11,9}\}=15,$

$C_{92}=B_{92}+P_{923}=15+5=20,$

$\mathrm{idl}_{23}=C_{92}=20.$

阶段3: $4\rightarrow 12\rightarrow 11\rightarrow 3\rightarrow 1\rightarrow 9.$

$B_{43}=C_{42}=6,$

$C_{43}=B_{43}+\min\{P_{431},P_{432},P_{433}\}=$
$6+\min\{3,4,2\}=8,$

$Y_{4133}=1,\mathrm{idl}_{33}=C_{43}=8;$

$B_{12,3}=C_{12,2}=11,$

$C_{12,3}=B_{12,3}+\min\{P_{12,3,1},P_{12,3,2}\}=$
$11+6=17,$

$Y_{12,1,3,2}=1,\mathrm{idl}_{32}=C_{12,3}=17;$

$B_{11,3}=C_{11,2}=13,$

$C_{11,3}=B_{11,3}+\min\{P_{11,3,1}\}=13+2=15,$

$Y_{11,1,3,1}=1,\mathrm{idl}_{31}=C_{11,3}=15;$

$\min\{\mathrm{idl}_{31},\mathrm{idl}_{32},\mathrm{idl}_{33}\}=8\Rightarrow Y_{3133}=1,$

$B_{33}=\max\{C_{32},\mathrm{idl}_{33}+S_{43}\}=14,$

$C_{33}=B_{33}+P_{333}=14+8=22,\mathrm{idl}_{33}=C_{33}=22;$

$\min\{\mathrm{idl}_{31},\mathrm{idl}_{32},\mathrm{idl}_{33}\}=15\Rightarrow Y_{1131}=1,$

$B_{13}=\max\{C_{12},\mathrm{idl}_{31}+S_{11,1}\}=19,$

$C_{13}=B_{13}+P_{131}=19+6=25,\mathrm{idl}_{31}=C_{13}=25;$

$\min\{\mathrm{idl}_{31},\mathrm{idl}_{32},\mathrm{idl}_{33}\}=17\Rightarrow Y_{9132}=1,$

$B_{93}=\max\{C_{92},\mathrm{idl}_{32}+S_{12,9}\}=20,$

$C_{93}=B_{93}+P_{932}=20+4=24,\mathrm{idl}_{32}=C_{93}=24.$

因此,按上述解码过程得到如图1所示的解码结果.

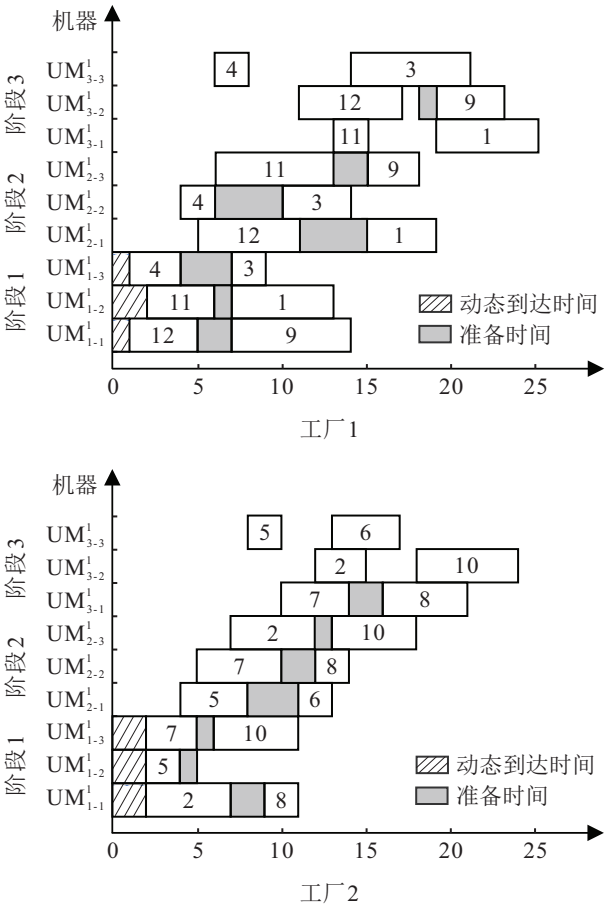


图 1　解码后的调度甘特图

2.2　初始工厂-工件序列群的产生

利用随机规则产生一个包含N个工件号的加工序列,然后引入均衡分派策略确定工件至工厂的分配,即取$\lfloor N/F\rfloor$作为每个工厂至少加工的工件数,随机产生$N-F\cdot\lfloor N/F\rfloor$个工厂,这些工厂处

理的工件数均为$\lfloor N/F\rfloor+1$,而其他工厂的工件数均为$\lfloor N/F\rfloor$,从而最终确定了各工厂加工的工件数n^f. 将加工序列的前n^1个工件分配至工厂1,第n^1+1至n^1+n^2个位置的工件归入工厂2,依次类推,第$n^1+n^2+\ldots+n^{F-1}+1$至$n^1+n^2+\ldots+n^F$个位置的工件归入工厂F. 最后,对每个工厂,将分配至该工厂的工件按照r_j升序排列形成最终的工件子序列τ_f.

2.3 引领蜂搜索阶段

引入DDE算法使引领蜂在当前工厂-工件序列群的邻域展开搜索,以获取新工厂-工件序列.

2.3.1 DE/best/1变异

采用“DE/best/1”变异方式,其中“DE”表示差分进化算法,“best”表示选择最优工厂-工件序列作为变异向量,“1”表示变异过程中差向量的个数为1. 令$H_a=[h_a(1),\ldots,h_a(k),\ldots,h_a(N)]$和$H_b=[h_b(1),\ldots,h_b(k),\ldots,h_b(N)]$为随机选择的两个工厂-工件序列,扩展Zhang等[6]的研究,得到变异算子

$$V_i^l=H_{\rm best}^{l-1}\oplus\sigma\otimes(H_a^{l-1}-H_b^{l-1})'. \tag{15}$$

其中:$a,b,i\in[1,{\rm NS}]$,NS为工厂-工件序列群规模;l为当前迭代数;σ为变异影响因子.

变异算子的执行如图2所示,包括3部分.

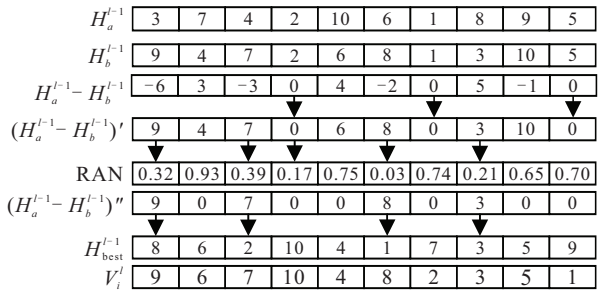


图2 变异算子

step 1: 对H_a^{l-1}和H_b^{l-1}进行差分算子得到差向量$(H_a^{l-1}-H_b^{l-1})'$. 将H_a^{l-1}和H_b^{l-1}内各对应元素相减得到向量$(H_a^{l-1}-H_b^{l-1})$,若其中有非零元素,则用H_b^{l-1}中对应位置的工件号取代该非零元素,即

$$\xi_i^l=(H_a^{l-1}-H_b^{l-1})'\Leftrightarrow$$
$$\xi_i^l(k)=\begin{cases}h_b^{l-1}(k),\ h_a^{l-1}(k)\neq h_b^{l-1}(k);\\0,\ \text{otherwise.}\end{cases} \tag{16}$$

其中$\xi=[\xi(1),\xi(2),\ldots,\xi(N)]$.

step 2: 根据σ和随机数rand,对差向量$(H_a^{l-1}-H_b^{l-1})'$做$\otimes$运算得到加权差向量$(H_a^{l-1}-H_b^{l-1})''$. 随机产生一个与工厂-工件序列相同维度的序列RAN,序列中元素在[0, 1]随机生成. 若${\rm rand}<\sigma$,则保留差向量ξ_i^l对应位置的元素不变. 否则,将差向量ξ_i^l对应位置的元素替换为0,即

$$\gamma_i^l=(H_a^{l-1}-H_b^{l-1})''=\sigma\otimes\xi_i^l\Leftrightarrow$$
$$\begin{cases}\xi_i^l(k),\ {\rm rand}<\sigma;\\0,\ \text{otherwise.}\end{cases} \tag{17}$$

其中$\gamma=[\gamma(1),\gamma(2),\ldots,\gamma(N)]$.

step 3: 将$(H_a^{l-1}-H_b^{l-1})''$与最优工厂-工件序列$H_{\rm best}^{l-1}$执行$\oplus$运算,得到一个变异工厂-工件序列V_i^l. 首先,初始化$k=1$;然后,当$\gamma_i^l(k)=0$或$\gamma_i^l(k)\neq0$且$\gamma_i^l(k)=h_{\rm best}^{l-1}(k)$时,保持$h_{\rm best}^{l-1}(k)$不变;当$\gamma_i^l(k)\neq h_{\rm best}^{l-1}(k)\neq0$时,在$h_{\rm best}^{l-1}$中寻找并记录工件号为$\gamma_i^l(k)$所处的位置$k'$,交换$h_{\rm best}^{l-1}$中处于位置$k$和$k'$的工件号$h_{\rm best}^{l-1}(k)$和$h_{\rm best}^{l-1}(k')$,更新$h_{\rm best}^{l-1}$;最后,令$k=k+1$,若$k\leqslant N$,重复前一过程,否则输出$V_i^l=h_{\rm best}^{l-1}$.

2.3.2 单点交叉

对V_i^l和前次迭代工厂-工件序列群中的H_i^{l-1}应用单点交叉算子. 如图3所示,交换H_i^{l-1}和V_i^l内位于交叉点μ之前的序列片段,形成子代序列$W1_i^l=w1_i^l(1),\ldots,w1_i^l(\mu-1),w1_i^l(\mu),\ldots,w1_i^l(N)$和$W2_i^l=w2_i^l(1),\ldots,w2_i^l(\mu-1),w2_i^l(\mu),\ldots,w2_i^l(N)$. 记录$W1_i^l$具有重复工件号的位置$\mu_1$和$\mu_2$,将工件号$w1_i^l(\mu_1)$替换成工件号$w2_i^l(\mu_2)$,重复该过程直至$W1_i^l$没有重复的工件号;对$W2_i^l$也执行上述修正过程.

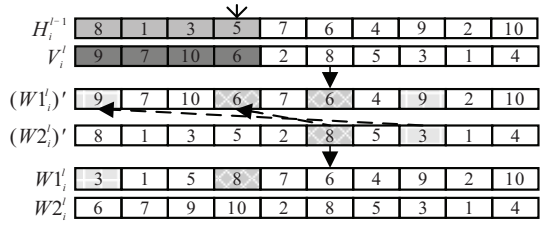


图3 交叉算子

2.3.3 概率选择

基于模拟退火算法提出以一定接受概率为准则的选择策略,使目标值大的序列也能进入当前迭代序列群成为试验序列H_i^l. 结合Khare等[8]的选择规则设计工厂-工件序列选择公式如下:

$$H_i^l=$$
$$\begin{cases}W_i^l,\ \Delta O<0\ \text{或}\ {\rm rand}'<{\rm Exp}(-\Delta O<T);\\H_i^{l-1},\ \text{otherwise.}\end{cases} \tag{18}$$

其中:$\Delta O={\rm TWET}(W_i^l)-{\rm TWET}(H_i^{l-1})$;${\rm rand}'\in(0,1)$;$T$为简化后的固定温度值,给定为

$$T=\frac{F\times\sum_{j=1}^{N}\sum_{t=1}^{G}\sum_{m=1}^{U_t}P_{jtm}}{10\times N\times F\times\sum_{t=1}^{G}U_t}. \tag{19}$$

2.4 引领蜂搜索阶段

跟随蜂采用轮盘赌规则在更新后的工厂-工件序列群中选择工厂-工件序列,试验序列H的适应度值和选择概率分别为

$$\mathrm{fit}(H)=1\Big/\sum_{j=1}^{N}(\alpha_jE_j+\beta_j,T_j), \tag{20}$$

$$\mathrm{FP}(H)=\mathrm{fit}(H)\Big/\sum_{i}\mathrm{fit}(i). \tag{21}$$

对所选序列执行VND算法,设计3种邻域产生机制:随机工厂内插入、关键/非关键工厂间交换和关键/非关键工厂内逆序反转. 依次执行这3种邻域产生机制,若新邻域序列优于当前序列,则替换当前序列,搜索结束,否则继续执行下一种邻域搜索.

2.4.1 随机工厂内插入

随机选择工厂f和工件$J_{f-\theta}(1\leqslant\theta\leqslant n^f)$,将$J_{f-\theta}$从$\tau_f$中移除,形成片段子序列$\tau_f'$;将$J_{f-\theta}$逐次插入到$\tau_f'$中的所有可能位置,选择具有最小目标值的子序列作为工厂f的最终工件子序列,如图4所示.

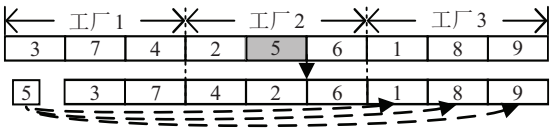


图4　随机工厂内插入

2.4.2 关键/非关键工厂间交换

设有最大目标值的工厂为关键工厂f_1,有最小目标值的工厂f_3为非关键工厂. 从f_1和f_3分别随机产生工件$J_{f_{1-\theta_1}}$和$J_{f_{3-\theta_2}}$并进行交换,如图5所示.

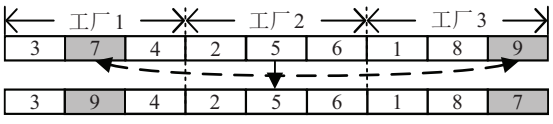


图5　关键/非关键工厂间交换

2.4.3 关键/非关键工厂内逆序反转

随机从关键工厂f_1和非关键工厂f_3各选一个工件$J_{f_{1-\sigma_1}}$和$J_{f_{3-\sigma_2}}$,以这两个工件所处位置作为变异点,将位于这两个变异点之间的工件进行逆序反转,如图6所示.

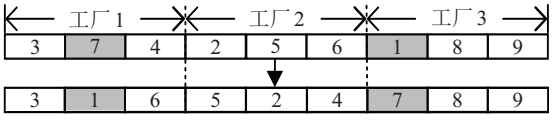


图6　关键/非关键工厂内逆序反转

2.5 侦察蜂搜索阶段

如果某个工厂-工件序列执行limit次迭代后适应度值仍未改进,则引领蜂转变为侦察蜂. 侦察蜂通过关键/非关键工厂间插入方式进行LS以获取新工厂-工件序列. 从关键工厂内随机选择一个工件J_ρ并移除,将J_ρ依次插入到非关键工厂的工件子序列中的所有可能位置,具有最小目标值的子序列即为非关键工厂的新工件子序列,如图7所示.

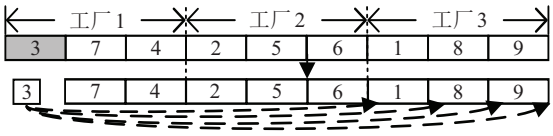


图7　关键/非关键工厂间插入

2.6 HDABC算法流程

基于上述描述得到HDABC算法的总体流程.

step 1: 参数初始化,设置工厂-工件序列群规模NS,最大迭代数MaxG,变异影响因子σ,最大尝试次数limit;

step 2: 采用随机规则和均衡分派策略初始化工厂-工件序列群;

step 3: 计算每个工厂-工件序列的适应度值,记录全局最优序列;

step 4: 当算法超过MaxG时,输出最优工厂-工件序列及其目标值,否则继续执行step 5;

step 5: 引领蜂执行离散差分进化算法,采用一定接受概率的选择策略产生新工厂-工件序列群;

step 6: 跟随蜂采用轮盘赌规则选择较优工厂-工件序列,进行变邻域下降算法;

step 7: 若某个工厂-工件序列在limit次迭代后仍未改进,则引领蜂转变为侦察蜂,继续step 8,否则转至step 3;

step 8: 侦察蜂对被放弃序列进行局域搜索,产生新工厂-工件序列,转至step 3.

3 实验仿真与分析

3.1 算例生成

为了测试HDABC算法的性能,随机生成不同规模的算例进行仿真实验. 数据产生如下:$F\in\{2,4,5,6\}$,$N\in\{10,20,40,60,90,120\}$,$G\in\{3,4,5\}$,$U_t\in\{3,4,5,6\}$;$P_{jtm}\in U[1,10]$,$S_{j'jtm}\in U[0,5]$,$r_j\in U[1,5]$. 定义工件在所有阶段的总平均处理时间$P_j^{\mathrm{avg}}$和总平均准备时间$S_j^{\mathrm{avg}}$如下:

$$P_j^{\mathrm{avg}}=\Big(\sum_{t=1}^{G}\sum_{m=1}^{U_t}P_{jtm}\Big)\Big/\sum_{t=1}^{G}U_t, \tag{22}$$

$$S_j^{\rm avg}=\sum_{t=1}^{G}\frac{\sum_{m=1}^{U_t}\sum_{j'=1,j'\leqslant j}^{N}S_{j'jtm}}{U_t\times(N-1)}. \tag{23}$$

进而,计算每个工件的交货期[19]

$$d_j=(P_j^{\rm avg}+S_j^{\rm avg})\times(1+\lambda\times 3),\lambda\in U[0,1]. \tag{24}$$

3.2 实验设计

将HDABC算法与变邻域搜索(VNS)、传统ABC算法、Zhang等[6]提出的混合局域搜索的DDE算法(HDDE)、轩华等[20]设计的自适应遗传算法(AGA)、Xuan等[4]提出的改进DABC算法(IDABC)进行对比,验证其求解性能. 所有算法均采用Matlab R2020a编程,在Inter Core i5-6200U CPU(2.30 GHz)的微机上运行. 为公平比较这6种算法,通过正交实验设置$\sigma=0.5,\lim=5,\mathrm{NS}=80;\mathrm{MaxG}=100$;将ABC算法运行100代,以最大CPU时间不超过1 000 s所需的CPU时间作为其他5种算法的停止时间.

$\{F,N,G,U_t\}$的不同组合共产生56种问题规模,每种规模随机产生10组算例,共测试560组算例. 定义相对百分比偏差(%)为

$$R_\varepsilon=\frac{\mathrm{TWET}_\varepsilon-\mathrm{TWET}_{\rm HDABC}}{\mathrm{TWET}_{\rm HDABC}}\times 100\,\%. \tag{25}$$

其中:$\mathrm{TWET}_{\rm HDABC}$为HDABC算法得到的平均目标值,$\mathrm{TWET}_\varepsilon$为当前算法$\varepsilon$($\varepsilon\in\{\mathrm{VNS},\mathrm{ABC},\mathrm{HDDE},\mathrm{AGA},\mathrm{IDABC}\}$)求解算例的平均目标值.

3.3 HDABC算法的有效性检验

设置100次迭代作为算法停止条件,取$N=\{40,60,90,120\}$且$F=G=U_t=4$的各规模问题作为一组算例,分别运行未引入DDE算法的HDABC算法(ABC1)、未结合VND算法的HDABC算法(ABC2)、未使用LS的HDABC算法(ABC3)以及HDABC算法. 如图8所示,HDABC算法的求解效果最好,其他3种算法的收敛性表现均不如HDABC算法,因此DDE、VND和LS这3种算法的引入能够提升HDABC的算法性能. 随着问题规模的增大,HDABC算法与ABC 1算法的迭代曲线差异较大,与ABC 3算法的差异较小,因此DDE算法对所提算法的影响程度最大,LS影响最小.

3.4 测试结果与分析

由前述HDABC、VNS、ABC、HDDE、AGA和IDABC六种算法求解不同规模算例得到如表3和表4所示的测试结果.

从表3和表4的数据可以看出:

对于中小规模问题,在平均CPU时间126.4 s内,由HDABC算法得到的平均值较VNS、ABC、HDDE、AGA和IDABC算法分别改进17.3 %、15.2 %、11.8 %、8.9 %和4.9 %;对于大规模问题的算例,在平均CPU时间641.3 s内,由HDABC算法得到的平均值较VNS、ABC、HDDE、AGA和IDABC算法分别改进14.6%、11.9%、10.7%、6.0%和2.7%. 因此,对于所有规模问题,在相同的运行时间内,由HDABC算法所获得的解的质量一致优于其他5种算法.

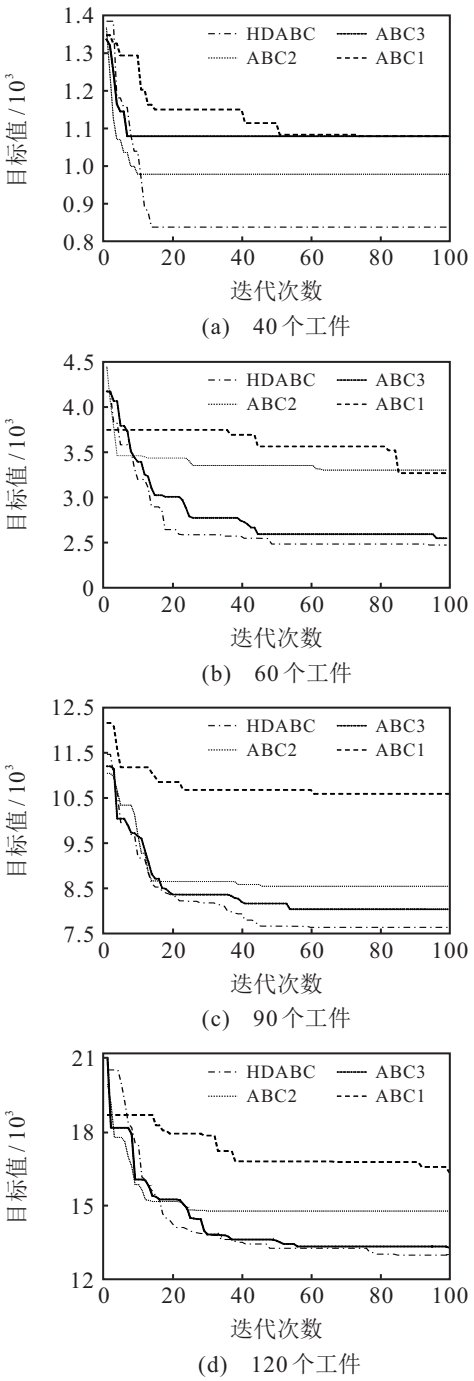


图8 各规模问题的目标值变化趋势

为了说明实验结果具有统计意义,在95% 的置信水平下,对实验结果进行统计学检验. 图9表明5种对比算法在不同算例测试下所得到的相对百分比偏差结果之间具有显著性差异(相对百分比偏差衡量

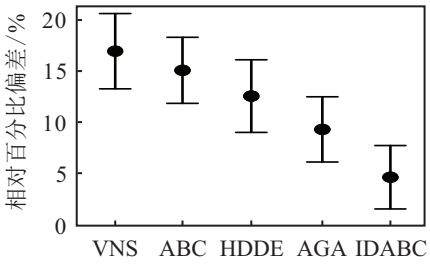
表 3　中小规模问题的测试结果

$F\times N\times G\times U_t$	VNS	ABC	HDDE	AGA	IDABC	HDABC	$R_{\rm VNS}$/%	$R_{\rm ABC}$/%	$R_{\rm HDDE}$/%	$R_{\rm AGA}$/%	$R_{\rm IDABC}$/%	CPU/s
2×10×3×3	100.0	109.2	103.8	98.6	98.2	96.8	3.3	13.4	7.2	1.8	1.4	42.2
2×10×3×4	89.0	97.8	81.2	85.0	80.8	79.4	12.0	23.1	2.3	7.1	1.8	49.8
2×10×4×3	85.3	84.2	80.6	83.8	86.4	76.4	11.6	10.2	0.05	9.7	13.1	54.8
2×10×4×4	200.2	196.8	194.2	190.4	176.8	152.2	31.5	29.3	27.5	25.1	16.2	66.1
2×20×3×3	519.2	508.2	488.2	483.0	466.8	455.8	13.9	11.5	7.1	5.9	2.4	77.6
2×20×3×4	280.8	269.2	261.6	255.0	241.4	233.2	20.4	15.4	12.2	9.3	3.5	91.8
2×20×4×3	812.4	794.4	789.4	776.2	756.6	753.2	7.9	5.4	4.8	3.1	0.5	105.1
2×20×4×4	190.8	280.8	295.6	257.6	253.0	248.8	16.9	12.9	18.9	3.5	1.7	122.1
2×40×3×3	3 374.6	3 249.2	3 349.6	3 286.4	3 216.8	3 100.6	8.8	4.8	8.0	5.9	3.7	143.5
2×40×3×4	2 072.0	1 967.6	2 011.2	2 037.4	1 945.8	1 854.2	11.7	6.1	8.5	9.9	4.9	174.2
2×40×4×3	3 260.2	3 211.2	3 172.2	3 076.6	3 000.2	2 900.6	12.4	10.7	9.4	6.1	3.4	190.4
2×40×4×4	2 183.8	2 032.8	2 081.8	1 981.6	1 902.0	1 805.4	20.9	12.6	15.3	9.8	5.4	229.8
4×20×3×3	301.6	287.6	265.0	264.6	249.8	239.0	20.1	20.3	10.9	5.9	4.5	84.1
4×20×3×4	230.0	222.0	204.8	205.6	203.6	203.4	13.1	9.1	0.7	1.1	0.1	93.1
4×20×4×3	290.6	283.6	259.6	252.2	226.8	224.4	29.5	26.4	15.7	12.4	1.1	108.9
4×20×4×4	305.2	298.8	275.6	238.6	243.2	233.0	30.9	28.2	18.3	2.4	4.4	129.5
4×40×3×3	2 200.8	2 125.6	2 157.0	2 246.8	2 051.0	1 793.4	22.7	18.6	20.3	25.3	14.4	149.6
4×40×3×4	1 224.6	1 167.0	1 175.6	1 138.2	1 064.0	1 021.2	19.9	14.3	15.1	11.5	4.2	182.1
4×40×4×3	2 554.2	2 473.6	2 346.2	2 229.6	2 206.8	2 076.6	22.9	19.1	13.0	7.4	6.3	195.4
4×40×4×4	1 190.4	1 171.4	1 256.6	1 194.2	1 091.4	1 025.4	16.1	14.2	22.5	16.5	6.4	237.5
平均	1 073.2	1 041.5	1 042.4	1 019.1	978.1	928.7	17.3	15.2	11.8	8.9	4.9	126.4

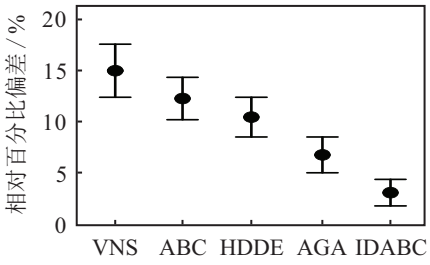
表 4　大规模问题的测试结果

$F\times N\times G\times U_t$	VNS	ABC	HDDE	AGA	IDABC	HDABC	$R_{\rm VNS}$/%	$R_{\rm ABC}$/%	$R_{\rm HDDE}$/%	$R_{\rm AGA}$/%	$R_{\rm IDABC}$/%	CPU/s
4×60×4×3	7 281.8	7 264.2	7 276.7	6 990.6	6 807.8	6 689.0	8.9	8.6	8.8	4.5	1.8	283.1
4×60×4×4	3 580.2	3 483.8	3 577.8	3 513.4	3 314.8	3 227.2	10.9	7.9	10.9	8.9	2.7	358.0
4×60×5×3	6 870.0	6 807.4	6 794.0	6 525.8	6 363.6	6 248.8	9.9	8.9	8.7	4.4	1.8	352.1
4×60×5×4	3 783.4	3 930.6	3 970.0	3 647.4	3 617.8	3 539.0	6.9	11.1	12.2	3.1	2.2	431.5
4×90×4×4	11 551.6	11 163.2	11 931.8	11 353.4	10 796.4	10 784.6	7.1	3.5	10.6	5.3	0.1	518.9
4×90×4×5	7 282.2	7 385.4	7 773.0	7 163.2	6 965.2	6 768.8	7.6	9.1	14.8	5.8	2.9	614.7
4×90×5×4	12 972.2	12 207.4	12 077.6	12 334.6	11 679.4	11 485.2	12.9	6.3	5.2	7.4	1.7	657.0
4×90×5×5	7 399.2	7 273.0	7 255.6	7 200.2	6 945.2	6 848.8	8.0	6.2	5.9	5.1	1.4	776.0
4×120×4×5	14 955.6	14 794.4	14 943.8	14 514.4	14 061.2	14 009.8	6.7	5.6	6.7	3.6	0.3	791.9
4×120×4×6	10 436.2	10 414.6	10 655.6	9 718.8	9 809.6	9 479.4	10.1	9.9	12.4	2.5	3.5	910.4
4×120×5×5	16 125.0	15 783.8	15 706.4	14 840.2	14 363.2	14 321.2	12.5	10.2	9.7	3.6	0.3	1 000.0
4×120×5×6	9 937.4	9 821.8	10 306.0	9 249.8	9 311.4	9 302.0	6.8	5.6	10.8	0.5	0.1	1 000.0
5×60×4×3	7 013.8	6 784.4	6 863.2	6 940.4	6 501.6	6 293.2	11.4	7.8	9.1	10.3	3.3	288.2
5×60×4×4	3 606.0	3 756.6	3 771.0	3 474.4	3 309.8	3121.2	15.5	20.4	20.8	11.3	6.0	348.5
5×60×5×3	7 647.0	7 567.6	7 384.8	6 760.8	6 812.2	6 505.4	17.5	16.3	13.5	3.9	4.7	353.7
5×60×5×4	3 753.8	3 726.6	3 770.0	3 650.8	3 324.2	3 142.4	19.4	18.6	20.0	16.2	5.8	435.1
5×90×4×4	11 241.6	11 284.2	10 933.8	10 307.4	9 993.8	9 227.2	21.8	22.3	18.5	11.7	8.3	517.2
5×90×4×5	7 029.0	6 865.8	6 866.2	6 520.8	6 302.0	6 265.8	12.1	9.5	9.6	4.1	0.6	607.9
5×90×5×4	12 272.6	12 225.0	12 114.8	11 809.2	11 105.4	10 614.4	15.6	15.2	14.1	11.3	4.6	644.4
5×90×5×5	7 564.2	7 550.0	7 430.0	6 766.8	6 713.8	6 652.0	13.7	13.5	11.7	1.7	0.9	759.3
5×120×4×5	14 678.2	14 345.0	14 771.8	14 053.0	13 769.6	13 714.4	7.0	4.6	7.7	2.4	0.4	807.1
5×120×4×6	9 830.6	9 648.2	8 749.8	9 045.6	8 679.2	8 637.4	31.8	13.8	1.3	4.7	0.5	919.1
5×120×5×5	15 569.6	15 138.8	14 813.2	14 243.2	14 103.4	13 998.6	11.2	8.1	5.8	1.7	0.7	997.3
5×120×5×6	10 980.4	10 610.6	10 492.8	10 331.2	9 967.6	9 878.0	11.4	7.4	6.2	4.5	0.9	1 000.0
6×60×4×3	6 694.4	6 567.6	6 501.8	6 315.6	5 922.2	5 652.2	18.4	16.2	15.0	11.7	4.8	290.4
6×60×4×4	3 349.4	3 313.2	3 161.4	3 160.6	2 995.8	2 779.2	20.5	19.2	13.8	13.7	7.8	353.8
6×60×5×3	6 731.4	6 542.0	6 216.4	5 658.8	5 891.2	5 546.8	21.3	17.9	12.1	2.0	6.2	362.4
6×60×5×4	4 300.0	4 044.8	3 548.6	3 500.2	3 375.6	3 330.6	29.1	21.4	6.5	5.1	1.4	446.7
6×90×4×4	11 310.6	11 057.0	10 652.0	9 847.4	9 666.2	9 380.8	20.6	17.9	13.6	4.9	3.0	514.6
6×90×4×5	7 707.2	7 315.6	7 081.8	6 823.0	6 688.0	6 624.2	16.3	10.4	6.9	2.9	0.9	607.3
6×90×5×4	12 416.2	11 885.2	12 030.0	11 209.0	11 074.6	10 922.6	13.2	8.8	10.1	2.6	1.4	637.6
6×90×5×5	7 586.4	7 524.0	6 957.6	7 214.4	6 866.0	6 602.8	14.9	14.0	5.4	9.3	4.0	748.2
6×120×4×5	16 736.2	16 257.6	15 949.6	14 611.8	14 880.4	14 214.4	17.7	14.3	12.2	2.8	4.7	819.7
6×120×4×6	11 688.8	11 397.0	11 089.4	10 718.6	10 111.0	9 804.8	19.2	16.2	13.1	9.3	3.1	934.2
6×120×5×5	16 658.0	16 093.0	15 903.4	15 360.4	14 343.8	13 852.8	20.2	16.2	14.8	10.9	3.5	1 000.0
6×120×5×5	12 020.4	10 737.8	10 614.0	10 398.2	10 203.2	10 067.8	19.4	6.7	5.4	3.3	1.3	1 000.0
平均	9 460.0	9 238.0	9 164.8	8 771.5	8 517.7	8 320.4	14.6	11.9	10.7	6.0	2.7	641.3

的是5种对比算法与所提算法HDABC得到的平均目标值的相对差,因此相对百分比偏差的计算只针对5种对比算法). 各算法的性能依次按照HDABC、IDABC、AGA、HDDE、ABC、VNS降序排列,因此实验结果具有可靠性.



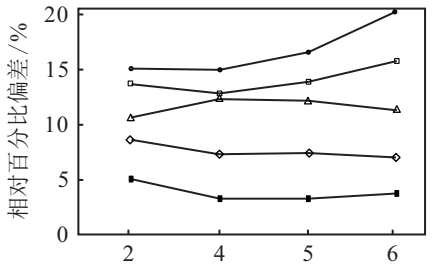
(a) 中小规模问题



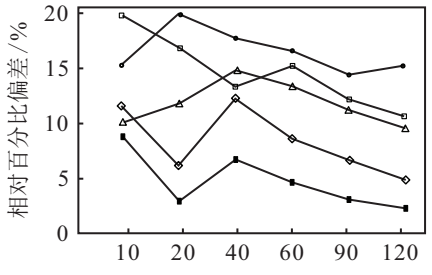
(b) 大规模问题

图9 5种算法求解不同问题的相对百分比偏差区间图

图10显示了参数N和F对算法性能的影响. 随着N的增加,问题的复杂度和解空间也相应增大,这导致多数算法的相对百分比偏差呈下降趋势,而且每种对比算法对N的变化都比较敏感;随着F的增加,F对算法性能的影响程度不显著,但总体上HDABC算法的求解性能最好.



(a) 工厂因素



(b) 工件因素

图10 5种算法和关键因素的交互作用

4 结 论

本文研究了考虑不相关并行机和顺序相关准备时间的DFFSP,以最小化总加权提前/拖期惩罚为目标,提出了混合离散人工蜂群算法. 结合随机规则和均衡分派策略产生了初始工厂-工件序列群;进而,引领蜂引入DDE算法设计了DE/best/1变异、单点交叉和概率选择算子以得到新序列,跟随蜂采用含随机工厂内插入、关键/非关键工厂间交换和关键/非关键工厂内逆序反转3种邻域产生机制的VND算法寻找更优邻域序列,侦查蜂利用LS产生新序列替代被放弃序列;最后,对不同规模问题进行仿真实验,结果表明了HDABC算法能在可接受的计算时间内得到较好的近优解. 未来可探讨含能耗约束的多目标DFFSP-UPM,也可尝试采用其他智能优化算法(如帝国竞争算法等)求解类似问题.

参考文献(References)

[1] Lohmer J, Lasch R. Production planning and scheduling in multi-factory production networks: A systematic literature review[J]. International Journal of Production Research, 2021, 59(7): 2028-2054.

[2] 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.
(Qin H X, Han Y Y, Chen Q D, et al. A double level mutation iterated greedy algorithm for blocking hybrid flow shop scheduling[J]. Control and Decision, 2022, 37(9): 2323-2332.)

[3] Sun Z W, Gu X S. A novel hybrid estimation of distribution algorithm for solving hybrid flowshop scheduling problem with unrelated parallel machine[J]. Journal of Central South University, 2017, 24(8): 1779-1788.

[4] Xuan H, Zhang H X, Li B. An improved discrete artificial bee colony algorithm for flexible flowshop scheduling with step deteriorating jobs and sequence-dependent setup times[J]. Mathematical Problems in Engineering, 2019, 2019: 8520503.

[5] Shao Z S, Pi D C, Shao W S. Hybrid enhanced discrete fruit fly optimization algorithm for scheduling blocking flow-shop in distributed environment[J]. Expert Systems with Applications, 2020, 145: 113147.

[6] Zhang G H, Xing K Y, Cao F. Discrete differential evolution algorithm for distributed blocking flowshop scheduling with makespan criterion[J]. Engineering Applications of Artificial Intelligence, 2018, 76: 96-107.

[7] Huang J P, Pan Q K, Miao Z H, et al. Effective constructive heuristics and discrete bee colony optimization for

distributed flowshop with setup times[J]. Engineering Applications of Artificial Intelligence, 2021, 97: 104016.

[8] Khare A, Agrawal S. Effective heuristics and metaheuristics to minimise total tardiness for the distributed permutation flowshop scheduling problem[J]. International Journal of Production Research, 2021, 59(23): 7266-7282.

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

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

[11] Shao W S, Shao Z S, Pi D C. Effective constructive heuristics for distributed no-wait flexible flow shop scheduling problem[J]. Computers & Operations Research, 2021, 136: 105482.

[12] 雷德明, 苏斌. 基于多班教学优化的多目标分布式混合流水车间调度[J]. 控制与决策, 2021, 36(2): 303-313.
(Lei D M, Su B. Multi-class teaching-learning-based optimization for multi-objective distributed hybrid flow shop scheduling[J]. Control and Decision, 2021, 36(2): 303-313.)

[13] Li Y L, Li X Y, Gao L, et al. A discrete artificial bee colony algorithm for distributed hybrid flowshop scheduling problem with sequence-dependent setup times[J]. International Journal of Production Research, 2021, 59(13): 3880-3899.

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

[15] 吴锐, 郭顺生, 李益兵, 等. 改进人工蜂群算法求解分布式柔性作业车间调度问题[J]. 控制与决策, 2019, 34(12): 2527-2536.
(Wu R, Guo S S, Li Y B, et al. Improved artificial bee colony algorithm for distributed and flexible job-shop scheduling problem[J]. Control and Decision, 2019, 34(12): 2527-2536.)

[16] Li J Q, Duan P Y, Cao J D, et al. A hybrid Pareto-based tabu search for the distributed flexible job shop scheduling problem with E/T criteria[J]. IEEE Access, 2018, 6: 58883-58897.

[17] 张梓琪, 钱斌, 胡蓉. 混合交叉熵算法求解复杂零等待流水线调度问题[J]. 控制理论与应用, 2021, 38(12): 1919-1934.
(Zhang Z Q, Qian B, Hu R. Hybrid cross-entropy algorithm for solving complex no-wait flow-shop scheduling problem[J]. Control Theory & Applications, 2021, 38(12): 1919-1934.)

[18] Al-Shayea A M, Saleh M, Alatefi M, et al. Scheduling two identical parallel machines subjected to release times, delivery times and unavailability constraints[J]. Processes, 2020, 8(9): 1025.

[19] Behnamian J, Zandieh M. A discrete colonial competitive algorithm for hybrid flowshop scheduling to minimize earliness and quadratic tardiness penalties[J]. Expert Systems with Applications, 2011, 38(12): 14490-14498.

[20] 轩华, 李冰, 罗书敏, 等. 基于总加权完成时间的可重入混合流水车间调度问题[J]. 控制与决策, 2018, 33(12): 2218-2226.
(Xuan H, Li B, Luo S M, et al. Reentrant hybrid flowshop scheduling problem based on total weighted completion time[J]. Control and Decision, 2018, 33(12): 2218-2226.)

作者简介

轩华(1979–), 女, 教授, 博士, 从事物流优化与控制、生产计划与调度、智能优化算法等研究, E-mail: hxuan@zzu.edu.cn;

李文婷(1998–), 女, 硕士生, 从事生产调度的研究, E-mail: liwentingg1003@163.com;

李冰(1976–), 男, 教授, 博士, 从事物流优化与控制、运输组织优化等研究, E-mail: lbing@zzu.edu.cn.

(责任编辑：齐　霁)