超启发式人工蜂群算法求解多场景鲁棒分布式置换流水车间调度问题

连 戈[1,2], 朱 荣[3†], 钱 斌[1,2], 吴绍云[4], 胡 蓉[1,2]

(1. 昆明理工大学 信息工程与自动化学院, 云南 昆明 650500;

2. 昆明理工大学 云南省人工智能重点实验室, 云南 昆明 650500;

3. 昆明理工大学 城市学院, 云南 昆明 650051; 4. 云南玉溪水松纸厂, 云南 玉溪 653100)

摘要: 本文考虑现实中广泛存在的加工时间不确定的分布式置换流水车间调度问题(DPFSP), 研究如何建立问题模型和设计求解算法, 方可确保算法最终获得的解在多个典型DPFSP场景下, 均具有能满足客户期望的较小优化目标值(即makespan值). 在问题建模方面, 首先, 采用场景法构建多个不同典型场景以组成场景集(每个场景对应1个具有不同加工时间的DPFSP), 并设定合适的makespan值作为场景阈值, 用于在评价问题解时从场景集中动态筛选出“坏”场景子集; 其次, 在常规优化目标makespan的基础上, 结合“坏”场景子集概念提出可实现鲁棒调度的新型优化目标, 用于引导算法每代加强对当前“坏”场景子集中每个DPFSP场景对应解空间的搜索; 然后, 结合所提的新型优化目标, 建立基于多场景的鲁棒DPFSP (MSRDPFSP). 在算法设计方面, 提出一种超启发式人工蜂群算法(HHABC)对MSRDPFSP进行求解. HHABC分为高、低两层结构, 其中低层设计6种启发式操作(HO), 高层采用人工蜂群算法控制和选择低层HOs来不断生成新的混合启发式算法, 从而实现在不同场景对应解空间中的较深入搜索. 在不同规模测试问题上的仿真实验与算法对比, 验证了HHABC的有效性.

关键词: 分布式置换流水车间调度问题; 多场景; 鲁棒调度; 人工蜂群算法; 超启发式算法

引用格式: 连戈, 朱荣, 钱斌, 等. 超启发式人工蜂群算法求解多场景鲁棒分布式置换流水车间调度问题. 控制理论与应用, 2023, 40(4): 713 – 723

DOI: 10.7641/CTA.2022.11026

Hyper-heuristic artificial bee colony algorithm for the multi-scenario-based robust distributed permutation flow-shop scheduling problem

LIAN Ge[1,2], ZHU Rong[3†], QIAN Bin[1,2], WU Shao-yun[4], HU Rong[1,2]

(1. School of Information Engineering and Automation,
Kunming University of Science and Technology, Kunming Yunnan 650500, China;
2. Yunnan Key Laboratory of Artificial Intelligence, Kunming University of Science and Technology, Kunming Yunnan 650500, China;
3. City College, Kunming University of Science and Technology, Kunming Yunnan 650051, China;
4. Tipping Paper Mill of Yuxi, Yuxi Yunnan 653100, China)

Abstract: This paper considers the widely existing distributed permutation flow shop scheduling problem (DPFSP) with uncertain processing time, and studies how to establish the problem model and design the solution algorithm, so as to ensure that the final solution obtained by the algorithm has a smaller optimization target value (i.e. makespan value) that can meet customer expectation in multiple typical scenarios of DPFSP. In terms of problem modeling, firstly, the scenario method is used to construct multiple different typical scenarios to form a scenario set in which each scenario corresponds to a DPFSP with different processing time, and the appropriate makespan value is selected as the scenario threshold to dynamically filter out the bad scenario subset from the scenario set when evaluating the problem's solution; secondly, based on the conventional optimization objective (i.e., makespan) and combined with the concept of "bad" scene subset, a new optimization objective that can realize robust scheduling is proposed to guide each generation of the algorithm to strengthen the search in the corresponding solution space of each DPFSP's scenario in the current "bad" scenario set; thirdly, combined with the proposed new optimization objective, a multi-scenario-based robust DPFSP (MSRDPFSP) is established. In terms

收稿日期: 2021−10−26; 录用日期: 2022−02−24.

†通信作者. E-mail: 1270030921@qq.com; Tel.: +86 13354640533.

本文责任编委: 王凌.

国家自然科学基金项目(62173169, 61963022), 云南省基础研究重点项目(202201AS070030)资助.

Supported by the National Natural Science Foundation of China (62173169, 61963022) and the Basic Research Key Project of Yunnan Province (202201AS070030).

of algorithm design, a hyper-heuristic artificial bee colony algorithm (HHABC) is proposed to solve the MSRDPFSP. The HHABC is divided into a high-level and low-level structure. The low level is designed with six heuristic operations (HO), and the high level utilizes the artificial bee colony algorithm to control and select low-level HOs to continuously generate new hybrid heuristic algorithms, which are used to realize in-depth search in the corresponding solution spaces of different scenarios. Simulation experiments and algorithm comparisons on the test problems with different scales verify the effectiveness of HHABC.

Key words: distributed permutation flow-shop scheduling problem; multi-scenario; robust scheduling; artificial bee colony algorithm; hyper-heuristic algorithm

Citation: LIAN Ge, ZHU Rong, QIAN Bin, et al. Hyper-heuristic artificial bee colony algorithm for the multi-scenario-based robust distributed permutation flow-shop scheduling problem. *Control Theory & Applications*, 2023, 40(4): 713 – 723

1 引言

随着经济全球化进程的不断推进, 分布式制造作为一种能有效分散生产能力与缓解生产压力的生产模式现已被广泛应用于多数企业. 因此, 近年来分布式置换流水车间调度问题(distributed permutation flow-shop scheduling problem, DPFSP)得到了广泛关注与研究. DPFSP由Naderi等[1]首次提出, 文献[2–6]针对DPFSP分别提出了分散搜索 (scatter search, SS)算法、有界搜索迭代贪婪 (bounded-search iterated greedy, BSIG)算法、化学反应优化(chemical reaction optimization, CRO)算法、模因离散差分进化(memetic discrete differential evolution, MDDE)算法、偏随机迭代局部搜索 (biased-randomized iterated local search, BR–ILS)算法进行求解.

在实际的生产过程中可能会出现各种不确定性因素, 如设备老化或故障、工人水平参差不齐等, 企业无法在生产前得知哪种因素将会发生, 继而无法确定具体的加工时间, 因此可以采用鲁棒调度中的场景法刻画由不同因素导致的加工时间, 建立针对问题的鲁棒调度模型, 鲁棒调度模型通常以优化某些鲁棒性能准则为优化目标而不是随机调度所要求的系统性能, 有益于获得更加可行的调度方案. 因此, 研究多场景鲁棒分布式置换流水车间调度问题 (multi-scenario-based robust DPFSP, MSRDPFSP)具有深远的经济价值和社会意义. 在计算复杂度上, DPFSP已被证明具有非确定多项式难(non-deterministic polynomial hard, NP-hard)的属性, 而该问题又约归为MSRDP–FSP, 故MSRDPFSP属于NP-hard问题. 因此, 研究MSRDPFSP及其求解算法具有重要的理论价值和实际意义. 目前已有学者对鲁棒调度相关问题进行研究. Mulvey等[7]首次提出使用鲁棒调度方法替代随机规划方法来处理具有不确定因素的调度的问题. Kouvelis等[8]首次提出了场景法, 此方法是描述鲁棒调度中不确定因素的重要工具之一. Daniels等[9]通过区间场景描述不确定加工时间, 提出了最小–最大(后悔) (min– max (regret))准则, 来解决具有不确定加工时间的单机调度问题. Wang等[10]提出了一种基于“坏”场景子集的鲁棒调度模型应用于作业车间调度问题(job shop scheduling problem, JSP). Wu等[11]针对优化目标为最小化两个场景中最大完工时间的两阶段装配车间调度问题, 提出了一种分支定界算法(branch and bound algorithm, BBA)进行求解. 由文献调研可知, 对于实际生产中广泛存在的一类问题, 即具有不确定加工时间的分布式置换流水车间调度问题, 目前尚无学者采用鲁棒调度的方法对其进行研究.

考虑到MSRDPFSP的NP-hard属性, 现有数学规划算法难以在较短时间内获取问题的满意解, 故采用智能算法进行求解. 超启发式算法(hyper-heuristic algorithm, HHA)是一种新兴的智能优化算法. HHA为两层结构, 高层的策略域包含某种搜索算法(随机或学习型算法), 即高层策略域(high-level strategy, HLS); 低层的问题域包含需要求解问题的模型、评价函数, 以及一组直接搜索问题解空间的启发式算法(low-level heuristics, LLHs). 在HHA的迭代过程中, HLS动态的控制LLHs来不断组合生成新的混合启发式算法, 可实现对问题解空间不同区域的较深入搜索. 该算法具有很强的通用性, 因此近年来HHA已被应用于多种调度问题上. Cowling等[12]最早提出了超启发式算法的概念并将其用于求解调度问题. Lin等[13]针对优化目标为最小化最大完工时间的分布式装配流水车间调度问题, 提出了一种基于回溯搜索的超启发式(backtracking search based hyper-heuristic, BS–HH)算法进行求解. 李尚函等[14]针对优化目标为最小化最大模糊完工时间的模糊柔性作业车间调度问题, 提出了一种混合超启发式遗传算法(hybrid hyper-heuristic genetic algorithm, HHGA)进行求解. Lin等[15]针对优化目标为最小化最大完工时间的半导体测试调度问题, 提出了一种基于Q-learning的超启发式(Q-learning based hyper-heuristic, QHH)算法进行求解. 由文献调研可知, 高层采用高效的策略作为HLS与低层针对问题模型特性构造有效的LLHs, 是设HHA的关键.

人工蜂群(artificial bee colony, ABC)算法是一种新颖的智能算法, 该算法通过模拟蜜蜂自主和协同觅食过程来对问题的解空间进行搜索, 具有操作简

单、参数少、易于实现等特点[16], 其三阶段搜索过程逐步递进的算法框架相比其他传统智能算法具有更高的灵活性. 在面对一些复杂调度问题求解时, 由于其出色的全局搜索能力, ABC及其改进算法已被应用于多种调度问题中. 郑小操等[17]针对优化目标为最小化最大模糊完工时间的模糊柔性作业车间调度问题, 提出了一种基于邻域搜索的改进人工蜂群算法进行求解. Li等[18]针对考虑工件恶化效应和并行分批的DPFSP, 提出了一种具有5种局部搜索和新型侦查蜂搜索过程的混合人工蜂群算法(hybrid artificial bee colony algorithm, HABC)进行求解. Gong等[19] 针对优化目标为最小化最大完工时间的柔性作业车间调度问题, 提出了一种混合人工蜂群算法进行求解. 由文献调研可知, 由于调度问题的组合特性, 多数学者采用离散ABC算法与其他算法相结合对问题进行求解. 考虑到MSRDPFSP中复杂的解空间, 高层采用ABC作为HHA的高层策略域, 可以通过对ABC算法3个阶段的设计来动态控制和选择低层启发式操作, 生成更多有效混合启发式算法对其进行求解, 从而增加算法的求解深度.

本文针对MSRDPFSP的建模与算法求解进行研究. 在建模方面, 面对具有诸多不确性因素的DPFSP, 决策者不能事先确定哪种因素会发生, 也无法为这些因素分配发生的概率, 因此本文采用鲁棒调度中的场景法, 综合考虑各种不确定因素, 构建多个不同典型场景以组成场景集, 其中每个场景对应一个具有不同加工时间的DPFSP, 通过一个场景阈值来从场景集中动态筛选“坏”场景子集而不是只考虑最“坏”场景, 并在常规优化目标makespan的基础上, 结合“坏”场景子集概念提出一个基于场景阈值的新型优化目标即“坏”场景子集总惩罚量, 首次建立MSRDPFSP模型. 在算法求解方面, 由于MSRDPFSP的解空间较为复杂, 因此设计了一种融合了ABC与HH方法的超启发式人工蜂群 (hyper-heuristic artificial bee colony, H-HABC)算法进行求解. HHABC算法由高层策略域和低层操作域构成, 其中高层策略域使用人工蜂群算法来控制和选择低层启发式操作, 从而得到一组关于低层操作的序列, 低层则根据此序列对解空间依次进行搜索. 此外, 本文在算法高层中的侦查蜂阶段设计了一种基于关键工厂的扰动策略, 以避免算法过早陷入局部最优. 最后, 通过仿真实验和算法比较验证了所提HHABC的有效性.

2　多场景鲁棒分布式置换流水车间调度问题

2.1　符号定义

关于本文所涉及的数学符号及定义如表1所示.

2.2　问题模型

由于在实际生产过程中伴随着很多不确定性因素导致工件拥有不同的加工时间, 因此MSRDPFSP可描述为: 相较于DPFSP问题拥有确定的加工时间, MSRDPFSP考虑实际情况, 每个工件的加工时间为多个可能的离散值, 决策者无法预知哪种情况会发生便无法为每种情况分配合适的发生概率, 因此决策者需要综合考虑所有工件可能的加工时间将其划分为Λ个场景构成场景集. 以$N=3, M=3, |\Lambda|=3$的MSRDPFSP为例, 场景示意表如表2所示, 在场景1中, 第1个工件在3台机器的加工时间分别为$\{7,-3,-4\}$; 在场景2中, 第1个工件在3台机器的加工时间分别为$\{9,-5,-2\}$; 在场景3中, 第1个工件在3台机器的加工时间分别为$\{6,-5,-4\}$, 以此类推. DPFSP则可以看作MSRDPFSP具有单一场景的一种特殊情况.

表 1　符号表

Table 1　Notations

符号	释义
f	工厂序列, $f=\{1,2,\cdots,F\}$.
j	机器序列, $j=\{1,2,\cdots,M\}$.
i	工件序列, $i=\{1,2,\cdots,N\}$.
N_f	工厂f里的工件数.
ps	问题域种群容量.
π	工件加工序列, $\pi=\{\pi_1,\pi_2,\cdots,\pi_N\}$.
π^f	工厂f的工件加工序, $\pi^f=\{\pi_1^f,\pi_2^f,\cdots,\pi_{N_f}^f\}$.
$p_{i,j}$	工件i在机器j上的加工时间.
$C_{i,j}$	工件i在机器j上的完工时间.
C_f	工厂f的完工时间.
C	所有工厂的完工时间.
T	场景阈值.
Λ	场景集合, $\Lambda=\{\lambda_1,\lambda_2,\cdots,\lambda_{\lvert\Lambda\rvert}\}$.
S	可行解集合.
$C(s,\lambda)$	某一可行解s在某一场景λ下的完工时间.
$\Lambda_T(s)$	“坏”场景子集.
$PT(s)$	某一可行解s在“坏”场景子集下的总惩罚量.

表 2　MSRDPFSP场景示意表

Table 2　Illustrations of MSRDPFSP

Job	场景1			场景2			场景3		
	M1	M2	M3	M1	M2	M3	M1	M2	M3
1	7	3	4	9	5	2	6	5	4
2	3	12	9	7	16	5	5	10	6
3	3	4	5	4	8	7	5	8	6

MSRDPFSP中调度产生的解可能在不同的场景下有不同性能表现, 为确保算法最终获得的解在多个场景下, 均具有客户期望的较小优化目标值(即makespan值), MSRDPFSP首先通过决策者给出的场景阈值T筛选出属于某个解s的“坏”场景子集$\Lambda_T(s)$, 其中T由决策者根据客户对系统性能下降风险的忍受程度来给出并应大于所有场景下的最优性能值, 小于所

有场景下的最差性能. 其次通过s在某一“坏”场景下的makespan与T差值的平方表示对该“坏”场景的惩罚量, 属于s的“坏”场景子集的惩罚量之和表示“坏”场景子集的总惩罚量. 最后以“坏”场景子集总惩罚量最小化作为优化目标来获得一个在所有场景里综合表现较好的鲁棒解而不是在某一场景下表现最好的解. 基于问题描述建立如下计算表达式:

$$C_{\pi_1^f,1}=p_{\pi_1^f,1},\ f=1,2,\cdots,F; \tag{1}$$

$$C_{\pi_i^f,1}=C_{\pi_{i-1}^f,1}+p_{\pi_i^f,1},$$
$$f=1,2,\cdots,F,\ i=2,\cdots,N_f; \tag{2}$$

$$C_{\pi_1^f,j}=C_{\pi_1^f,j-1}+p_{\pi_1^f,j},$$
$$f=1,2,\cdots,F,\ j=2,\cdots,M; \tag{3}$$

$$C_{\pi_i^f,j}=\max\{C_{\pi_{i-1}^f,j},C_{\pi_i^f,j-1}\}+p_{\pi_i^f,j},$$
$$f=1,2,\cdots,F,\ i=2,\cdots,N_f,\ j=2,\cdots,M; \tag{4}$$

$$C_f=C_{\pi_{N_f}^f,M},\ f=1,2,\cdots,F; \tag{5}$$

$$C=\max_{f=1,2,\cdots,F}\{C_f\}; \tag{6}$$

$$\Lambda_T(s)=\{\lambda|C(s,\lambda)\geqslant T,\lambda\in\Lambda\}; \tag{7}$$

$$PT(s)=\sum_{\lambda\in\Lambda_T(s)}[C(s,\lambda)-T]^2; \tag{8}$$

$$\min_{s\in S}PT(s), \tag{9}$$

其中: 式(1)–(5)用于计算各个工厂在某一场景下的完工时间, 式(6)用于计算所有工厂在某一场景下的总完工时间, 式(7)通过一个初始设定的场景阈值筛选所有场景中的“坏”场景子集, 式(8)用于计算“坏”场景子集的总惩罚量, 式(9)通过HHABC算法寻找到一个“坏”场景子集总惩罚最小的可行解即优化目标. HHABC算法将在第3节进行详细介绍.

3 超启发式人工蜂群算法(HHABC)

在HHABC中, 首先在高层策略域利用ABC算法对6种邻域操作的组合顺序(每个组合顺序为一个高层种群个体)进行排列优化, 从而获得高层策略域种群, 然后在低层问题域将每个高层策略域个体作为独立的启发式算法控制低层对应个体进行变邻域局部搜索. HHABC的两层结构示意图如图1所示.

3.1 编码与解码

对于高层策略域, 种群中每个个体在算法不同阶段可以由LLH1–LLH5混合构成或由LLH6单独构成, 其表现形式为各启发式操作对应编号所组成的序列, 编码时允许包含LLH1–LLH5中相同的低层启发式操作, 编码长度越长可表示的混合启发式算法越多, 但进行搜索时花费的时间也越多, 因此为满足LLH1–LLH5可以在高层个体中同时出现两次, 将编码长度设为10, 由于一个高层策略域个体控制一个低层问题域个体, 因此高层策略域种群容量与低层问题域种群容量为ps. 解码高层策略域个体时, 将相应低层问题域个体从左到右依次执行高层策略域个体每一位所代表的低层启发式操作, 每执行完一个低层启发式操作, 就根据总惩罚量对操作前后的两个解进行比较, 若新解优于旧解则使用新解代替旧解, 并继续执行剩余低层启发式操作; 否则, 保留旧解并继续执行剩余低层启发式操作直至执行完高层策略域个体中所有启发式操作, 高层个体的适应值即为其对应更新的低层问题域个的适应值. 高层策略域个体示意图如图2所示.

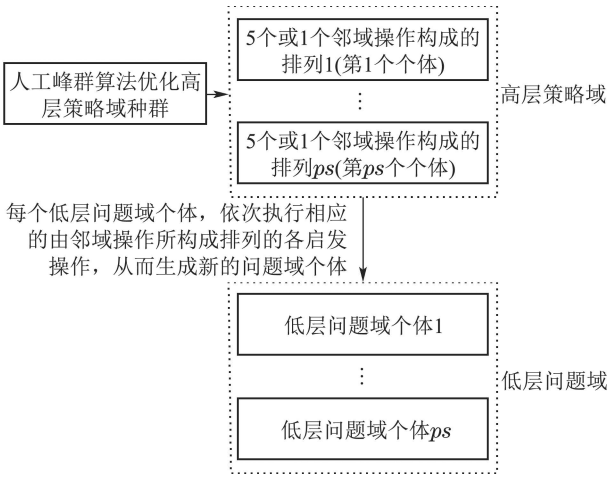

图 1 HHABC的两层结构示意图

Fig. 1 Illustrations of two-layer structure of HHABC

图 2 高层策略域个体示意图

Fig. 2 Illustrations of high-level's individual

对于低层问题域, 每个个体就是原问题的一个可行解. 解的表示是优化算法的重要问题, FSP解的编码通常为n个作业的排列, 然而在MSRDPFSP中, 为了方便高层策略域个体控制低层问题域个体进行变邻域局部搜索, 本文采用二维数组表示一个低层问题域个体, 该数组由F行组成, 每一行由一部分工件序列组成, 序列表示作业在给定工厂里的加工顺序. 以$F=2, N=10$的问题为例, 低层问题域个体示意图如图3所示, 即第1个工厂里工件加工顺序为$\{2-5-4-8-9-10\}$, 第2个工厂里工件加工顺序为$\{3-1-7-6\}$.

3.2 种群初始化

本文采用随机初始化方法生成高层策略域初始种群, 由于初始解的质量对解决组合优化问题有较大的

影响, 因此对于低层问题域的前$ps-1$个初始种群个体, 本文先随机生成一组工件序列再使用NEH2 (Nawaz-Enscore-Ham)规则的后3个步骤生成初始解, 对于第ps个初始种群个体, 则采用NEH2启发式规则生成初始解. 由于在MSRDPFSP中工件具有多个场景下不同的加工时间, 因此, 低层问题域在初始化种群时将随机选择一个场景下的加工时间用于初始化来避免解的分布过于集中.

F_1	2	5	4	8	9	10
F_2	3	1	7	6		

图 3 低层问题域个体示意图

Fig. 3 Illustrations of low-level's individual

3.3 低层启发式操作

本文设计了如下6种邻域操作, 作为低层启发式操作. 由于MSRDPFSP拥有多个场景, 某个场景下拥有最大或最小完工时间的工厂在其他场景下不一定也是拥有最大或最小完工时间的工厂, 因此本文所采用的邻域操作LLH1–LLH5是对随机工厂里的工件进行操作, LLH6是对关键工厂里的工件进行操作, 其中关键工厂的定义为, 通过第2.2节中的式(7)寻找属于低层问题域个体的"坏"场景子集, 若存在某个工厂在较多个"坏"场景下, 皆为拥有最大完工时间的工厂, 则将此工厂作为关键工厂; 若存在多个工厂在相同数量的"坏"场景下, 皆为拥有最大完工时间的工厂, 则随机选择其中一个工厂作为关键工厂; 若每个"坏"场景下拥有最大完工时间的工厂皆不相同, 则选择惩罚量最大场景下拥有最大完工时间的工厂作为关键工厂. LLH6 与 LLH1–LLH5相比, LLH6可以对MSRDPFSP庞大且不规则的解空间进行较深入的搜索, 但时间复杂度较高, 因此在满足侦查蜂阶段条件时才会在此阶段进行调用. LLH1–LLH6相关描述如下:

1) LLH1: 同一工厂内交叉操作. 随机选择一个工厂, 再从该工厂内随机选择两个工件进行交换.

2) LLH2: 同一工厂内插入操作. 随机选择一个工厂, 再从该工厂内随机选择选择两个工件, 并将工件2插入到工件1之前.

3) LLH3: 不同工厂间交叉操作. 随机选择两个工厂, 再分别从两个工厂内选择一个工件进行交换.

4) LLH4: 不同工厂间插入操作. 随机选择两个工厂, 再分别从两个工厂内随机选择两个工件, 并将工件2插入到工件1之前.

5) LLH5: 同一工厂内逆序操作. 随机选择一个工厂, 再从该工厂内工件加工序列中随机选择两个位置, 并将两个位置之间的工件逆序排列.

6) LLH6: 寻找低层问题域个体的关键工厂, 并将关键工厂中的工件依次插入到所有工厂的全部可能位置直至搜索到更优质的低层问题域个体.

3.4 人工蜂群算法

本文采用人工蜂群算法对高层策略域种群进行优化, 在ABC算法过程中3种人工蜜蜂即雇佣蜂(employed bees)、跟随蜂(onlooker bees)和侦察蜂(scout bees)将围绕食物源(food source)即启发式操作排列顺序的邻域结构依次进行搜索, 并以MSRDPFSP模型中的$PT(s)$作为3个阶段判断是否更新低层问题域个体的指标, 3个搜索阶段和邻域结构将在下文进行详细描述.

3.4.1 邻域结构

结合高层策略域个体结构特点, 本文设计了4种邻域结构, 定义如下:

1) N1: 交换邻域即从高层策略域个体序列中随机选择两个位置进行交换.

2) N2: 前插邻域即从高层策略域个体序列中随机选择两个位置并将位置编号大的元素插入到位置编号小的元素之前.

3) N3: 逆序邻域即从高层策略域个体序列中随机选择两个位置并将包含所选两位及其之间的元素逆序排列.

4) N4: 序对交换邻域即从高层策略域个体序列中随机选择两个位置并将两个位置及其之间元素前后对应依次交换.

3.4.2 雇佣蜂阶段

雇佣蜂的主要任务是在食物源附近搜索更好的食物源即寻找更加有效的启发式算法. 传统的ABC算法中的雇佣蜂操作算子用于处理连续问题, 并不适用于MSRDPFSP. 本文给出一种离散的ABC算法, 雇佣蜂阶段过程如下:

步骤 1　为高层策略域种群中每个个体分配一个雇佣蜂;

步骤 2　计算高层策略域个体对应低层问题域个体s的总惩罚$PT(s)$;

步骤 3　在第3.4.1节中给出的4种邻域结构中随机选择一种邻域结构, 用于更新高层策略域个体, 通过更新后的高层策略域个体控制s进行变邻域局部搜索得到s'并计算其总惩罚$PT(s')$;

步骤 4　若$PT(s') < PT(s)$, 更新高层策略域个体与低层问题域个体, 否则保留原个体.

3.4.3 跟随蜂阶段

跟随蜂的主要任务是在更新后的种群中以概率选择的方式循环对优质个体进行进一步搜索. 采用轮盘赌注选择方式, 需要比较每个低层问题域个体的总惩罚$PT(s)$的大小, 对于大规模问题而言, 时间复杂度

较高且对于问题求解帮助较小. 因此本文采用一种较为简单的锦标赛跟随蜂选择策略, 跟随蜂阶段过程如下:

步骤 1 在更新后的低层问题域种群中随机选择两个不同个体s_1和s_2;

步骤 2 分别计算$PT(s_1)$和$PT(s_2)$并选择具有较小总惩罚的个体s为其分配跟随蜂;

步骤 3 在第3.4.1节中给出的4种邻域结构中随机选择一种邻域结构, 用于更新s对应的高层策略域个体, 通过更新后的高层策略域个体控制s进行变邻域局部搜索得到s'并计算其总惩罚$PT(s')$;

步骤 4 若$PT(s') < PT(s)$, 更新高层策略域个体与低层问题域个体, 否则保留原个体;

步骤 5 判断循环次数是否小于ps/2, 若小于则执行步骤1, 否则结束跟随蜂阶段.

3.4.4 侦察蜂阶段

侦察蜂的主要任务是在低层问题域种群迭代过程中长时间没有得到更新时采用一种有效扰动策略避免算法过早陷入局部最优. Pan等[20]分析了最优个体携带的优质解信息较多, 围绕其进行搜索有可能发现更多优质解. 因此本文设计了一种改进侦察蜂算法, 当低层问题域最优个体连续η代未更新, 意味着算法可能陷入了局部最优, 此时派出侦察蜂更新低层问题域个体种群, 侦察蜂阶段过程如下:

步骤 1 保留总惩罚量较小的前$\theta \times ps$个优秀低层问题域个体, 为后$(1-\theta) \times ps$个每个低层问题域个体分配一个侦察蜂;

步骤 2 侦察蜂直接调用第3.3节中给出的LLH6邻域操作控制低层问题域个体进行邻域搜索得到$(1-\theta) \times ps$个新低层问题域个体;

步骤 3 用新生成的$(1-\theta) \times ps$个低层问题域个体替换掉总惩罚量较大的后$(1-\theta) \times ps$个较差低层问题域个体;

步骤 4 重新生成每个低层问题域个体所对应的高层策略域个体.

3.4.5 算法流程

根据上述算法描述, 整个算法具体流程描述如下:

步骤 1 初始化高层策略域与低层问题域种群, 种群容量为ps;

步骤 2 根据场景阈值T筛选“坏”场景子集并根据第2.2节中的式(8)评价低层问题域种群中每个个体.

步骤 3 派出雇佣蜂更新高层策略域种群, 将高层策略域个体中每一位所对应的启发式操作依次对应更新对应低层问题域个体, 每当执行完一个低层启发式操作, 就将生成的新解与旧解进行比较, 若新解的总惩罚小于旧解的总惩罚则用新解代替旧解, 并继续执行下一个低层启发式操作, 否则保留旧解, 并继续执行下一个低层启发式操作. 当所有低层启发式操作完成后, 更新低层问题域个体, 该高层个体的适应值即为其对应更新的低层问题域个体的适应值.

步骤 4 采用锦标赛策略选择优质低层问题域个体, 并派出跟随蜂更新其对应高层策略域个体, 通过更新后高层策略域个体控制低层启发式操作更新低层问题域个体.

步骤 5 判断最优个体连续未更新迭代次数是否大于η, 若大于则派出侦察蜂更新低层问题域个体与高层策略域个体, 否则执行步骤6.

步骤 6 判断是否满足终止条件. 若不满足则跳转至步骤3, 否则终止循环并输出当前最优个体.

算法流程图如图4所示.

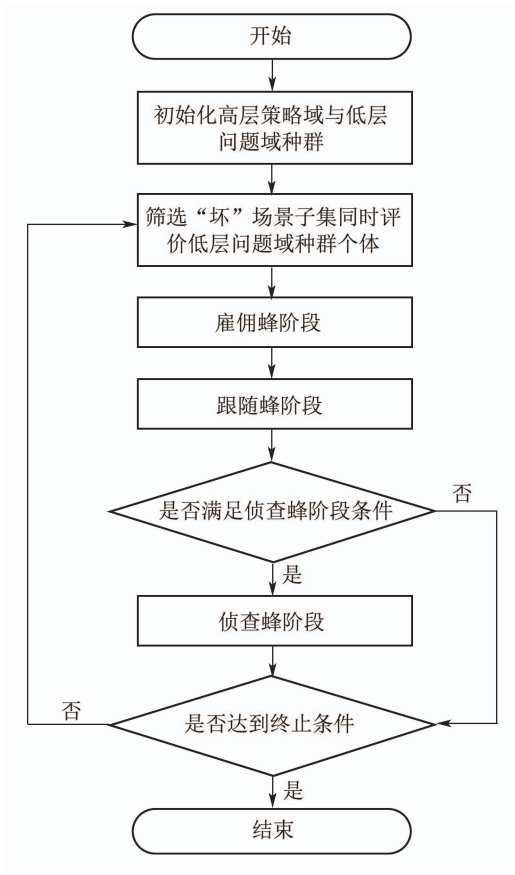

图 4 HHABC流程图

Fig. 4 The flow chart of HHABC

4 实验设计与分析

由于目前没有适合MSRDPFSP的标准算例, 本文所有测试算例均在由Naderi等[2]为解决DPFSP所提供的测试算例的基础上生成. 本文共有27个测试算例, 其中: $F = \{2, 4, 6\}$, $N = \{20, 50, 100\}$, $M = \{5, 10, 20\}$, $|\Lambda| = 20$按照$N \times M \times F$的形式组合. 每种

算法对每个测试算例均在相同时间下独立运行21次. 每次输出的结果为当前最优解的“坏”场景子集的总惩罚量, 性能指标为“坏”场景子集的总惩罚量的相关数学指标WST, BST, AVG. 其中: WST为算法独立运行21次输出结果的最差值, BST为算法独立运行21次输出结果的最优值, AVG为算法独立运行21次输出结果的平均值.

4.1　参数设置

MSRDPFSP的关键参数为种群规模ps, 当前最优解迭代未更新次数η, 优秀个体占比θ. 本节将针对一组中等规模问题($100 \times 10 \times 2 \times 20$)采用实验设计方法(design of experiment, DOE)[21]进行实验分析, 进而确定MSRDPFSP的参数设置. 表3给出了3个关键参数所对应的4个水平, 并建立规模为$L16(4^3)$的正交实验表, 每种参数组合下的测试问题均进行21次独立实验. 每个参数组合下的运行时间为$20 \times N \times M \times F \times |\Lambda|$ ms.

表 3　关键参数水平表

Table 3　Key parameters and levels

参数	水平			
	1	2	3	4
ps	60	80	100	120
η	20	30	40	50
θ	0.2	0.3	0.4	0.5

实验结果由如表4和表5所示, 同时由表5可得到图5所示的各参数响应趋势曲线. 由图5可知, 各参数不同的取值对算法性能有较大的影响, 具体而言, η取值过小可能会导致种群积累的优质信息丢失, 过大则不利于算法跳出局部最优; ps取值过大会导致算法在相同时间内迭代次数减少而难以发现真正的优质解区域; θ取值过大会导致算法搜索方向过于依赖优质解中的信息, 过小则会导致算法搜索效率降低. 基于实验结果及对其分析, 将HHABC的参数设置为$ps = 60$, $\eta = 30$, $\theta = 0.4$时, 算法性能较好.

4.2　仿真结果与比较

对于不同规模问题, 设定每种算法的运行时间均为$20 \times N \times M \times F \times |\Lambda|$毫秒. 所有规模问题均进行21次独立实验. 实验有WST, BST, AVG 3个指标, 每个问题对应最优结果用粗体表示.

4.2.1　验证高层策略的有效性

为验证MSRDPFSP中选择ABC算法作为高层策略的有效性, 将采用分布式估计算法(estimation of distribution algorithm, EDA)作为高层策略的超启发式分布估计算法 (hyper-heuristic estimation of distribution algorithm, HHEDA)和采用遗传算法(genetic algorithm, GA)作为高层策略的超启发式遗传算法(hyper-heuristic genetic algorithm, HHGA)与HHABC算法进行对比. 为保证实验公平性, 这3种算法进行对比时除高层策略不同外, 其他部分均相同, 对比结果如表6所示.

表 4　正交表和AVG统计值

Table 4　Orthogonal table of parameters settings

参数组合	参数水平			AVG
	ps	η	θ	
1	1	1	1	378.460
2	1	2	2	353.483
3	1	3	3	369.575
4	1	4	4	380.050
5	2	1	2	373.413
6	2	2	1	369.065
7	2	3	4	390.464
8	2	4	3	359.597
9	3	1	3	375.934
10	3	2	4	368.187
11	3	3	1	394.465
12	3	4	2	384.393
13	4	1	4	389.315
14	4	2	3	378.070
15	4	3	2	395.266
16	4	4	1	388.872

表 5　各参数响应值

Table 5　Average response values of each parameter

水平	参数		
	ps	η	θ
1	370.392	379.280	382.716
2	373.135	367.013	376.714
3	380.745	387.443	370.794
4	387.881	378.228	382.004
极差	17.489	20.43	11.922
影响力排名	2	1	3

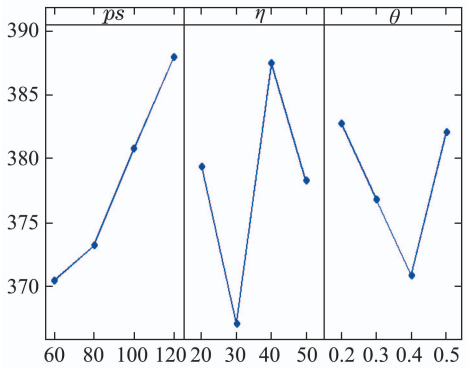

图 5　各参数响应趋势

Fig. 5　The influence trend of parameters

表 6 验证HHABC高层策略的有效性

Table 6 ABC's effectiveness verification

问题规模	HHGA			HHEDA			HHABC		
	WST	BST	AVG	WST	BST	AVG	WST	BST	AVG
20 × 5 × 2	48.6	46.6	47.8	47.3	44.8	45.2	**45.7**	**43.5**	**44.3**
20 × 10 × 2	178.2	172.1	174.1	174.4	165.2	168.3	**167.3**	**162.6**	**164.4**
20 × 20 × 2	88.3	85.6	87.0	89.6	**84.6**	86.2	**87.7**	85.3	**86.1**
50 × 5 × 2	143.6	100.9	123.3	135.1	103.6	118.4	**127.4**	**90.5**	**104.5**
50 × 10 × 2	249.8	238.8	242.0	253.6	237.7	240.9	**247.8**	**232.8**	**236.8**
50 × 20 × 2	220.4	183.1	210.3	207.1	199.1	195.0	**204.1**	**178.7**	**192.9**
100 × 5 × 2	651.6	562.8	578.3	557.5	490.4	532.8	**405.4**	**386.7**	**389.6**
100 × 10 × 2	721.2	532.5	641.6	643.5	346.2	514.2	**378.5**	**343.2**	**359.3**
100 × 20 × 2	1086.7	925.0	947.4	954.3	871.7	905.2	**736.3**	**665.7**	**693.8**
20 × 5 × 4	134.2	91.8	116.4	95.9	84.3	88.7	**80.9**	**72.5**	**75.4**
20 × 10 × 4	370.7	241.1	292.5	352.0	209.6	309.6	**273.1**	**237.3**	**257.5**
20 × 20 × 4	305.7	231.3	287.4	283.9	191.3	259.6	**192.3**	**187.3**	**190.7**
50 × 5 × 4	663.5	414.2	533.4	522.7	**296.2**	457.1	**408.4**	343.6	**368.5**
50 × 10 × 4	1259.6	966.7	1134.1	1170.3	875.1	1034.5	**882.0**	**816.7**	**855.4**
50 × 20 × 4	1321.3	558.6	719.7	1033.2	734.8	896.3	**613.4**	**547.9**	**587.1**
100 × 5 × 4	2761.8	1346.3	1867.4	1523.5	1298.6	1400.2	**1363.5**	**1108.3**	**1226.3**
100 × 10 × 4	1792.3	1321.4	1516.1	**1492.7**	1279.5	1410.3	1536.5	**1193.3**	**1343.7**
100 × 20 × 4	2084.4	1364.3	1630.5	1855.6	1169.3	1549.0	**1273.8**	**984.5**	**1099.3**
20 × 5 × 6	178.1	98.0	122.8	158.3	**58.0**	105.3	**80.6**	67.3	**72.7**
20 × 10 × 6	288.5	214.9	198.1	**147.5**	141.9	143.3	151.5	**136.1**	**146.8**
20 × 20 × 6	376.1	256.1	306.2	**172.1**	**120.6**	**131.7**	185.3	149.1	164.5
50 × 5 × 6	177.4	154.2	163.5	154.8	119.5	148.9	**132.9**	**113.2**	**128.3**
50 × 10 × 6	781.4	477.9	653.7	679.6	562.1	584.9	**545.2**	**347.5**	**404.7**
50 × 20 × 6	891.2	549.0	646.6	**480.2**	425.0	473.8	485.3	**348.5**	**426.3**
100 × 5 × 6	1229.2	650.7	983.3	983.9	457.1	719.8	**577.3**	**412.9**	**483.8**
100 × 10 × 6	3793.6	1992.9	2782.5	2844.7	1851.2	2082.5	**1276.0**	**1064.4**	1135.6
100 × 20 × 6	1960.6	1186.5	1806.4	1813.6	1133.3	1510.2	**1556.5**	**1123.7**	**1367.4**
Average	879.6	554.2	697.1	697.3	487.1	596.7	**519.1**	**437.5**	**466.9**

通过对HHABC与HHEDA, HHGA在不同算例下的测试结果进行方差分析(analysis of variance, ANOVA)以更为直观的形式呈现出HHABC与HHEDA, HHGA之间存在的性能差异. 图6显示出3种算法的均值变化线及95%置信度下的Tukey's HSD检验的置信区间.

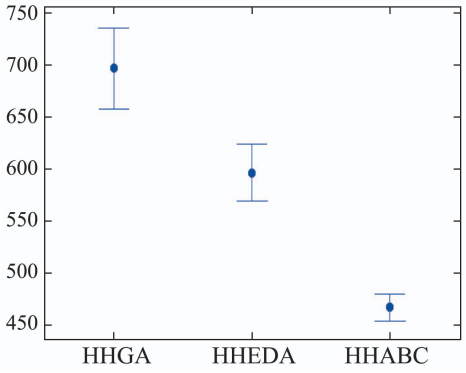

图 6 HHABC与HHEDA, HHGA的方差分析图

Fig. 6 Analysis of variance chart between HHABC and HHEDA, HHGA

由表6和图6可知HHABC在上述实验中表现出比HHEDA与HHGA更为优异的性能. ABC算法中的雇佣蜂阶段通过4种领域结构生成了多种搜索策略, 跟随蜂阶段则对优质个体进行更深一步搜索, 相较于GA作为高层策略, 可以在一定程度避免GA这类传统进化方法中普遍存在的搜索深度有限与对较优解模式破坏的问题; 相较于EDA作为高层策略, ABC算法中的侦察蜂阶段通过对部分较优进行操作, 结合了保优与重构机制, 使算法更容易跳出局部最优. 因此, ABC算法是作为HHA高层策略的有效算法.

4.2.2 验证本文算法的有效性

为证验证HHABC的有效性, 将HHABC与SS[2], BSIG[3], CRO[4], MDDE[5], BR–ILS[6] 5种算法进行对比. 这5种算法是近年来用于求解分布式置换流水车间相关调度问题的有效算法. 由于目前尚无学者对MSRDPFSP的进行研究, 因此在进行对比试验时, 对上述3种算法进行适当修改使其适用于MSRDPFSP. 为保证实验公平性, 对比算法所采用的参数取值与来源参考文献保持一致. 对比结果如表7所示.

表 7 HHABC与CRO, BSIG, SS, MDDE, BR–ILS的对比结果

Table 7 Comparison results of HHABC, CRO, BSIG, SS, MDDE, BR–ILS

问题规模	CRO			BSIG			SS			HHABC			MDDE			BR–ILS		
	WST	BST	AVG	WST	BST	AVG	WST	BST	AVG	WST	BST	AVG	WST	BST	AVG	WST	BST	AVG
20 × 5 × 2	147.3	53.3	72.4	73.2	53.3	69.2	123.3	**36.3**	70.7	**45.7**	43.5	**44.3**	77.4	57.2	63.5	69.4	52.7	65.2
20 × 10 × 2	304.4	251.4	277.0	287.4	183.4	213.5	237.4	198.4	212.8	**167.3**	**162.6**	**164.4**	253.6	170.1	253.6	236.7	168.9	189.5
20 × 20 × 2	138.2	101.6	121.9	104.6	92.6	98.5	126.9	101.6	113.7	**87.7**	**85.3**	**86.1**	120.4	96.7	106.3	136.3	93.7	114.5
50 × 5 × 2	**123.5**	114.8	117.2	197.8	123.3	174.6	166.8	120.8	135.7	127.4	90.5	**104.5**	233.7	**78.0**	124.8	156.3	127.8	137.8
50 × 10 × 2	535.7	317.8	424.1	340.8	**202.5**	275.3	357.8	244.8	319.2	**247.8**	232.8	**236.8**	345.2	259.6	280.4	374.2	218.3	287.8
50 × 20 × 2	327.1	215.3	279.6	249.1	188.1	213.5	422.1	217.1	263.4	**204.1**	178.7	**192.9**	271.9	**162.4**	205.8	258.4	195.7	213.7
100 × 5 × 2	952.0	490.8	652.9	735.9	492.4	545.9	745.3	407.2	557.1	**405.4**	386.7	**389.6**	547.6	392.6	459.8	693.7	**357.7**	493.6
100 × 10 × 2	751.3	484.5	577.5	711.6	372.2	518.2	1368.6	481.2	799.2	**378.5**	**343.2**	**359.3**	672.9	483.2	542.7	737.4	417.6	578.6
100 × 20 × 2	1257.7	929.5	1029.6	547.6	**335.1**	**419.0**	1096.5	733.5	875.4	736.3	665.7	693.8	**529.4**	473.6	492.4	648.5	467.2	524.6
20 × 5 × 4	118.4	**63.8**	92.7	143.4	84.3	115.2	171.9	73.8	123.1	**80.9**	72.5	**75.4**	162.7	78.2	121.7	96.4	74.0	86.3
20 × 10 × 4	514.2	325.0	476.5	362.6	256.6	304.4	335.2	**229.7**	271.9	**273.1**	237.3	**257.5**	378.2	253.9	292.4	415.8	243.1	368.9
20 × 20 × 4	289.4	202.3	221.6	241.8	215.3	225.1	244.3	194.3	202.4	**192.3**	**187.3**	**190.7**	231.6	193.5	217.6	269.4	216.9	234.5
50 × 5 × 4	515.5	385.6	392.2	597.5	384.6	450.1	544.1	416.1	447.7	**408.4**	**343.6**	**368.5**	472.8	411.0	446.9	538.2	428.6	457.6
50 × 10 × 4	1494.3	919.8	1105.7	1952.3	1268.8	1560.4	1528.9	1200.8	1310.2	**882.0**	**816.7**	**855.4**	1064.6	835.4	893.7	1279.8	879.4	1064.2
50 × 20 × 4	1965.7	1483.9	1625.7	1271.6	732.7	985.3	1096.0	720.6	810.2	**613.4**	547.9	**587.1**	673.8	579.2	623.5	975.2	**508.3**	715.7
100 × 5 × 4	2366.6	1410.2	1838.6	1813.3	1281.3	1457.1	1512.9	1120.8	1335.4	**1363.5**	1108.3	**1226.3**	1762.1	1293.7	1499.4	1715.8	**1058.4**	1374.9
100 × 10 × 4	3532.9	1729.2	2067.2	2475.8	1252.1	1497.6	4076.2	2737.2	3483.5	1536.5	**1193.3**	**1343.7**	**1479.6**	1264.8	1398.7	1973.9	1367.8	1542.7
100 × 20 × 4	1950.3	1443.6	1613.5	1654.2	1348.9	1467.6	2254.8	**712.4**	1861.5	**1273.8**	984.5	**1099.3**	1576.4	1167.2	1249.2	1467.4	978.2	1182.6
20 × 5 × 6	270.7	155.0	203.7	298.5	**59.0**	132.4	281.7	84.0	127.0	**80.6**	67.3	**72.7**	189.4	127.5	152.3	137.5	97.2	107.1
20 × 10 × 6	322.5	215.3	240.7	562.5	149.5	352.2	249.7	145.3	199.6	**151.5**	**136.1**	**146.8**	275.8	148.5	193.4	259.8	143.5	213.7
20 × 20 × 6	274.1	166.1	207.1	325.5	**108.1**	219.3	261.8	169.1	207.1	**185.3**	149.1	**164.5**	273.4	157.3	219.4	209.4	137.5	174.8
50 × 5 × 6	472.9	213.7	404.6	295.2	125.0	197.2	237.5	127.7	190.6	**132.9**	113.2	**128.3**	184.3	**109.2**	152.2	187.1	123.9	159.6
50 × 10 × 6	938.9	504.7	696.2	800.6	657.6	727.4	768.4	500.7	665.3	**545.2**	**347.5**	**404.7**	632.5	492.9	538.1	734.6	553.6	637.9
50 × 20 × 6	1082.1	538.3	758.4	835.6	404.8	518.9	786.5	575.6	621.1	**485.3**	**348.5**	**426.3**	659.0	508.7	552.8	573.5	415.7	521.4
100 × 5 × 6	1771.5	1082.	1416.8	965.4	574.7	626.5	1046.5	787.3	824.8	**577.3**	**412.9**	**483.8**	744.5	595.4	673.8	758.4	569.3	623.9
100 × 10 × 6	1992.4	1237.8	1536.3	2389.2	1520.0	1771.6	2026.0	1538.9	1841.3	**1276.0**	**1064.4**	**1135.6**	1483.5	1274.5	1358.3	2179.5	1497.7	1749.3
100 × 20 × 6	1427.8	1157.9	**1234.2**	3042.8	1796.6	2543.2	1787.3	1274.2	1545.8	1556.5	**1123.7**	1367.4	**1378.4**	1187.6	1273.5	2767.8	1708.2	2193.7
Average	956.9	599.7	729.0	862.1	528.3	655.8	833.5	561.1	719.1	**519.1**	**423.8**	**466.9**	617.6	476.0	532.8	735.2	485.2	593.2

通过对 HHABC 与 SS, BSIG, CRO, MDDE, BR–ILS在不同算例下的测试结果进行方差分析(ANOVA), 进一步验证各算法性能的差异. 图7显示3种算法的均值变化线及95%置信度下的Tukey's HSD检验的置信区间.

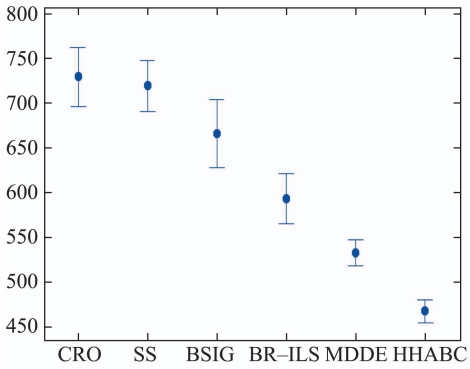

图 7 HHABC与CRO, SS, BSIG, BR–ILS, MDDE的方差分析图

Fig. 7 Analysis of variance chart of HHABC, CRO, SS, BSIG, BR–ILS, MDDE

由表7和图7可知, HHABC算法在上述实验中表现出比CRO, BSIG, SS, MDDE, BR–ILS算法更为优异的性能. 针对MSRDPFSP, 上述5种对比算法都有一定缺陷, 其中CRO算法本质上是每代采用一种基于交叉、变异等邻域操作生成新个体, 以实现对问题解空间的搜索, 这种搜索方式搜索邻域数量有限且搜索方式较为单一, 导致其搜索深度有限. BSIG算法搜索深度较高, 但其也有搜索邻域数量有限的问题, 且其时间复杂度较高在求解大规模问题时效率不高. SS算法则过于倾向于每代较优解忽视了较差解其潜在邻域导致其较容易陷入局部最优. MDDE算法同样存在过于依赖较优解信息的问题, 但其设计多种邻域结构并根据搜索反馈的结果进行变邻域搜索, 因此具有相对较好的实验结果. BR–ILS算法设计的扰动策略的时间复杂度较大且无法利用优质解中的信息, 导致其虽具备搜索到较优解的能力, 但搜索效率不高. HHABC算法高层利用ABC算法中的雇佣蜂、跟随蜂、侦察蜂3个阶段动态控制低层个体进行包含多种邻域操作组合的启发式搜索, 有效避免了采用单一搜索策略或者简单邻域操作对解空间进行搜索时所导致的搜索深度有限的问题, 此外为避免算法过早陷入局部最优, 侦察蜂调用结合保优和重构机制的特殊策略对种群进行更新, 增强了算法跳出局部最优的能力, 进而更容易发现复杂解空间的较优解. 因此, HHABC是求解MSRDPFSP的有效算法.

5 结论

本文在DPFSP的基础上, 进一步考虑实际生产过程中普遍存在的由于设备老化或故障、工人水平参差不齐等因素所导致工件加工时间不确定的DPFSP, 首次建立MSRDPFSP的模型. 该模型通过采用鲁棒调度中的场景法将MSRDPFSP划分为多个具有确定加工时间的典型问题场景, 并通过设定合适的场景阈值来动态确定不同问题解的"坏"场景子集, 进而将最小化所定义的"坏"场景子集总惩罚量作为优化目标, 以确保求解算法有可能获得在在所有场景下均表现良好的鲁棒解. 考虑到MSRDPFSP的复杂性和现有智能算法的不足, 提出一种HHABC算法进行求解. HHABC算法具有如下特点: 1) 结合MSRDPFSP特点设计的编解码规则, 有助于合理限定搜索空间; 2) 采用基于NEH规则的方法来初始化种群, 可提高初始种群的质量; 3) 算法高层在HHABC的雇佣蜂与跟随蜂阶段, 动态控制低层的6种启发式操作来持续生成新的混合启发式算法, 可对问题解空间进行较深入的搜索; 4) 算法高层在侦察蜂阶段采用特定策略以避免算法过早陷入局部最优, 有利于进一步增加算法搜索深度. 后续工作将进一步考虑面向节能降耗的不确定调度问题, 研究相关问题建模和基于HHABC的有效求解算法.

参考文献:

[1] NADERI B, RUIZ R. The distributed permutation flowshop scheduling problem. *Computers & Operations Research*, 2010, 37(4): 754 – 768.

[2] NADERI B, RUIZ R. A scatter search algorithm for the distributed permutation flowshop scheduling problem. *European Journal of Operational Research*, 2014, 239(2): 323 – 334.

[3] VICTOR F, FRAMINAN J M. A bounded-search iterated greedy algorithm for the distributed permutation flowshop scheduling problem. *International Journal of Production Research*, 2015, 53(4): 1111 – 1123.

[4] BARGAOUI H, DRISS O B, GHÉDIRA K. A novel chemical reaction optimization for the distributed permutation flowshop scheduling problem with makespan criterion. *Computers & Industrial Engineering*, 2017, 111: 239 – 250.

[5] ZHAO F Q, HU X, WANG L, et al. A memetic discrete differential evolution algorithm for the distributed permutation flow shop scheduling problem. *Complex & Intelligent Systems*, 2021, 8(1): 141 – 161.

[6] FERONE D, HATAMI S, ELIANA M, et al. A biased-randomized iterated local search for the distributed assembly permutation flowshop problem. *International Transactions in Operational Research*, 2020, 27(3): 1368 – 1391.

[7] MURVEY J M, VANDERBEI R J, ZENIOS S A. Robust optimization of large-scale systems. *Operations Research*, 1995, 43(2): 264 – 281.

[8] KOUVELIS P, YU G. *Robust Discrete Optimization and Its Applications*. Boston: Springer, 1997.

[9] DANIELS R L, KOUVELIS P. Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science*, 1995, 41(2): 468 – 509.

[10] WANG B, WANG X Z, XIE H X. Bad-scenario-set robust scheduling for a job shop to hedge against processing time uncertainty. *International Journal of Production Research*, 2019, 57(10): 3168 – 3185.

[11] WU C C. Robust scheduling for a two-stage assembly shop with scenario-dependent processing times. *International Journal of Production Research*, 2021, 59(17): 5372 – 5387.

[12] COWLING P, KENDALL G, SOUBEIGA E. A hyper heuristic approach to scheduling a sales summit. *International Conference on the Practice and Theory of Automated Timetabling*. Berlin: Springer, 2000.

[13] LIN J, WANG Z J, LI X. A backtracking search hyper-heuristic for the distributed assembly flow-shop scheduling problem. *Swarm and Evolutionary Computation*, 2017, 36: 124 – 135.

[14] LI Shanghan, HU Rong, QIAN Bin, et al. Hyper-heuristic genetic algorithm for solving fuzzy flexible job shop scheduling problem. *Control Theory & Applications*, 2020, 37(2): 316 – 330.
(李尚函, 胡蓉, 钱斌, 等. 超启发式遗传算法求解模糊柔性作业车间调度. 控制理论与应用, 2020, 37(2): 316 – 330.)

[15] LIN J, LI Y Y, SONG H B. Semiconductor final testing scheduling using Q-learning based hyper-heuristic. *Expert Systems with Applications*, 2021, DOI: 10.1016/j.eswa.2021.115978.

[16] KARABOGA D, BASTURK B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, 2007, 39(3): 459 – 471.

[17] ZHENG Xiaocao, GONG Wenyin. An improved artificial bee colony algorithm for fuzzy flexible job-shop scheduling problem. *Control Theory & Applications*, 2020, 37(6): 1284 – 1292.
(郑小操, 龚文引. 改进人工蜂群算法求解模糊柔性作业车间调度问题. 控制理论与应用, 2020, 37(6): 1284 – 1292.)

[18] LI J Q, SONG M X, WANG L, et al. Hybrid artificial bee colony algorithm for a parallel batching distributed flow-shop problem with deteriorating jobs. *IEEE Transactions on Cybernetics*, 2020, 50(6): 2425 – 2439.

[19] GONG G L, DENG Q W, GONG X T, et al. A hybrid artificial bee colony algorithm for flexible job shop scheduling with worker flexibility. *International Journal of Production Research*, 2020, 58(14): 4406 – 4420.

[20] PAN Q K. An effective co-evolutionary artificial bee colony algorithm for steelmaking-continuous casting scheduling. *European Journal of Operational Research*, 2016, 250(3): 702 – 714.

[21] MONTGOMERY D C. *Design and Analysis of Experiments*. Hoboken: John Wiley & Sons, 2005.

作者简介:

连　戈　硕士研究生, 目前研究方向为智能算法与优化调度, E-mail: 513796249@qq.com;

朱　荣　教授, 硕士生导师, 目前研究方向为智能优化调度、物流优化, E-mail: 1270030921@qq.com;

钱　斌　教授, 博士生导师, 目前研究方向为优化调度理论与方法、智能优化方法, E-mail: bin.qian@vip.163.com;

吴绍云　工程师, 目前研究方向为智能化数字工厂构建, E-mail: hongtawsy@163.com;

胡　蓉　教授, 博士生导师, 目前研究方向为智能优化调度、物流优化, E-mail: ronghu@vip.163.com.