基于启发式方法对车间调度问题的最优化方法

刘洪毅

摘要：调度问题是一类经典的 NP-hard 问题，本文将用三种不方法——启发式规则，元启发，超启发规则进行探讨问题的较优解。本文启发式规则使用先到先服务（FIFC）+最短工件用时（SRPT）共同决策，直接通过启发式规则得到加工工序序列。本文元启发基于粒子群算法（PSO），并加入由启发式规则得到的工序序列作为初始粒子之一，优化初始解。本文超启发规则基于遗传算法（GA），并加入了不同染色体有自适应的变异，交叉概率，以提高染色体存活概率和解的质量。

关键词：车间调度问题 启发式规则 元启发规则 粒子群算法 遗传算法 超启发规则

1. 引言

车间调度问题（Job-shop Scheduling Problem, JSP）是一类典型的生产调度问题，具有很强的工程背景，许多实际工程问题均可与之相转化。同时，Job Shop 调度问题也是非常难的 NP-hard（Non-Deterministic Polynomial）问题，因此卡法求解 Job Shop 调度问题的有效算法一直是调度和优化领域的重要课题。目前研究 Job Shop 调度问题的方法包括神经网络、禁忌搜索、启发式算法、登山算法、模拟退火、粒子群算法、蚁群算法、遗传算法以及超启发算法等算法。

虽然对于车间调度问题的研究已经有几十年的历史，提出了许多最优化求解方法，但由于其本身的复杂性，至今尚未形成系统的理论与方法。

启发式规则（heuristics）是一种经验规则，它是基于有限的知识（或“不完整的信息”）在短时间内找到问题较优解决方案的一种技术。其优点是能够短时间内找到较优解，一般启发式规则比较简单，明了易于实现。其缺点是解决方案有可能会偏离最佳方案，不同的启发式规则适应不同的环境，目前没有一种启发式规则适合全部环境。本文以先到先服务（FIFC）及最短加工时间（SRPT）为启发式规则，对启发式规则进行测试。

元启发算法（meta-heuristic）是一种利用随机数搜索的搜索算法，是一种通用寻找较优解的方法。基于仿生学，有很多种元启发算法,例如：登山算法、模拟退火、蚁群算法、遗传算法、粒子群算法等。其优点：使用环境广泛，搜索结果最优。其缺点：搜索所用时间较长。本文基于 PSO 算法（粒子群算法）实现元启发算法。

超启发算法（hyper-heuristic）是一种新兴算法，其基于前两者的优点，以元启发框架搜索适于当前环境的启发式规则组合（不同的机器上的启发式规则可能不同）。其优点：其搜索速度快于元启发算法，且基于不同的环境搜索出来的启发式规则组合是不同的，有其自适应的启发式规则。其缺点：搜索结果并不如元启发算法得到的结果好。本文基于 GA（遗传算法）框架，及 FIFC+SPT（最短已加工时间优先）、FIFC+SRPT、FIFC+TIS（最长等待时间优先）三种启发

式规则实现超启发算法。

2. 问题描述及模型建立

2.1 问题描述

典型的 Job Shop 调度问题可描述为：n 个工件和 m 台机器上加工，事先给定各工件在各台机器上的加工工序，和各工件在当前工序在当前机器上的加工时间。要求确定各机器上所有工件的加工次序，以使某些加工性能指标（如总加工时间）最优。

最典型的 Job Shop 问题一般会有以下约束条件：

1. 同一时刻每台机器只能加工一个工序，且每个工序只能被一台机器加工，且加工过程不可中断。

2. 在整个加工过程中，每个工件不能在同一台机器上加工多次。

3. 各工件必须按照工艺路线以指定的次序在机器上加工，工件 i 的第 j 道工序必须在第$(j-1)$道工序完成后才能开始。

4. 不考虑工件的优先权，允许操作等待。

5. 除非特殊说明，工件的加工时间事先给定，且在整个加工过程中保持不变。

2.2 数学模型

设车间作业系统中：

1. 有n个工件： $J=(j_1,j_2,j_3,\dots j_n)$，$j_i$ 为第 i个工件。
2. 有m台机器：$M=(m_1,m_2,\dots m_m)$ m_j 为第j台机器。
3. 各工件各工序对应机器信息:$S=(S_1,S_2,\dots S_n)^T$,S_i为第i个工件加工工序所需机器。$S_i=(s_{i1},s_{i2},\dots s_{im})$,$s_{ik}$为第$i$个工件第$k$个工序所需的机器。

4. 各工件各工序对应时间信息$T=(T_1,T_2,\dots T_n)^T$,T_i为第i个工件加工所需要时间。$T_i=(t_{i1},t_{i2},\dots,t_{ik})$,$t_{ik}$为第$i$个工件第k个工序所需加工时间。

2.3 约束条件

- $start_{ik}+t_{ik}\le start_{i(k+1)}$
- $t_{ik}\ge 0$,
- $1\le i\le n$
- $1\le j,k\le m$

其中$start_{ij}$表示第i个工件第k个工序的开始时间

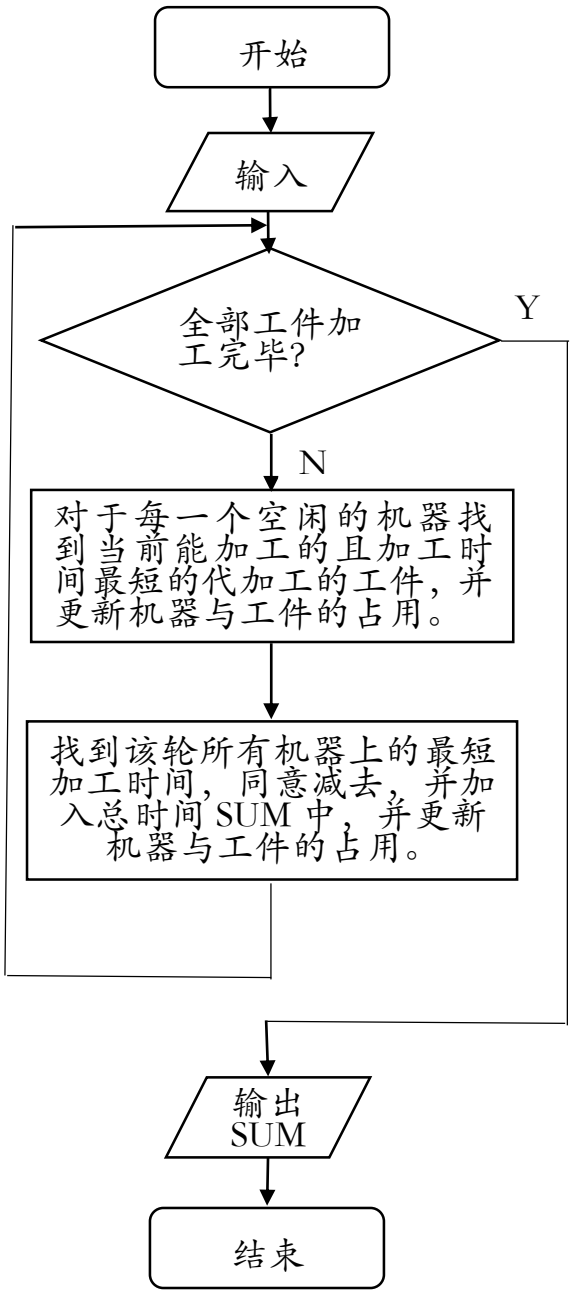
2.4 目标函数

车间调度问题优化目标函数为最小化最大完工时间，即

$$T=\min\{\max(T_1,T_2,\dots T_n)\};$$

3. FIFC+SRPT 启发式规则

3.1 流程图



3.2 算法核心

该启发式规则每一轮会对所有空闲的机器进行更新，当某一空闲机器M_o有多个工件可以选择时，例如：$J_1, J_2, \ldots J_r$，M_o会选择其中所需加工时间最短的工件进行加工。

3.3 优缺点分析

优点：

- 时间复杂度低。
- 逻辑简单，易于实现。

缺点：

- 通用性差，不能适应所有的车间调度问题。

4. PSO 粒子群算法

粒子群优化算法（PSO）是一种基于群体智能理论的优化算法，由 J. Kennedy 和 R. Eberhart 在 1995 年提出的，该算法通过模拟鸟类群体调整自身飞行速度和飞行方向，将所有个体移动到适应度好的环境中，从而抽象出一种可以求解具有复杂解空间性质问题的优化算法. 同时在进化过程中 该算法保留位置X与速度V上的信息，其概念和参数调整简单而且容易编程实现，它既保持传统进化算法深刻的群体智慧背景，同时又有自己许多良好的优化性能。

4.1 编码

传统 PSO 是面向连续变化问题的，然而车间调度问题的工序是离散化的问题，所以需要建立函数映射，将空间搜索上的连续变化映射为离散值。

本文利用PPP[1]编码方式，即基于粒子坐标位置排列编码。

例如，对于 3 个工件 3 台机器加工排序问题，表 1 给出了粒子位置对应 JSP 问题解的表达形式。表 1 中粒子位置最小的坐标值对应数字 1，第二小的坐标值对应数字 2，依次类推。如表 1，根据粒子

位置$X_i = (2.40,1.71,-0.89,3.10,-2.34,-1.20,2.45,0.26,1.20)$，我们得到这样一组排列$Rank = (7,6,3,9,1,2,8,4,5)$。粒子$X_i$映射为唯一的一组排列，然后根据这个排列映射出唯一工序序列。我们定义在$Rank$中越小的值对应的工件序号越小，及前m小个$Rank$值对应 1 号工件，第二m小个对应 2 号工件，然后我们得到加工工件序列$OrderList = (3,2,1,3,1,1,3,2,2)$这种解的表示方法使得粒子中每一位置坐标唯一一个工序序列。这样就能使对粒子产生正反馈，使解越来越好。

表格 1 JSP 问题中的粒子表示（编码）

粒子维数	1	2	3	4	5	6	7	8	9
粒子位置	2.40	1.71	-0.89	3.10	-2.34	-1.20	2.45	0.26	1.20
工件排列	7	6	3	9	1	2	8	4	5
工件序列	3	2	1	3	1	1	3	2	2

4.2 数学描述

在n维空间中有N个粒子组成的种群即

$$Swarm = (X_1, X_2, \dots, X_N)^T;$$

每个粒子坐标为

$$X_i = \left(X_{i1}, X_{i2}, \dots, X_{i(n*m)}\right)$$

同时n维空间中有与其对应的N个速度

$$Speed = (V_1, V_2, \dots V_N)^T;$$

对应的每个粒子的速度：

$$V_i = \left(V_{i1}, V_{i2}, \dots, V_{i(n*m)}\right);$$

$Pbest$（局部最优解）记录每个粒子所经过的地方中的最优解：

$$Pbest = (P_1, P_2, \dots, P_N);$$

$$P_i = (P_{i1}, P_{i2}, \dots, P_{i(n*m)});$$

$Gbest$（全局最优解）记录全部粒子所经过的地方中的最优解：

$$Gbest = (G_1, G_2, \dots, G_{n*m});$$

注：由上面的编码方式可知每个粒子的长度为$n * m$;

4.3 算法描述

对于每一次迭代，利用粒子自身的$Pbest$和全局$Gbest$来更新粒子自身速度V，然后用自身速度V来更新自身位置X。

更新公式：

$$V_{ij}(t+1) = \omega V_{ij}(t) + c_1 r_1\left(P_{ij}(t) - X_{ij}(t)\right) + c_2 r_2\left(G_j(t) - X_{ij}(t)\right); \quad (1)$$

$$X_{ij}(t+1) = X_{ij}(t) + V_{ij}(t+1); \quad (2)$$

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{t_{\max}} * \mathrm{t}; \quad (3)$$

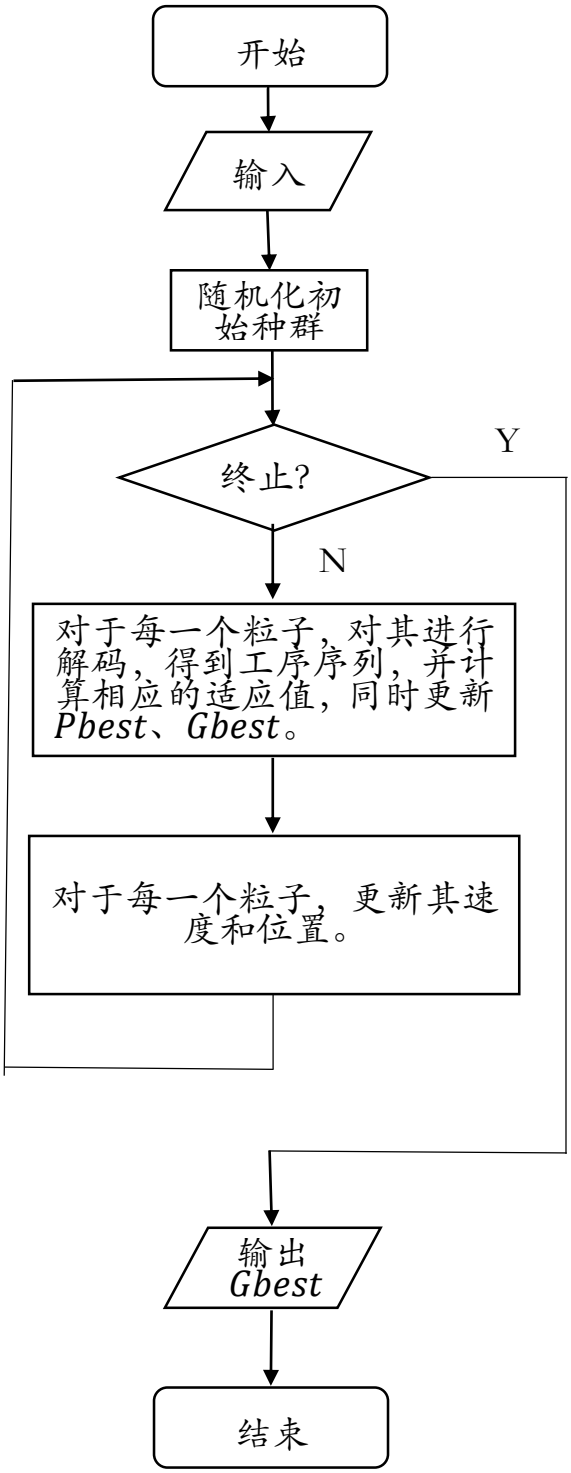
如果$V_{ij} > V_{max}$，则$V_{ij} = V_{max}$；

如果$V_{ij} < -V_{max}$，则$V_{ij} = -V_{max}$；

注：式中ω表示粒子惯性权重，ω_{max}、ω_{min}为惯性权重的初始值和最终值，t_{max}是最大迭代次数，i表示第i个粒子，j表示第j个维度，c_1, c_2都是正常数，称为加速系数，r_1, r_2是两个在$[0,1]$范围内变化的随机数。V_{max}是常数，限制了速度的最大值。当V_{max}较大时，粒子飞行速度较大，有利于全局搜索，但有可能飞过最优解；当V_{max}较小时，粒子可以再较小区域内进行局部精细搜索，但容易陷入局部最优解。

在本文计算实例中$\omega_{max} = 1.2$，$\omega_{min} = 0.4$，加速系数 c_1, c_2均取 2。

4.4 流程图



注：

终止条件：迭代次数达到上限，或者解的精度达到要求。

适应值：相应工序的加工时间。

优化初始解：将其中一个初始解，赋值为由启发式规则得到的工序序列。

4.5 优缺点分析

优点：

- 通用性强，可以适应各种问题环境
- 公式简单。

缺点：

- 容易收敛过早，陷入局部最优。
- 代码实现复杂。

5. HHGA（超启发遗传算法）

超启发式算法作为最新的、最有前景的组合优化算法，采用高层用元启发算法搜索低层启发式规则的方法，兼并寻优能力于计算效率。本文使用 GA（遗传算法）为高层元启发框架，使用 4 个低层规则。

5.1 数学描述

作为群智能算法的一种，也像其他的一样，存在并行搜索，不断优化结果的核心思想。这里的种群时**染色体**。染色体上每一个基因的编码为某个机器M_i所对应启发式规则。染色体的基本操作有**交叉、突变、计算自适应值、计算加工所用时间**。

加工所用时间 $UsingTime$：由该染色体决定的加工时间。

自适应值 F：是一个[0,1]的常数，与加工所用时间成反关系。

匹配集 $MatchSet$：选择自适应值前 N 的染色体进入匹配集。进入匹配集代表将会在下一代染色体出现，该染色体较为适应当前环境。

交叉集 $CrossSet$：对于每一个染色体，基于自身交叉概率通过轮盘赌的方式判断是否进行交叉操作。将即将进行交叉操作的染色体放入交叉集。

轮盘赌：随机产生$Random(0,1)$，如果$R < P$则选中该目标。

设有N个染色体：

$$Chromosome = (C_1, C_2, \dots, C_N);$$

每个染色体上有m个基因，对应m台机器上的启发式规则：

$$C_i = (C_{i1}, C_{i2}, \dots, C_{im});$$

每个染色体有与其对应的交叉概率：

$$P_c = (P_{c1}, P_{c2}, \dots, P_{cN});$$

每个染色体有与其对应的突变概率：

$$P_m = (P_{m1}, P_{m2}, \dots, P_{mn});$$

记录最优解：

$$Best;$$

5.2 算法描述

适应度计算：

$$F(i) = 1 - \frac{UsingTime(i)}{UsingTime_{max}}$$

自适应遗传算法交叉、突变率计算公式：

设前一代和新生的一代一共有Z个染色体：

$$f_{max} = \max(f_1, f_2, \dots f_z);$$

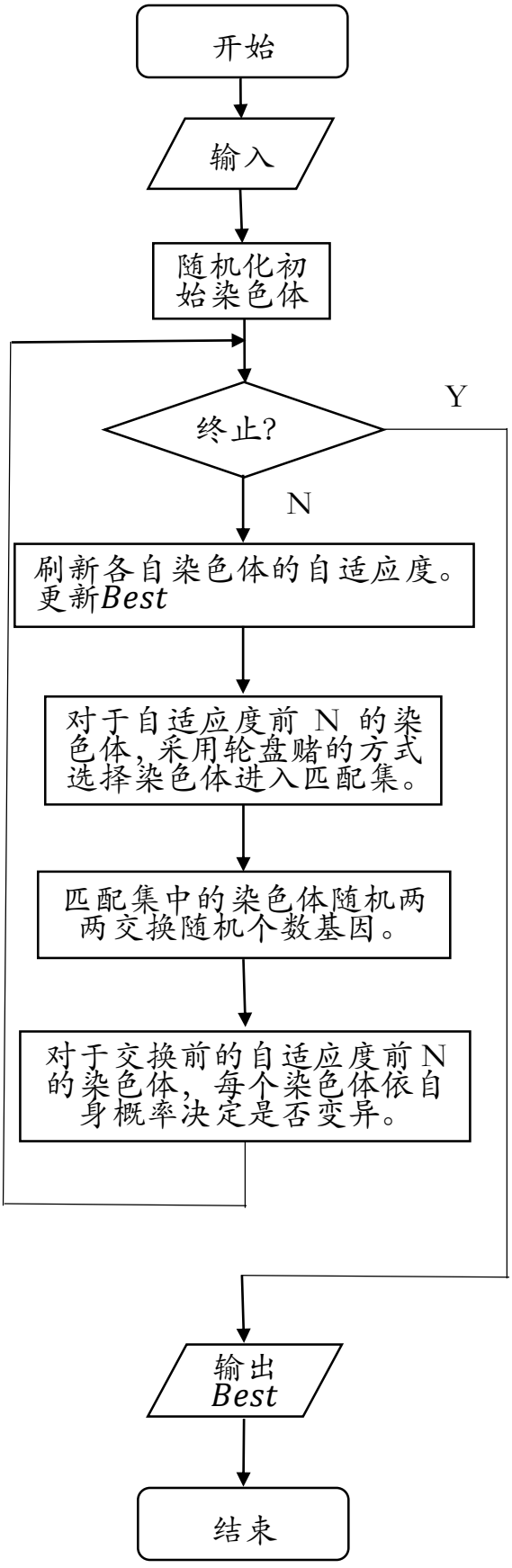
$$f_{avg} = \frac{\sum_{i=1}^{z} f_i}{z} \ ;$$

$$P_{ci} = \begin{cases} k_1 \sin\left(\frac{\pi}{2} \frac{f_{\max} - f_i}{f_{\max} - f_{avg}}\right), & f_i > f_{avg} \\ k_2, & f_i \le f_{avg} \end{cases}$$

$$P_{mi} = \begin{cases} k_3 \sin\left(\frac{\pi}{2} \frac{f_{\max} - f_i}{f_{\max} - f_{avg}}\right), & f_i > f_{avg} \\ k_4, & f_i \le f_{avg} \end{cases}$$

注：式中一般取$k_1 = 1.0$，$k_2 = 1.0$，$k_3 = 0.5$，$k_4 = 0.5$；

5.3 流程图



终止条件：达到迭代次数上限。

表格 2 启发式规则集

编号	规则名	描述
0	FIFC+SPT	最短加工时间优先（已加工时间最短）
1	FIFC+SRPT	最短加工时间优先（下一个操作加工用时最短）
2	FIFC+TIS	最长等待时间优先(当前用时减去在工件上加工的用时)

5.4 优缺点分析

优点:

- 超启发算法兼并了元启发算法寻优能力强和启发式规则的计算效率高的优点
- 本文的超启发算法较为简单，易于实现。
- 通用性强，有与元启发算法相同的适应范围。

缺点:

- 本文所用启发式规则较少，不能搜索到整个解空间
- 本文超启发算法不能生成新的启发式规则。

6. 算法实验分析

6.1 实验环境

硬件平台:

CPU：i7-4510u 2.0-2.6GHz

RAM：8GB

软件平台:

操作系统：Windows 10 专业版

编译器：Microsoft Visual Studio Ultimate 2013 版本 12.0.21005.1 REL

6.2 实验结果

为了验证三启发式算法求解 JSP 的性能，我们采用一个典型 JSP 测试用例（LA 类），这些测试问题在文献中被广泛用于测试。FT 类问题有 Fisher and Thompson 给出，LA 类问题有 Lawrence 给出，在此选取其中 8 个不同规模的问题: FT06, FT10, FT20, LA01, LA05, LA10, LA12,LA20 各算法均独立运行 50 次。其中迭代次数上限均为 300，种群规模 100，若连续迭代 50 次最优解无变化则推出迭代。其余参数按前文叙述设置，实验结果如下:

6.3 实验结果分析

实验结果表明:

① 超启发算法算出来的解优于启发式规则和元启发规则;

② 由于元启发式算法加入了启发式规则的解, 所以最差解不可能小于启发式规则。

三种算法结果对比:

单纯启发式规则运算速度快，比其他两者快了 3 个数量级。而元启发规则由于实现问题，并没有体现出其结果的高质量性。超启发的结果明显优于其他两者，并且在大规模问题下有明显优势。

7. 总结

通过最优化方法课程，深刻的学习了如何解决 NP 问题，接触了解了很多智能调度与优化的知识。本文表述了解决车间调度问题的算法进化过程。明显看出, PSO 算法实现并不容易，且并没有达到理想的

效果，但是超启发遗传算法效果拔群。因此，在解决车间调度问题时推荐使用超启发遗传算法。超启发算法的发展也不会止步，其是求解 NP 问题的最有前景的算法，将会是今后的重点研究方向之一。

表格 3 实验结果

问题	算法	最优解(s)	平均解(s)	最差解(s)	平均用时(s)
$FT06_{6*6}$	FIFC+SRPT	88	88	88	$1.58e^{-6}$
	PSO	67	72	77	0.440
	HHGA	65	65	65	0.360
$FT10_{10*10}$	FIFC+SRPT	1074	1074	1074	$6.8e^{-7}$
	PSO	1074	1074	1074	0.680
	HHGA	1037	1051	1042	1.30
$FT20_{10*10}$	FIFC+SRPT	1267	1267	1267	$8.6e^{-7}$
	PSO	1267	1267	1267	0.400
	HHGA	941	943	951	0.860
$LA01_{5*5}$	FIFC+SRPT	751	751	751	$3.6e^{-7}$
	PSO	751	751	751	0.320
	HHGA	717	720	731	0.420
$LA05_{10*5}$	FIFC+SRPT	610	610	610	$3.8e^{-7}$
	PSO	610	610	610	0.300
	HHGA	593	593	593	0.500
$LA10_{10*5}$	FIFC+SRPT	1049	1049	1049	$5e^{-7}$
	PSO	1049	1049	1049	0.460
	HHGA	958	958	958	0.480
$LA12_{20*5}$	FIFC+SRPT	1203	1203	1203	$6e^{-7}$
	PSO	1203	1203	1203	0.640
	HHGA	1039	1039	1039	0.620
$LA20_{10*10}$	FIFC+SRPT	1000	1000	1000	$1.42e^{-6}$
	PSO	1000	1000	1000	0.400
	HHGA	1267	1267	1267	0.480

参考文献

[1] 常桂娟.改进粒子群算法在作业车间调度问题中的应用.[J].四川师范大学学报(自然科学版)，2009,32(1)：139-142.

[2] 黄慧，黎向锋，左敦稳，薛善良.基于改进粒子群算法的车间作业排序的优化设计.[J].中国制造业信息化，2011,40(21)：6-9.

[3] 何利，刘永贤，谢华龙，刘笑天.基于粒子群算法的车间调度与优化.[J].东北大学学报(自然科学版)，2008,29(4)：565-568.

[4] 李小华，熊禾根.基于粒子群算法的车间作业调度问题.[J].信息技术，2009(7)：19-21.
[5] 周驰，高亮，高海兵.基于 PSO 的置换流水车间调度算法.[J].电子学报，2006(11)：2008-2011.
[6] 范华丽，熊禾根，蒋国璋，李公法.车间作业调度问题中调度规则算法的设计方法研究进展.[J].科技导报，2016,34(2)：33-38.
[7] 王万良，吴启迪，宋毅.求解作业车间调度问题的改进自适应遗传算法.[J].系统工程理论与实践，2004(2)：58-62.
[8] 周泓,姬彬.求解作业排序问题的通用混合遗传算法研究.[J].系统工程理论与实践，2001(12)：66-71.
[9] 余有明，刘玉树.，阎光伟.遗传算法的编码理论与应用.[J].计算机工程与应用，2006(3)：86-89.
[10] Dongni Li, Rongxin Zhan, Dan Zheng, Miao Li, and Ikou Kaku. A Hybrid Evolutionary Hyper-Heuristic Approach for Intercell Scheduling Considering Transportation Capacity IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, VOL. 13, NO. 2, APRIL 2016.

Thanks to $DongNi\ \ Li, RongXin\ \ Zhan$.
2016/06/08 16:10