文章编号: 1001-0920 (2017) 01-0012-09 DOI: 10.13195/j.kzyjc.2015.1568

求解 PFSP 的双种群协同学习算法

亓祥波[1,2†], 朱云龙[1], 张丁一[1]

(1. 中国科学院 沈阳自动化研究所，沈阳 110016；2. 中国科学院大学，北京 100049)

摘 要: 在人工蜜蜂群算法的基础上, 提出一种双种群协同学习算法. 该算法根据个体适应度高低把蜜蜂群划分为两个子群, 并重新定义子群的学习交流机制. 在 10 个常用的基准测试函数上与其他 4 个常用的群体智能算法进行比较, 比较结果表明, 所提出算法的性能有明显改进. 采用双种群协同学习算法求解置换流水车间调度问题, 在一些著名的中大规模测试问题包括 21 个 Reeves 实例和 40 个 Taillard 实例上进行测试, 结果表明, 所提出的算法优于其他算法, 能有效解决置换流水车间调度问题.

关键词: 协同学习；置换流水车间调度；智能算法

中图分类号: TP301 **文献标志码:** A

Double population co-learning algorithm for permutation flow-shop scheduling problems

QI Xiang-bo[1,2†], *ZHU Yun-long*[1], *ZHANG Ding-yi*[1]

(1. Shenyang Institute of Automation，Chinese Academy of Sciences，Shenyang 110016，China；2. University of Chinese Academy of Sciences，Beijing 100049，China)

Abstract: Based on the artificial bee colony(ABC) algorithm, a double population co-learning(DPCL) algorithm is proposed. A population is divided into two populations according to their fitness. The individuals of each population are updated according to the given learning rules. With a test on ten benchmark functions, the proposed DPCL algorithm is proved to have significant improvement over canonical ABC and several other comparison algorithms. The DPCL algorithm is then employed for permutation flow-shop scheduling problem(PFSP). Twenty-one Reeves instances and forty Taillard instances are used. The results show that the DPCL algorithm can obtain better results than other algorithms, and is a competitive approach for PFSP.

Keywords: co-learning；permutation flow-shop scheduling；intelligent algorithm

0 引 言

置换流水车间调度问题 (PFSP) 是经典的组合优化问题. 典型的 PFSP 问题描述如下: n 个工件在 m 台设备上加工, 每台机器上所有工件的加工顺序都相同, 任意时刻每台机器只能加工一个工件, 任意时刻一个工件只能在一台机器上进行加工. 已知每个工件在每台设备上的加工时间, 求调度序列使某项生产指标最小化. 研究表明, 3 台机器以上的 PFSP 问题属于 NP-hard 难题, 求解困难.

随着智能优化算法的发展, PFSP 逐渐受到研究者的关注. 许多研究人员都是受到大自然中生物行为的启发, 从生物系统中获得灵感, 模仿它们内在的机制获取解决复杂优化问题的方法. 比如, 粒子群算法 (PSO) 模仿了飞鸟和鱼聚群的行为[1], 改进的 PSO 常用于求解各类车间调度问题. 宋存利等提出了一种两阶段混合粒子群算法求解作业车间调度问题[2]; 刘延凤等[4]利用基于 NEH[3]的贪婪随机自适应搜索 (GRASP) 算法得到初始最优解, 对最优解进行局部搜索, 提出了一种改进粒子群算法用于求解 PFSP; Tasgetiren 等设计了一个重要的规则 SPV, 将连续变量转换为工件的排列顺序, 提出了求解 PFSP 的基于最小位置值的粒子群 (PSOspv) 算法[5-6], 取得了不错

收稿日期: 2015-12-18；**修回日期:** 2016-02-23.

基金项目: 国家杰出青年科学基金项目(61174164, 51205389)；辽宁省自然科学基金项目(2015020163).

作者简介: 亓祥波 (1981−), 男, 博士生, 从事智能优化算法及应用的研究；朱云龙 (1967−), 男, 研究员, 博士生导师, 从事 CIMS、分布式智能技术等研究.

†通信作者. E-mail: qixb@sia.cn

的效果. PSO 在参数设置方面, 除了评估次数和种群大小两个参数外, 还需要设置两个学习因子和惯性权重. 然而, 在解决实际工程问题时, 多个控制参数导致很难找到适当的值满足求解质量. 人工蜜蜂群算法 (ABC) 模仿了蜜蜂觅食的行为[7], 算法需要设置的参数少, 优化效果突出, 常用于置换车间调度优化问题. Tasgetiren 等提出了一种离散版人工蜜蜂群算法用于求解总流经时间最小的 PFSP[8], 该算法保留了 ABC 的 3 个种群, 在初始化阶段的第 1 个解是用 NEH 方法生成的[5], 除第 1 个解外的其他解是随机生成的, 在雇佣蜂阶段和跟随蜂阶段采用多种基于插入与交换的邻域搜索策略生成新解, 在跟随蜂阶段采用锦标赛选择方法选择跟随的个体, 在侦察蜂阶段通过锦标赛选择方法选择需要替换的个体, 雇佣蜂数量设置为种群大小, 跟随蜂数量设置为两倍种群大小, 侦察蜂数量设置为 0.2 倍种群大小. 晏晓辉等提出了一种基于趋化的混合人工蜜蜂群算法 (HABC) 求解 PFSP[9], 该算法将细菌的趋化行为引入到雇佣蜂阶段和跟随蜂阶段来加强其邻域搜索, 与当前较新的调度算法相比, 该算法具备较好的优化能力. 教与学算法 (HTLBO) 是近年来较新的智能算法, 文献 [10] 将变邻域搜索策略融合到教与学算法求解 PFSP, 取得了不错的结果.

本文在人工蜜蜂群算法的基础上提出一种基于双种群协同学习的算法, 按照个体适应度排序后将种群划分为两个种群, 分别为每个种群设置生成新解的策略, 高适应度种群自学习, 低适应度种群向高适应度种群学习, 从而实现高适应度种群引导优化的“精英”指导策略. 在 10 个常用的单峰和多峰基准测试函数上与其他 4 个常用的群体智能算法比较, 所提出的算法在多数函数上无论是收敛速度还是求解精度都比原始的人工蜜蜂群算法有明显的改进. 采用该算法求解最小化最大完工时间的 PFSP, 并利用变邻域搜索作为改进策略进行局部开发以提高局部解的质量. 为了验证该算法的优化能力, 将其在 21 个 Reeves 实例和 40 个 Taillard 实例上进行测试, 并与较新的智能算法进行比较, 结果验证了所提出算法在解决 PFSP 问题上的有效性和优越性.

1　双种群协同学习算法

在原始 ABC 算法中, 生成新食物源的蜜蜂是随机选择的. 然而, 蜂群从开始优化到当前, 搜索得到的所有食物源的适应度都是已知的, 特别是适应度高的食物源具有最好的基因结构和优良特性, 对全局搜索更具影响力. 所以, 在后续优化过程中, 用高适应度的个体指导优化过程对全局收敛具有重要的作用.

鉴于上述分析, 对原始的 ABC 算法进行改进, 形成双种群协同学习算法 (DPCL), 将人工蜜蜂群按照它们的适应度高低分为两组, 具有较好适应度的人工蜜蜂构成雇佣蜂, 专门负责新食物源的开发, 雇佣蜂中的个体向除了自己之外的其他雇佣蜂个体学习. 具有较差适应度的人工蜜蜂构成跟随蜂, 跟随蜂中的个体也是向雇佣蜂中的个体学习, 专门负责新食物源的探索. 高适应度种群主要的角色是开发较优个体的区域, 而低适应度种群主要的角色是向高适应度种群中的个体学习并探索新的区域; 雇佣蜂和跟随蜂两者协作实现高适应度种群引导优化过程.

与 ABC 算法一致, 在 DPCL 算法中, 雇佣蜂和跟随蜂的维度学习策略不变, 雇佣蜂和跟随蜂都是按下式生成新食物源:

$$v_{i,j} = x_{i,j} + \varphi_{i,j}(x_{i,j} - x_{k,j}). \tag{1}$$

其中: $i \in [1, 2, \cdots, \mathrm{SN}]$, SN 是食物源的数量, 即蜜蜂群数量的一半; $j \in [1, 2, \cdots, D]$ 是随机选择的维度, D 是要解决问题的维度大小; $k \in [1, \mathrm{SN}]$ 是随机选择的食物源, $k \neq i$; $\varphi_{i,j}$ 是 [−1, 1] 之间的随机数. 在 DPCL 算法中, 式 (1) 中的 k 都是从雇佣蜂中随机选择的.

DPCL 算法仅保留 ABC 算法的雇佣蜂和跟随蜂, 但 DPCL 算法中的雇佣蜂数量与跟随蜂数量不是相等的, 为了保证高适应度个体发挥作用, 雇佣蜂占蜂群的 60%; 而在 ABC 算法中, 雇佣蜂数量与跟随蜂数量是相等的, 各占蜂群的 50%. DPCL 算法中没有侦察蜂阶段. 但是, 在跟随蜂生成的新食物源的适应度不高于原来食物源的适应度时, 算法按照下式生成新食物源:

$$v_{i,j} = \delta \cdot v_{i,j} + (x_j^{\max} - x_j^{\min}) \cdot \mathrm{step} \cdot \mathrm{rand}(), \tag{2}$$

$$\delta = \frac{\Delta_p}{\sqrt{\Delta_p^{\mathrm{T}} \Delta_p}}. \tag{3}$$

其中: $i \in [1, 2, \cdots, \mathrm{SN}]$, $j \in [1, 2, \cdots, D]$, step 是步长, Δ 是随机生成的 D 维向量, p 是 [1, D] 之间随机数. 式 (3) 是在当前食物源的基础上随机改变一个方向, 式 (2) 是在当前食物源改变方向的基础上随机移动一定步长. 式 (2) 和 (3) 的设计是为了让算法跳出局部最优解, 提高算法搜索的广度. DPCL 算法伪代码如下.

1) 初始化食物源.

　　计算食物源的适应度.

2) repeat.

3) 按照适应度对食物源排序.

4) 选择 60% 适应度较好的蜜蜂构成雇佣蜂, 余下的构成跟随蜂.

5) 雇佣蜂阶段.

遍历每个雇佣蜂.

随机选择一个雇佣蜂 k;

按照式 (1) 生成一个新食物源 v_i;

计算新食物源的适应度;

对新食物源和原食物源采用贪婪选择;

结束遍历.

6) 雇佣蜂阶段.

遍历每个跟随蜂.

随机选择一个雇佣蜂 k;

按照式 (1) 生成一个新食物源 v_i;

计算新食物源的适应度;

If 新食物源适应度好于原食物源适应度,

对新食物源和原食物源采用贪婪选择;

else

按式 (2) 生成新食物源替换原食物源.

End if

结束遍历.

7) until 终止条件.

2 实验与结果分析

为了测试DPCL算法的性能, 选择一组常用标准测试函数, 即 sphere(f_1), powers(f_2), sumsquares(f_3), rosenbrock(f_4), quadric(f_5), zakharov(f_6), dixon & price (f_7), schwefel(f_8), rastrigrin(f_9), griewank(f_{10}). 这 10 个函数常用作基准测试[11-12], 其中前 7 个函数是单峰连续函数, 后 3 个函数是多峰连续函数. 将 DPCL 算法与 ABC、PSO、CPSO[13]和遗传算法 (GA) 进行比较.

2.1 参数设置

在实验中, 所有算法都在 30 维的函数上测试, 每种算法运行 30 次, 种群大小设置为 40. 在 ABC 算法中, 雇佣蜂和跟随蜂数量都是种群大小的 50%, 雇佣蜂适应度没有改进的数量阈值设置为 100, 侦察蜂数量设置为 1. 在 CPSO 和 PSO 算法中, 惯性权重随着迭代次数由 0.9 线性下降至 0.4, 学习因子 C_1 和 C_2 都设置为 2; 在 GA 算法中, 交叉概率设置为 0.9, 变异概率设置为 0.1; 在 DPCL 算法中, 雇佣蜂数量设置为种群大小的 60%, 跟随蜂数量设置为种群大小的 40%, step 随着迭代次数由 0.1 线性下降至 0.000 1. 算法终止条件采用函数评估次数, 所有算法在评估次数为 100 000 次时终止, 所有算法采用 Matlab R2010a 实现. 计算机配置为 Intel core i5-2450M CPU, 2.5 GHz, 2 GB 内存, 操作系统是 Windows 7.

2.2 实验结果及分析

实验结果包括算法独立运行 30 次后函数的平均值和方差, DPCL、ABC、PSO、CPSO 和 GA 的实验数据如表 1 所示, 每个函数上的最好结果加粗显示. 算法的收敛情况如图 1 ~ 图 10 所示.

表 1 DPCL、ABC、PSO、CPSO 和 GA 在测试函数上的优化

Function		DPCL	ABC	PSO	CPSO	GA
f_1	Mean	**6.568 852e-016**	7.213 468e-016	6.506 300e-005	1.607 142e-006	1.556 852e+000
	Std	**9.533 839e-017**	1.363 389e-016	5.441 858e-005	1.011 650e-006	6.419 853e-001
f_2	Mean	**8.868 602e-018**	3.138 795e-014	1.137 226e-016	7.702 572e-010	7.136 000e-003
	Std	**7.192 063e-018**	3.847 084e-014	3.513 930e-016	1.849 204e-009	4.647 327e-003
f_3	Mean	9.349 789e-016	**8.112 22e-016**	1.051 227e-002	9.401 491e-005	5.218 730e+001
	Std	1.576 856e-016	**1.456 30e-016**	7.267 093e-003	5.250 847e-005	2.152 344e+001
f_4	Mean	**1.076 027e-005**	2.188 909e-001	4.222 103e+001	1.390 806e+001	1.371 071e+003
	Std	**1.149 772e-005**	3.584 574e-001	3.055 575e+001	2.518 993e+001	7.428 071e+002
f_5	Mean	**2.425 821e-005**	6.452 912e+001	6.552 226e-002	2.533 629e+001	2.517 306e+002
	Std	**2.464 051e-005**	1.268 983e+001	2.862 387e-002	3.319 334e+001	4.967 288e+001
f_6	Mean	**5.714 000e-006**	2.332 798e+002	4.935 099e+001	2.358 768e+002	4.282 034e+002
	Std	**8.368 011e-006**	3.522 437e+001	5.642 599e+001	7.456 155e+001	9.275 371e+001
f_7	Mean	1.813 071e-003	**3.594 541e-005**	7.657 381e-001	4.162 826e-003	7.284 596e+001
	Std	1.176 627e-003	**5.832 195e-005**	1.157 811e-001	2.432 101e-003	4.250 173e+001
f_8	Mean	**3.818 270e-004**	4.004 110e+000	6.125 663e+003	1.184 391e+002	1.564 399e+003
	Std	**2.756 842e-019**	2.161 507e+001	7.755 152e+002	1.319 606e+002	6.160 345e+002
f_9	Mean	**8.337 035e-014**	5.178 038e-008	6.608 378e+001	1.622 153e-003	1.655 100e+002
	Std	**2.884 328e-014**	2.783 074e-007	1.700 460e+001	1.751 115e-003	3.851 081e+001
f_{10}	Mean	**5.565 822e-012**	2.169 533e-007	1.027 076e-001	2.583 422e-002	5.555 759e+000
	Std	**9.334 256e-012**	8.130 993e-007	4.221 921e-002	3.168 992e-002	1.610 693e+000

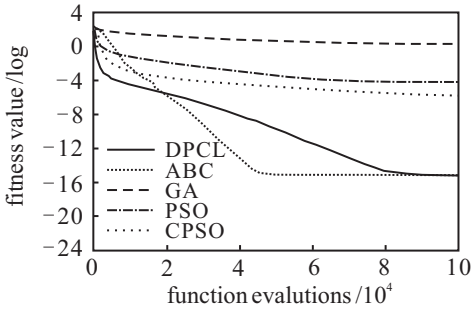

图 1　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (sphere(f_1))

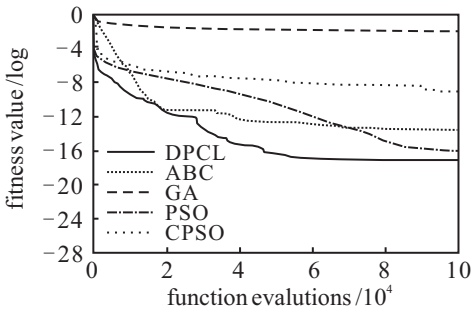

图 2　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (powers(f_2))

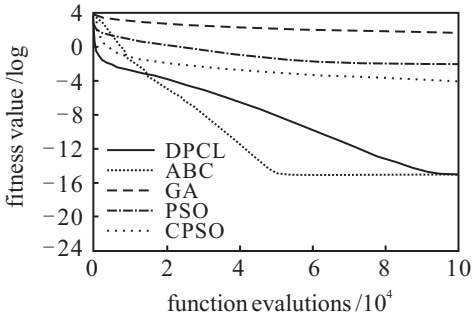

图 3　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (sumsquares (f_3))

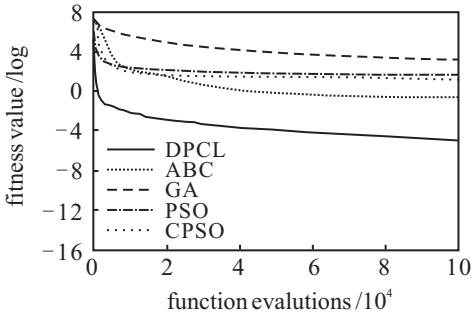

图 4　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (rosenbrock(f_4))

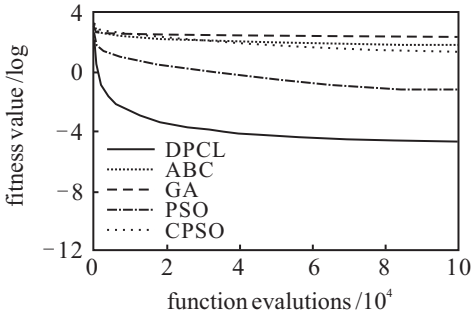

图 5　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (quadric(f_5))

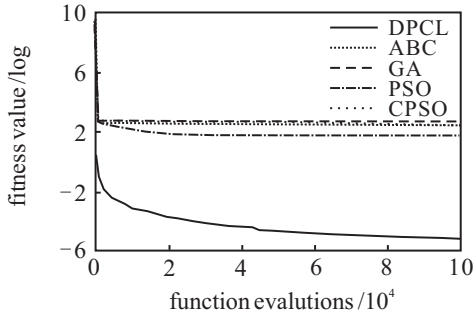

图 6　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (zakharov(f_6))

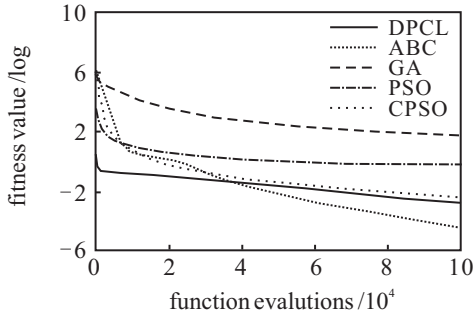

图 7　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (dixon & price(f_7))

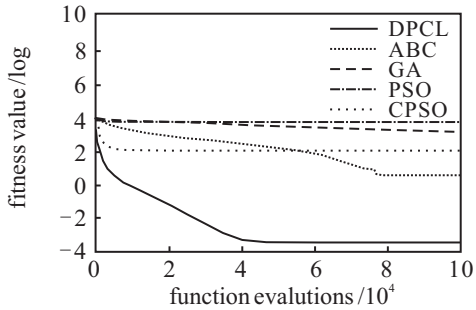

图 8　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (schwefel(f_8))

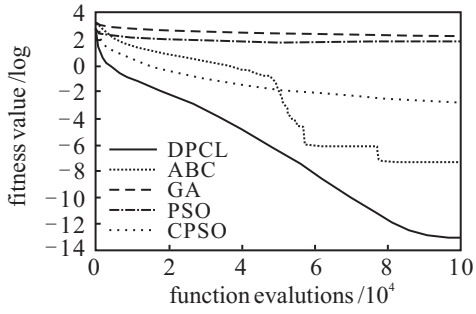

图 9　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (rastrigrin(f_9))

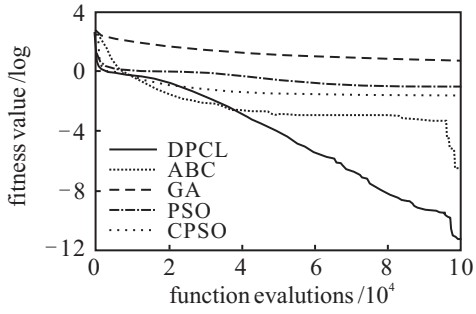

图 10　DPCL、ABC、GA、PSO 和 CPSO 的平均收敛 (griewank(f_{10}))

在sphere函数上, DPCL和ABC取得了较好的结果, DPCL取得了最好的结果, ABC稍逊一点, 在函数评估次数大约为50 000次时, ABC陷入了局部最优, CPSO和PSO取得的结果比DPCL低10～11个数量级, GA表现最差.

在powers函数上, 除了GA外, 所有算法都取得了不错的结果. 从图2可以看出, DPCL在函数评估次数为70 000时陷入局部最优, 但是已经取得了最好的结果.

在sumsquares函数上, DPCL和ABC都取得了不错的结果, DPCL算法稍逊于ABC, 但是与ABC为同样的数量级别, GA、PSO和CPSO表现不好, GA最差.

在rosenbrock函数上, 除了DPCL外, 所有算法都表现不好, GA、PSO和CPSO在开始阶段收敛速度快, 但是很快陷入局部最优, ABC在函数评估次数到达大约50 000次时陷入局部最优.

在quadric函数上, DPCL性能最好, 取得的结果比排名第2的PSO高3个数量级, 其他3种算法表现不好.

在zakharov函数上, DPCL性能表现最好, ABC、PSO、CPSO和GA取得的结果比DPCL的结果差.

在dixon & price函数上, ABC表现最好, DPCL的性能与CPSO很相似, 不过CPSO稍逊于DPCL, GA和PSO表现不好.

在schwefel函数上, DPCL开始收敛很快, 在函数评估次数到达40 000次左右时陷入局部最优, 但DPCL取得了最好的结果, 其他4种算法表现不佳.

在rastrigrin函数上, DPCL表现很好, 持续对解进行改善最终取得满意结果, 比排名第2的ABC高6个数量级, ABC也取得了可接受的结果, 但是在函数评估次数约为60 000次左右时陷入局部最优, PSO、CPSO和GA在该函数上表现不佳.

在griewank函数上, DPCL的表现非常类似于在rastrigin函数上的表现, 持续对解进行改善, 最终取得了最好的结果, 比排名第2的ABC高5个数量级, 其他4种算法表现不好.

由上述分析可知, DPCL算法在多数测试函数上都取得了比其他4种算法更好的结果, 也说明了该算法在数值优化问题上是一种很有效的算法. 与原始ABC算法比较, 无论在求解精度方面还是在收敛速度方面都有明显提高, 取得该成果的主要原因是对ABC算法进行了改进: 一是设置了明确的"精英"指导搜索策略, 将种群按照适应度高低划分为两个种群, 用高适应度种群引导整个优化过程, 该策略在一定程度上加强了算法的开采能力; 二是在跟随蜂的适应度没有得到改善时采用式(2)产生新解, 使得算法的搜索范围加大, 在一定程度上加强了算法的探索能力. 在sphere、sumsquares和dixon & price这3个测试函数上, DPCL开始收敛很快, 在一定评估次数后没有ABC收敛速度快是因为DPCL为了提高搜索的广度, 在跟随蜂的适应度没有改善时随机产生新解, 放慢了收敛速度, 但是从它们的收敛图上可以看出其收敛趋势很好, 随着迭代次数的增加, 最终也可以找到理想的解.

3 DPCL算法求解PFSP问题

3.1 PFSP问题描述

假设$\pi=(\pi_1,\pi_2,\cdots,\pi_n)$为PFSP问题的一个调度序列, $P(\pi_i,j)\,(i=1,2,\cdots,n,j=1,2,\cdots,m)$为工件$\pi_i$在机器$j$上的加工时间, $C(\pi_i,j)$为工件π_i在机器j上的完工时间. PFSP问题的完工时间计算过程如下:

$$C(\pi_1,1)=P(\pi_1,1);$$

$$C(\pi_i,1)=C(\pi_{i-1},1)+P(\pi_i,1),i=2,3,\cdots,n;$$

$$C(\pi_1,j)=C(\pi_1,j-1)+P(\pi_1,j),j=2,3,\cdots,m;$$

$$C(\pi_i,j)=\max\{C(\pi_{i-1},j),C(\pi_i,j-1)\}+P(\pi_i,j),$$
$$i=2,3,\cdots,n,\ j=2,3,\cdots,m.$$

目标就是找到一个调度序列π^*, 使得

$$C_{\max}(\pi^*)\leqslant C(\pi_n,m).$$

3.2 求解PFSP问题的DPCL算法流程

DPCL算法在连续的搜索空间有很好的优化效果, 对于离散的搜索空间不能直接使用, 需要对其进行改进. 给出算法流程前先给出PFSP问题的个体表达方式. 对于基于种群的算法, 必须对个体编码进行设计以表达调度问题的解. 由于PFSP问题中所有机器上工件的加工顺序相同, 解为被加工工件的一个排列, 对于工件数为n的PFSP问题, 本文采用直接编码, 解的维度与工件数相同, 可用一个n维向量表示. 每个维度上的值直接代表工件序号, 如对于6个工件的PFSP问题, 个体[6, 5, 3, 2, 4, 1]代表其加工顺序, 即为工件6, 5, 3, 2, 4, 1. 这种解的编码直接简单, 每个个体出现且仅出现一次, 否则将为无效解.

算法流程如下:

初始化种群, 计算种群中所有个体的适应度.

在没有达到算法终止条件下, 循环执行Step 1～Step 4.

Step 1: 按照适应度大小进行排序;

Step 2: 按照一定比例 60%选择适应度较好的个体作为雇佣蜂, 余下的个体作为跟随蜂;

Step 3: 对于雇佣蜂, 每个个体随机地向雇佣蜂中的其他个体学习, 如果适应度得到改善, 则更新当前个体;

Step 4: 对于跟随蜂, 每个个体随机地向雇佣蜂中的某个个体学习, 对于跟随蜂中的个体在向雇佣蜂中的个体学习后, 如果适应度得到改善, 则更新当前个体, 否则进行自我学习.

在该算法中, 自我学习采用变异算子, 其操作过程是随机选择两个工件并交换它们的位置.

在该算法中, 向其他个体学习有两种方式: 一是采用交叉算子, 二是为了在被挑选的学习个体附近进行较大范围的搜索, 采用交叉算子与变异算子的组合. 值得说明的是, 本节中双种群协同学习算法有两个版本, 将采用交叉算子进行学习的算法记作 DPCL, 将采用交叉算子与变异算子组合进行学习的算法记作 DPCLM.

交叉算子的操作过程如图 11 所示.

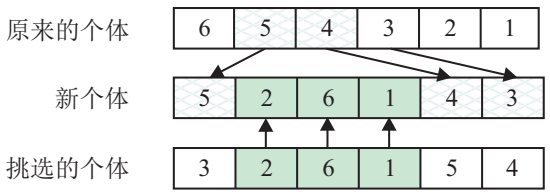

图 11　交叉学习

交叉算子的具体操作过程如下:

1) 随机选择两个交叉点.

2) 将被挑选的学习个体的两个交叉点之间的部分复制到新个体相应位置.

3) 将学习个体中余下部分依次复制到新个体剩余位置.

为了改进局部解的质量, 可以在算法中加入局部搜索的方法. 基于这种思想, 在雇佣蜂阶段增加基于插入算子的局部搜索[8], 在跟随蜂阶段后增加基于变邻域搜索的局部搜索[1]. 将加入局部搜索的 DPCLM 算法记作 DPCLMLS. 其中, 变邻域搜索中的算子包括如下内容.

交换: 随机选择两个工件交换它们的位置;

向前插入: 随机选择两个工件, 把位置小的工件插入到位置大的前面;

向后插入: 随机选择两个工件, 把位置大的工件插入到位置小的前面;

逆转: 随机选择两个工件, 把两个工件之间的所有工件逆序排列;

邻居交换: 随机选择一个工件, 把它与下一个邻居工件交换位置. 如果选择的工件是最后的工件, 则把它与第一个工件交换位置.

3.3　实验结果及分析

为了验证 DPCL 算法求解 PFSP 的性能, 在一些著名的基准测试问题上, 以最小化完工时间为目标进行仿真实验. Reeves 实例和 Taillard 实例常作为中到大规模标准的测试问题[14-15], 在本实验中采用 21 个 Reeves 实例和 40 个 Taillard 实例. 其中: Reeves 实例包括 20×5、20×10、20×15、30×10、30×15、50×10 和 75 × 20 七组, 每组 3 个实例; Taillard 实例包括 20 × 5、20 × 10、20 × 20 和 50 × 5 四组, 每组 10 个实例. 实验结果用最佳相对偏差 BRE (Best Relative Error) 和平均相对偏差 ARE (Average Relative Error) 表示.

最佳相对偏差 BRE 表示为

$$\mathrm{BRE} = \frac{C_{\mathrm{best}} - C^*}{C^*} \times 100.$$

平均相对偏差 ARE 表示为

$$\mathrm{ARE} = \frac{\sum_{i=1}^{k} \dfrac{C_{\mathrm{best}} - C^*}{C^*}}{k}.$$

在该实验中, 对 PSOspv[6]、DPCLM、DPCLMLS、DPCL、HABC[9]、HTLBO[1]方法进行了比较.

种群大小设置为 $10 \times m$, 终止条件是运行时间为 $0.1 \times n \times m \times 1\,000$ ms. 算法在每个实例上运行 10 次. DPCLM、DPCLMLS 和 DPCL 选择种群中具有较好适应度的个体的 60% 构成雇佣蜂, 余下的个体构成跟随蜂.

表 2 给出各个算法在 21 个 Reeves 测试实例上的最佳相对偏差, 每个函数上的最好结果加粗显示.

从表 2 可以看出, DPCLMLS 在 16 个测试实例上取得了最好的效果, DPCLM 在 13 个测试实例上取得了最好的效果, HABC 在 7 个测试实例上取得了最好的效果.

表 3 给出了各个算法在 Reeves 测试实例上的平均相对偏差, 每个函数上的最好结果加粗显示.

从表 3 可以看出, 在 20×5、20×10、20×15、30×10、30×15 和 50×10 六组实例上, DPCLMLS 都取得了最好的效果, 在 75×20 这组实例上, DPCLM 取得了最好的效果, DPCLMLS 排名第 3, 在 20×10 这组实例上, 除了 PSOspv 算法外的所有算法都取得了最好的效果.

表 4 给出了各算法在 40 个 Taillard 测试实例上的最佳相对偏差, 每个函数上的最好结果加粗显示.

表 2 各算法在 Reeves 测试实例上的最佳相对偏差

instance	n	m	C^*	DPCL	DPCLM	DPCLMLS	PSOspv	HABC	HTLBO
rec 1	20	5	1 247	0.160 385	**0**	**0**	3.688 853	0.160 385	0.160 385
rec 3	20	5	1 109	0.180 343	**0**	**0**	2.164 112	0.180 343	0.180 343
rec 5	20	5	1 242	**−0.080 520**	**−0.080 520**	**−0.080 520**	1.690 821	**−0.080 520**	**−0.080 520**
rec 7	20	10	1 566	**0**	**0**	**0**	4.150 702	**0**	**0**
rec 9	20	10	1 537	**0**	**0**	**0**	2.147 040	**0**	**0**
rec 11	20	10	1 431	**0**	**0**	**0**	1.886 792	**0**	**0**
rec 13	20	15	1 930	0.310 881	0.259 067	**0**	2.901 554	0.259 067	0.207 254
rec 15	20	15	1 950	0.102 564	**0**	**0**	2.102 564	0.051 282	0.666 667
rec 17	20	15	1 902	0.630 915	0.368 034	**−0.210 3**	2.681 388	0.630 915	0.368 034
rec 19	30	10	2 093	1.290 014	0.430 005	**0.286 670**	5.733 397	0.430 005	0.955 566
rec 21	30	10	2 017	1.437 779	1.437 779	**1.189 886**	4.709 965	1.636 093	1.487 357
rec 23	30	10	2 011	1.093 983	**0.447 539**	**0.447 539**	2.436 599	**0.447 539**	0.497 265
rec 25	30	15	2 513	1.352 965	0.517 310	**0.477 517**	4.655 790	0.676 482	1.233 585
rec 27	30	15	2 373	1.137 800	0.800 674	**0.252 845**	3.286 979	1.222 082	1.011 378
rec 29	30	15	2 287	1.792 742	1.049 41	0.787 057	7.127 241	**0.174 902**	1.049 410
rec 31	50	10	3 045	1.904 762	**0.985 222**	1.379 310	4.794 745	1.576 355	2.495 895
rec 33	50	10	3 114	0.738 601	0.834 939	**0.224 791**	4.206 808	0.834 939	0.834 939
rec 35	50	10	3 277	**0**	**0**	**0**	3.539 823	**0**	**0**
rec 37	75	20	4 951	4.120 380	**2.565 138**	4.847 506	9.008 281	5.554 433	3.797 213
rec 39	75	20	5 087	2.555 534	**1.828 190**	4.088 854	8.452 919	3.499 115	2.280 322
rec 41	75	20	4 960	3.548 387	**3.245 968**	5.120 968	8.911 290	5.181 452	3.689 516

表 3 各算法在 Reeves 测试实例上的平均相对偏差

问题	20 × 5	20 × 10	20 × 15	30 × 10	30 × 15	50 × 10	75 × 20
PSOspv	0.754 378	0.818 453	0.768 550	1.287 996	1.507 000	1.254 137	2.637 249
DPCL	0.026 021	**0**	0.104 435	0.382 177	0.428 350	0.264 336	1.022 430
DPCLM	**−0.008 051**	**0**	0.062 710	0.231 532	0.336 739	0.182 016	**0.763 929**
DPCLMLS	**−0.008 051**	**0**	**−0.021 030**	**0.192 409**	**0.151 742**	**0.160 410**	1.405 733
HABC	0.026 021	**0**	0.094 126	0.251 363	0.207 346	0.241 129	1.423 500
HTLBO	0.026 021	**0**	0.124 195	0.294 019	0.329 437	0.333 083	0.976 705

表 4 各算法在 Taillard 测试实例上的最佳相对偏差

instance	n	m	C^*	DPCL	DPCLM	DPCLMLS	PSOspv	HABC	HTLBO
ta 001	20	5	1 278	**0**	**0**	**0**	1.486 697	**0**	1.486 698
ta 002	20	5	1 359	**0**	**0**	**0**	**0**	**0**	0.515 085
ta 003	20	5	1 081	**0**	**0**	**0**	2.682 701	**0**	0.647 549
ta 004	20	5	1 293	**0**	0.309 358	**0**	2.861 562	**0**	**0**
ta 005	20	5	1 235	**0**	**0**	**0**	1.214 574	**0**	1.214 575
ta 006	20	5	1 195	**0**	**0**	**0**	5.271 966	1.255 23	0
ta 007	20	5	1 239	0.242 13	0.968 523	**0**	1.614 205	**0**	0.968 523
ta 008	20	5	1 206	**0**	**0**	**0**	2.487 562	**0**	**0**
ta 009	20	5	1 230	**0**	**0**	**0**	2.764 227	**0**	1.869 919
ta 010	20	5	1 108	**0**	**0**	**0**	2.075 812	**0**	**0**
ta 011	20	10	1 582	**0.063 211**	**0.063 211**	**0.063 211**	1.453 855	0.252 844	0.189 633
ta 012	20	10	1 659	1.145 268	0.060 277	**0**	3.918 022	0.301 386	0.964 436
ta 013	20	10	1 496	0.401 069	0.267 379	**0**	2.205 882	0.802 139	0.935 829
ta 014	20	10	1 378	0.217 706	**0**	**0**	3.193 033	0.507 982	0.725 689
ta 015	20	10	1 419	0.070 472	0.422 832	**0**	4.228 329	**0**	0.704 722
ta 016	20	10	1 397	0.357 909	**0**	**0**	2.219 04	0.644 237	0.572 656
ta 017	20	10	1 484	0.539 083	**0**	**0**	2.021 563	**0**	0.606 469
ta 018	20	10	1 538	0.650 195	**0.390 117**	**0.390 117**	2.470 741	0.780 234	0.455 137
ta 019	20	10	1 593	0.062 774	0.251 098	**0**	2.824 858	0.564 971	0.941 62
ta 020	20	10	1 591	0.439 974	**0**	**0**	3.331 238	0.565 681	1.068 51
ta 021	20	20	2 297	0.522 42	0.130 605	**0**	1.872 006	0.478 885	0.174 14
ta 022	20	20	2 100	0.523 809	0.238 095	**−0.047 619**	2.523 809	0.523 809	0.571 429
ta 023	20	20	2 326	0.429 922	**0.085 984**	**0.085 984**	1.805 674	0.515 907	0.558 899
ta 024	20	20	2 223	**0**	0.269 905	**0**	2.968 96	0.269 905	0.269 906
ta 025	20	20	2 291	0.567 437	0.218 245	**0**	1.615 015	0.305 543	0.218 245
ta 026	20	20	2 226	0.134 77	0.179 694	**0.089 847**	1.347 708	0.134 77	0.134 771
ta 027	20	20	2 273	**0**	**0**	0.219 974	1.979 762	0.307 963	0.351 958
ta 028	20	20	2 200	0.590 909	0.227 272	**0**	1.363 636	**0**	**0**
ta 029	20	20	2 237	0.223 513	**0**	**0**	2.950 379	0.223 513	0.223 514
ta 030	20	20	2 178	0.229 568	0.045 913	**0**	2.754 82	0.505 05	0.229 568

续表

instance	n	m	C^*	DPCL	DPCLM	DPCLMLS	PSOspv	HABC	HTLBO
ta 031	50	5	2 724	**−0.440 528**	**−0.440 528**	**−0.440 528**	0.220 264	**−0.440 528**	**−0.440 528**
ta 032	50	5	2 834	0.494 001	**0.141 143**	**0.141 143**	2.258 292	**0.141 143**	0.635 145
ta 033	50	5	2 621	0.038 153	0.038 153	0.038 153	2.747 043	**0**	0.038 153
ta 034	50	5	2 751	**0.072 7**	**0.072 7**	**0.072 7**	2.908 033	**0.072 7**	0.399 855
ta 035	50	5	2 863	0.034 928	**0**	**0**	2.514 844	**0**	0.034 928
ta 036	50	5	2 829	**0**	0.106 044	**0**	2.297 631	0.106 044	0.106 045
ta 037	50	5	2 725	0.256 88	**0**	**0**	1.541 284	**0**	0.256 881
ta 038	50	5	2 683	**0**	**0**	**0**	1.751 77	0.409 988	0.782 706
ta 039	50	5	2 552	0.470 219	0.352 664	**0**	1.959 247	0.352 664	0.470 219
ta 040	50	5	2 782	**0**	**0**	**0**	1.689 432	**0**	**0**

表 5　各算法在 Taillard 测试实例上的平均相对偏差

问题	20 × 5	20 × 10	20 × 20	50× 5
PSOspv	2.245 93	2.786 656	2.118 177	1.988 784
DPCL	0.024 213	0.394 766	0.322 235	0.092 635
DPCLM	0.127 788	0.145 491	0.139 571	0.027 017
DPCLMLS	**0**	**0.045 333**	**0.034 819**	**−0.018 853**
HABC	0.125 523	0.441 947	0.326 534	0.064 201
HTLBO	0.670 235	0.716 47	0.273 243	0.228 34

从表 4 可看出, DPCLMLS 在 38 个测试实例上取得了最好的效果, DPCLM 在 24 个测试实例上取得了最好的效果, HABC 在 19 个测试实例上取得了最好的效果, DPCL 在 17 个测试实例上取得了最好效果.

表 5 给出了各个算法在 Taillard 测试实例上的平均相对偏差, 每个函数上的最好结果加粗显示. 从表 5 可以看出, 在 20 × 5、20 × 10、20 × 20 和 50×5 这四组测试实例上, DPCLMLS 算法都取得了最好的效果.

图 12 为各算法在 Reeves 实例上的平均相对偏差的折线图; 图 13 为各算法在 Taillard 实例上的平均相对偏差的折线图.

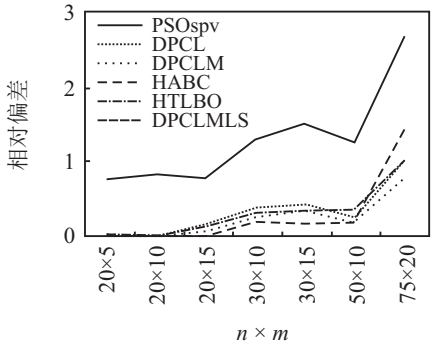

图 12　算法在 Reeves 实例上的平均相对偏差折线

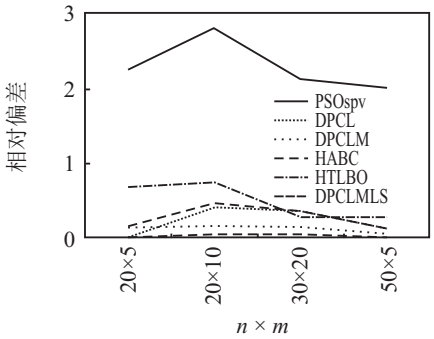

图 13　算法在 Taillard 实例上的平均相对偏差折线

通过对 Reeves 系列和 Taillard 系列的实例测试可以看出, 加入局部搜索策略的 DPCLMLS 算法取得了排名第 1 的性能, 它在更多的实例上取得了比其他算法更好的效果, 验证了它在 PFSP 问题上具有较好的优化能力, 也说明了局部搜索策略对 DPCLM 提高了解的质量. 从表 3 和表 4 可以看出, 没有加入局部搜索策略的 DPCLM 算法取得了排名第 2 的性能, 这也验证了所提出算法在求解 PFSP 问题上的有效性, 其原因也正是对原始 ABC 算法的改进: 一是设置了明确的“精英”指导搜索策略, 将种群按照适应度高低划分为两个种群, 用高适应度种群引导优化过程; 二是在跟随蜂的适应度没有得到改善时采用变异算子产生新解以跳出局部最优解.

4　结　论

本文提出了一种双种群协同学习算法, 根据食物源的适应度对人工蜜蜂群算法的雇佣蜂和跟随蜂重新划分, 并对跟随蜂的学习机制给出新的规则, 新规则充分利用了高适应度的个体对全局搜索的影响力. 为了验证新算法的优化能力, 在 10 个常用基准测试函数上与 ABC、PSO、CPSO 和 GA 进行了比较, 测试结果表明, 无论在收敛速度还是求解精度上, DPCL 在 8 个测试函数上的表现都超出了其他 4 种算法. 由于 DPCL 在数值优化上具有很好的效果, 将其应用于求解 PFSP. 在一些著名的中大规模测试实例 (包括 21 个 Reeves 系列的实例和 40 个 Taillard 系列) 上进行求解, 并与较新的几种智能算法进行比较测试, 结果表明, 加入局部搜索策略的 DPCL 在更多的测试实例上取得了最好的效果.

本文中求解调度问题的目标仅限于最小化完工时间, 生产实际中的调度问题可能存在其他优化目标或者多目标, 在这些问题上进行探索是下一步的研究方向.

参考文献(References)

[1] Kennedy J, Eberhart R. Particle swarm optimization[C]. IEEE Int Conf Neural Networks. Perth: IEEE Computer Society, 1995: 1942-1945.

[2] 宋存利, 时维国. 求解车间调度问题的2阶段混合粒子群优化算法[J]. 信息与控制, 2012, 41(2): 193-196.
(Song C L, Shi W G. A two stage hybrid particle swarm optimization algorithm for job-shop scheduling problem [J]. Information and Control, 2012, 41(2): 193-196.)

[3] Nawaz M, Enscore J E, Ham I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem[J]. Omega-Int J of Management Science, 1983, 11(1): 91-95.

[4] 刘延风, 刘三阳. 改进微粒群优化求解置换流水车间调度问题[J]. 计算机集成制造系统, 2009, 15(10): 1968-1972.
(Liu Y F, Liu S Y. Improved particle swarm optimization for permutation flow shop scheduling problems[J]. Computer Integrated Manufacturing Systems, 2009, 15(10): 1968-1972.)

[5] Tasgetiren M F, Sevkli M, Liang Y C, et al. Particle swarm optimization algorithm for permutation flow shop sequencing problem[J]. Lecture Notes in Computer Science, 2004, 3172: 382-389.

[6] Tasgetiren M F, Liang Y C, Sevkli M G, et al. A particle swarm optimization algorithm for makespan and total flow time minimization in the permutation flow shop sequencing problem[J]. European J of Operational Research, 2007, 77(3): 1930-1947.

[7] Karaboga D. An idea based on honey bees warm for numerical optimization[R]. Kayseri: Erciyes University, 2005.

[8] Tasgetiren M F, Pan Q K, Suganthan P N, et al. A discrete artificial bee colony algorithm for the total flowtime minimization in permutation flow shops[J]. Information Sciences, 2011, 181(16): 3459-3475.

[9] 晏晓辉. 群体智能算法研究及其在调度优化中的应用[D]. 北京: 中国科学院大学, 2012.
(Yan X H. Research of swarm intelligence algorithms and its application on scheduling optimization[D]. Beijing: University of Chinese Academy of Sciences, 2012.)

[10] Xie Z P, Zhang C Z, Shao X Y, et al. An effective hybrid teaching-learning-based optimization algorithm for permutation flow shop scheduling problem[J]. Advances in Engineering Software, 2014, 77: 35-47.

[11] Niu Bei, Zhu Yunlong, He Xiaoxian, et al. MCPSO: A multi-swarm cooperative particle swarm optimizer[J]. Applied Mathematics and Computation, 2007, 185(2): 1050-1062.

[12] Karaboga D, Akay B. A comparative study of artificial bee colony algorithm[J]. Applied Mathematics and Computation, 2009, 214(1): 108-132.

[13] Van Den Bergh F, Engelbrecht A P. A cooperative approach to participle swam optimization[J]. IEEE Trans on Evolutionary Computation, 2004, 8(3): 225-239.

[14] Zhang Y, Li X, Wang Q. Hybrid genetic algorithm for permutation flowshop scheduling problems with total flowtime minimization[J]. European J of Operational Research, 2009, 196(3): 869-876.

[15] Wang L, Pan Q K, Suganthan P N. A novel hybrid discrete differential evolution algorithm for blocking flow shop scheduling problems[J]. Computers & Operations Research, 2010, 37(3): 509-520.

(责任编辑：闫 妍)

《控制与决策》被评为“2016中国国际影响力优秀学术期刊”和“2016年度中国高校百佳科技期刊”

本刊讯 2016年11月21日～23日, 由中国期刊协会等5家单位联合主办、同方知网承办的“中国学术期刊未来论坛”在京隆重举行. 在此次大会上, 中国学术期刊(光盘版)电子杂志社有限责任公司、清华大学图书馆和中国学术文献国际评价中心联合发布了《2016中国学术期刊国际国内影响力研究报告》, 公布了2016年度“中国最具国际影响力学术期刊”和“中国国际影响力优秀学术期刊”名单. 按期刊国际影响力指数排序, 东北大学主办的《控制与决策》被评为“2016中国国际影响力优秀学术期刊”.

中国高校科技期刊研究会第20届年会于2016年11月23日在京举办, 颁发了2016年度中国高校百佳科技期刊等各类奖项. 东北大学主办的《控制与决策》被评为“2016年度中国高校百佳科技期刊”.

又讯 2016年10月12日, 中国科学技术信息研究所主办的中国科技论文统计结果发布会在北京国际会议中心召开, 会上发布了“中国百篇最具影响国内学术论文”名单.《控制与决策》于2014年第29卷第1期上发表的论文“基于局部搜索的人工蜂群算法”(作者刘三阳等)名列其中.