控制与决策
Control and Decision

求解多目标不相关并行机调度问题的多群体人工蜂群算法

雷德明, 杨海

引用本文:
雷德明,杨海. 求解多目标不相关并行机调度问题的多群体人工蜂群算法[J]. 控制与决策, 2022, 37(5): 1174−1182.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2020.0775

求解多目标不相关并行机调度问题的多群体人工蜂群算法

雷德明†, 杨 海

(武汉理工大学 自动化学院，武汉 430070)

摘 要: 针对具有预防性维修(PM)和顺序相关准备时间(SDST)的不相关并行机调度问题,提出一种多群体人工蜂群算法(MABC)以同时最小化完工时间和总延迟时间. 该算法将雇佣蜂分割成s个雇佣蜂群,除最差雇佣蜂群外,每个雇佣蜂群都对应1个跟随蜂群. 结合2个目标函数、PM和SDST的特征设计3种邻域搜索,采用全局搜索和邻域搜索的不同组合实现雇佣蜂阶段和跟随蜂阶段,并引入两种淘汰过程. 通过大量实验测试MABC新策略和搜索性能,计算结果验证了新策略的有效性和MABC的搜索优势.

关键词: 预防性维修；顺序相关准备时间；不相关并行机调度；人工蜂群算法

中图分类号: TP273 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2020.0775

引用格式: 雷德明,杨海. 求解多目标不相关并行机调度问题的多群体人工蜂群算法[J]. 控制与决策, 2022, 37(5): 1174-1182.

Multi-colony artificial bee colony algorithm for multi-objective unrelated parallel machine scheduling problem

LEI De-ming†, YANG Hai

(College of Automation，Wuhan University of Technology，Wuhan 430070，China)

Abstract: To solve the unrelated parallel machine scheduling problem (UPMSP) with preventive maintenance (PM) and sequence dependent setup time (SDST), a multi-colony artificial bee colony (MABC) algorithm is proposed to minimize makespan and total tardiness simultaneously. In this algorithm, employed bees are divided into s colonies. Except for the worst employed bee colony, each employed bee colony corresponds to a onlooker bee colony. Combined with the characteristics of two objective functions, PM and SDST, three kinds of neighborhood searches are designed. Different combinations of global search and neighborhood search are used to implement the employed bee phase and the onlooker bee phase, and two elimination processes are applied. Experimental research on the strategy and search performance of the MABC is carried out, and computational results demonstrate the effectiveness of the proposed strategy and the search advantage of the MABC.

Keywords: preventive maintenance；sequence dependent setup time；unrelated parallel machine scheduling；artificial bee colony algorithm

0 引 言

并行机调度问题(PMSP)是一种典型的调度问题[1],广泛存在于轮胎行业[2]、纺织行业[3]和烟草行业[4]等实际制造业. 它可以看作扩展的单机调度或者柔性多阶段生产系统的基本单元[5-6]. 根据并行机功能,PMSP可分为同速PMSP、异速PMSP和UPMSP. 由于工件在不同机器上的加工速度各异,UPMSP求解难度更大,是目前PMSP研究的焦点,考虑各种实际加工约束,如PM和SDST的UPMSP受到研究者广泛关注.

通常假设并行机在整个生产过程中持续可用,然而,在实际制造过程中,经常因为随机故障或可能失效等问题导致机器不可用. 由于维修能有效预防机器的潜在故障和严重事故,有必要研究考虑机器维修可用性约束的UPMSP. 近年来,相关研究取得了一些进展[7-12]. 自Parker等[13]的开创性工作之后,具有SDST的UPMSP引起广泛关注[14-24].

以上考虑PM或SDST的UPMSP研究以单目标优化[7-8, 11, 14-22]和优化最大完成时间为主,很少考虑以总延迟时间等为目标的UPMSP. 多目标UPMSP

收稿日期: 2020-06-15；**录用日期:** 2021-01-19.
基金项目: 国家自然科学基金项目(61573264).
责任编委: 唐加福.
†通讯作者. E-mail: deminglei11@163.com.

研究取得了一定进展[9, 12, 24-25], 主要应用于遗传算法[24]、蚁群算法[24]和教学优化算法[25]等优化最大完成时间和提前/延迟惩罚总成本等目标.

由于实际UPMSP通常存在多个目标, 如最大完成时间和总延迟时间, 这些目标分别描述了生产系统的性能和按时交货水平, 而生产性能和按时交货水平等往往彼此冲突, 使得相应的目标函数彼此冲突, 需要在目标间协调折中, 从而增加了问题的求解难度. 此外, 现有研究往往单独处理具有PM的UPMSP或具有SDST的UPMSP, 很少将PM与SDST同时结合到UPMSP中[11-12]. 实际上, 这两种约束在生产制造过程中常见且往往同时存在. 综上所述, 有必要研究具有PM和SDST的多目标UPMSP.

与遗传算法和蚁群算法相比, ABC具有参数少、实现简单等特点. 该算法已广泛应用于各种调度问题, 包括PMSP[17-18, 26]、流水车间调度[27-28]、作业车间调度[29-31]等以及其他优化问题[32-33]的求解. 然而, 尽管ABC在具有PM或SDST的UPMSP的求解方面具有一些应用, 但该算法较少用于解决考虑PM和SDST两种约束的多目标UPMSP.

针对具有PM和SDST的UPMSP, 本文提出一种MABC算法以同时最小化完工时间和总延迟时间. 该算法采用启发式初始化方法, 将雇佣蜂分割成s个雇佣蜂群, 除最差雇佣蜂群外, 每个雇佣蜂群对应1个跟随蜂群, 结合2个目标、PM和SDST的特征设计3种邻域搜索, 运用全局搜索和邻域搜索的不同组合实现雇佣蜂阶段和跟随蜂阶段, 并引入两种淘汰过程. 通过大量实验验证了新策略对MABC性能的影响以及MABC在UPMSP求解方面的搜索优势.

1　问题描述

具有PM和SDST的UPMSP描述如下: 有n个工件$J_1, J_2, \ldots, J_n$由m台不相关并行机$M_1, M_2, \ldots, M_m$加工, 工件可在m台机器中任何一台机器加工. 工件J_j的加工时间p_{kj}取决于其处理机器M_k的性能, 不同机器加工时间通常不同, d_j表示J_j的交货期.

为了保证机器持续可用, 在计划阶段确定固定的维修周期, 即对于每台机器M_k, 每运行一段时间进行1次PM活动, w_k表示每次维修的持续时间, 工件只能在两个连续维修活动之间的时间间隔内加工, 间隔时间为u_k, 该机器的PM周期表示为T_k, $T_k = w_k + u_k$[11]. 如果某工件无法在间隔时间内完成加工, 则安排其在下一个时间间隔内完成加工.

对于SDST, st_{kij}表示机器M_k在工件J_i加工完毕后再加工工件J_j的准备时间, st_{k0j}表示机器M_k维修完成后第1个加工工件J_i的准备时间, st_{ki0}表示机器M_k加工完工件J_i后执行维修活动的准备时间. 此外, 具有如下工件和机器约束: 1) 任意工件和机器在零时可用; 2) 每个工件只加工1次; 3) 工件加工操作和机器PM不可被中断; 4) 每个机器在同一时间只能加工1个工件.

具有PM和SDST的UPMSP由机器分配和调度两个子问题组成, 同时考虑如下两个目标:

$$\min f_1 = C_{\max} = \max\{C_j | j = 1, 2, \ldots, n\}; \quad (1)$$

$$\min f_2 = \sum_{j=1}^{n} \max\{C_j - d_j, 0\}. \quad (2)$$

其中: f_1为最大完成时间; f_2为总延迟时间; C_j为工件j的完成时间.

2　MABC求解考虑PM和SDST的UPMSP

ABC将蜜蜂分为雇佣蜂、跟随蜂和侦察蜂3类, 包括雇佣蜂阶段、跟随蜂阶段和侦查蜂阶段3个阶段.

雇佣蜂阶段, 对于每个解x, 产生新解y, 有

$$y = x + \phi(x - x'). \quad (3)$$

其中: ϕ为$[-1, 1]$内随机数的向量, $x' \neq x$为从种群中随机选择的解.

跟随蜂阶段, 每个跟随蜂通过轮盘赌选择食物源, 然后根据式(3)产生新解. 若某个雇佣蜂连续未更新的次数超过规定的阈值limit, 则雇佣蜂变成侦查蜂, 随机产生一个食物源, 替代该雇佣蜂.

用贪心选择确定新解是否替代旧解: 若新解y的花蜜量大于x的花蜜量, 则y替代x, 否则仍保留x.

针对所研究的UPMSP提出一种MABC, 该算法将所有雇佣蜂分割为s个雇佣蜂群, 构建$s-1$个跟随蜂群, 并采用全局搜索和邻域搜索的不同组合实现这些蜂群内的搜索.

2.1　初始化

具有PM和SDST的UPMSP由机器分配和调度两个子问题组成, 常使用双串[34-37]描述问题的解. 本文采用双串表示法. 对于具有n个工件和m台机器的UPMSP, 每个解由机器分配串$[M_{\theta_1}, M_{\theta_2}, \ldots, M_{\theta_n}]$和调度串$[\pi_1, \pi_2, \ldots, \pi_n]$组成. 其中: M_{θ_j}为分配给工件J_j的并行机, $1 \leqslant \theta_j \leqslant m$, $\pi_i \in \{1, 2, \ldots, n\}$.

解码过程如下: 首先根据机器分配串确定为每个工件所分配的机器; 然后根据调度串和PM条件, 处理工件加工和每台机器的维修活动.

EB表示所有雇佣蜂的集合, 由N个初始解组成, 每个初始解采用如下方式产生: 随机选择一个目标,

对于选定的目标f_i,通过启发式构造解[12].

2.2 多群体产生

MABC中,N个雇佣蜂分割成s个雇佣群EB_g,$g=1,2,\ldots,s$. 具体划分过程如下:首先,根据Deb等[38]的方法确定所有解的rank和拥挤距离;然后对所有解根据rank进行升序排序,相同rank的解按拥挤距离降序排序;最后,从第1个解开始,前$|\mathrm{EB}_1|$个解分配到EB_1中,接着$|\mathrm{EB}_2|$个解构成EB_2,依次进行,直至所有解都加入相应的雇佣蜂群中. 显然,$\mathrm{EB}_1,\mathrm{EB}_2,\ldots,\mathrm{EB}_s$也是基于rank和拥挤距离降序排列的,$\mathrm{EB}_s$由EB的最差解组成,使得其本身也是$s$个雇佣蜂群中最差的群体.

通常跟随蜂的数量等于雇佣蜂的数量. 本文跟随蜂数量为$\bar{N}$,$\bar{N}\neq N$,所有跟随蜂划分为$s-1$个跟随蜂群OB_h,$h=1,2,\ldots,s-1$,$\bar{N}=\sum\limits_{h=1}^{s-1}|\mathrm{OB}_h|$. OB_g只是跟随EB_g,即OB_g中的每个跟随蜂只选择EB_g中的解,而不是EB_h,$h\neq g$,这样,EB_s被排除在跟随蜂的选择之外,即跟随蜂不选择$x\in\mathrm{EB}_s$作为食物源,从而避免了在一些低质量解上浪费计算资源. 因为至少需要1个跟随蜂群即$(s-1)\geqslant 1$,所以至少需要2个雇佣蜂群.

由于采用多个雇佣蜂群和跟随蜂群,雇佣蜂阶段和跟随蜂阶段可实现差异化搜索,为此,引入全局搜索和邻域搜索的多种组合以实现各蜂群内的搜索.

2.3 雇佣蜂阶段

雇佣蜂阶段描述如下(每个雇佣蜂$x\in\mathrm{EB}$):

1) 如果$x\in\mathrm{EB}\backslash\mathrm{EB}_s$,则若$x\in\bigcup\limits_{g=1}^{\tau}\mathrm{EB}_g$,则随机选择$y\in\mathrm{EB}_1$,否则随机选择$y\in\bigcup\limits_{g=1}^{\tau}\mathrm{EB}_g$. 依次执行$x$的局部搜索$\mathrm{NS}_1$以及$x$与$y$之间的全局搜索,得到新解$z$,如果替换条件成立,则$x=z$,用$x$更新档案$\Omega$.

2) 如果$x\notin\mathrm{EB}\backslash\mathrm{EB}_s$,则若随机数$\alpha<0.5$,则随机选择$y\in\mathrm{EB}\backslash\mathrm{EB}_s$,对$x$执行$\mathrm{NS}_1$和$x$与$y$之间的全局搜索,如果新解$z$满足替换条件,则$x=z$,用$x$更新$\Omega$;否则对$x$执行从$\mathrm{NS}_2$和$\mathrm{NS}_3$随机选择的邻域搜索,得到新解$z$. 如果替换条件成立,则$x=z$,用$x$更新$\Omega$.

3) 更新γ_x,并对x应用第1种淘汰过程. 其中γ_x、τ为整数,如果$s=2$,则$\tau=1$;如果$s>2$,则$\tau=2$. 如果x被新解更新替代,则$\gamma_x=0$,否则$\gamma_x=\gamma_x+1$.

替换条件:若$z\succ x$或z与x彼此非劣,则z替换x,同时运用z更新Ω. 其中,$z\succ x$表示z支配x,若$i=1,2$,$f_i(z)\leqslant f_i(x)$,且$\exists i=1$或2,$f_i(z)<f_i(x)$,则$z\succ x$. 若z不支配x也不被x支配,则z与x彼此非劣.

外部档案Ω保留MABC搜索过程产生的非劣解,其更新方式如下:将x加入Ω,根据Pareto支配比较Ω内所有解,并从Ω中移除所有受支配解.

邻域搜索NS_1依次作用于x的两个串. 调度串中,从π_1开始,对于每个π_i,若随机数$\alpha<\eta$,则互换π_i和随机选中的π_j,$j\neq i$;分配串中,对于每个M_{θ_i},若随机数$\alpha<\eta$,则M_{θ_i}替换为随机选取的并行机. 通过多次实验$\eta=0.1$.

邻域搜索NS_2描述如下:生成新解$z\in\mathcal{N}_1(x)$,若z满足替换条件,则z替换x并更新Ω,搜素结束,否则产生$z\in\mathcal{N}_2(x)$. 若替换条件满足,则利用z替代x并更新Ω,否则$z\in\mathcal{N}_3(x)$. 如果替换条件成立,则更新x和Ω. 其中:$\mathcal{N}_i$为邻域结构,$\mathcal{N}_i(x)$为对x执行邻域结构$\mathcal{N}_i$而得到的x的邻域解集.

邻域结构$\mathcal{N}_1$的具体过程如下:从f_1、f_2随机选择一个目标,若f_1被选中,则确定完成时间最大的机器M_a和最小的机器M_b,$\tilde{C}_a$为M_a的完成时间. 从M_a随机选择工件J_i,并计算$\bar{C}_{i,l,b}$,得到l^*,它满足$\bar{C}_{i,l^*,b}=\min\limits_{l}\{\bar{C}_{i,l,b}\}$,其中$\bar{C}_{i,l,b}$表示工件$J_i$被插入机器$M_b$的位置$l$时$M_b$的完成时间. 若$f_2$被选中,则确定总延迟时间最大的机器$M_a$和最小的机器$M_b$,选取工件$J_i$,计算$\bar{T}_{i,l,b}$,得到满足$\bar{T}_{i,l^*,b}=\min\limits_{l}\{\bar{T}_{i,l,b}\}$的位置$l^*$,其中$\bar{T}_{i,l,b}$为工件$J_i$被插入机器$M_b$的位置$l$时$M_b$的总延迟时间. 最后,将工件$J_i$插入到$M_b$的位置$l^*$.

$\mathcal{N}_1$中,当工件J_i从M_a上移除后,剩余工件的加工顺序保持不变,$\tilde{C}'_a$为剩余工件的最大完工时间,可知$\tilde{C}'_a<\tilde{C}_a$. 当J_i从M_a转移到M_b后,若$\bar{C}_{\min,b}<\tilde{C}_a$,则新解$z\in\mathcal{N}_1(x)$最大完成时间小于$x$,$z$至少与$x$互不支配. 同理,$\tilde{T}'_a$为$J_i$从$M_a$上移除后剩余工件的总延迟时间,若$(\tilde{T}'_a+\bar{T}_{\min,b})<(\tilde{T}_a+\tilde{T}_b)$,则$f_2(z)<f_2(x)$. 这样,利用$\mathcal{N}_1$具体过程中的计算结果容易判断新解能否满足替换条件,从而简化计算.

邻域结构$\mathcal{N}_2$的具体步骤如下:随机选择1个目标,若f_1被选中,则与$\mathcal{N}_1$一样,确定具有完成时间$\tilde{C}_a$最大的机器M_a和完成时间$\tilde{C}_b$最小的机器M_b,从M_a中随机选择工件J_i,从M_b中随机选定工件J_j,并从各自机器上移除. 计算满足$\bar{C}_{\min,a}=\bar{C}_{j,q^*,a}$和$\bar{C}_{\min,b}=\bar{C}_{i,l^*,b}$的位置$q^*$和$l^*$,其中前者在机器$M_a$上,后者在机器$M_b$上. 若选中$f_2$,则分别计算$\bar{T}_{\min,b}$和$\bar{T}_{\min,a}$,确定位置$q^*$和$l^*$. 最后,将$J_i$插入$M_b$的位置$l^*$,$J_j$插入$M_a$的位置$q^*$.

同样,若$\bar{C}_{\min,b}<\tilde{C}_a$,$\bar{C}_{\min,a}<\tilde{C}_a$,则$f_1(z)<f_1(x)$;或若$(\bar{T}_{\min,a}+\bar{T}_{\min,b})<(\tilde{T}_a+\tilde{T}_b)$,则$f_2(z)<f_2(x)$.

$\mathcal{N}_3$与$\mathcal{N}_1$类似. 选择1个目标,确定M_a和J_i,计算$\bar{C}_{\min,a}$或$\bar{T}_{\min,a}$,得到M_a上的位置q^*,将J_i插入M_a的位置q^*. $\mathcal{N}_3$只是将机器M_a上的工件J_i插入到该机器的新位置上.

邻域搜索NS_3的主要步骤包括:随机选定1台并行机M_k和1个目标,假设选定目标为f_1. 若M_k加工$G\geqslant 3$个工件,则重复如下过程R次:随机选择3个工件$J_i, J_j, J_r, (i\neq j\neq r)$,存在6种不同的组合,计算每种组合对应的$f_1$,确定使$f_1$最小的组合并根据该组合调整工件的加工顺序. 通过大量实验确定,当$G<4$时,$R=1$;当$4\leqslant G<10$时,$R=2$;当$G\geqslant 10$时,$R=3$.

全局搜索采用循环交叉(cycle crossover, CX[39]). 对于解x和y,利用CX得到新解z,如果$\pi_k^z\neq\pi_k^x$,即π_k^z来自y的调度串,则令$M_{\theta_{\pi_k^z}}=M_{\theta_{\pi_k^y}}$,否则令$M_{\theta_{\pi_k^z}}=M_{\theta_{\pi_k^x}}$.

第1种淘汰过程:对于$x\in \mathrm{EB}_g$且$\gamma_x\geqslant\mu_g$,随机选择$y\in\Omega$,并随机选择NS_2和NS_3中的一个作用于y,生成新解z,若满足替换条件,则用z替代x,同时令$\gamma_x=0$. 其中μ_g是整数,经实验得到如下取值:

$$\mu_g=\begin{cases}20,\ 1\leqslant g\leqslant\tau;\\ 15,\ \tau<g\leqslant s-1;\\ 10,\ g=s.\end{cases}\tag{4}$$

如上所述,NS_1和全局搜索作用于x,只生成一个解z. 对于执行全局搜索所需的解y,当$x\in\mathrm{EB}_1$时,$y\in\mathrm{EB}_1$;当$x\in\mathrm{EB}_g(g>1)$时,$y\in\mathrm{EB}_j,\ j=1,2,\ldots,g-1$,即$x\in\mathrm{EB}_g(g>1)$总是与质量更好的$y$交叉,避免所选择的$y$比$x$差,这样可以有效地改善$x$.

2.4　跟随蜂阶段

对于跟随蜂群OB_g,跟随蜂阶段中,ϕ为实数,根据大量实验确定取值为0.8. 有

$$\mathrm{fit}_{x\in\mathrm{EB}_g}=1-\beta_x/|\mathrm{EB}_g|+\max\{0,1-\gamma_x/u_g\},\tag{5}$$

$$P_x=\mathrm{fit}_x\Big/\sum_{y\in\mathrm{EB}_g}\mathrm{fit}_y.\tag{6}$$

其中:β_x表示x是EB_g中的第β_x个解,fit_x为解x的适应度. 显然,fit_x由解的质量和进化质量决定,β_x越小,fit_x越高. 若x频繁被更新,则γ_x值为0或接近0,式(5)第2部分值更大,进化质量直接影响fit_x. 而现有ABC研究中,适应度通常仅由解的质量决定[30-31]. 其主要步骤如下.

step 1: 计算每个$x\in\mathrm{EB}_g$如式(6)所示的轮盘赌选择概率P_x,令$t=1$.

step 2: 使用轮盘赌选择一个$y\in\mathrm{EB}_g$.

step 3: 若$g=1$,则对y执行从NS_2和NS_3随机选择的邻域搜索;当$1<g\leqslant\tau$时,如果随机数$\alpha<\phi$,则对y执行NS_2,否则对y执行NS_3;若$g>\tau$,则对y执行NS_2. 得到新解z.

step 4: 如果替换条件成立,则运用z替代y并更新Ω. 更新γ_y,并对y执行第1种淘汰过程.

step 5: 若$t\leqslant|\mathrm{OB}_g|$,则$t=t+1$,转至step 2.

跟随蜂阶段主要采用邻域搜索,雇佣蜂阶段主要采用全局搜索,$\bar{N}\geqslant N$,EB_s被排斥在跟随蜂阶段之外,OB_g中的每个跟随蜂仅从对应的EB_g中选择一个解,从而加强对优秀解的搜索,淘汰EB_s以充分利用计算资源. 这是跟随蜂阶段新的实现方法,很少出现在以前的ABC中[30-31].

2.5　算法描述

MABC的主要步骤描述如下.

step 1: 采用启发式方法生成N个初始解构成EB,对于任意$x\in\mathrm{EB}$,令$\gamma_x=0$,Ω为空.

step 2: 构建s个雇佣蜂群$\mathrm{EB}_1,\mathrm{EB}_2,\ldots,\mathrm{EB}_s$.

step 3: 执行雇佣蜂阶段.

step 4: 对EB执行非劣排序和拥挤距离计算.

step 5: 执行跟随蜂阶段.

step 6: 执行第2种淘汰过程.

step 7: 对EB执行非劣排序和拥挤距离计算.

step 8: 重新调整外部档案Ω.

step 9: 若满足终止条件,则搜索结束,否则转至step 2.

算法中,终止条件是目标函数评价的最大值. 第2种淘汰过程如下:构造一个集合$\Lambda=\{x\in\Omega\,|\,x\notin\mathrm{EB}\}$,如果$|\Lambda|>1$,则随机选择一个元素$y\in\Lambda$,替换EB的最差解.

外部档案重新调整的过程如下:对于每个非劣解$x\in\mathrm{EB}$,如果$x\notin\Omega$,则将其加入Ω. 依据Pareto支配,比较Ω中所有元素,然后移除所有受支配解.

MABC具有以下特点:1) 构建了多个雇佣蜂群和跟随蜂群,OB_g跟随EB_g,EB_s没有跟随蜂跟随,雇佣蜂群EB_g $(g>1)$的解向优于该群的其他雇佣蜂群的解学习;2) 采用多样化搜索方式实现雇佣蜂阶段和跟随蜂阶段;3) 采用两种淘汰过程,其中第1种淘汰过程相当于侦查蜂阶段,第2种淘汰过程淘汰EB的最差解.

3　计算实验

为了验证MABC在求解考虑PM和SDST的UPMSP方面的有效性和优势,进行大量计算实验,所有测试实验均采用Microsoft visual C++ 2019编程实现,在8.0 G RAM, 2.30 GHz CPU环境下运行.

3.1 测试实例、对比算法以及评价指标

采用48个实例测试. 小规模实例$n=20, m=2$. 中规模实例$n=30, m\in\{2,3,4\}$. 大规模实例$n\in\{50,100,150,200\}, m\in\{10,15,20\}$. $p_{kj}\in[1,99]$. 有3种准备时间,分别是$[1,9]$、$[1,99]$和$[1,124]$. 小规模与大规模的加工时间和准备时间来自http://www.ci-ma.uadec.mx/instancias/, 中规模两种时间随机生成[11]. $w_k\in[1,99]$, PM间隔u_k和交货期d_j按照文献[12]的公式设定如下:

$$u_k=\max_{j=1,2,\cdots,n}\{s_{k0j}+p_{kj}+s_{kj0}\},\tag{7}$$

工件J_j的交货期为

$$d_j=(0.5+2\delta)\times\sum_{k=1}^{m}\frac{p_{kj}}{m},\tag{8}$$

其中δ为介于0到1的随机实数.

采用如下2个性能测试指标. DI_R[40]通过计算非劣解集Ω_l相对于参考集合Ω^*的距离衡量算法的收敛性能.

为了计算DI_R,所有算法非劣解集合$\bigcup_l\Omega_l$中的非劣解构成Ω^*, $\mathrm{DI}_R(\Omega_l)$越小,该算法收敛性越好, $\mathrm{DI}_R(\Omega_l)=0$表明算法产生的解都在参考集$\Omega^*$内. ρ_l指标[41]表示集合$\{x\in\Omega_l\,|\,x\in\Omega^*\}$与$|\Omega^*|$的大小比例. DI_R和ρ_l经常应用于度量调度算法的性能[12,28].

选用多目标多点模拟退火算法[42](MOMSA), 多目标和声搜索[43](MOHS)与新型帝国竞争算法(NICA[12])作为对比算法.

MOMSA用于求解多目标UPMSP. 在解码过程中,加入准备时间和维修后,MOMSA可以直接应用于考虑PM和SDST的多目标UPMSP求解. MOHS中,应用规则将实数调度串转换为离散工件排列. 机器分配通过对机器分配串元素取整来确定,在解码过程中采用本文方法处理准备时间和维修后,MOHS能够直接求解本文问题. NICA本身用来求解具有SDST和PM的UPMSP,可直接用作对比算法.

MABC具有以下参数: $N, \bar{N}, s, \max\_\mathrm{it}$. 对于$s$, 如前所述, $s\geqslant 2$,如果$s=2$,则跟随蜂阶段的多样性搜索无法进行,因此$s\geqslant 3$. 为了处理方便,限定$s\leqslant 4$, 这样s应等于3或4. 由于s限定为4,采用如下方法确定各蜂群大小: 令$|\mathrm{EB}_s|=N/6$,若$s=4$,则$|\mathrm{EB}_1|=|\mathrm{EB}_2|$, $|\mathrm{EB}_3|=|\mathrm{EB}_4|$, $|\mathrm{OB}_1|:|\mathrm{OB}_2|:|\mathrm{OB}_3|=7:4:1$; 若$s=3$,则$|\mathrm{EB}_1|=|\mathrm{EB}_2|$, $|\mathrm{OB}_1|:|\mathrm{OB}_2|=7:4$.

对算法其他主要参数进行田口实验,取$n=30$, $m=4$, SDST为$[1,124]$. 实验结果表明,当$N=30$, $\bar{N}=60, s=4, \max_\mathrm{it}=10^5$时, MABC性能最好,故选择该组参数.

关于对比算法,通过实验发现,当$\max_\mathrm{it}=10^5$时,几个对比算法均无法产生更好的计算结果,同时考虑比较的公平性,对比算法均采用与MABC一样的终止条件. 限于篇幅,以上算法参数实验的相关表格未列出.

3.2 MABC策略有效性

将MABC与MABC1和MABC2进行比较,以验证MABC的策略有效性. MABC1的参数设置与MABC相同,其与MABC不同之处在于初始种群随机产生. MABC2中, $\bar{N}=N, s=2, \mathrm{EB}_2$为空,即没有多群体和全局搜索与邻域搜索的多样化搜索, MABC2其他步骤和参数与MABC相同.

每个算法独立运行10次. 表1和表2描述了MABC及其两个变体的计算结果. 结果表明: MABC关于41个实例获得的DI_R小于MABC1,只是对于4个实例所得的ρ小于MABC1,这表明,启发式初始

表1 MABC及其两个变体关于指标DI_R的计算结果

index	MABC	MABC1	MABC2	index	MABC	MABC1	MABC2	index	MABC	MABC1	MABC2
1	3.79	0.00	12.12	17	**0.00**	19.65	37.97	33	3.20	1.80	101.63
2	**0.64**	1.62	10.63	18	**0.00**	3.22	62.87	34	**0.00**	43.34	48.85
3	**0.00**	49.75	41.70	19	**0.00**	53.50	15.02	35	**0.00**	8.34	95.31
4	6.99	1.94	18.30	20	**0.00**	7.87	41.71	36	**0.00**	13.30	61.82
5	**1.17**	8.45	24.62	21	**0.00**	27.39	22.34	37	**0.00**	21.22	20.16
6	12.48	7.26	33.74	22	**0.00**	13.71	50.68	38	**0.03**	2.98	92.05
7	**0.00**	17.69	3.57	23	3.50	0.00	82.85	39	**0.00**	16.71	74.22
8	**0.03**	4.08	13.11	24	1.31	1.02	62.76	40	**0.00**	12.87	67.04
9	**0.00**	13.84	48.06	25	**0.00**	23.44	26.37	41	**0.00**	12.40	99.10
10	**6.02**	7.23	10.40	26	**0.00**	8.18	88.11	42	**0.00**	2.68	90.74
11	2.66	0.78	23.45	27	**0.00**	6.79	70.69	43	**0.00**	65.73	57.84
12	**0.00**	15.68	47.46	28	**0.00**	28.38	31.99	44	**0.00**	6.10	59.86
13	**0.00**	2.95	11.19	29	**0.37**	2.47	33.24	45	**0.00**	11.47	102.91
14	**0.00**	7.71	39.85	30	**0.00**	11.27	51.16	46	**0.00**	118.99	36.87
15	**0.00**	20.88	50.60	31	**0.00**	13.03	54.67	47	**0.00**	8.78	81.43
16	**0.00**	79.03	26.53	32	**0.00**	7.22	103.28	48	**0.00**	14.36	83.69

表 2　MABC及其两个变体关于指标ρ的计算结果

index	MABC	MABC1	MABC2	index	MABC	MABC1	MABC2	index	MABC	MABC1	MABC2
1	0.50	1.00	0.50	17	**1.00**	0.00	0.00	33	0.33	0.67	0.00
2	**0.78**	0.56	0.00	18	**1.00**	0.00	0.00	34	**1.00**	0.00	0.00
3	**1.00**	0.00	0.00	19	**1.00**	0.00	0.00	35	**1.00**	0.00	0.00
4	**0.75**	0.25	0.00	20	**1.00**	0.00	0.00	36	**1.00**	0.00	0.00
5	**0.80**	0.20	0.00	21	**1.00**	0.00	0.00	37	**1.00**	0.00	0.00
6	0.50	0.50	0.00	22	**1.00**	0.00	0.00	38	**0.91**	0.09	0.00
7	**1.00**	0.00	0.00	23	0.00	1.00	0.00	39	**1.00**	0.00	0.00
8	**0.88**	0.13	0.00	24	0.50	0.50	0.00	40	**1.00**	0.00	0.00
9	**1.00**	0.00	0.00	25	**1.00**	0.00	0.00	41	**1.00**	0.00	0.00
10	**0.87**	0.33	0.00	26	**1.00**	0.00	0.00	42	**1.00**	0.00	0.00
11	0.33	0.67	0.00	27	**1.00**	0.00	0.00	43	**1.00**	0.00	0.00
12	**1.00**	0.00	0.00	28	**1.00**	0.00	0.00	44	**1.00**	0.00	0.00
13	**1.00**	0.50	0.00	29	**0.75**	0.25	0.00	45	**1.00**	0.00	0.00
14	**1.00**	0.00	0.00	30	**1.00**	0.00	0.00	46	**1.00**	0.00	0.00
15	**1.00**	0.00	0.00	31	**1.00**	0.00	0.00	47	**1.00**	0.00	0.00
16	**1.00**	0.00	0.00	32	**1.00**	0.00	0.00	48	**1.00**	0.00	0.00

化方法能够改善MABC的性能. MABC2关于所有实例所得的DI_R大于或等于MABC的相应结果, 关于44个实例, MABC2的DI_R至少比MABC的DI_R大10, 即MABC2的非支配解远离MABC的非支配解. MABC关于ρ也显著优于MABC2, 表明多蜂群和差异化搜索是必要且有效的.

3.3　结果与分析

3个对比算法的终止条件与MABC一样以保证公平比较. 每个算法关于每个实例独立运行10次. 表3～5列出了MABC与3种算法的对比结果和计算时间. 图1显示了4种算法的非支配解分布. 图2描述了4种算法的箱形图.

表 3　MABC与对比算法关于指标DI_R的计算结果

index	MABC	MOHS	MOMSA	NICA	index	MABC	MOHS	MOMSA	NICA	index	MABC	MOHS	MOMSA	NICA
1	46.50	3.50	82.21	39.28	17	24.84	20.13	105.14	6.68	33	**0.00**	111.07	39.04	9.20
2	**2.24**	13.93	46.88	5.17	18	8.53	6.92	90.85	1.91	34	12.62	96.31	40.71	0.00
3	**0.00**	53.03	85.17	43.82	19	10.09	0.00	100.00	5.59	35	9.51	92.19	99.00	1.40
4	**0.00**	46.99	27.58	24.29	20	**10.38**	10.49	122.59	25.72	36	**7.46**	92.29	29.68	10.07
5	**0.00**	50.94	24.64	34.71	21	11.11	11.29	85.80	5.80	37	**0.24**	95.17	41.56	8.24
6	**0.00**	50.65	113.48	18.82	22	**0.00**	84.27	14.52	15.50	38	31.74	74.88	44.69	0.00
7	**0.00**	13.11	22.13	4.81	23	**0.00**	104.86	44.76	23.27	39	**0.00**	71.90	47.40	31.71
8	**2.36**	58.71	49.48	10.86	24	**0.00**	117.37	57.95	27.53	40	**0.27**	74.78	10.49	2.27
9	**0.11**	58.83	91.25	8.38	25	**0.00**	60.13	65.19	19.81	41	**0.00**	81.33	18.04	3.06
10	**0.00**	13.63	66.96	15.46	26	**3.34**	79.73	33.75	5.13	42	**0.00**	92.07	33.58	6.12
11	**0.00**	40.89	90.72	17.99	27	**0.00**	99.42	84.62	32.78	43	**0.61**	103.02	62.83	1.44
12	**0.00**	59.70	78.47	8.24	28	10.86	33.54	42.07	0.00	44	10.13	122.02	41.27	0.93
13	**0.10**	28.31	64.56	2.74	29	78.69	31.78	104.95	31.78	45	**0.00**	112.51	84.68	10.12
14	20.99	65.15	119.35	0.00	30	14.38	70.84	90.96	1.56	46	**0.00**	69.17	27.97	6.96
15	**0.00**	38.43	110.73	25.95	31	4.01	103.05	59.24	0.00	47	57.29	86.69	60.41	0.00
16	**0.00**	1.29	100.00	5.18	32	**0.00**	102.78	41.55	23.52	48	63.56	125.34	21.50	8.00

表 4　MABC与对比算法关于指标ρ的计算结果

index	MABC	MOHS	MOMSA	NICA	index	MABC	MOHS	MOMSA	NICA	index	MABC	MOHS	MOMSA	NICA
1	0.17	0.67	0.00	0.17	17	0.20	0.40	0.00	0.40	33	**1.00**	0.00	0.00	0.00
2	**0.63**	0.00	0.00	0.38	18	0.17	0.33	0.00	0.50	34	0.00	0.00	0.00	1.00
3	**1.00**	0.00	0.00	0.00	19	0.00	1.00	0.00	0.00	35	0.20	0.00	0.00	0.80
4	**1.00**	0.00	0.00	0.00	20	**0.67**	0.33	0.00	0.00	36	**0.50**	0.00	0.17	0.33
5	**1.00**	0.00	0.00	0.00	21	0.33	0.17	0.00	0.50	37	**0.83**	0.00	0.00	0.17
6	**1.00**	0.00	0.00	0.00	22	**1.00**	0.00	0.00	0.00	38	0.00	0.00	0.00	1.00
7	**1.00**	0.00	0.00	0.00	23	**1.00**	0.00	0.00	0.00	39	**1.00**	0.00	0.00	0.00
8	**0.67**	0.00	0.00	0.33	24	**1.00**	0.00	0.00	0.00	40	**0.83**	0.00	0.00	0.17
9	**0.83**	0.00	0.00	0.17	25	**1.00**	0.00	0.00	0.00	41	**1.00**	0.00	0.00	0.00
10	**1.00**	0.00	0.00	0.00	26	**0.60**	0.00	0.00	0.40	42	**1.00**	0.00	0.00	0.00
11	**1.00**	0.00	0.00	0.00	27	**1.00**	0.00	0.00	0.00	43	**0.75**	0.00	0.00	0.25
12	**1.00**	0.00	0.00	0.00	28	0.00	0.00	0.00	1.00	44	0.25	0.00	0.00	0.75
13	**0.67**	0.00	0.00	0.33	29	0.00	0.50	0.00	0.50	45	**1.00**	0.00	0.00	0.00
14	0.00	0.00	0.00	1.00	30	0.25	0.00	0.00	0.75	46	**1.00**	0.00	0.00	0.00
15	**1.00**	0.00	0.00	0.00	31	0.00	0.00	0.00	1.00	47	0.00	0.00	0.00	1.00
16	**1.00**	0.00	0.00	0.00	32	**1.00**	0.00	0.00	0.00	48	0.00	0.00	0.33	0.67

表 5 4种算法的运行时间

index	MABC	MOHS	MOMSA	NICA	index	MABC	MOHS	MOMSA	NICA	index	MABC	MOHS	MOMSA	NICA
1	1.84	2.35	1.30	1.58	17	2.31	3.82	2.84	2.65	33	7.15	5.34	4.85	4.88
2	2.02	2.18	1.39	1.39	18	2.60	2.96	3.29	2.25	34	5.81	5.50	5.42	4.62
3	2.07	2.09	0.82	1.67	19	1.75	4.22	3.47	2.90	35	6.35	5.35	5.72	4.57
4	2.46	2.40	1.07	1.53	20	2.03	3.92	2.83	2.71	36	6.48	5.71	5.29	5.23
5	2.52	1.79	0.90	1.28	21	2.18	3.64	2.76	2.91	37	5.94	6.20	6.94	6.11
6	2.57	1.83	1.20	1.46	22	3.98	4.62	2.97	3.46	38	6.70	5.85	5.83	5.84
7	2.21	2.40	1.11	1.59	23	4.41	4.35	2.86	3.24	39	6.23	5.86	5.44	5.78
8	2.33	2.14	0.88	1.34	24	4.40	4.89	3.08	3.50	40	9.74	7.98	7.77	6.70
9	2.40	2.49	1.08	1.60	25	3.72	4.69	3.52	4.34	41	11.35	7.34	7.06	6.07
10	2.05	2.54	1.12	1.69	26	4.06	4.44	3.00	4.23	42	10.87	7.05	6.79	5.98
11	2.21	2.71	1.67	1.56	27	4.16	4.81	3.28	4.29	43	8.75	7.88	7.27	7.26
12	2.16	2.35	1.09	1.40	28	3.41	5.51	6.40	5.62	44	9.67	7.67	7.64	7.49
13	2.45	4.09	3.10	2.55	29	4.32	5.09	3.37	4.96	45	9.71	8.09	6.61	7.67
14	2.58	4.04	2.93	2.31	30	4.04	5.01	3.36	5.03	46	8.74	8.25	8.19	8.21
15	2.54	3.86	2.03	2.62	31	6.55	5.71	3.74	5.17	47	9.67	8.54	7.29	8.07
16	1.84	3.22	3.43	2.83	32	7.53	5.49	5.13	5.03	48	9.70	8.28	6.95	8.05

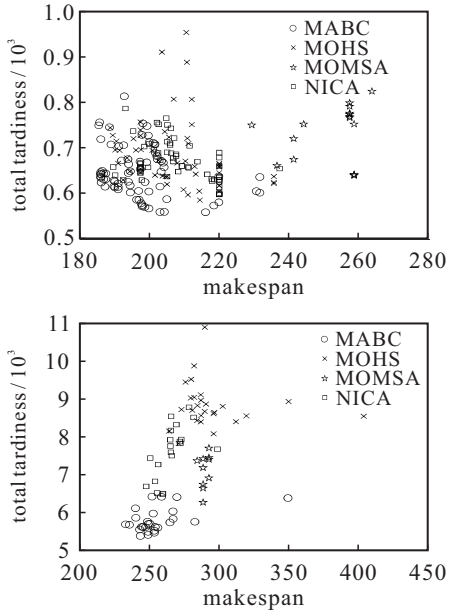

图 1 4种算法在2个实例上的非劣解图

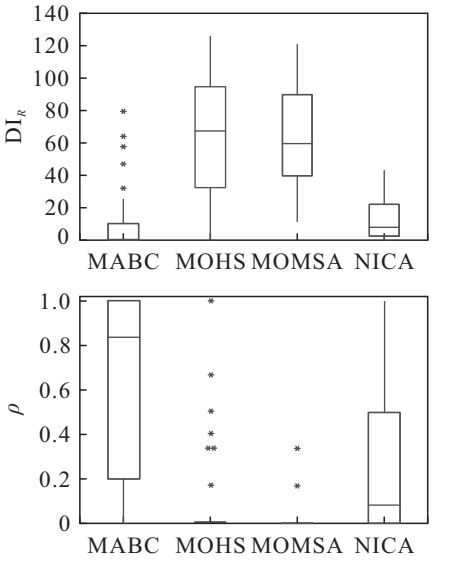

图 2 两种指标的箱线图

如表 3 和表 4 所示, 关于 47 个实例, MABC 的 DI_R 和 ρ 优于 MOMSA 的相应结果. 对于 MOHS, 仅关于 5 个实例取得了优于 MABC 的 DI_R, 关于 5 个实例, MOHS 取得了优于 MABC 的 ρ 值. MABC 关于 32 个实例所得的 DI_R 小于 NICA, 关于 34 个实例的 ρ 大于 NICA. 此外, MABC 关于 22 个实例的 DI_R 等于 0, 且关于 22 个实例提供了集合 Ω^* 的所有元素. 如图 1 和图 2 所示, MABC 的搜索性能优于对比算法. 因此, 在相同的计算时间内, MABC 在大多数情况下都能得到比 3 种比较算法更好的结果, 具有较强的搜索优势.

MABC 中, 启发式算法解决了初始解质量低的问题, 多雇佣蜂群和多跟随蜂群的引入以及全局搜索和三邻域搜索的差异化组合有效地避免搜索陷入局部最优, 两个淘汰过程提高了某些差解的质量. 这些特点使得差解不断被淘汰并充分利用了优秀解, 有效实现了全局搜索和局部搜索的良好平衡. 因此, MABC 有效地解决了具有 PM 和 SDST 的 UPMSP.

4 结 论

PM 和 SDST 通常同时存在并行机制造过程, 故研究包含 PM 和 SDST 的双目标 UPMSP 非常必要, 然而, 目前关于这类问题的研究进展较少. 为此, 本文提出了一种 MABC 算法. 该算法将所有雇佣蜂分为 s 个雇佣蜂群, 除最差雇佣蜂群外, 每个雇佣蜂群都对应 1 个跟随蜂群. 采用全局搜索和 3 种邻域搜索的不同组合实现雇佣蜂阶段和跟随蜂阶段, 并引入了两种淘汰方法. 通过大量实验测试 MABC 的新策略和性能, 计算结果表明, MABC 在求解具有 PM 和 SDST 的 UPSMP 方面具有较强的搜索优势.

具有SDST等实际加工约束的UPMSP是一个重要的调度研究主题,广泛存在于实际的制造系统中. 下一步将继续关注这类问题,并应用蛙跳算法和教-学优化算法等进行解决,同时多工厂分布式绿色调度也是未来的研究课题.

参考文献(References)

[1] 吴楚格, 王凌, 郑晓龙. 求解不相关并行机调度的一种自适应分布估计算法[J]. 控制与决策, 2016, 31(12): 2177-2182.
(Wu C G, Wang L, Zheng X L. An adaptive estimation of distribution algorithm for solving the unrelated parallel machine scheduling[J]. Control and Decision, 2016, 31(12): 2177-2182.)

[2] Jans R, Degraeve Z. An industrial extension of the discrete lot-sizing and scheduling problem[J]. IIE Transactions, 2004, 36(1): 47-58.

[3] Silva C, Magalhaes J M. Heuristic lot size scheduling on unrelated parallel machines with applications in the textile industry[J]. Computers & Industrial Engineering, 2006, 50(1/2): 76-89.

[4] Pattloch M, Schmidt G, Kovalyov M Y. Heuristic algorithms for lotsize scheduling with application in the tobacco industry[J]. Computers & Industrial Engineering, 2001, 39(3/4): 235-253.

[5] Wang H B, Alidaee B. Effective heuristic for large-scale unrelated parallel machines scheduling problems[J]. Omega, 2019, 83: 261-274.

[6] Li G, Liu M Q, Sethi S P, et al. Parallel-machine scheduling with machine-dependent maintenance periodic recycles[J]. International Journal of Production Economics, 2017, 186: 1-7.

[7] Yang D L, Cheng T C E, Yang S J, et al. Unrelated parallel-machine scheduling with aging effects and multi-maintenance activities[J]. Computers & Operations Research, 2012, 39(7): 1458-1464.

[8] Tavana M, Zarook Y, Santos-Arteaga F J. An integrated three-stage maintenance scheduling model for unrelated parallel machines with aging effect and multi-maintenance activities[J]. Computers & Industrial Engineering, 2015, 83: 226-236.

[9] Wang S J, Liu M. Multi-objective optimization of parallel machine scheduling integrated with multi-resources preventive maintenance planning[J]. Journal of Manufacturing Systems, 2015, 37: 182-192.

[10] Gara-Ali A, Finke G, Espinouse M L. Parallel-machine scheduling with maintenance: Praising the assignment problem[J]. Europeab Journal of Operational Research, 2016, 252(1): 90-97.

[11] Avalos-Rosales O, Angel-Bello F, Alvarez A, et al. Including preventive maintenance activities in an unrelated parallel machine environment with dependent setup times[J]. Computers & Industrial Engineering, 2018, 123: 364-377.

[12] Wang M, Pan G H. A novel imperialist competitive algorithm with multi-elite individuals guidance for multi-object unrelated parallel machine scheduling problem[J]. IEEE Access, 2019, 7: 121223-121235.

[13] Parker R G, Deane R H, Holmes R A. On the use of a vehicle routing algorithm for the parallel processor problem with sequence dependent changeover costs[J]. AIIE Transactions, 1977, 9(2): 155-160.

[14] Kurz M E, Askin R G. Heuristic scheduling of parallel machines with sequence-dependent set-up times[J]. International Journal of Production Research, 2001, 39(16): 3747-3769.

[15] Arnaout J P, Rabadi G, Musa R. A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times[J]. Journal of Intelligent Manufacturing, 2010, 21(6): 693-701.

[16] Vallada E, Ruiz R. A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times[J]. European Journal of Operational Research, 2011, 211(3): 612-622.

[17] Lin S W, Ying K C. ABC-based manufacturing scheduling for unrelated parallel machines with machine-dependent and job sequence-dependent setup times[J]. Computers & Operations Research, 2014, 51: 172-181.

[18] Caniyilmaz E, Benli B, Ilkay M S. An artificial bee colony algorithm approach for unrelated parallel machine scheduling with processing set restrictions, job sequence-dependent setup times, and due date[J]. The International Journal of Advanced Manufacturing Technology, 2015, 77(9/10/11/12): 2105-2115.

[19] Diana R O M, De França Filoh M F, De Souza S R, et al. An immune-inspired algorithm for an unrelated parallel machines' scheduling problem with sequence and machine dependent setup-times for makespan minimisation[J]. Neurocomput, 2015, 163: 94-105.

[20] Wang L, Wang S Y, Zheng X L. A hybrid estimation of distribution algorithm for unrelated parallel machine scheduling with sequence-dependent setup times[J]. IEEE/CAA Journal of Automatica Sinica, 2016, 3(3): 235-246.

[21] Ezugwu A E, Akutsah F. An improved firefly algorithm for the unrelated parallel machines scheduling problem with sequence-dependent setup times[J]. IEEE Access, 2018, 6: 54459-54478.

[22] Fanjul-Peyro L, Ruiz R, Perea F. Reformulations and an exact algorithm for unrelated parallel machine scheduling problems with setup times[J]. Computers & Operations Research, 2019, 101: 173-182.

[23] Bektur G, Sarac T. A mathematical model and heuristic algorithms for an unrelated parallel machine scheduling problem with sequence-dependent setup times, machine eligibility restrictions and a common server[J]. Computers & Operations Research, 2019, 103:

46-63.

[24] Afzalirad M, Rezaeian J. A realistic variant of bi-objective unrelated parallel machine scheduling problem: NSGA-II and MOACO approaches[J]. Applied Soft Computing, 2017, 50: 109-123.

[25] 宋强. 求解异构并行机调度问题的混合多目标教-学优化算法[J]. 控制理论与应用, 2020, 37(10): 2242-2256.
(Song Q. A hybrid multi-objective teaching-learning-based optimization algorithm for unrelated parallel machine scheduling problem[J]. Control Theory & Applications, 2020, 37(10): 2242-2256.)

[26] 王凌, 周刚, 许烨, 等. 求解不相关并行机混合流水线调度问题的人工蜂群算法[J]. 控制理论与应用, 2012, 29(12): 1551-1557.
(Wang L, Zhou G, Xu Y, et al. An artificial bee colony algorithm for solving hybrid flow-shop scheduling problem with unrelated parallel machines[J]. Control Theory & Applications, 2012, 29(12): 1551-1557.)

[27] Han Y Y, Gong D W, Sun X Y. A discrete artificial bee colony algorithm incorporating differential evolution for the flow-shop scheduling problem with blocking[J]. Engineering Optimization, 2015, 47(7): 927-946.

[28] 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.

[29] 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.)

[30] Lei D M, Guo X P. Scheduling job shop with lot streaming and transportation through a modified artificial bee colony[J]. International Journal of Production Research, 2013, 51(16): 4930-4941.

[31] Li X Y, Xiao S Q, Wang C Y, et al. Mathematical modeling and a discrete artificial bee colony algorithm for the welding shop scheduling problem[J]. Memetic Computing, 2019, 11(4): 371-389.

[32] 匡芳君, 徐蔚鸿, 金忠. 自适应Tent混沌搜索的人工蜂群算法[J]. 控制理论与应用, 2014, 31(11): 1502-1509.
(Kuang F J, Xu W H, Jin Z. Artificial bee colony algorithm based on self-adaptive Tent chaos search[J]. Control Theory & Applications, 2014, 31(11): 1502-1509.)

[33] 暴励, 曾建潮. 一种双种群差分蜂群算法[J]. 控制理论与应用, 2011, 28(2): 266-272.
(Bao L, Zeng J C. A bi-group differential artificial bee colony algorithm[J]. Control Theory & Applications, 2011, 28(2): 266-272.)

[34] Zhou J J, Yao X F, Chan F T S, et al. An individual dependent multi-colony artificial bee colony algorithm[J]. Information Sciences, 2019, 485: 114-140.

[35] Xiang Y, Zhou Y R. A dynamic multi-colony artificial bee colony algorithm for multi-objective optimization[J]. Applied Soft Computing, 2015, 35: 766-785.

[36] Gao J Q, He G X, Wang Y S. A new parallel genetic algorithm for solving multiobjective scheduling problems subjected to special process constraint[J]. The International Journal of Advanced Manufacturing Technology, 2009, 43(1/2): 151-160.

[37] Afzalirad M, Rezaeian J. A realistic variant of bi-objective unrelated parallel machine scheduling problem: NSGA-II and MOACO approaches[J]. Applied Soft Computing, 2017, 50: 109-123.

[38] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182-197.

[39] Oliver I M, Smith D J, Holland J R C. A study of permutation crossover operators on the traveling salesman problem[C]. Proceedings of the 2nd International Conference on Genetic Algorithms on Genetic Algorithms and Their Application. 1987: 224-230.

[40] Knowles J, Corne D. On metrics for comparing nondominated sets[C]. Proceedings of 2002 Congress on Evolutionary Computation. Piscataway: IEEE, 2002, 711-716.

[41] Lei D. Pareto archive particle swarm optimization for multi-objective fuzzy job shop scheduling problems[J]. The International Journal of Advanced Manufacturing Technology, 2008, 37(1/2): 157-165.

[42] Lin S W, Ying K C. A multi-point simulated annealing heuristic for solving multiple objective unrelated parallel machine scheduling problems[J]. Internation Journal of Production Research, 2015, 53(4): 1065-1076.

[43] Shahidi-Zadeh B, Tavakkoli-Moghaddam R, Taheri-Moghadam A, et al. Solving a bi-objective unrelated parallel batch processing machines scheduling problem: A comparison study[J]. Computers Operations Research, 2017, 88: 71-90.

作者简介

雷德明 (1968−), 男, 教授, 博士生导师, 从事智能系统优化与控制等研究, E-mail: deminglei11@163.com;

杨海 (1996−), 男, 硕士生, 从事制造系统智能优化与调度的研究, E-mail: YangHai_2049@163.com.

(责任编辑：魏 冰)