doi: 10.3969/j.issn.1005-3026.2014.09.026

求解带缓冲区和机器可用性约束的非置换流水车间调度

郑永前, 李 燕
(同济大学 机械与能源工程学院, 上海 201804)

摘 要: 为得到非置换流水车间更好的调度方案,考虑到缓冲区、机器可用性约束和序列相关换模时间,以最小化最大完工时间为目标,建立数学模型和析取图模型,构造了一种面向NPFS的列表启发式算法.算法通过允许列表和候选列表记录启发式过程信息,采用量子蚁群和SPT启发式规则搜索并选择析取边的可行移动方案,得到一个没有冲突的有向非循环图.通过正交试验法验证了算法关键参数,实例验证了算法求解和CPLEX的精确解相同.同时采用8组Demirkol测试问题,与MHD-ACS和ACO算法比较评估,验证了算法的有效性和鲁棒性.

关 键 词: 缓冲区;非置换;机器可用性;析取图;量子蚁群
中图分类号: TP 18 **文献标志码**: A **文章编号**: 1005-3026(2014)09-1329-07

Solution for Non-permutation Flow Shop Scheduling with Buffers and Machine Availability Constraints

ZHENG Yong-qian, *LI Yan*
(School of Mechanical Engineering, Tongji University, Shanghai 201804, China. Corresponding author: LI Yan, E-mail: ly.127@126.com)

Abstract: To get a better non-permutation flow shop (NPFS) scheduling, buffers, machine availability constraints and sequence-dependent setup time were considered in building the mathematical model and disjunctive graph model. A NPFS-oriented list heuristic algorithm was constructed to minimize the makespan. Allowed list and candidate list were built to record heuristic process information. Quantum ant colony and SPT heuristic rule were applied to search and select the feasible movement of related disjunction arcs to get an acyclic conjunctive graph. The algorithm parameters were verified through orthogonal test. The optimal disjunctive scheduling solution was the same with the exact CPLEX solution. The proposed algorithm has been critically compared and assessed by eight benchmark problems from Demirkol with the MHD-ACS and ACO ones, which confirms its good effectiveness and robustness.

Key words: buffers; non-permutation; machine availability; disjunctive graph; quantum ant colony

非置换流水车间(NPFS, non-permutation flow shop)调度中,为了满足工件间的交叉操作,机器间的缓冲区是必须的[1].当机器数 $m=3$ 时,置换流水车间(PFS, permutation flow shop)调度就已被证明是NPC(non-deterministic polynomial complete)组合优化问题,有 $n!$ 种调度方案,而NPFS则增加到 $(n!)^m$ 种.一方面,PFS的工件加工顺序相同,得到的调度方案通常较差.另一方面,缓冲区在制造系统中的广泛应用,使NPFS更有研究意义.

Liao和Ying等[2-4]对比研究了PFS和NPFS,并考虑了最小化延迟时间和工件流动时间,采用禁忌搜索和遗传算法得到了NPFS更低的下界,是目前为止最好的解.Lin和Ying则采

收稿日期: 2013-07-12
基金项目: 国家自然科学基金资助项目(71071115,61273035).
作者简介: 郑永前(1965-),男,河南商丘人,同济大学副教授.

用禁忌搜索、混合模拟退火[5],将 PFS 的解作为 NPFS 调度的初始解,构造了一种迭代贪婪启发式算法[6]和蚁群结合多种启发式算法[7]求解 NPFS 问题. Mehravaran 等[8]在优化供应链的 NPFS 调度中,考虑了序列相关换模时间和双标准. Rossi 等[9]则采用析取图法提出了一种本地启发式算法求解 NPFS 调度问题.

可见,目前研究 NPFS 主要集中在对元启发式算法的改进和对目标函数的讨论,缺少对问题模型的深入研究. 另一方面,现有 NPFS 模型多根据传统 PFS 模型释放工件加工顺序约束,或直接采用 PFS 模型,将 PFS 调度结果作为初始解并通过改变工件加工顺序得到 NPFS 解. 该方法的优点为求解快捷,能直观比较改进效果,但是缺少对 NPFS 问题本身的研究和缓冲区等需求的考虑,而 PFS 初始解对求解质量有较大影响,无法比较改进算法的有效性和鲁棒性等.

本文针对带缓冲区需求和机器可用性约束的 NPFS 调度问题,建立了混合整数线性规划模型,采用析取图构造了一种面向问题的列表启发式算法. 通过量子蚁群在析取图模型中搜索 SPT 规则,得到调度方案,并采用正交实验法对算法的参数进行分析,通过 Demirkol 基准问题测试算法性能.

1 数学模型

工件 $i \in \{1,2,\cdots,n\}$,在机器 $j \in \{M_1, M_2, \cdots, M_m\}$ 上加工,其中在 M_j 和 M_{j+1} $(j=1,2,\cdots,m-1)$ 之间存在 $m-1$ 个缓冲区 B_j,缓冲区大小为 b_j. 且当机器 $j=m$ 时,缓冲区大小 $b_j=n$,即输出缓冲区满足所有的工件量需求. 工件加工时间 $p_{i,j}$,释放时间 r_i,交货期 d_i,序列相关换模时间为 $\mathrm{Set}_{i,k,j}$,表示在机器 j 上工件 i 在工件 k 前加工时的换模时间. 同时,对每个机器存在可用时间 a_t. 如果工件 i 跳过机器 j, $f_{i,j}=0$;反之,若工件 i 在机器 j 上加工, $f_{i,j}=1$. 每个工件由 m 个操作 $O_{i,j}$ 组成,且操作 $O_{i,j}$ 在机器 M_j 上加工满足非置换约束. 每个工件都存在一个加工关系,可用加工路径表示:$O_{i,1} \rightarrow O_{i,2} \rightarrow \cdots \rightarrow O_{i,m}$.

工件 i 在机器 j 上的开始加工时间为 $S_{i,j}$,完工时间为 $C_{i,j}$,二进制变量 $Y_{i,k,j}$ 为 1 时表示在机器 j 上工件 i 在工件 k 前加工,反之则为 0. 机器非抢占,且工件加工过程不允许中断. 建立 MILP 模型如下:

$$\min C_{\max}, \tag{1}$$

s. t.

$$S_{i,j} \geqslant a_t f_{i,j}, i=1,2,\cdots,n, j=1,2,\cdots,m; \tag{2}$$

$$S_{i,j} \geqslant r_i f_{i,j} - \mathrm{Set}_{0,i,j} Y_{0,i,j}, i=1,2,\cdots,n, j=1,2,\cdots,m; \tag{3}$$

$$S_{i,j} \leqslant M f_{i,j}, i=1,2,\cdots,n, j=1,2,\cdots,m; \tag{4}$$

$$S_{i,j} - C_{k,j} + M(1 - Y_{i,k,j}) \geqslant 0, i,k=1,2,\cdots,n, i<k, j=1,2,\cdots,m; \tag{5}$$

$$S_{k,j} - C_{i,j} + M(1 - Y_{k,i,j}) \geqslant 0, i,k=1,2,\cdots,n, k<i, j=1,2,\cdots,m; \tag{6}$$

$$C_{i,j} \geqslant S_{i,j} + f_{i,j} p_{i,j} + \mathrm{Set}_{0,i,j} Y_{0,i,j}, i=1,2,\cdots,n, j=1,2,\cdots,m; \tag{7}$$

$$C_{i,j} - S_{i,j} + M(1 - Y_{i,k,j}) \geqslant f_{i,j} p_{i,j}, i,k=1,2,\cdots,n, i<k, j=1,2,\cdots,m; \tag{8}$$

$$C_{k,j} - S_{k,j} + M(1 - Y_{k,i,j}) \geqslant f_{k,j} p_{k,j}, i,k=1,2,\cdots,n, k<i, j=1,2,\cdots,m; \tag{9}$$

$$C_{i,j} \leqslant M f_{i,j}, i=,1,2,\cdots,n, j=1,2,\cdots,m; \tag{10}$$

$$\sum_{i=1,k\neq i}^{n} Y_{i,k,j} \leqslant 1, k=0,1,2,\cdots,n, j=1,2,\cdots,m; \tag{11}$$

$$\sum_{k=0,k\neq i}^{n} Y_{i,k,j} \leqslant f_{i,j}, i=1,2,\cdots,n, j=1,2,\cdots,m. \tag{12}$$

目标函数(1)表示最小化最大完工时间. 约束式(2)~式(4)计算当工件 i 在机器 j 上第一个加工时的开始时间,式(2)表示在工件释放后机器可用或机器可用后工件释放,但释放时间在该工件换模完成前的情况下,工件的开始加工时间为 a_t;式(3)表示工件在换模完成后释放,机器闲置,$r_i - \mathrm{Set}_{0,i,j}$ 表示工件最迟开始时间;式(4)保证了工件 i 跳过机器 j 时的开始加工时间为 0;式(5)~式(6)保证了在同一时间机器不能同时加工 2 个工件,且计算了工件(非第一个工件)的开始加工时间;式(7)~式(10)计算工件的完成时间,式(7)表示第一个加工工件 i 的完工时间;式(8)~式(9)表示非第一个工件的完工时间至少等于工件的开始加工时间加上加工时间;式(10)保证了当工件 i 跳过机器 j 时的完工时间为 0;式(11)保证了机器一次只能加工一个工件;式(12)保证了工件在加工顺序中的位置唯一.

2 带缓冲区的 NPFS 析取图模型

对 n 个工件 m 台机器的 NPFS 可用图 1 所示的析取图来描述.

虚拟节点 S,T 是两个虚拟操作,表示加工开始和结束. 合取弧(箭头线)表示 n 个工件在 m 台机器上从开始到结束的加工路径;析取边(虚线)表示同一机器上加工各个操作的链接.

析取图 $G=(V,A,E,W)$，其中：V 为所有加工操作构成的顶点集；A 为 n 条合取弧构成的弧集；E 为 m 条析取边构成的边集；W 为节点权重，表示工件加工时间和换模时间，$W(O_{i,j})=p_{i,j}+$

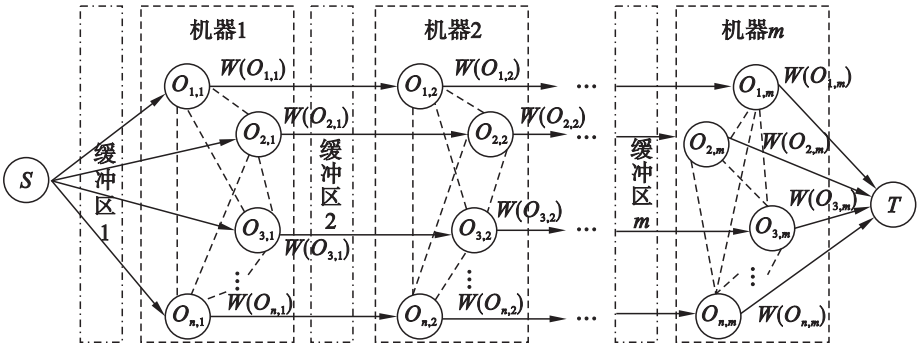

图 1　带缓冲区的 NPFS 调度析取图模型

Fig. 1　The disjunctive graph model of NPFS scheduling with buffers

$\mathrm{Set}_{i,j}$. 对 NPFS 的求解归结为找到各弧上优先决策的操作的一组先后顺序，选择析取边，从而得到一个各操作之间没有冲突的有向非循环图，其关键路径长度即为最大完成时间.

3　启发式算法构造

3.1　基于 NPFS 析取图的启发式算法

算法通过计算每个节点的完工时间，采用允许列表和候选列表记录启发式算法过程的信息，得到一个可行的调度方案. 具体步骤如下：

1）输入 NPFS 调度析取图 $G=(V,A,E,W)$；2）初始化候选节点，为当前操作 $O_{1,j}$ 建立允许列表 $\mathrm{AL}_{i,j}$，将连接该操作的前后节点加入到允许列表；3）通过非延迟调度优化准则限制允许列表，对限制的允许列表建立候选列表 $\mathrm{CL}_{i,j}$；4）初始化可行移动方案，将机器 j 的候选列表 $\mathrm{CL}_{i,j}$ 的最后一个操作 $O_{i,j}$ 的析取边 E_j 标记为可行移动；5）选择 E_j 的下一个可行移动；6）删除操作 $O_{i,j}$ 的所有剩余的析取边；7）权重转移，将操作节点 $O_{i,j}$ 的权重 $W(O_{i,j})$ 转移到机器 j 的下一个节点和工件 i 的加工路径，计算完工时间；8）删除操作 $O_{i,j}$，更新析取图结构；9）对下一个操作，转入步骤 2），直到访问完所有的操作；10）输出 NPFS 调度方案.

节点 S 是所有调度方案的起点，采用 SPT 启发式规则随机选择可行移动方案，通过将权重转移到同一个工件的加工路径和同一台机器的加工顺序，保证最终虚拟节点 T 的权重最大，得到关键路径.

3.2　量子蚁群算法

量子蚁群算法利用蚂蚁寻优和量子计算原理实现优化，能有效提高算法的搜索效率和效果. 为适于求解 NPFS 问题，对算法中的参数进行重新定义. 设蚂蚁 a 的相邻节点的状态转移概率为

$$p_{i,j}^{a}=\begin{cases}\dfrac{[\tau_{i,j}{}^{a}][\eta_{i,u}]^{\beta}v_{i,j}^{2}}{\sum_{u\in\mathrm{allowed}_a}[\tau_{i,j}]^{a}[\eta_{i,u}]^{\beta}v_{i,j}^{2}}, & \text{若}j\in\mathrm{allowed}_a;\\ 0, & \text{其他}.\end{cases}\tag{13}$$

其中候选解集合 $\mathrm{allowed}_a=\{0,1,\cdots,n-1\}$，表示第 a 个蚂蚁下一步可选节点的集合；tabu_a 表示禁忌表集合，用来存放蚂蚁 a 已经过的节点；α 和 β 分别表示蚂蚁在运动过程中所累积的信息及启发式因子对蚂蚁决策的影响，通常 $0\leqslant\alpha,\beta\leqslant5$；$v_{i,j}$ 表示边弧 (i,j) 的量子信息强度. 在 NPFS 中，可行顶点集就是当前所有工件未完成的第一道工序的集合，启发式信息按照工件排序完工时间进行选取，以加工时间的倒数作为启发信息，即 $n_{i,j}=1/p_{i,j}$.

信息素强度的更新如下：

$$\tau_{i,j}^{\mathrm{new}}=\rho\tau_{i,j}^{\mathrm{old}}+\sum{}_{a}\Delta\tau_{i,j}^{a}.\tag{14}$$

其中：ρ 为信息素残留系数；$\Delta\tau_{i,j}^{a}$ 为蚂蚁 a 在边弧 (i,j) 上留下的单位长度轨迹信息素数量，则：

$$\Delta\tau_{i,j}^{a}=\begin{cases}\dfrac{Q}{p_{i,j}},\text{若}(i,j)\text{在最优路径上};\\ 0,\text{其他}.\end{cases}\tag{15}$$

其中 Q 是体现蚂蚁所留下轨迹数量的一个常数，通常取 $1\leqslant Q\leqslant10\ 000$. 采用量子位进行编码，将蚂蚁经过路径的信息素强度的增量转变为量子旋转门旋转角. 量子非门实现蚂蚁变异，以避免算法出现早熟收敛. 量子信息更新公式如下：

$$\begin{bmatrix}u'_{i,j}\\ v'_{i,j}\end{bmatrix}=\begin{bmatrix}\cos(\theta_{i,j}) & -\sin(\theta_{i,j})\\ \sin(\theta_{i,j}) & \cos(\theta_{i,j})\end{bmatrix}\begin{bmatrix}u_{i,j}\\ v_{i,j}\end{bmatrix}.\tag{16}$$

其中 $\theta_{i,j}$ 为旋转角,其正负决定着算法收敛的方向,幅度决定算法收敛的速度和效率,$\theta_{i,j}$ 的更新使用如下的动态变化策略:

$$\theta_{i,j}=\Delta\theta\times\mathrm{sign}(u_{i,j}\times v_{i,j}),i=1,2,\cdots,m. \quad (17)$$

蚂蚁按照状态转移概率进行移动,计算蚂蚁目标函数值,找到当前最优解,然后修改轨迹强度和量子信息,经过数次迭代后,输出当前最优解.

3.3 面向 NPFS 的列表启发式算法描述

采用析取图 $G=(V,A,E,W,\Gamma)$ 来表示流水车间模型,Γ 表示析取弧相关的信息素分布. 蚂蚁在图 G 中搜索,按信息素踪迹及启发信息的指引遍历图中所有的节点,将问题转换为在析取图中寻找最佳路径的问题. 算法流程如下:

1) 设置算法参数,生成析取图;2) 初始化候选节点;3) 把蚂蚁 a 放到候选节点,把候选节点放入 tabu_a;4) 建立允许列表,根据状态转移概率公式(13),确定蚂蚁 a 的下一步所走的路径;5) 建立候选列表,将蚂蚁 a 放到所选节点,并将该节点放入蚂蚁 a 的 tabu_a,更新候选点集合 $\mathrm{allowed}_a$;6)初始化可行移动方案;7)选择可行移动方案,蚂蚁 a 走完所有路径,计算各蚂蚁的目标函数值,记录当前最好解,按式(14)修改轨迹信息素强度,按式(16)更新量子信息;8)当蚁群算法循环次数达到最大次数且无退化解,删除析取边;否则,转入步骤 3);9)转移权重,更新析取图结构;10)访问完所有操作节点,输出调度方案;否则,转入步骤 2).

4 试验与算法比较

4.1 算法参数分析

由于量子蚁群算法的参数会影响搜索空间和搜索过程,本文采用正交试验法,测试算法的 4 个因素:信息素衰减系数 α、启发式信息权重 β、信息素挥发系数 ρ 和信息素总量 Q. 以 Ta005 问题为例,采用四因素三水平正交表 $L_9(3^4)$,如表 1 所示.

表 1 因素水平正交表

Table 1 Orthogonal table of factors levels

水平	因素			
	A(α)	B(β)	C(ρ)	D(Q)
1	0.2	0.0	0.1	50
2	0.5	1.0	0.2	100
3	0.8	2.0	0.4	200

采用信噪比公式 $S/N=-10\lg[\frac{1}{n}\sum_{i=1}^{n}y_i^2]$ 计算分析,y_i 为经过 n 次参数组合得到最小的最大完工时间,每个参数值对应 9 次试验,得到最好的方案为 $\alpha=0.8,\beta=2,\rho=0.1,Q=100$,如表 2 所示.

表 2 正交试验数据分析表

Table 2 Data analysis of orthogonal test

水平	A	B	C	D
1	-91.970 9	-91.271 2	-91.128 7	-91.306 1
2	-91.935 7	-91.253 3	-91.280 9	-91.200 7
3	-91.850 2	-91.199 0	-91.206 3	-91.208 9
极差	0.120 7	0.072 2	0.152 2	0.105 4
最优	A3	B3	C1	D2

4.2 算例描述

仿真环境为 Pentium IV 3.0 GHz,RAM 2GB,Windows7 操作系统,Matlab7.1 编译软件. 参数设置:蚂蚁数目为 8;最大迭代次数 $T_{\max}=200$;$\alpha=0.8,\beta=2,\rho=0.1,Q=100$;算法独立运行 20 次. 设 5 工件 4 机器的 NPFS 调度中,工件 1 跳过机器 2 和机器 4,工件信息如表 3 ~ 表 4 所示.

表 3 工件加工信息表

Table 3 Job process information table

i	M_1	M_2	M_3	M_4	r_i	d_i
	$a_t(72)$	$a_t(105)$	$a_t(139)$	$a_t(168)$		
1	32	0	35	0	46	216
2	25	40	37	26	40	171
3	34	28	26	33	25	243
4	36	24	39	27	8	170
5	37	38	35	31	2	281

表 4 工件序列相关换模时间

Table 4 Job sequence-dependent setup time

i	M_1					M_2					M_3					M_4				
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	0	32	35	11	10	0	0	0	0	0	0	12	13	25	37	0	0	0	0	0
2	24	0	29	25	27	0	0	16	28	30	9	0	5	13	20	0	0	30	35	23
3	15	30	0	27	38	0	3	0	39	28	40	16	0	34	22	0	10	0	19	28
4	6	25	10	0	22	0	13	37	0	10	2	2	1	0	31	0	26	18	0	23
5	15	6	23	36	0	0	22	31	15	0	3	11	13	17	0	0	31	30	32	0
R	13	4	3	10	8	0	14	16	11	25	32	4	40	31	27	0	7	16	27	40

表 4 中 R 行表示工件是机器上第一个加工工件时的换模时间. 采用本文算法,调度结果为机器 1 中 J4 – J1 – J3 – J5 – J2,机器 2 中 J3 – J1 – J4 – J5 – J2,机器 3 中 J3 – J1 – J4 – J5 – J2,机器 4 中 J2 – J1 – J4 – J5 – J3,不同的虚线表示不同机器上的候选路径,且工件 1 跳过机器 2、机器 4,图中相对操作 $O_{1,2}$,$O_{1,4}$用虚线表示,且加工时间为 0. 设置缓冲区内初始存储的工件 $B_1 = 4,1,3$, $B_2 = 3$, $B_3 = 3,1$, $B_4 = 2,4,5$,结余的缓冲区空间为空. 调度结果析取图如图 2 所示.

$J4M_1 - J3M_2 - J1M_1 - J3M_3 - J4M_2 - J3M_1 - J2M_4 - J4M_4 - J1M_3 - J5M_2 - J5M_1 - J2M_2 - J5M_4 - J4M_3 - J3M_4 - J2M_1 - J5M_3 - J2M_3$ 为最终加工序列,最大完工时间最优值为2 844.

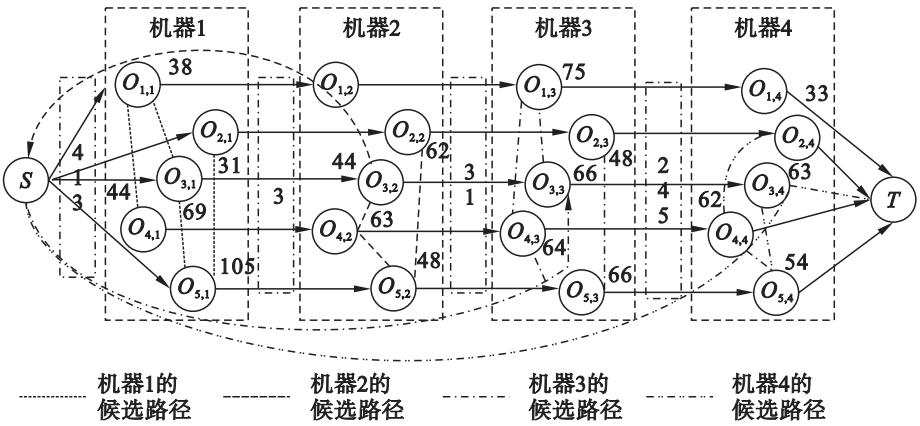

图 2　调度结果析取图

Fig. 2　Disjunctive graph for scheduling result

具体加工开始时间和完成时间如表 5 所示.

同时采用 ILOG CPLEX Optimization Studio. (2009)计算的优化结果也是 2 844. 本文提出的面向 NPFS 的启发式算法能有效解决带缓冲区和机器可用性约束,考虑工件序列相关换模时间的 NPFS 问题,并给出具体的调度方案.

表 5　工件加工调度时间表

Table 5　Job processing and dispatching schedule

M	ST	CT	ST	CT	ST	CT	ST	CT	ST	CT
M_1	J4		J1		J3		J5		J2	
	72	116	116	154	154	223	223	328	328	359
M_2	J3		J1		J4		J5		J2	
	105	149	–	–	149	212	212	260	260	322
M_3	J3		J1		J4		J5		J2	
	139	205	205	280	280	344	344	410	410	458
M_4	J2		J1		J4		J5		J3	
	168	201	–	–	201	263	263	317	317	380

4.3　算法对比与分析

采用 Demirkol 问题比较算法有效性,将已知最优解作为问题的上界(UB),将通过松弛(m – 1)台机器的能力约束的单机调度问题,得到的解作为下界(LB). 松弛不同的机器得到的下界不同,用 k 表示. 根据 $\mathrm{Gap}_{n,m,k} = (\mathrm{UB}_{n,m,k} - \mathrm{LB}_{n,m,k})/\mathrm{LB}_{n,m,k}$,将降序排列得到每组问题的第一个算例作为本文的基准问题. 评价指标为:1)算法得到的最优值和相应的计算时间;2)算法得到的平均最优解和已知最优解的相对偏差,如式(18)所示:

$$\%\ \mathrm{Gap}_{\mathrm{avg}} = 100\,\frac{\sum_{i=1}^{10} \mathrm{C}_{\mathrm{avg}} - \sum_{i=1}^{10} \mathrm{UB}}{\sum_{i=1}^{10} \mathrm{UB}}. \qquad (18)$$

通过正交试验对每个算例进行 10 次仿真统计,平均最优值如表 6 所示.

根据表 6 得到面向 NPFS 的启发式算法(N – QACO)的参数,蚂蚁数目为 $n \times m/5$,最大迭代次数为 $T_{\max} = 500$, $\Delta\theta = 0.05\pi$,算法独立运行 10 次. 对比算法采用 Ying 等[7]提出的 MHD – ACS 算法和蚁群优化(ACO)算法. 记录得到的最

表 6 通过正交试验得到的最优参数设置

Table 6 The optimal parameters through orthogonal test

问题	α	β	ρ	Q
flcmax_20_15_3	0.2	2.0	0.10	100
flcmax_20_20_1	0.1	2.2	0.12	100
flcmax_30_15_3	0.3	2.2	0.10	150
flcmax_30_20_3	0.1	2.0	0.12	100
flcmax_40_15_5	0.2	2.0	0.12	150
flcmax_40_20_3	0.1	2.3	0.12	100
flcmax_50_15_6	0.2	2.0	0.15	100
flcmax_50_20_2	0.2	2.0	0.12	100

优值和计算最优值所用的时间,具体如表 7 所示.

可见采用 N – QACO 算法除 flcmax_40_20_3 问题,得到的解均不劣于 MHD – ACS 和 ACO 算法,所有的解均低于上界(UB),具有较高的求解质量,如表中黑体数据所示,明显比 ACO 算法改进了求解结果和速度. 由于计算时间受硬件和软件的影响,并不能据此比较算法的有效性,但对于大规模问题的求解,N – QACO 算法仍能在合理的时间消耗内得到优质的解,说明了算法在实际问题中的可应用性.

N – QACO 算法对所有算例的平均相对偏差为 – 4.35% ,最优改进偏差为 – 8.10% ,均优于 MHD – ACS 的 – 0.62% 和 – 3.2% ,如图 3 所示.

表 7 算例计算结果和对比

Table 7 Comparison and analysis of benchmark problem

问题	基准问题			MHD – ACS		ACO		N – QACO		
	LB	UB	Time/s	C_{best}	Time/s	C_{best}	Time/s	C_{best}	Time/s	C_{avg}
flcmax_20_15_3	3 354	4 437	69.93	4 420	46	4 945	430	**4 077**	241	4 134.1
flcmax_20_20_1	3 776	4 821	159.12	4 819	59	5 029	282	**4 802**	29	4 929.6
flcmax_30_15_3	4 020	5 226	148.99	5 210	93	5 378	535	**4 973**	402	5 110.3
flcmax_30_20_3	4806	6 183	0.24	5 987	121	6 639	261	**5 719**	98	6 048.9
flcmax_40_15_5	5 560	6 986	155.97	6 972	154	7 219	763	**6 625**	167	6 692.3
flcmax_40_20_3	5 693	7 154	615.49	**7 132**	210	7 645	1314	7 151	24	7 304.5
flcmax_50_15_6	6 290	7 673	313.15	7 631	238	8 988	1266	**7 458**	131	7 603.1
flcmax_50_20_2	6 740	8 838	798.83	8 836	312	10 421	1507	**8 307**	697	8 484.4

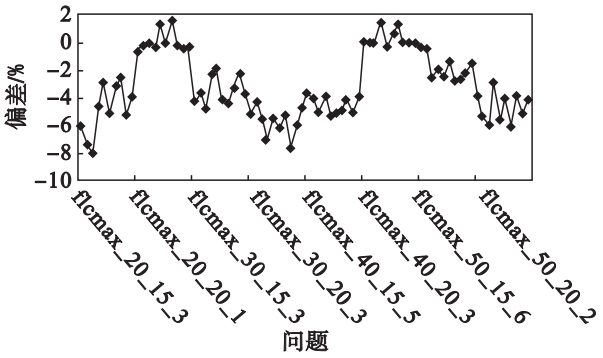

图 3 N – QACO 算法改进偏差

Fig. 3 Improvement deviation of N-QACO

在这 80 个算例解中 74 个低于上界,达到最优解的次数约为 92.5% ,说明 N – QACO 求解的鲁棒性,求解标准差为 0.010 1 ~ 0.054 5,平均标准差为 0.024 0,说明算法具有一定的稳定性. 当工件数相同,机器数从 15 台增加至 20 台时,标准差相应增加. 可见,算法得到解的稳定性受到操作复杂程度的影响,在工件数相同时,随机器数增加稳定性相对减弱.

5 结 论

本文针对 NPFS 调度问题,考虑了缓冲区,机器可用性约束和序列相关换模时间,以最小化最大完工时间为目标,采用 SPT 启发式规则和量子蚁群算法,构造了一种面向 NPFS 的列表启发式算法——N – QACO 算法,改进求解效率和质量. 采用正交试验法,测试并分析了算法参数. 通过实例验证了 N – QACO 算法的有效性. 并通过 8 组 Demirkol 问题,和 MHD – ACS,ACO 算法比较发现,N – QACO 在解的质量、求解速度和鲁棒性上均有一定的提高,并分析了算法稳定性和问题规模的关系. 在解决现实问题,尤其是大规模问题上,N – QACO 算法能在合理的计算时间成本内得到相对优质的解,并为非置换流水车间调度的进一步研究提供了方向.

(下转第 1345 页)