Journal of System Simulation

Volume 35 | Issue 7 Article 12

8-25-2023

Improved Particle Swarm Algorithm of Unrelated Parallel Batch Scheduling Optimization

Lizhen Du
School of Mechanical Engineering and Automation,Wuhan Textile University, Wuhan 430200, China, dlzay@wtu.edu.cn

Tao Ye
School of Mechanical Engineering and Automation,Wuhan Textile University, Wuhan 430200, China

Yuhao Wang
School of Mechanical Engineering and Automation,Wuhan Textile University, Wuhan 430200, China

Yajun Zhang
School of Mechanical Engineering and Automation,Wuhan Textile University, Wuhan 430200, China

Improved Particle Swarm Algorithm of Unrelated Parallel Batch Scheduling Optimization

Abstract

Abstract: To address the problems of population diversity loss and the tendency to fall into local optimality in the PSO (particle swarm optimization)algorithm in dealing with unrelated parallel batch scheduling problems, an improved scheduling optimization algorithm for PSO is proposed for minimizing the maximum completion time solution. *A real number encoding based on the sequence of artifacts is used for the encoding operation. A new strategy based on J_B local search is designed based on the mixed integer programming model of the problem. The Metropolis criterion of the simulated annealing algorithm isintroduced into the individual extreme value search of the population particles.The performance of the algorithm is tested with randomly generated small,medium and large instances and compared with proposed metaheuristic for this scheduling problem and three other metaheuristics.* The experimental results and statistical tests shows that the algorithm performs significantly better than the comparison algorithm.

Keywords

unrelated parallel, batch scheduling, local search strategy, particle swarm algorithm, simulated annealing

第 35 卷第 7 期 系统仿真学报© Vol. 35 No. 7
2023 年 7 月 Journal of System Simulation Jul. 2023

改进粒子群算法的不相关并行批处理调度优化

杜利珍，叶涛，王宇豪，张亚军，宣自风
(武汉纺织大学 机械工程及自动化学院，湖北 武汉 430200)

摘要：针对粒子群优化(particle swarm optimization，PSO)算法在处理不相关并行批处理调度问题中存在的种群多样性丢失、易陷入局部最优等问题，提出了一种改进PSO的调度优化算法，用于最小化最大完工时间求解。*采用基于工件序列的实数编码方式进行编码操作；基于该问题的混合整数规划模型，设计了一种J_B局部搜索的新策略；将模拟退火算法的Metropolis准则引入种群粒子的个体极值搜索。通过随机生成的小型、中型和大型实例对该算法的性能进行了测试，并与针对该调度问题提出的元启发式算法和其他3种元启发式算法进行了比较。*实验结果和统计测试表明，该算法的性能明显优于对比算法。

关键词：不相关并行；批调度；局部搜索策略；粒子群算法；模拟退火

中图分类号：TP391.9 文献标志码：A 文章编号：1004-731X(2023)07-1549-13

DOI: 10.16182/j.issn1004731x.joss.22-0367

引用格式: 杜利珍, 叶涛, 王宇豪, 等. 改进粒子群算法的不相关并行批处理调度优化[J]. 系统仿真学报, 2023, 35(7): 1549-1561.

Reference format: Du Lizhen, Ye Tao, Wang Yuhao, et al. Improved Particle Swarm Algorithm of Unrelated Parallel Batch Scheduling Optimization[J]. Journal of System Simulation, 2023, 35(7): 1549-1561.

Improved Particle Swarm Algorithm of Unrelated Parallel Batch Scheduling Optimization

Du Lizhen, *Ye Tao*, *Wang Yuhao*, *Zhang Yajun*, *Xuan Zifeng*

(School of Mechanical Engineering and Automation,Wuhan Textile University, Wuhan 430200, China)

Abstract: To address the problems of population diversity loss and the tendency to fall into local optimality in the PSO (particle swarm optimization)algorithm in dealing with unrelated parallel batch scheduling problems, an improved scheduling optimization algorithm for PSO is proposed for minimizing the maximum completion time solution. *A real number encoding based on the sequence of artifacts is used for the encoding operation. A new strategy based on J_B local search is designed based on the mixed integer programming model of the problem. The Metropolis criterion of the simulated annealing algorithm isintroduced into the individual extreme value search of the population particles.The performance of the algorithm is tested with randomly generated small,medium and large instances and compared with proposed metaheuristic for this scheduling problem and three other metaheuristics.* The experimental results and statistical tests shows that the algorithm performs significantly better than the comparison algorithm.

Keywords: unrelated parallel; batch scheduling; local search strategy; particle swarm algorithm; simulated annealing

收稿日期：2022-04-18 修回日期：2022-07-21

基金项目：国家重点研发计划(2019YFB1706300)

第一作者：杜利珍(1975-)，女，副教授，博士，研究方向为智能调度、系统建模仿真与优化。E-mail：dlzay@wtu.edu.cn

0 引言

随着企业智能化改革的进一步推进，智能排产技术在制造业中得到广泛应用[1]，传统人工手排严重阻碍制造业的智能化发展，为提高生产效率，大多数企业都引进了智能制造系统和数控设备。并行批处理调度问题(parallel batch processing scheduling problem，PBPSP)具有很高的灵活性，其计算复杂性远大于并行调度问题(parallel scheduling problem，PSP)，区别在于前者将产品按生产需求策略化的组批式生产，帮助企业降低了生产时间成本。在实际生产过程中，存在诸多不确定性因素，例如：产品种类繁多、机器加工属性不同等，因此延伸出了不相关并行批处理调度问题(unrelated parallel batch processing scheduling problem，UPBPSP)。该调度问题与PBPSP和PSP的主要区别在于机器加工属性不同，即相同工件在不同机器上的加工时间不同，3种调度问题的相同点在于工件在一道工序加工过程中时有多台机器可以选择。UPBPSP的生产模式在金属加工、半导体制造等行业应用广泛，且国内研究学者对该调度问题的研究尚少。在保证调度方案稳定性、实时性、鲁棒性[2]的条件下，促进企业生产满足市场需求、物料资源利用更加高效，是UPBPSP的主要目标。同时，在以最大完工时间最小化为目标来评估整个调度方案的有效性时，尽可能降低不确定性因素对其造成的损失是不相关并行批处理调度问题研究中的一个热点。

对于PBPSP研究，国内许多学者提出了各类启发式算法针对此类问题进行了求解。文献[3]针对具有任意容量的并行批处理机调度问题，以总加权交付时间最小化为目标，提出了两种启发式算法和一种蚁群优化(ant colony optimization，ACO)算法来求解该问题，实验结果验证了该算法在求解性能上优于其他对比算法。文献[4]针对模糊环境下的并行批处理调度问题，以最大完工时间最小化为目标，提出模糊蚁群算法解决具有不同作业大小和模糊处理时间的调度问题，实验结果证明该算法相比几种最新算法在规定时间内可找到最优解。

对于UPBPSP的研究，大多数采用给出调度问题下界或是引入局部优化策略进行问题的求解。例如，文献[5]提出一种迭代贪婪(iteraticn greedy，IG)算法，给出调度问题的下界和混合整数规划(mixed integer programming，MIP)模型，解决具有不同作业大小和非零准备时间的不相关并行批处理调度问题。文献[6]针对不同作业规模和权重的无关并行批处理调度问题，以总加权完工时间最小化为目标，提出结合局部优化策略的蚁群算法，解决了考虑作业权重和作业大小的调度问题，实验结果证明该算法在规定时间内可求得最优解。文献[7]针对具有电价价格的不相关并行批处理的绿色批量调度问题，提出单种群遗传算法和多种群遗传算法，引入自适应参数调整策略，通过随机生成测试案例进行实验，实验结果证明该算法在求解性能上优于商业解算器。文献[8]针对具有退化和学习效应的无关并行批处理调度问题，提出一种结合偏置随机密钥遗传算法和差分进化的混合算法来求解该调度问题，计算实验验证了该算法的有效性和高效性。文献等[9]针对机器具有不同容量、不同加工速率的UPBPSP，以最大完工时间最小化为目标，提出了两类启发式算法进行问题求解，通过调度问题的下界来评估该算法的质量，计算实验验证了该算法明显在求解性能方面优于其他对比算法。

综上，大部分研究UPBPSP是基于调度问题的特征提出一种混合算法或是引入局部优化策略和自适应参数调整策略，以使整个调度方案更具鲁棒性和稳定性，故本文基于模型约束条件之下，提出一种新的变邻域搜索策略，用于在模型约束条件下的可行搜索空间中搜寻比当前最优解更好的解。根据UPBPSP的实际生产特征，引入了J_B局部搜索策略和模拟退火(simulated annealing，SA)算法相结合的混合粒子群算法，应用此改进策略不仅提高了粒子在模型约束的可行搜索空间中寻优的能力，并

DOI: 10.16182/j.issn1004731x.joss.22-0367

有效地避免了粒子出现早熟收敛的现象，最后运用随机生成的400个测试案例进行算法寻优性能上的对比分析，实验结果证明混合粒子群优化(hybrid particle swarm optimization，HPSO)算法相比于其他对比算法在求解性能上更具优越性、高效性。

1 问题描述及建模

1.1 问题描述

UPBPSP由组批阶段和批调度阶段组成。首先将工件按照一定规则组批，然后将批次按调度规则分配到相互独立的批处理机器上加工。机器具有不同的加工容量Q_k，作业具有不同的加工尺寸s_i、加工时间p_i，对所有作业i，每台机器都有一个固定的速度V_k，使得作业i在机器k上的加工时间p_i/V_k。所有批次需在k台不相关并行批处理机上完成加工，此外，允许机器零时刻进行加工。

该问题由文献[10]提出的标准三元组法$\alpha|\beta|\gamma$为

$$R_m|p-batch, s_i, p_i, Q_k, V_k|C_{\max}$$

式中：R_m为m台不相关并行批处理机器；$p-batch$为工件以组批的方式在批处理器上进行加工；$C_{\max}$为所有批次加工完后的机器最长加工时间。

1.2 基本假设

为便于建模及求解，做出如下假设：

(1) 一旦开始加工，整个加工过程不能中断，并且在所有批次加工完成之前，不能将其他批次中的作业向正在被处理批次进行添加或移动；

(2) 同一时刻同一台批处理机器上只允许加工一个批次，上一批次完成加工后下一批次才可加工；

(3) 批加工时间为批中作业最大加工时间，批次总尺寸为该批次中所有作业尺寸之和。

1.3 模型建立

对于调度过程中相邻批次的切换时间及机器设置时间均忽略不计，以最大完工时间最小化作为优化目标，该问题的MILP如下。

目标(1)是最大完工时间最小化。

约束(2)确保每个作业被分配到一个批处理，并且在一台机器上处理。

约束(3)保证在一台机器上处理的批次中所有作业尺寸总大小不超过机器的容量限制。

约束(4)确保一个批次的开始时间必须大于或等于前一个批次的完成时间。

约束(5)表示第b个批次在机器k上的加工时间大于或等于该批中作业在机器k上最大加工时间。

约束(6)表示每台批处理机器从零时刻开始加工。

约束(7)表示最大完工时间等于每台并行批处理机上所有最后一批的最大完工时间。

约束(8)确保决策变量具有非负性。

约束(9)是布尔变量的取值范围。

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

s.t.

$$\sum_{b\in B}\sum_{m\in M} x_{ibk}=1,\ \forall i\in N \tag{2}$$

$$\sum_{i\in N} s_i x_{ibk}\leqslant Q_k,\ \forall b\in B, k\in M \tag{3}$$

$$S_{(b+1)k}\geqslant S_{bk}+P_{bk},\ \forall b\in B, k\in M \tag{4}$$

$$P_{bk}\geqslant (p_i/V_k)x_{ibk},\ \forall i\in N, b\in B, k\in M \tag{5}$$

$$C_{1k}=P_{1k}V_k,\ \forall k\in M \tag{6}$$

$$C_{\max}\geqslant \sum_{b\in B} P_{bk}V_k,\ \forall k\in M \tag{7}$$

$$P_{bk}\geqslant 0, C_{bk}\geqslant 0, S_{bk}\geqslant 0,\ \forall b\in B, k\in M \tag{8}$$

$$x_{ibk}\in\{0,1\},\ \forall i\in N, b\in B, k\in M \tag{9}$$

式中：P_{bk}为第b个批在机器k上的加工时间；S_{bk}为第b个批在机器k上的开始加工时间；s_i为工件的加工尺寸；Q_k为批加工机器的加工容量；p_i为工件加工时间，V_k为批处理机器的加工速率；x_{ibk}为布尔变量，即第b批中的第i个工件在机器k上加工则为1，否则为0；C_{bk}为第b批在机器k上的完成时间；N为作业集合，$N=\{1,2,\cdots,n\}$；B为批次集合，$B=\{1,2,\cdots,b\}$；M为批处理机器集合，$M=\{1,2,\cdots,m\}$。

2 HPSO求解UPBPSP

2.1 PSO简介

粒子群优化(particle swarm optimization, PSO)算法受鸟类觅食行为过程的启发，具有收敛速度快的特点。种群中各粒子以特定速度在多维搜索空间中搜寻最优解，所有粒子将自身搜寻的最优解与种群中其他粒子共享，通过找到最优个体极值作为种群当前全局最优解。所有粒子根据当前个体极值和种群共享的当前全局最优解进行比对，从而调整自身搜索速度和当前位置。

2.2 种群编码

所研究的UPBPSP属于离散优化问题，PSO属于连续优化方法，应用基于工件序列的实数编码[11]方式将离散型问题转化为连续型问题进行求解，其编码原理是用一组不重复的N维随机数向量表征问题可行解，通过排序确定工件加工顺序，向量长度为总加工工件数，具体编码如表1所示。

表1 工件序列的实数编码
Table 1 Real encoding of the artifact sequence

数组	位置K	工件	数组	位置K	工件
0.11	2	1	0.65	8	5
0.20	4	2	0.45	6	6
0.32	5	3	0.56	7	7
0.05	1	4	0.12	3	8

表1中第一行是随机产生的数组，通过第二行排序确定第三行工件$\{1,2,3,4,5,6,7,8\}$加工顺序：4、1、8、2、3、6、7、5。

工件组批过程中引入布尔型数组对工件状态进行编码操作，其编码如表2所示。

表2 工件状态编码
Table 2 Workpiece state coding

工件	1	2	3	4	5	6	7	8
工作状态变量	0	0	1	0	1	1	1	0

表中第一行是工件编号，第二行是工件状态变量，由表2可知,工件$\{1,2,4,8\}$未被组批处理，工件$\{3,5,6,7\}$已被组批处理。

2.3 初始种群

粒子初始位置根据式(10)随机生成，特定参数$X_{\max}=4.0$，$X_{\min}=0$，r_1和r_2均为0~1之间随机数，粒子初始速度由式(11)随机生成，其中$V_{\min}=-4.0$，$V_{\max}=4.0$。

$$X_{ij}^0=X_{\min}+(X_{\max}-X_{\min})r_1 \tag{10}$$

$$V_{ij}^0=V_{\min}+(V_{\max}-V_{\min})r_2 \tag{11}$$

2.4 个体更新

粒子更新公式与文献[12]提出的公式一样。在N维空间中，粒子每一次迭代有3种可能性选择：

(1) 继续遵循当前所搜索的模式进行下去；

(2) 根据自身的经验回到它以前最佳的位置；

(3) 根据自身经验和粒子间合作与知识共享，回到种群中最佳粒子的位置。

粒子当前搜寻的最好位置记作p，作为粒子自身的飞行经验，整个种群迄今为止搜索到的最优位置记作g，视为粒子同伴经验。所有粒子都是依赖这两个值来更新自身搜索速度和位置，更新公式如下：

$$V_{ij}^t=w^{t-1}\cdot V_{ij}^{t-1}+c_1r_1(p_{ij}-X_{ij}^{t-1})+c_2r_2(g_j-X_{ij}^{t-1}) \tag{12}$$

$$X_{ij}^t=X_{ij}^{t-1}+V_{ij}^t \tag{13}$$

式中：ω为惯性权重，用于控制前一次迭代所产生的粒子速度对本次迭代速度的影响，表示多大程度上保持原来的粒子搜索的速度；c_1，c_2分别为自身认知系数和社会学习系数，通常取值为2；r_1和r_2均为0~1之间的随机数；V_{ij}^t和V_{ij}^{t-1}为在第t次和$t-1$次迭代中粒子i飞行速度矢量的第j维分量；X_{ij}^t和X_{ij}^{t-1}为第t次和$t-1$次迭代时粒子i位置矢量的第j维分量；p_{ij}为粒子i在第j维分量上最佳适应度值相关联的最佳位置；g_j为整个种群中适应度值最好粒子所对应的最佳位置。

2.5 J_B局部搜索机制

基于不同搜索机制组排实验[13]，本文提出的局部搜索策略分为2部分：①与当前最优解对应

机器具有相同加工容量机器间的批次交换[14]；②经过批次交换后，对新的最优解对应机器上各批次间进行作业交换[15]。该策略伪代码如下所示。

算法1：J_B局部搜索策略

输入：U_c, M_c, ID_b, Q_s

输出：最优解U_c'

1: if $ID\_b \geqslant 2$ then

2: $d_1, n = ((n_f - 1)n_f)/2$

3: while $d < n$ do

4: $M_c(n_f) \rightarrow B_a, B_b$

5: if $E(Q(B_a \rightleftarrows Bb) \subset Q_s)$ then

6: $U_d = \min(F(B_a, B_b))$;

7: $U_c' \longleftarrow U_d$;

8: end if

9: $d_1 \longleftarrow d_1 + 1$;

10: end while

11: $R \longleftarrow J(U_c', ID\_YOU, TD\_CM, JOB\_ID)$

12: if $ID\_YOU == 0$ then

13: $d_2, m_1 = ((JOB\_ID - 1) \cdot JOB\_ID)/2$

14: while $d_2 < m_1$ do

15: $M_c(n_f) \rightarrow L_a, L_b$

16: $\{L_a, L_b\} \xrightarrow{\text{respectively}} j_a, j_b$

17: if $E(Q(j_a \rightleftarrows j_b) \subset Q_s)$ then

18: $U_e = \min(F(j_a, j_b))$

19: $U_c' \longleftarrow U_e$

20: end if

21: $d_2 \longleftarrow d_2 + 1$;

22: end while

23: else

24: $d_3, m_2 = ((JOB\_ID - 1)JOB\_ID)/2$

25: while $d_3 < m_2$ do

26: $TD\_CM(n_m) \rightarrow F_a, F_b$

27: $\{F_a, F_b\} \xrightarrow{\text{respectively}} h_a, h_b$

28: if $E(Q(h_a \rightleftarrows h_b) \subset Q_s)$ then

29: $U_f = \min(F(h_a, h_b))$;

30: $U_c' \longleftarrow U_f$;

31: end if

32: $d_3 \longleftarrow d_3 + 1$;

33: end while

34: end if

35: end if

伪代码中，U_c为算法当前最优解；M_c为当前最优解对应的机器；ID_b为与M_c具有相同加工容量的机器数；Q_s为机器的加工容量；n_f为与M_c具有相同加工容量机器的总批次数；$M_c(n_f) \rightarrow$为从与M_c具有相同加工容量机器的总批次中按一定规则$(a<b)$抽取批次；$E(Q(B_a \rightleftarrows Bb) \subset Q_s)$为批次$B_a$, B_b交换后，各机器没有超出加工容量Q_s；$\min(F(B_a, B_b))$表示批次B_a, B_b交换后所得最小可行解；U_c'为经过算法1得到的最优解；R为经过批次交换后输出解的集合；ID_YOU为U_c'对应的交换批次；TD_CM为U_c'对应的机器编号；JOB_ID为TD_CM机器上的作业总数；$\xrightarrow{\text{respectively}} j_a, j_b$为从批次$L_a, L_b$抽取作业$j_a, j_b$；$\min(F(j_a, j_b))$为作业$j_a, j_b$交换所得最小可行解；$TD_CM(n_m)$为$U_c'$对应机器$TD_CM$上的总批次集合；$h_a, h_b$为从批次$F_a$, F_b上抽取的作业；$\min(F(h_a, h_b))$为作业h_a, h_b交换所得最小可行解。

算法1中，第1行：判断与机器M_c具有相同加工容量的机器数是否大于等于2；第2~10行：将所有与机器M_c具有相同加工容量机器上的总批次按规则$(a<b)$抽取后进行机器之间的批次交换，输出比当前最优解更优的解U_d；第12行：判定批次交换后是否存在优于当前最优解的批次交换情况；第13~22行：此情况不发生，在当前最优解对应机器上进行批次间工件的交换，输出比当前最优解更优的解U_e；第24~33行：此情况发生，找出诸多情况中最小最优解对应的机器，在该机

器上按规则($a<b$)抽取更新后各批次的作业进行交换，输出比U_d更优的解U_f，通过比较U_f和U_e确定最终输出U_c'。

2.6 种群个体极值代替机制

粒子搜寻过程中会保留当前搜索到的最优解，作为粒子的认知部分，使得粒子具有返回自身最优位置的能力，此方法虽可以指导粒子向全局最优解靠近，但随迭代次数的递增，种群多样性会快速下降，易使算法陷入局部最优。故本文在将J_B局部搜索机制应用到粒子寻优基础之上，引入模拟退火算法的Metropolis准则，即一定概率K接受比当前最优位置差的解，更新后的粒子进入下一代种群：

$$K=\exp\left(-\frac{E_p-E_x}{T_k}\right) \tag{14}$$

$$T_k=wT_{k-1} \tag{15}$$

式中：E_p为第p个粒子更新位置后对应的适应度值；E_x为第x个粒子当前最优解对应的适应度值；T_k为降温函数；w为温度衰减参数；k为降温次数。

2.7 批处理形成与调度

调度过程中存在两个相互依赖的决策：工件组批和批次调度。为优化整个调度过程，本文基于文献[16]提出了一种新的启发式方法，该方法具体步骤如下：

步骤1：给定工件序列。

步骤2：找出首台可用机器k，若存在多台可用机器，按机器加工容量与其加工速率乘积($Q_k\cdot V_k$)降序排列，从而确定使用顺序，创建第一个批次b。

步骤3：按工件序列依次选择，根据首台机器加工容量进行工件组批，超出容量限制时则考虑后一作业，重复步骤3，直到工件组完批。

步骤4：在步骤2确定好各批处理机器的首个批次后，根据ECT(earliest completion time)规则确定哪台机器将会被安排批次进行调度处理。

步骤5：重复步骤2~4，直到所有的批次b安排到对应批处理机器k上。

图1所示为作业数为20、机器数为3的调度甘特图，其作业序列、作业大小、处理时间、机器容量和加工速率相关数据，如表3和表4所示。

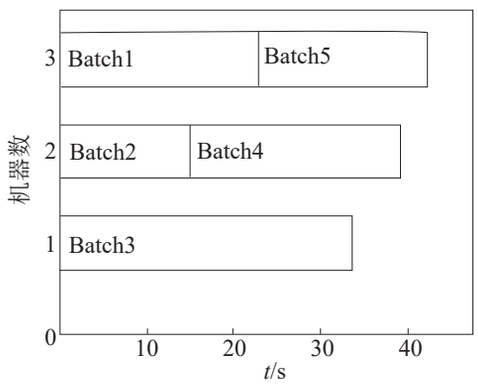


图1 调度甘特图
Fig. 1 Scheduling gant chart

表3 工件加工数据
Table 3 Workpiece processing data

作业	尺寸	PT_1	PT_2	PT_3
19	3.0	9.3	6.5	7.2
20	8.0	8.0	21.4	15.0
15	17.0	33.6	23.5	26.1
18	14.0	29.3	20.5	22.8
4	6.0	17.9	12.5	13.9
7	13.0	22.1	15.5	17.2
12	10.0	17.9	12.5	13.9
13	7.0	20.0	23.6	16.5
1	20.0	23.6	16.5	18.3
3	6.0	7.1	5.0	5.6
16	8.0	15.0	10.5	11.7
6	10.0	20.7	14.5	16.1
5	16.0	12.9	9.0	10.0
17	7.0	19.3	13.5	15.0
10	14.0	26.4	18.5	20.6
2	16.0	17.9	12.5	13.9
14	11.0	34.3	24.0	26.7
8	13.0	31.4	22.0	24.4
9	15.0	25.0	17.5	19.4
11	4.0	22.9	16.0	17.8

DOI: 10.16182/j.issn1004731x.joss.22-0367

表4 机器加工属性
Table 4 Machine processing attributes

机器	加工容量	加工速率	乘积
1	40	1.4	56
2	40	2.0	80
3	50	1.8	90

该问题的可行解是通过每台批处理机器上不同批次中包含的作业编号集进行编码，该示例的可行解方案为批处理机器1中给定的加工批次为(1, 16, 6, 2, 14, 8)，批处理机器2给定的加工批次为(7, 12, 13, 3, 11)、(5, 17, 10)，批处理机器3中给定的加工批次为(19, 20, 15, 18, 4)、(9)。

2.8 改进粒子群算法求解流程

改进粒子群(hybrid particle swarm optimization, HPSO)算法流程如图2所示。

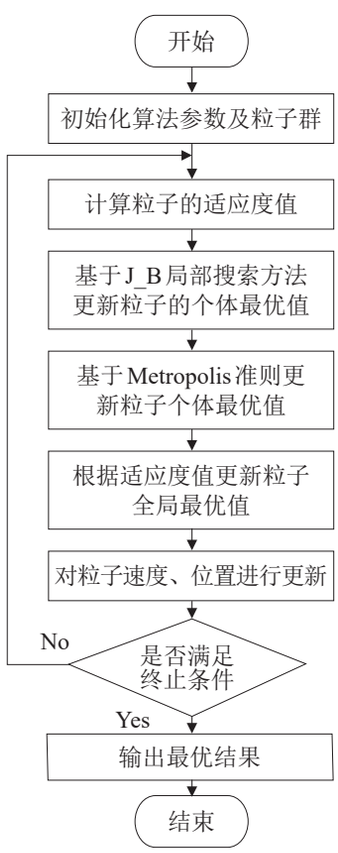


图2 改进粒子群算法流程图
Fig. 2 Flow chart of improved particles swarm algorithm

具体步骤如下：

步骤1：对算法参数初始化，主要包括：粒子种群规模、种群粒子移动速度、最大迭代终止次数$T_{\max}$。

步骤2：对种群粒子个体进行适应度值的计算。

步骤3：基于J_B局部搜索方法对种群粒子个体极值在模型约束条件下的解集空间中进行寻优。

步骤4：在局部搜索方法寻优的基础上，引入Metropolis准则对已更新的个体极值再次进行替代更新，并根据粒子的适应度值对全局最优解进行更新。

步骤5：对种群粒子的速度和位置进行更新。

步骤6：判断是否满足最大迭代终止次数，若满足则输出最优结果，否则转步骤2。

2.9 算法复杂度分析

在HPSO每次迭代中，时间复杂度主要体现在参数初始化、算法迭代和局部搜索操作。参数初始化时间复杂度为$O(n^2)$，粒子完成f次迭代的时间复杂度为$O(P\times f\times n^2)$，J_B局部搜索机制的时间复杂度为$O(n)$，即本文算法的时间复杂度为$O(P\times f\times n^2)$。

3 实验仿真与分析

为了测试HPSO在求解UPBPSP方面的性能，进行了大量仿真实验，所有算法和测试程序均用Matlab R2016b编程实现，操作系统为Win 10，处理器为Intel(R) Xeon(R) W-2133 CPU 3.6 GHz，内存为32 GB环境下运行。

3.1 测试实例和比较算法

文献[16]随机产生了40个组合算例，每个组合算例生成10个问题实例并用各算法分别求解了5次，本文按照文献[16]相同的方法随机生成仿真测试实例进行相同次数的求解。该调度问题的实例参数参照文献[16]设定，其具体参数如表5所示。

表5 问题实例参数
Table 5 Problem instance parameters

变量	符号	数值
工件数量	$n\in\{J_1,J_2,J_3,J_4,J_5\}$	20, 50, 100, 200, 300
工件尺寸	$s_j\in\{S_1,S_2\}$	U[1, 20], U[10, 30]
工件加工时间	p_j	U[8, 48]
机器数量	$m\in\{M_1,M_2,M_3,M_4\}$	2, 3, 4, 5
机器容量	H	{40, 50, 60}
机器速率	V_k	1, 1.2, 1.4, 1.6, 1.8, 2.0

本文选用平均百分比提升率*IMPR*作为算法求解质量的评价指标，其计算公式为

$$IMPR_{\mathrm{DDE}}=\frac{\bar{C}_{\max,\mathrm{DDE}}-\bar{C}_{\max,\mathrm{HPSO}}}{\bar{C}_{\max,\mathrm{DDE}}} \tag{16}$$

式中：$\bar{C}_{\max}$为算法所求当前最优解的平均值。选用 DDE(discrete differential evolution)[16]，LSPSO (local search particle swarm optimization)[14]，PSO 和 TS(tabu search)作为对比算法，通过求解 UPBPSP 所得评价指标来凸显 HPSO 在求解该问题上的有效性。其中 LSPSO 是文献[14]提出的 LS2 局域搜索策略与 PSO 算法的结合。

3.2 参数优化

采取正交实验法对 $J_3M_1S_{1\_1}$~$J_3M_3S_{2\_1}$ 共 6 个实例进行计算。其中“J_3”表示工件数为 100，“M_1”表示加工机器数为 2 台，“S_1”表示工件加工范围[1,20]，“S_2”表示工件加工尺寸范围[10,30]，末尾的 1 表示该组合算例的第一个问题实例。

HPSO 算法的参数初始化为 $X_{\max}=4$、$X_{\min}=0$，$V_{\max}=4$、$V_{\min}=-4$。对参数 P, $T_{\max}$, C_1, C_2 进行组合优化时，各参数的设定范围：$P\in\{20,50,100\}$、$T_{\max}\in\{100,150,200\}$、$C_1\in\{0.5,1.0,2.0\}$、$C_2\in\{0.5,1.0,2.0\}$。

对所有参数组合进行编号：LN_1~LN_9，将每个参数组合对 6 个实例独立运行 10 次，计算平均 $C_{\max}$ 值；用其正交表 $L_9(3^4)$ 将每个参数组合对应的 $C_{\max}$ 平均值再取平均值得 AMCT(avearge maximum completion time)，不同参数组合下的仿真结果、正交表及 AMCT 值如表 6，7 所示。以表 7 的 AMCT 值作为参数绘制各个参数折线图，如图 3 所示。

由图 3 结果可知，种群数为 50、最大迭代次数为 100 时，*AMCT*值相对较大，故算法具体参数如表 8 所示。表中：P_c 为离散差分进化算法的交叉概率；P_m 为离散差分进化算法的变异概率；T_{abuL} 为禁忌长度，C_{num} 为候选集个数。

3.3 实验结果与比较

为了验证 HPSO 求解性能，将 HPSO 与文献[16]中的 DDE 及其他 3 种算法(PSO，LSPSO[14]，TS)进行求解性能的对比分析，测试过程中：标黑加粗数字代表每组测试数据的较优值，Min 为每组算例所求 50 个解中的最优解，Avg 为每组算例所求 50 个解中的平均值，Max 为每组算例所求 50 个解中的最差解。测试结果如表 9 所示。

表6 不同参数组合下的仿真结果
Table 6 Simulation results under different parameter combinations

算例	编号								
	LN_1	LN_2	LN_3	LN_4	LN_5	LN_6	LN_7	LN_8	LN_9
$J_3M_1S_{1\_1}$	225.19	232.64	234.60	231.31	229.45	224.17	227.07	221.01	222.06
$J_3M_1S_{2\_1}$	387.03	383.25	387.19	383.97	402.25	382.72	392.28	370.22	376.56
$J_3M_2S_{1\_1}$	160.07	158.85	154.12	158.77	155.25	156.38	159.02	152.27	152.75
$J_3M_2S_{2\_1}$	213.12	230.30	223.40	223.35	227.00	217.55	224.50	220.05	219.16
$J_3M_3S_{1\_1}$	118.39	111.89	110.25	115.40	126.37	111.48	124.48	114.33	114.95
$J_3M_3S_{2\_1}$	179.51	186.81	185.76	183.35	191.31	172.29	189.68	169.48	171.76
$\bar{C}_{\max}$	213.88	217.29	215.89	216.03	221.94	210.77	219.50	207.89	209.54

DOI: 10.16182/j.issn1004731x.joss.22-0367

表7 正交表及AMCT值

Table 7 Orthogonal table and AMCT value

编号	Number	P	T_{max}	C_1	C_2	$AMCT$
LN_1	1	20	100	0.5	0.5	213.88
LN_2	2	20	150	1.0	1.0	217.29
LN_3	3	20	200	2.0	2.0	215.89
LN_4	4	50	100	1.0	2.0	216.03
LN_5	5	50	150	2.0	0.5	221.94
LN_6	6	50	200	0.5	1.0	210.77
LN_7	7	100	100	2.0	1.0	219.50
LN_8	8	100	150	0.5	2.0	207.89
LN_9	9	100	200	1.0	0.5	209.54

由表9可知，对于本文所考虑的Min，Avg和Max指标，工件数为20时，HPSO相比于LSPSO和DDE在解的求解质量上略差，工件数为50、100、200、300时，HPSO相比于LSPSO、PSO、TS和DDE解的求解质量上明显优越。这表明所提出的J_B局部搜索机制和Metropolis准则结合的HPSO算法在工件数多的情况下解的质量更好，同时证实提出的优化方法能够在模型约束条件下的可行解集空间中寻得比当前最优解更好的解，引导算法整体搜索到最优解对应的区域空间中去。

由表9数据分别作出工件数、机器数和机器加工容量对平均完工时间改进影响图，如图4所示。

表8 算法参数设置

Table 8 Algorithm parameter setting

DDE[16]	HPSO	LSPSO[14]	TS	PSO
P=40	P=50	P=50	T_{max}=200	P=50
T_{max}=100	T_{max}=100	T_{max}=100	T_{abuL}=10	T_{max}=100
Pc=0.1	ω=0.6	ω=0.6	C_{num}=10	ω=0.9
P_m=0.5	C_1=2.0	C_1=2.0		C_1=2.05
	C_2=1.0	C_2=1.0		C_2=2.05

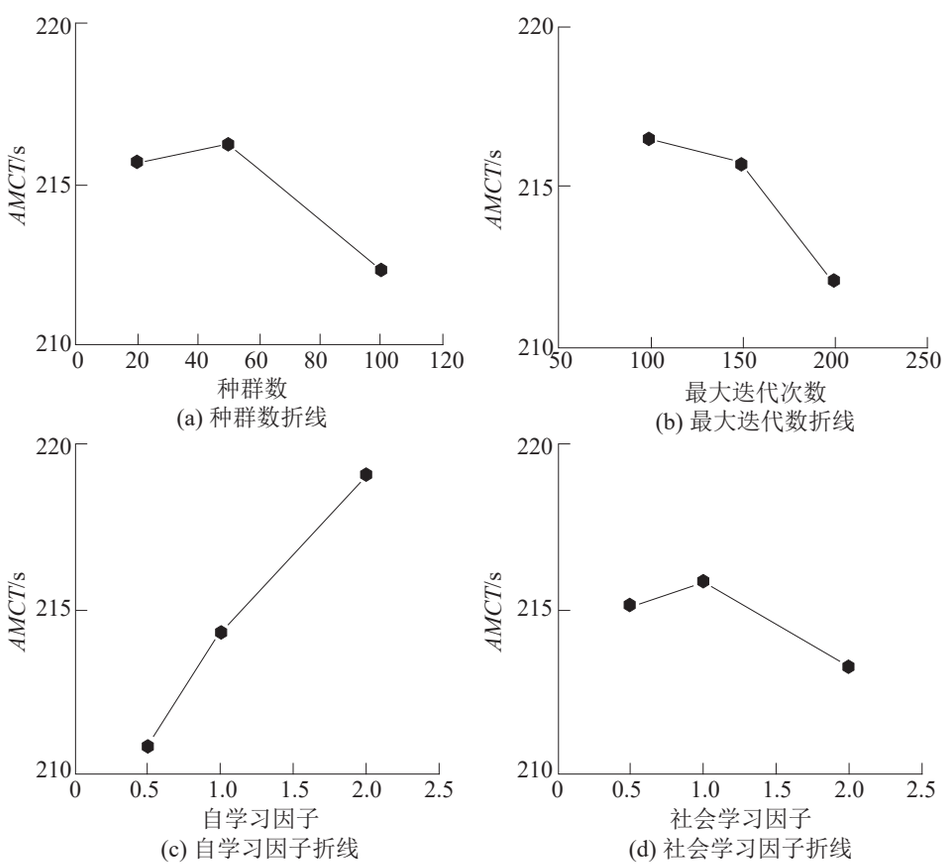


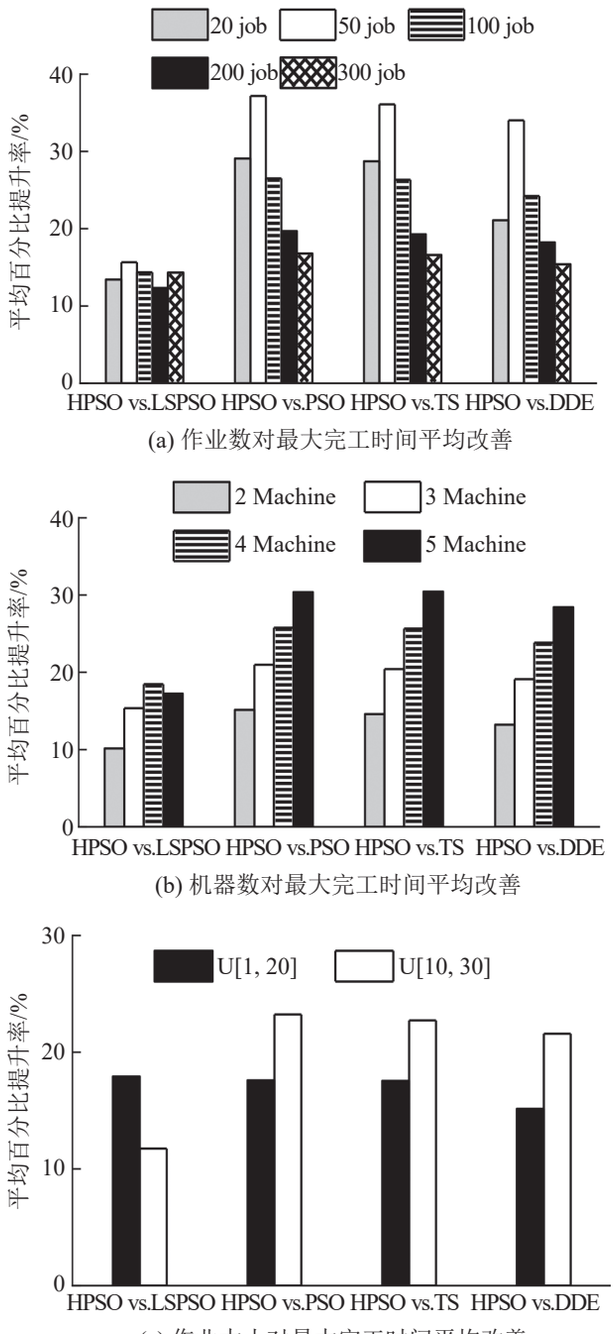
图3 参数影响图

Fig. 3 Parameter influence diagram

表9 仿真结果
Table 9 Simulation results

算例	HPSO			LSPSO[14]			PSO			TS			DDE[16]		
	Min	*Avg*	*Max*	*Min*	*Avg*	*Max*	*Min*	*Avg*	*Max*	*Min*	*Avg*	*Max*	*Min*	*Avg*	*Max*
$J_1M_1S_1$	**25**	**44**	**73**	38	50	74	37	57	87	33	57	105	34	50	74
$J_1M_1S_2$	**36**	**66**	97	65	72	**88**	65	103	145	61	98	157	64	95	133
$J_1M_2S_1$	**21**	**33**	56	34	38	**45**	31	44	62	30	47	76	30	40	56
$J_1M_2S_2$	**36**	**42**	**54**	40	48	**54**	49	65	92	43	65	126	43	59	86
$J_1M_3S_1$	**21**	**26**	36	24	30	**34**	26	34	42	23	34	53	24	29	**34**
$J_1M_3S_2$	**28**	**36**	55	36	42	**50**	40	51	67	37	49	76	37	46	59
$J_1M_4S_1$	**20**	**24**	**29**	24	29	32	22	30	35	22	32	46	21	25	**29**
$J_1M_4S_2$	**27**	**33**	**40**	31	36	42	40	45	53	36	45	85	37	41	46
$J_2M_1S_1$	**48**	**108**	**152**	76	124	178	88	137	188	74	137	263	82	130	173
$J_2M_1S_2$	**127**	**185**	258	145	199	**232**	187	249	347	172	242	405	185	241	326
$J_2M_2S_1$	**35**	**61**	**93**	48	78	**93**	67	96	137	63	93	149	64	90	127
$J_2M_2S_2$	**70**	**112**	**147**	89	126	154	151	181	225	138	177	248	149	174	210
$J_2M_3S_1$	**26**	**42**	**66**	41	56	75	61	75	93	54	76	107	56	70	86
$J_2M_3S_2$	**49**	**71**	**96**	67	78	**96**	114	130	156	106	128	166	112	125	149
$J_2M_4S_1$	**28**	**31**	**42**	34	42	53	60	68	77	52	69	100	56	64	72
$J_2M_4S_2$	**24**	**46**	**69**	44	55	74	91	106	130	87	102	133	88	100	122
$J_3M_1S_1$	**180**	**242**	**341**	198	278	354	220	282	376	199	279	434	214	273	370
$J_3M_1S_2$	**292**	**370**	**463**	314	394	475	374	462	548	349	460	665	368	450	534
$J_3M_2S_1$	**113**	**141**	**175**	120	177	199	153	184	209	143	183	263	145	178	202
$J_3M_2S_2$	**180**	**237**	**323**	197	256	345	279	337	426	271	333	439	269	328	414
$J_3M_3S_1$	**76**	**104**	**129**	97	142	168	116	149	183	110	152	194	112	144	176
$J_3M_3S_2$	**124**	**168**	**215**	135	189	243	212	257	299	204	264	336	205	249	286
$J_3M_4S_1$	**54**	**76**	**104**	78	98	134	97	117	137	91	115	153	94	112	132
$J_3M_4S_2$	**98**	**123**	**144**	104	135	189	174	199	233	167	196	251	169	193	224
$J_4M_1S_1$	**410**	**530**	**736**	452	580	786	485	589	818	462	585	788	476	579	807
$J_4M_1S_2$	**690**	**889**	**1 169**	742	945	1 224	862	1 070	1 353	835	1 062	1 439	840	1 053	1 326
$J_4M_2S_1$	**262**	**338**	**526**	354	388	578	309	401	622	301	401	633	307	392	605
$J_4M_2S_2$	**418**	**551**	**653**	478	610	687	536	697	795	503	679	858	528	684	778
$J_4M_3S_1$	**212**	**245**	**314**	245	294	382	261	306	380	259	307	386	261	300	372
$J_4M_3S_2$	**309**	**380**	**449**	354	432	487	422	515	596	410	515	783	426	506	580
$J_4M_4S_1$	**142**	**188**	**217**	189	240	340	194	249	279	187	249	305	193	244	269
$J_4M_4S_2$	**230**	**291**	**327**	321	344	371	361	424	488	346	430	575	358	415	476
$J_5M_1S_1$	646	**725**	**830**	702	845	945	682	788	920	674	785	963	**553**	767	910
$J_5M_1S_2$	**1 087**	**1 326**	1 575	1 124	1 453	**1 543**	1 257	1 549	1 838	1 220	1 548	2 222	1 241	1 531	1 803
$J_5M_2S_1$	**382**	**484**	**642**	420	568	741	432	567	731	425	569	752	430	557	716
$J_5M_2S_2$	**706**	**826**	**1 072**	842	957	1 124	820	989	1 219	806	989	1 306	811	977	1 200
$J_5M_3S_1$	**304**	**366**	**423**	345	446	512	357	430	502	344	428	515	356	423	489
$J_5M_3S_2$	**484**	**626**	**688**	524	724	898	665	819	965	658	809	1 050	651	805	929
$J_5M_4S_1$	**250**	**284**	**323**	**250**	320	425	299	361	415	298	362	534	304	354	411
$J_5M_4S_2$	**426**	**461**	**546**	435	513	587	534	621	742	525	621	769	531	610	721

DOI: 10.16182/j.issn1004731x.joss.22-0367



(a) 作业数对最大完工时间平均改善

(b) 机器数对最大完工时间平均改善

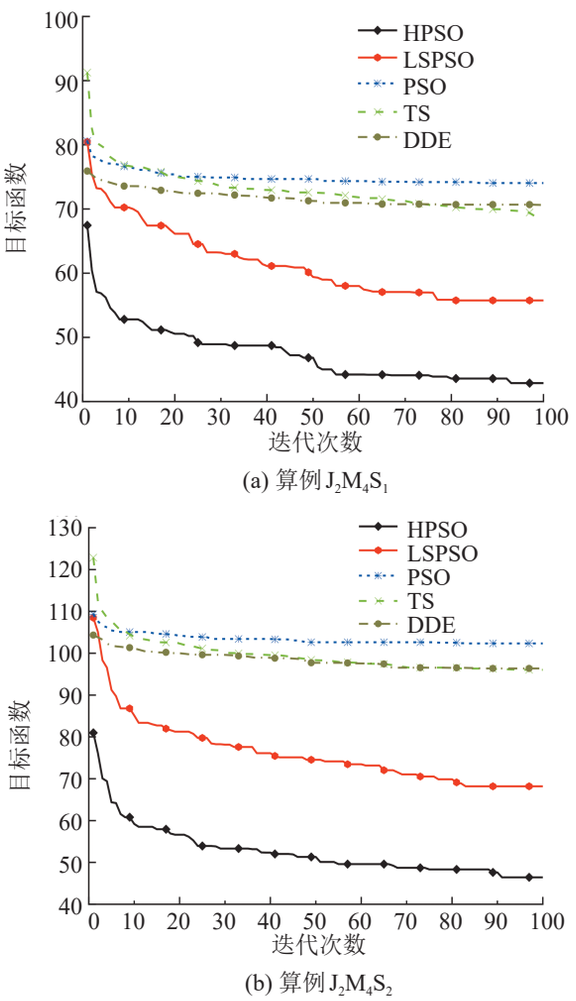
(c) 作业大小对最大完工时间平均改善

图4　3种因素对完工时间改进的影响

Fig. 4　Influence of three factors on completion time improvement

由图4数据可知，HPSO算法相比于其他4种算法在作业数、机器数、作业大小3种不同条件下的平均百分比改进率分别为21.97%，20.57%，18.44%。HPSO比LSPSO在3种不同条件下的平均百分比提升率比其他3种算法的平均百分比提升率相对较小，其平均优化程度为14.72%。

基于问题类别$J_2M_4S_1$和$J_2M_4S_2$，本文给出两类问题的收敛曲线图，如图5所示。



(a) 算例$J_2M_4S_1$

(b) 算例$J_2M_4S_2$

图5　收敛曲线

Fig. 5　Convergence curves

由图5可知，HPSO算法在$J_2M_4S_1$和$J_2M_4S_2$情况下，分别在第93、91次迭代收敛且目标函数值均小于其他对比算法。综上，HPSO在求解性能方面明显优于其他对比算法，同时证实了提出的优化方法在求解以最大完工时间最小化为目标的UPBPSP上，其优化效果明显高于其他对比算法。

4　结论

本文针对以最大完工时间最小化为目标的UPBPSP，建立了该问题的混合整数规模模型，提

出了一种混合粒子群算法进行求解。首先，基于适应值函数所求当前最优解，引入一种基于J_B的局部搜索策略，该策略用于在模型约束条件下的可行解集空间中搜寻比当前最优解更好的解，从而提高算法求解性能；然后，将Metropolis准则引入种群粒子个体的极值替换中，避免粒子出现早熟收敛的现象，增强粒子对解的全局搜索能力；最后，通过大量随机生成的仿真测试案例进行算法求解性能上的对比分析，计算实验验证了HPSO算法在求解UPBPSP上的有效性和优越性。

不相关并行批处理调度问题是近些年来比较热门的研究方向，普遍存在于实际工业制造系统中，未来的研究工作是针对带配送的不相关并行批处理调度问题展开研究，设计合理有效的分配规则及行之有效的寻优策略，更加贴近实际生产进行研究，并尝试着应用一些新的智能算法，比如灰狼算法、人工蜂群算法和萤火虫算法来求解此类问题。

参考文献：

[1] 陈魁, 毕利, 王文雅. 柔性作业车间AGV与机器双资源集成调度研究[J]. 系统仿真学报, 2022, 34(3): 461-469.
Chen Kui, Bi Li, Wang Wenya. Research on Integrated Scheduling of AGV and Machine in Flexible Job Shop[J]. Journal of System Simulation, 2022, 34(3): 461-469.

[2] 尤一琛, 王艳, 纪志成. 基于博弈论的柔性作业车间动态调度研究[J]. 系统仿真学报, 2021, 33(11): 2579-2588.
You Yichen, Wang Yan, Ji Zhicheng. Research on Flexible Job-shop Dynamic Scheduling Based on Game Theory[J]. Journal of System Simulation, 2021, 33(11): 2579-2588.

[3] Jia Zhaohong, Huo Siyun, Li Kai, et al. Integrated Scheduling on Parallel Batch Processing Machines With Non-identical Capacities[J]. Engineering Optimization, 2020, 52(4): 715-730.

[4] Jia Zhaohong, Yan Jianhai, Leung J Y T, et al. Ant Colony Optimization Algorithm for Scheduling Jobs With Fuzzy Processing Time on Parallel Batch Machines With Different Capacities[J]. Applied Soft Computing, 2019, 75: 548-561.

[5] José Elias C Arroyo, Leung J Y T. An Effective Iterated Greedy Algorithm for Scheduling Unrelated Parallel Batch Machines With Non-identical Capacities and Unequal Ready Times[J]. Computers & Industrial Engineering, 2017, 105: 84-100.

[6] Jia Zhaohong, Zhang Han, Long Wentao, et al. A Meta-heuristic for Minimizing Total Weighted Flow Time on Parallel Batch Machines[J]. Computers & Industrial Engineering, 2018, 125: 298-308.

[7] Tan Mao, Yang Huali, Su Yongxin. Genetic Algorithms With Greedy Strategy for Green Batch Scheduling on Non-identical Parallel Machines[J]. Memetic Computing, 2019, 11(4): 439-452.

[8] Kong Min, Liu Xinbao, Pei Jun, et al. A BRKGA-DE Algorithm for Parallel-batching Scheduling With Deterioration and Learning Effects on Parallel Machines Under Preventive Maintenance Consideration[J]. Annals of Mathematics and Artificial Intelligence, 2020, 88(1): 237-267.

[9] Zarook Y, Rezaeian J, Mahdavi I, et al. Efficient Algorithms to Minimize Makespan of the Unrelated Parallel Batch-processing Machines Scheduling Problem with Unequal Job Ready Times[J]. Rairo Operations Research, 2021, 55(3): 1501-1522.

[10] Graham R L, Lawler E L, Lenstra J K, et al. Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey[J]. Annals of Discrete Mathematics, 1979, 5: 287-326.

[11] 张源, 王加冕. 改进粒子群算法求解置换流水车间调度问题[J]. 软件, 2020, 41(6): 108-111, 131.
Zhang Yuan, Wang Jiamian. An Improved Particle Swarm Optimization Algorithm is Proposed to Solve the Displacement Flow Shop Scheduling Problem[J]. Computer Engineering & Software, 2020, 41(6): 108-111, 131.

[12] Lee C Y. Minimizing Makespan on a Single Batch Processing Machine With Dynamic Job Arrivals[J]. International Journal of Production Research, 1999, 37 (1): 219-236.

[13] Cakici E, Mason S J, Fowler J W, et al. Batch Scheduling on Parallel Machines With Dynamic Job Arrivals and Incompatible Job Families[J]. International Journal of Production Research, 2013, 51(8): 2462-2477.

[14] Jiang Wei, Shen Yilan, Liu Lingxuan, et al. A New Method for a Class of Parallel Batch Machine Scheduling Problem[J]. Flexible Services and Manufacturing Journal, 2022, 34(2): 518-550.

[15] José Elias C Arroyo, Leung J Y T, Ricardo Gonçalves Tavares. An Iterated Greedy Algorithm for Total Flow Time Minimization in Unrelated Parallel Batch Machines With Unequal Job Release Times[J]. Engineering

DOI: 10.16182/j.issn1004731x.joss.22-0367

Applications of Artificial Intelligence, 2019, 77: 239-254.

[16] Zhou Shengchao, Liu Ming, Chen Huaping, et al. An Effective Discrete Differential Evolution Algorithm for Scheduling Uniform Parallel Batch Processing Machines With Non-identical Capacities and Arbitrary Job Sizes[J]. International Journal of Production Economics, 2016, 179: 1-11.