

合作型协同进化遗传算法求解分布式柔性作业车间调度问题*

董博文^① 王有远^{②③}

(①南昌航空大学飞行器工程学院, 江西 南昌 330063; ②南昌航空大学工业工程研究所, 江西 南昌 330063; ③南昌市航空复杂系统与智能科学重点实验室, 江西 南昌 330063)

摘要: 针对以最小化最大完工时间为优化目标的分布式柔性作业车间调度问题, 提出一种合作型协同进化遗传算法。采用工厂分配和工序排序解耦编码, 基于机器负荷解码并基于工厂负荷初始化种群, 使算法在较优的解空间内迭代搜索。利用分而治之的思想, 将问题分解为多个子问题, 通过随机协同机制促进子种群协同进化并提高全局搜索能力。使用基于关键工厂的多重局部扰动策略, 提高算法的局部开发能力。在基准实例上进行实验, 并与其他算法进行对比, 验证了所提算法的有效性。

关键词: 分布式柔性作业车间调度; 合作型协同进化算法; 遗传算法; 最大完工时间

中图分类号: TP18 **文献标识码:** A

DOI: 10.19287/j.mtmt.1005-2402.2023.12.026

Cooperative co-evolutionary genetic algorithm for distributed flexible job shop scheduling problem

DONG Bowen^①, WANG Youyuan^{②③}

(①School of Aircraft Engineering, Nanchang Hangkong University, Nanchang 330063, CHN; ②Institute of Industrial Engineering, Nanchang Hangkong University, Nanchang 330063, CHN; ③Nanchang Key Laboratory of Aviation Complex Systems and Intelligence Science, Nanchang 330063, CHN)

Abstract: A cooperative co-evolutionary genetic algorithm is proposed for a distributed flexible job shop scheduling problem with the optimization objective of minimizing the maximum completion time. A decoupled encoding of factory assignment and operation sequencing is used, based on machine load decoding as well as initializing the population based on factory load, so that the algorithm iteratively searches in a better solution space. Using the divide-and-conquer idea, the problem is decomposed into multiple sub-problems, and a random collaboration mechanism is used to promote the subpopulations to co-evolve and improve the global exploration capability. Multiple local perturbation strategy based on key factory is used to improve the local exploitation capability. Experiments are conducted on benchmark instances and compared with other algorithms to verify the effectiveness of the proposed algorithm.

Keywords: distributed flexible job shop scheduling; cooperative co-evolutionary algorithm; genetic algorithm; maximum completion time

随着经济全球化发展, 许多制造企业由集中式制造向分布式制造转变, 生产制造不再局限于在单一工厂进行, 而是多工厂协同制造。生产调度在分布式制造中进一步延伸, 在机械加工等制造场景中,

产生了分布式柔性作业车间调度问题 (distributed flexible job shop scheduling problem, DFJSP)。DFJSP属于NP(难问题), 对其进行优化有助于缩短制造期、提高机器利用率, 因此其研究具有重要的学

* 国家自然科学基金(71761028); 航空基金(2022Z069056002)

术意义和应用价值。

国内外学者对 DFJSP 进行了相关研究。Chan F T S 等^[1]最早研究了 DFJSP, 提出了一种基于支配基因的遗传算法, 对问题解空间进行了完全编码; Giovanni L D 等^[2]使用不完全编码方式, 只编码了工厂分配和工序排序信息, 通过解码确定机器选择, 提出一种改进遗传算法 (IGA) 求解 DFJSP; Lu P H 等^[3]和 Wu M C 等^[4]分别使用工件序列 (GA_JS) 和工序序列 (GA_OP) 一维编码, 设计启发式规则进行解码; Marzouki B 等^[5-6]使用分散禁忌搜索 (DTSMA) 和化学反应优化算法 (CRO) 求解 DFJSP; 吴锐等^[7]提出一种改进人工蜂群算法, 引入基于关键路径的局部搜索算子提高算法局部搜索能力; 吴秀丽等^[8]设计了一种改进差分进化算法 (IDESAA), 利用模拟退火提高算法的局部搜索能力; 孟磊磊等^[9]提出了一种混合蛙跳算法 (HSFLA), 结合变邻域搜索提高算法的局部搜索能力; Li X Y 等^[10]使用工厂分配和工序排序双层编码, 提出一种改进灰狼优化算法 (IGWO), 使用基于关键工厂和关键路径的邻域搜索策略。

上述研究主要通过部分解空间搜索来降低对 DFJSP 的求解难度, 随着问题规模增大解空间急剧扩大, 算法全局搜索能力不足, 寻优能力变差。本文提出一种合作型协同进化遗传算法 (cooperative co-evolutionary genetic algorithm, CCGA) 求解 DFJSP, 利用分而治之的思想将问题分解为多个低维子问题, 使用随机协同机制促进种群协同进化以提高算法全局搜索能力。为弥补协同进化全局搜索能力强而局部开发能力弱的缺点, 使用基于关键工厂的多重局部扰动策略, 提高算法的局部开发能力。通过对公共基准算例进行测试并与其他算法对比, 验证了所提算法的有效性。

1 分布式柔性作业车间调度问题

DFJSP 描述如下: 给定 n 个待加工工件 $J = \{J_1, J_2, \dots, J_i, \dots, J_n\}$, 需要将其分配给 q 个工厂 $U = \{U^1, U^2, \dots, U^l, \dots, U^q\}$ 加工。工件 i 有 n_i 个工序 $J_i = \{O_{i1}, O_{i2}, \dots, O_{ij}, \dots, O_{in_i}\}$, 若工件 i 分配到工厂 l 加工, 则该工件所有工序均在该工厂内加工。工厂 l 有 m^l 台机器 $M^l = \{M^{l1}, M^{l2}, \dots, M^{lk}, \dots, M^{lm^l}\}$, 可加工工序 O_{ij} 的机器集合为 M_{ij}^l , $M_{ij}^l \subseteq M^l$ 。每台机器在同一时刻最多只能加工一个工序, 任一工件在任一时刻只能在一台机器上加工。工序一旦开始加工便不能

中断, 同一工件的所有工序需满足优先约束。所有工厂、机器和工件都在 0 时刻可用, 加工时间在不同工厂以及机器上可能不同。DFJSP 的目标是将工件分配给工厂、选择加工工序的机器和确定各机器上的加工顺序, 使得调度的最大完工时间最短。DFJSP 的数学模型可参考 Meng L 等^[11]归纳的 4 种主要模型。

2 合作型协同进化遗传算法求解 DFJSP

2.1 编码与解码

为使算法在解空间的较优区域搜索, 采用工厂分配和工序排序解耦编码方式, 在解码时基于机器负荷确定机器选择。一个 DFJSP 实例见表 1, 表中“—”表示该工序无法在该机器加工。该实例的一个编码方案如图 1 所示, 工厂分配编码 FA 长度与工件数相同, 第 i 个基因表示工件 i 所分配的工厂, 如第 1 个基因表示工件 J_1 分配到工厂 2。工序排序编码 OS 长度与总工序数相同, 使用工件号编码, 工件号出现的次数表示第几个工序, 如第 4 个基因为 3, 第二次出现, 表示为工序 O_{32} 。

表 1 DFJSP 实例

工件	工序	U^1		U^2	
		M^{11}	M^{12}	M^{21}	M^{22}
J_1	O_{11}	1	2	4	2
	O_{12}	4	3	2	—
	O_{13}	2	—	4	—
J_2	O_{21}	3	2	—	3
	O_{22}	4	2	3	4
	O_{23}	1	3	4	3
J_3	O_{31}	—	1	4	2
	O_{32}	6	3	2	3
	O_{33}	4	—	1	2

对应工件	J_1	J_2	J_3						
工厂分配编码 FA	2	1	2						
工序排序编码 OS	2	1	3	3	2	1	1	2	3
对应工序	O_{21}	O_{11}	O_{31}	O_{32}	O_{22}	O_{12}	O_{13}	O_{23}	O_{33}

图 1 DFJSP 实例的一个编码方案

染色体解码依 OS 顺序依次解码, 根据 FA 编码确定工序所在工厂, 进而确定可用机器集。选择最大完工时间最小的机器, 当具有相同的最大完工

时间时,选择加工时间短的机器,若加工时间也相同,则随机选择。在将工序分配给机器时,使用贪心策略,将工序安排在尽可能早地符合要求的机器空闲中,使得完工时间最小。图1所示的第一个基因表示工序 O_{21} ,根据FA可知工件 J_2 分配到工厂1,对应有两台机器 M^{11} 和 M^{12} 可加工,工时分别为1和2,显然在机器 M^{11} 上的完工时间比在 M^{12} 上短,因此选择机器 M^{11} 加工工序 O_{21} 。图1编码方案对应的调度甘特图如图2所示。

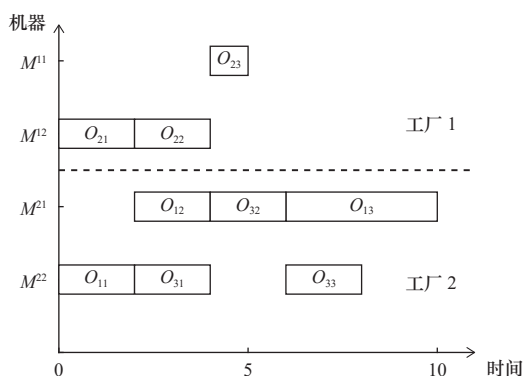


图2 编码方案对应的调度甘特图

2.2 协同机制

CCGA的思想是将问题分解为多个子问题,分而治之。子问题之间的协同机制是CCGA的一个关键因素,子种群中个体的适应度评估需要与其他子种群中的个体协作。常见的协同机制包括最优个体协同选择、最差个体协同选择、随机个体协同选择等。为提高算法全局搜索能力,使用随机个体协同机制,以尽可能多地搜索解空间。对于工厂分配子种群中的一个个体 P_i ,随机选择工序排序子种群中的一个个体,组合成完整染色体,通过解码得到的最大完工时间即为 P_i 的适应度,工序排序子种群中的个体适应度评估类似。

2.3 种群初始化与进化算子

初始种群的质量对于算法迭代搜索有一定的影响,好的初始种群能够帮助算法找到更优解。为提高初始种群质量,随机生成OS部分编码,使用Lu P H等^[3]提出的基于工厂负荷的工厂分配方法确定FA部分编码。

选择算子用于模拟适者生存的原则,能够帮助种群向最优个体收敛,使用三元锦标赛进行选择,以平衡随机协同机制收敛速度慢的影响。

交叉有助于增加种群多样性,工厂分配编码使用单点交叉,工序排序编码使用两点顺序交叉^[4],

如图3所示。两个交叉点将个体分为3段,子代 C_1 保留父代 P_1 的第一段和第三段,在父代 P_2 中删除这些基因,剩余部分即为 C_1 第二段。这样能保证得到的子代依然是可行解,且最大限度继承父代的编码信息。子代 C_2 使用类似的方法产生。

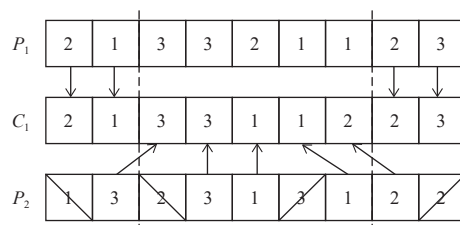


图3 工序排序两点顺序交叉

变异能在一定程度上避免算法陷入局部最优解,使用逆转变异算子对FA与OS进行变异,即任意选取两点,逆转两点之间的基因顺序。

2.4 多重局部扰动策略

DFJSP中完工时间最大的工厂称为关键工厂,关键工厂中具备最大完工时间的路径称为关键路径,关键路径上的工序称为关键工序。只有关键工厂和关键路径发生变化,才可能打破原有调度得到更优解。CCGA将问题分解为多个子问题,分而治之,优势是能够更加充分搜索解空间,缺点是局部开发能力较差。为提高所提算法的局部开发能力,设计如下6个邻域扰动算子。

邻域1:将关键工厂中的一个工件分配到完工时间最小的工厂中。

邻域2:将关键工厂的工序分成 k 段,每一段第一个基因与其他基因交换。

邻域3:将关键工厂的工序分成 k 段,重排 k 段的顺序。

邻域4:随机选择一个关键工序,与关键工厂的其他工序交换。

邻域5:随机选择一个关键工序,插入到关键工厂工序序列的某个位置。

邻域6:对于关键工厂中关键路径,采用移动一个工序的邻域搜索方法。

邻域2和3改编自Zhu J等^[12]提出的部分成对交换和简单插入,将均匀分为 k 段改为随机点位分段。邻域4和5改编自Valente J M S等^[13]提出的邻近非邻近交换和插入,将随机选择一个工序改为选择一个关键工序。邻域6为Mastrolilli M等^[14]针对FJSP提出的移动一个关键工序邻域结构。

多重局部扰动过程如下:

Step1: 随机选择一个邻域扰动算子。

Step2: 根据邻域结构生成一个新解。

Step3: 若新解优于原解, 以新解为中心返回 Step1, 否则返回 Step2, 直到邻域结构全部搜索完。

为提高计算效率, 设定只有当新个体适应度不超过最优解 d 时才进行多重局部扰动, 同时为避免进化后期种群收敛造成大量个体满足上述要求导致计算量增大, 设定每个种群最多 15% 的个体能够进行多重局部扰动, 且增加选择概率, $rand < 0.5$ 时才进行多重局部扰动, 提高局部搜索的均匀性。

2.5 算法整体框架

CCGA 算法伪代码如算法 1 (图 4), 4~10 行表示工厂分配子种群和工序排序子种群分别进化; 11~23 行表示使用随机协同策略评估个体适应度, 并对满足条件的个体进行多重局部扰动; 26~28 行表示在算法陷入局部最优时引入随机个体, 提高种群多样性继续搜索, 以帮助算法跳出局部最优。

算法1: CCGA

符号: 种群规模 N , 最大迭代次数 G , 工厂分配和工序排序子种群 P_1 、 P_2 , 最优个体 S_{best} , 最优个体适应度 f_{best}

```

1  随机初始化  $P_1$ , 基于工厂负荷生成  $P_2$ 
2  基于协同机制评估子种群个体, 得到最优个体  $S_{best}$  及其适应度  $f_{best}$ 
3  for  $g = 1$  to  $G$  do
4      for  $i = 1$  to 2 do
5          for  $j = 1$  to  $N/2$  do
6               $p1, p2 \leftarrow$  3元锦标赛选择 ( $P_i$ )
7               $c1, c2 \leftarrow$  交叉变异 ( $p1, p2$ )
8              将  $c1, c2$  添加到子种群  $P'_i$ 
9          end for
10         end for
11         for  $i = 1$  to 2 do
12             for  $j = 1$  to  $N$  do
13                  $S_{ij} \leftarrow P_i$  的第  $j$  个个体与另一子种群的随机个体形成一个完整解
14                  $f_{ij} \leftarrow S_{ij}$  的适应度
15                 if  $f_{ij} - f_{best} < f_{best} * d$  then
16                     对  $S_{ij}$  进行多重局部扰动并更新  $f_{ij}$ 
17                 end if
18                 if  $f_{ij} < f_{best}$  then
19                      $f_{best} = f_{ij}$ 
20                      $S_{best} = S_{ij}$ 
21                 end if
22             end for
23         end for
24          $P_i \leftarrow P'_i$ 
25          $P'_i \leftarrow \{\}$ 
26         if 连续20次迭代  $f_{best}$  无变化 then
27             随机生成  $N/2$  个体替换  $P_1$ 、 $P_2$  中个体
28         end if
29     end for
30 输出最优个体  $S_{best}$  及其适应度  $f_{best}$ 

```

图 4 CCGA 算法伪代码

3 实验验证

采用 De Giovanni L 和 Pezzella F^[2] 基于 23 个 FJSP 基准 (la01-la20、mt06、mt10 和 mt20) 拓展到 2/3/4 工厂情形下生成的 69 个 DFJSP 基准验证 CCGA 的有效性。使用 Matlab R2020b 编程, 取种群规模 $N = 200$ 、最大迭代次数 $G = 150$ 、交叉概率 $P_c = 0.95$ 、变异概率 $P_m = 0.01$ 、局部扰动比例参数 $d = 0.2$, 每个基准独立运行 30 次。

3.1 多重局部扰动策略有效性验证

为验证多重局部扰动策略的有效性, 使用 2 工厂下的 23 个基准进行对比实验, 对比结果见表 2。

表 2 CCGA-non 与 CCGA 对比实验

基准	工件数×工序数	CCGA-non		CCGA	
		C_m	AVG	C_m	AVG
la01	10×5	413	413.0	413	413.0
la02	10×5	394	394.0	394	394.0
la03	10×5	349	349.0	349	349.0
la04	10×5	369	369.0	369	369.0
la05	10×5	380	380.0	380	380.0
la06	15×5	426	443.9	420	434.2
la07	15×5	402	413.6	395	406.1
la08	15×5	420	427.4	395	415.5
la09	15×5	461	472.6	447	463.6
la10	15×5	443	447.1	443	443.4
la11	20×5	554	563.6	545	554.6
la12	20×5	484	495.7	475	486.0
la13	20×5	539	546.5	526	537.8
la14	20×5	551	566.6	544	557.3
la15	20×5	570	582.4	555	570.5
la16	10×10	717	717.0	717	717.0
la17	10×10	646	646.0	646	646.0
la18	10×10	663	663.0	663	663.0
la19	10×10	617	618.0	617	617.0
la20	10×10	756	756.0	756	756.0
mt06	6×6	47	47.0	47	47.0
mt10	10×10	655	655.0	655	655.0
mt20	20×5	544	559.9	526	546.3

表 2 中 CCGA 表示带多重局部扰动策略, CCGA-non 表示不带多重局部扰动策略, C_m 表示所求得的最优最大完工时间, AVG 表示平均最大完工时间。23 个基准中有 10 个基准 (la06-la09、la11-la15、mt20) CCGA 求得的最优解和平均值都比 CCGA-non 好, 剩余 13 个基准所求得的最优解相同, 其

中 CCGA 在基准 la10 和 la19 上的平均值比 CCGA-non 好。使用 Minitab 19.1 对两个算法在 23 个基准上求得的最优值和平均值进行 Wilcoxon 符号秩检验, 置信区间为 95%, 得到 p 值为 0.006 和 0.003, 均小于 0.05, 说明 CCGA 在统计意义上显著优于 CCGA-non, 即多重局部扰动策略是有效的。

3.2 CCGA 算法有效性验证

将 CCGA 与其他求解 DFJSP 的算法进行比较, 所求得的最优解对比结果见表 3 和表 4, 表中最后一行为使用 Wilcoxon 符号秩检验得到 p 值, 最后一列带 “*” 号表示 CCGA 找到基准下界。在 2 工厂情形时, 所有算法在 la01~la05、la16~la20、mt06 以及 mt10 基准上均能找到下界。剩余 11 个基准中, CCGA 求得的最优解均优于 IGA、DTSMA、CRO 和 GA_JS, 且 Wilcoxon 符号秩检验得到 p 值均小于 0.05, 证明 CCGA 优于 IGA、DTSMA、CRO 和 GA_JS 这 4 个算法。对于 GA_OP 和 IDESAA, CCGA

表 3 CCGA 和其他算法计算结果对比(2 工厂)

基准	LB	IGA	DTSMA	CRO	GA_JS	GA_OP	IDESAA	IGWO	HSFLA	CCGA
la01	413	413	413	413	413	413	413	413	413	413*
la02	394	394	394	394	394	394	394	394	394	394*
la03	349	349	349	349	349	349	349	349	349	349*
la04	369	369	369	369	369	369	369	369	369	369*
la05	380	380	380	380	380	380	380	380	380	380*
la06	413	445	424	424	424	424	419	413	413	420
la07	376	412	398	398	398	390	392	389	386	395
la08	369	420	406	420	406	397	399	393	391	395
la09	382	469	463	473	447	444	446	439	438	447
la10	443	445	445	445	443	443	443	443	443	443*
la11	413	570	553	600	548	541	543	539	538	545
la12	408	504	500	504	480	474	479	471	471	475
la13	382	542	551	537	533	529	530	523	522	526
la14	443	570	581	576	542	544	546	538	538	544
la15	378	584	597	584	562	554	555	550	550	555
la16	717	717	717	717	717	717	717	717	717	717*
la17	646	646	646	646	646	646	646	646	646	646*
la18	663	663	663	663	663	663	663	663	663	663*
la19	617	617	617	617	617	617	617	617	617	617*
la20	756	756	756	756	756	756	756	756	756	756*
mt06	47	47	—	—	47	47	47	47	47	47*
mt10	655	655	—	—	655	655	655	655	655	655*
mt20	387	560	—	—	529	525	523	519	518	526
p 值	0.004	0.006	0.006	0.013	0.554	0.594	0.006	0.006	0.006	/

分别在 3 个和 4 个基准上找到更优, Wilcoxon 符号秩检验得到 p 值均大于 0.05, 说明 GA_OP 和 IDESAA 在统计意义上与 CCGA 无显著差异。对于 IGWO 和 HSFLA, 求解结果和 p 值均表明其优于 CCGA。

表 4 CCGA 和其他算法计算结果对比(3 工厂)

基准	LB	IGA	DTSMA	CRO	GA_JS	GA_OP	IDESAA	IGWO	HSFLA	CCGA
la01	413	413	413	413	413	413	413	413	413	413*
la02	394	394	394	394	394	394	394	394	394	394*
la03	349	349	349	349	349	349	349	349	349	349*
la04	369	369	369	369	369	369	369	369	369	369*
la05	380	380	380	380	380	380	380	380	380	380*
la06	413	413	413	413	413	413	413	413	413	413*
la07	376	376	376	381	376	376	376	376	376	376*
la08	369	369	369	403	369	369	369	369	369	369*
la09	382	382	382	437	382	382	382	382	382	382*
la10	443	443	445	443	443	443	443	443	443	443*
la11	413	425	417	590	413	413	413	413	413	413*
la12	408	408	423	483	408	408	408	408	408	408*
la13	382	419	401	419	398	395	392	382	382	383
la14	443	443	462	470	443	443	443	443	443	443*
la15	378	451	434	451	420	417	389	396	390	408
la16	717	717	717	717	717	717	717	717	717	717*
la17	646	646	646	646	646	646	646	646	646	646*
la18	663	663	663	663	663	663	663	663	663	663*
la19	617	617	617	617	617	617	617	617	617	617*
la20	756	756	756	756	756	756	756	756	756	756*
mt06	47	47	—	—	47	47	47	47	47	47*
mt10	655	655	—	—	655	655	655	655	655	655*
mt20	387	439	—	—	408	397	387	387	387	389
p 值	0.100	0.036	0.014	0.181	0.181	0.789	0.181	0.181	0.181	/

在 3 工厂实验中, 由于生产资源增加, 但是生产任务没有变化, 因此比较容易找到最优解。从表 4 可以看出, CCGA 在 3 工厂情形下求解结果比 IGA、DTSMA、CRO、GA_JS 和 GA_OP 要好。与剩余算法求解结果主要在基准 la13、la15 和 mt20 上不同, Wilcoxon 符号秩检验 p 值均大于 0.05, 表明 CCGA 与这些算法在统计意义上没有显著差异。

4 工厂情形时可用生产资源进一步增加, CCGA 和 GA_JS、GA_OP、IDESAA、IGWO、HSFLA 均能找到 23 个基准的最优解, 在此不再列出实验结果。从以上结果分析可知, CCGA 在求解 DFJSP 时是有效的。图 5 所示为 2 工厂情形时基准 la08 的最优调度甘特图。

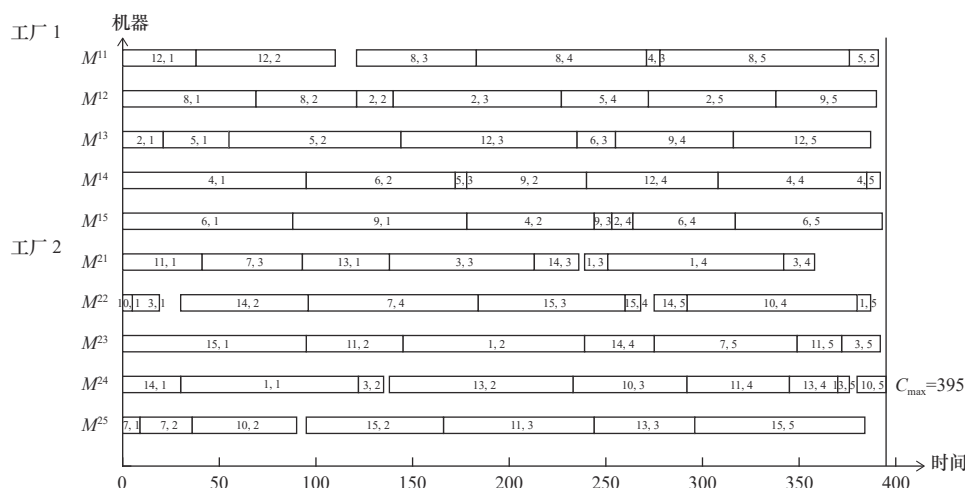


图 5 2 工厂情形 1a08 基准调度甘特图

4 结语

本文提出了一种合作型协同进化遗传算法求解 DFJSP，利用分而治之的思想提高全局搜索能力，并设计多重局部扰动策略提高算法的局部开发能力。在公共基准上的实验结果以及 Wilcoxon 符号秩检验表明多重局部扰动策略的有效性。在与其他算法对比中，所提算法优于 IGA、DTSMA、CRO 和 GA_JS 这 4 个算法，与 GA_OP 和 IDESAA 性能相近，略差于 IGWO 和 HSFLA，表明所提算法求解 DFJSP 是有效的。

DFJSP 这类问题通常包含多个子问题，与合作型协同进化算法的分而治之思想相贴合，因此具有一定的研究潜力。未来可以使用不同的协同机制，或在求解子问题时使用不同的进化算法来设计合作型协同进化算法。也可以尝试使用合作型协同进化算法求解多目标 DFJSP 以及其他车间调度问题。

参 考 文 献

- [1] Chan F T S, Chung S H, Chan P L Y. Application of genetic algorithms with dominant genes in a distributed scheduling problem in flexible manufacturing systems[J]. *International Journal of Production Research*, 2006, 44(3): 523-543.
- [2] Giovanni L D, Pezzella F. An improved genetic algorithm for the distributed and flexible job-shop scheduling problem[J]. *European Journal of Operational Research*, 2010, 200(2): 395-408.
- [3] Lu P H, Wu M C, Tan H, et al. A genetic algorithm embedded with a concise chromosome representation for distributed and flexible job-shop scheduling problems[J]. *Journal of Intelligent Manufacturing*, 2018, 29(1): 19-34.
- [4] Wu M C, Lin C S, Lin C H, et al. Effects of different chromosome representations in developing genetic algorithms to solve DFJS scheduling problems[J]. *Computers & Operations Research*, 2017, 80: 101-112.
- [5] Marzouki B, Driss O B, Ghedira K. Decentralized tabu searches in multi-agent system for distributed and flexible job shop scheduling problem[C].

2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA). IEEE, 2017: 1019-1026.

- [6] Marzouki B, Driss O B, Ghedira K. Solving distributed and flexible job shop scheduling problem using a chemical reaction optimization metaheuristic[J]. *Procedia Computer Science*, 2018, 126: 1424-1433.
- [7] 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.
- [8] 吴秀丽, 刘夏晶. 差分进化算法求解分布式柔性作业车间调度问题[J]. *计算机集成制造系统*, 2019, 25(10): 2539-2558.
- [9] 孟磊磊, 张彪, 任亚平, 等. 求解分布式柔性作业车间调度的混合蛙跳算法[J]. *机械工程学报*, 2021, 57(17): 263-272.
- [10] Li X Y, Xie J, Ma Q J, et al. Improved gray wolf optimizer for distributed flexible job shop scheduling problem[J]. *Science China Technological Sciences*, 2022, 65(9): 2105-2115.
- [11] Meng L, Zhang C, Ren Y, et al. Mixed-integer linear programming and constraint programming formulations for solving distributed flexible job shop scheduling problem[J]. *Computers & Industrial Engineering*, 2020, 142: 106347.
- [12] Zhu J, Li X P, Wang Q. Complete local search with limited memory algorithm for no-wait job shops to minimize makespan[J]. *European Journal of Operational Research*, 2009, 198(2): 378-386.
- [13] Valente J M S, Gonçalves J F, Alves R A F S. A hybrid genetic algorithm for the early/tardy scheduling problem[J]. *Asia-Pacific Journal of Operational Research*, 2006, 23(3): 393-405.
- [14] Mastrolilli M, Gambardella L M. Effective neighbourhood functions for the flexible job shop problem[J]. *Journal of Scheduling*, 2000, 3(1): 3-20.

第一作者: 董博文, 男, 1996 年生, 硕士研究生, 研究方向为车间调度与智能优化算法。E-mail: dongbowen2016@126.com

通信作者: 王有远, 男, 1965 年生, 博士, 教授, 研究方向为智能制造与制造业信息化。E-mail: yyywnc@sina.com

(编辑 高扬)

(收修改稿日期: 2023-04-20)

文章编号: 20231227

如果您想发表对本文的看法, 请将文章编号填入读者意见调查表中的相应位置。