

# 并行多机开放车间调度问题的模型与算法

陈亚绒 黄佩钰 李 沛 周富得 黄沈权

温州大学机电工程学院,温州,325035

**摘要:**发光二极管制造过程中,晶粒分类拣选工序的调度问题是典型的并行多机开放车间调度问题,属于 NP-hard 问题。研究了该调度问题以最小化总加权完工时间为目标的求解模型与算法。根据问题特性构建了可获得最优解的混合整数规划模型,并设计了同时考虑质量与求解效率的启发式算法和改进粒子群优化算法。仿真结果显示,启发式算法和改进粒子群优化算法都能在合理的时间内迅速有效地获得较佳的调度解。

**关键词:**开放车间调度问题;发光二极管;加权完工时间;改进粒子群优化算法;晶粒分类拣选

**中图分类号:**F224

**DOI:**10.3969/j.issn.1004-132X.2018.22.003

**开放科学(资源服务)标识码(OSID):**



## Model and Algorithm of Parallel Multiprocessor Open Shop Scheduling Problem

CHEN Yarong HUANG Peiyu LI Pei CHOU Fuhder HUANG Shengquan

College of Mechanical and Electronic Engineering, Wenzhou University, Wenzhou, Zhejiang, 325035

**Abstract:** Grain sorting scheduling problem in LED manufacturing was a typical multiprocessor open shop scheduling problem, which belonged to NP-hard problem. Model and algorithm for solving this kind of scheduling problems with the objective of minimizing total weighted completion time were studied herein. According to the characteristics of the problem, a mixed integer-programming model was constructed to obtain the optimal solution, and a heuristic algorithm and an improved particle swarm optimization algorithm were designed. Simulation results show that both heuristic algorithm and improved particle swarm optimization algorithm may obtain good scheduling solutions quickly and effectively in a reasonable time.

**Key words:** open shop scheduling problem; light emitting diode (LED); weighted completion time; modified particle swarm optimization; grain sorting

## 0 引言

开放车间调度问题(open shop scheduling problem, OSSP)一般描述为将有限集合  $J = \{j_1, j_2, \dots, j_n\}$  中的  $n$  个工件安排在车间的  $s$  个阶段(或道次)的机器上加工,  $k$  阶段机器的数量为  $m_k$  ( $k = 1, 2, \dots, s$ )。每个工件都包含  $s$  道工序,每道工序在每台机器上的加工时间确定。在给定的时间内,每台机器只能加工一个工件,且每个工件只能由一台机器加工。同一台机器上工件的加工顺序任意,每个工件的加工顺序也无限制。若  $k$  阶段机器的数量  $m_k = 1$ ,则这类调度问题称为传统开放车间调度问题;若  $m_k \geq 1$  且  $s \geq 2$ ,同时每个阶段的机器为同类等效机,则此类问题称为并行多机 OSSP<sup>[1-3]</sup>。

晶粒分类拣选工序是发光二极管(light emitting diode, LED)制造过程中的关键工序,该工序的生产调度问题是典型的并行多机 OSSP,其调度算法是否有效直接影响 LED 制造工厂的生产<sup>[4]</sup>。

开放车间调度问题具有 NP-hard 属性<sup>[5]</sup>,因此许多学者开发启发式算法<sup>[6-7]</sup>以及智能优化算法如遗传算法<sup>[8-10]</sup>、蚁群算法<sup>[11]</sup>与粒子群算法<sup>[12-13]</sup>,以期在较短的时间内得到近似最优解。考虑到现实生产的动态与随机性,采用仿真方法的研究日益受到关注。如文献[14]研究了一种基于仿真的实时调度方法,来优化释放时间随机的非抢占开放车间调度问题。研究的调度目标主要是最小化 Makespan 或总完工时间<sup>[7-8, 10-12]</sup>、最小化平均流程时间<sup>[6]</sup>、最小化总拖期<sup>[9]</sup>。比较而言,有关并行多机 OSSP 的研究相对较少,BAI 等<sup>[15]</sup>研究了基于 GDS 求解大规模 OSSP 的启发式算法,以及求解中等规模 OSSP 的差分进化算法;针对实际问题中工件批量大

**收稿日期:**2018-04-28

**基金项目:**国家自然科学基金资助项目(51705370, 71501143);

浙江省自然科学基金资助项目(LY18G010012, LY19G010007);

温州市公益性科技计划资助项目(G20170006)

于 1 的需求, WANG 等<sup>[16]</sup>提出了分割求解工件批量的并行多机 OSSP 的差分进化算法; MATTA<sup>[17]</sup>提出了一种求解并行多机 OSSP 的计算机搜索方法; CHEN 等<sup>[18]</sup>研究了每个阶段两台机器的 OSSP, 提出了求解该问题的多项式时间近似算法。文献[19]提出了一种基于网络流求解并行多机可中断 OSSP 的调度算法。这些文献的调度目标均是完工时间最小化。以当前的研究文献为基础, 结合晶粒分类拣选工序的生产特性分析, 本文以最小化总加权完工时间为目标, 构建了并行多机 OSSP 的混合整数规划模型, 设计了可以快速有效获得较佳解的启发式算法及改进粒子群优化算法, 并通过实验仿真比较了不同算法的性能。

### 1 晶粒分类拣选工序的调度问题描述

调研 LED 制造工厂得知, 根据客户标准, 晶圆片上的每一颗晶粒都可以归类到 128 种晶粒等级中的一种。每一台晶粒分类拣选设备/机台最多只能分类 32 种等级的晶粒。已设定分类某些等级晶粒的设备更改为其他等级时, 需耗费 8h 的程序设定时间。实际生产过程中, 管理者一般依照经验将 128 种晶粒等级归类成 4~8 群, 每个群最多包含 32 种特定的晶粒等级; 然后将若干台分类拣选设备设定成专门分类拣选“专属群”的晶粒等级。每一片晶圆都要历经 4 道以上不同“专属群”的加工工序(没有先后限制), 每道工序的加工设备是多台同类平行设备(图 1), 因此该调度问题是典型的并行多机开放车间调度问题  $O(P_{m1}, P_{m2}, \dots, P_{ms})/r_j/\sum w_j c_j$ 。

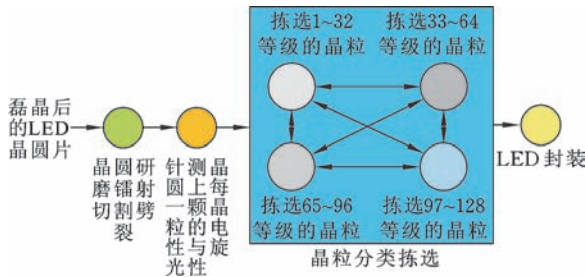


图 1 晶粒分类拣选工序加工示意图

Fig.1 Schematic diagram of grain sorting process

根据上述的生产作业环境, 本文探讨的晶粒分类拣选工序生产调度问题描述如下: ①磊晶工序之后的每一片晶圆都是一个独立的工件; ②经过电性与旋光性针测工序, 晶圆  $j$  上的晶粒等级、位置与数量确定且已知; ③晶圆  $j$  释放(或抵达)到晶粒分类拣选工序的时间为  $r_j$ ; ④晶圆  $j$  要经过  $s$  道晶粒分类拣选设备的加工(没

有先后限制), 工序  $k$  包含  $m_k$  台同型设备; ⑤每台设备从晶圆片上拣选 1 颗晶粒所需的时间确定且已知; ⑥每片晶圆在工序  $k$  上的作业时间等于这道工序设备拣选的晶粒总数量与拣选 1 颗晶粒所需时间的乘积; ⑦每台晶粒分类拣选设备一次最多只能分拣 1 片晶圆, 且不允许抢占; ⑧调度期间内, 不考虑晶粒分类拣选设备发生故障停机的情形; ⑨晶圆  $j$  的经营管理绩效重要程度即权重确定且已知; ⑩生产绩效是最小化总加权完工时间。

### 2 晶粒分类拣选调度问题的混合整数规划模型

针对晶粒分类拣选工序的并行多机、加工工序没有先后限制的生产特性, 构建了求解此类调度问题的混合整数规划(mixed integer programming, MIP)模型。

晶粒分类拣选工序调度问题 MIP 模型的目标是最小化所有工件的总加权完工时间。

最小化总加权完工时间为

$$\min \sum_{j=1}^n w_j F_j \quad (1)$$

式中,  $w_j$  为工件  $j$  的权重系数;  $F_j$  为工件  $j$  的完工时间;  $n$  为晶圆或工件数量。

约束条件如下:

(1)每个工件在各个道次间执行的先后顺序关系为

$$\sum_{k=1}^s \sum_{l=1}^s X_{jkl} = \frac{s(s-1)}{2} \quad (2)$$

$$X_{jkl} = \begin{cases} 1 & \text{工件 } j \text{ 的 } k \text{ 道次加工在 } l \text{ 道次前} \\ 0 & \text{其他} \end{cases} \quad (3)$$

(2)一个工件的一个道次中只允许在该道次的多台并行机台中的一台机器上加工, 即有

$$\sum_{g=1}^{m_k} Z_{jkg} = 1 \quad (4)$$

$$Z_{jkg} = \begin{cases} 1 & \text{工作 } j \text{ 在道次 } k \text{ 的机台 } g \text{ 上加工} \\ 0 & \text{其他} \end{cases} \quad (5)$$

(3)每个工件在每个道次中的完工时间必须不小于此工件的释放时间与此道次加工时间之和, 即

$$C_{jk} \geq r_j + \sum_{g=1}^{m_k} Z_{jkg} p_{jk} \quad (6)$$

式中,  $C_{jk}$  为工件  $j$  在道次  $k$  的完工时间;  $r_j$  为工件  $j$  的释放时间;  $p_{jk}$  为工件  $j$  在道次  $k$  的加工时间。

(4)每个工件最终的完工时间必须不小于此工件在每个道次的完工时间, 即

$$F_j \geq C_{jk} \quad (7)$$

(5) 每个工件在任意两个道次间的完工时间必须满足先后顺序关系,该先后顺序关系的数学表述为

$$C_{jl} \geq C_{jk} + \sum_{g=1}^{m_k} Z_{jkg} p_{jk} - M(1 - X_{jkl}) \quad (8)$$

式中,  $M$  为很大的正整数;  $C_{jl}$  为工件  $l$  的工序完工时间。

(6) 任意两个工件在同一道次的同一台机器上加工时,必定有先后顺序的关系;反之,则没有先后顺序的关系,该先后顺序关系的数学表述为

$$\left. \begin{aligned} Y_{ijk_g} + Y_{jik_g} &\leq 1 \\ (Z_{ik_g} + Z_{jk_g}) - 2(Y_{ijk_g} + Y_{jik_g}) &\geq 0 \\ (Z_{ik_g} + Z_{jk_g}) - (Y_{ijk_g} + Y_{jik_g}) &\leq 1 \end{aligned} \right\} \quad (9)$$

$$Y_{ijk_g} = \begin{cases} 1 & \text{道次 } k \text{ 的机台 } g \text{ 上的工件 } i \\ & \text{在工件 } j \text{ 之前加工} \\ 0 & \text{其他} \end{cases} \quad (10)$$

(7) 任意两个工件在同一道次的同一台机器上同时加工,两个工件之间的完工时间必须满足先后顺序关系,该先后顺序关系的数学表述为

$$C_{jk} \geq C_{ik} + \sum_{g=1}^{m_k} Z_{jkg} p_{jk} - M(1 - Y_{ijk_g}) \quad (11)$$

晶粒分类拣选工序的生产调度问题属于 NP-hard 问题,建立的 MIP 模型主要用于验证后续设计的启发式算法与改进粒子群优化算法结果的正确性,并将 MIP 模型求得的解作为评估调度算法求解质量的基准。

### 3 晶粒分类拣选调度问题的启发式算法

尽管 MIP 模型可以求解晶粒分类拣选工序的生产调度问题,但当工件数量或晶粒分类拣选机器的数量急剧扩大时,MIP 模型无法在可接受的时间内获得最优解,甚至无法获得任何一组可行解。考虑求解效率与现实运作要求,有必要设计迅速且有效获得满意可行调度解的启发式算法。

本文提出的启发式算法将整个求解过程细分成多个阶段,每个阶段只针对一台晶粒分类拣选机器与一个工件进行指派。阶段与阶段之间相互关联,即  $k+1$  阶段的求解是在  $k$  阶段获得的可行解基础上进行的,具体执行步骤如下:

(0) 令时间初始值  $t \leftarrow 0$ 。

(1) 从所有晶粒分类拣选机器中找出还有工件需要加工的机器,将每台满足机器可用时间  $m_a \leq t$  的晶粒分类拣选机器放到候选晶粒分类拣选机器集合。

(2) 若候选晶粒分类拣选机器集合为空集,则令  $t \leftarrow t+1$ ,执行步骤(1);否则,根据晶粒分类拣选机器的优先处理规则,从此候选晶粒分类拣选

机器集合中选择晶粒分类拣选机器。

(3) 筛选所有还需要在步骤(2)选择的晶粒分类拣选机器上加工处理的工件,将每个满足工件释放时间  $w_r \leq t$  的工件纳入到候选工件集合之中。

(4) 若候选工件集合为空集,则令  $t \leftarrow t+1$ ,执行步骤(1);否则,根据工件的优先处理规则,从此候选工件集合中选择工件。

(5) 更新此阶段晶粒分类拣选机器的可用时间  $m_a$ 、工件的释放时间  $w_r$ 。

(6) 若尚有工件未指派处理完成,则返回执行步骤(1);否则计算目标函数值,结束启发式算法。

晶粒分类拣选机器的优先处理规则是“未来剩余工件的平均负荷大者优先指派”。若晶粒分类拣选机器的平均负荷相同,则机器序号小者优先。工件的优先处理规则是“ $p_j/w_j$  小者优先指派”。若某道次工序当中的  $p_j/w_j$  相同,则工件序号小者优先指派。

### 4 晶粒分类拣选调度问题的改进粒子群优化算法

改进粒子群优化(modified particle swarm optimization, MPSO) 算法的基本概念与传统 PSO 相同,只在运行机制上进行了改进。

#### 4.1 编码

传统 PSO 算法主要适用于求解连续空间域的优化问题。针对晶粒分类拣选调度问题的离散空间域组合优化问题特征,本文采用随机数大小顺序值排列(rank order value, ROV)编码方式,将连续空间域的粒子位置映像转换为离散空间域的调度解粒子<sup>[20]</sup>。

晶粒分类拣选调度问题的可行解包含有  $n$  个工件在  $s$  道次机台上的工序加工顺序决策,因此,粒子  $i$  在时间  $t$  时必须具备  $sn$  维元素的向量  $X_i(t) = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(n)}, x_i^{(n+1)}, \dots, x_i^{(sn)})$ 。某 5 个工件在 4 道工序进行加工范例的连续空间域的粒子编码为(0.859, 0.600, 0.022, 0.393, 0.800, 0.326, 0.963, 0.984, 0.751, 0.149, 0.403, 0.065, 0.842, 0.097, 0.429, 0.138, 0.439, 0.602, 0.970, 0.339), ROV 映像转换后,离散空间域的编码为(3, 12, 14, 16, 10, 6, 20, 4, 11, 15, 17, 2, 18, 9, 5, 13, 1, 7, 19, 8)。ROV 映像转换的编码方式简单、易懂、可行,但存在多对一映像转换的问题。连续空间域中处于不同位置的 2 个粒子

在 ROV 映像转换后,离散空间域编码有可能相同。对此,本文提出的解决方案如下:

(1)对每个粒子使用 ROV 方法,将连续空间域的粒子位置映像转换成离散空间域编码。

(2)针对经过 ROV 映像转换后的每个粒子,利用 Hash 函数来计算此粒子的类似索引码,通过此数值来判断 2 个粒子的离散空间域编码是否重复。

(3)若粒子的离散空间域编码重复,则基于粒子多样性的需求,运用随机数重新产生一个粒子来取代此粒子。

#### 4.2 解码

ROV 映像转换后,粒子的离散空间域编码还需结合晶粒分类拣选调度问题的特性进行解码,通过解码程序来决定如何指派每个工件在各个加工道次的先后顺序,以及如何指派每

个工件在一台机器设备上的先后加工顺序。本文提出的解码方式如下:将 ROV 映像转换后的粒子的离散空间域编码利用(道次  $s =$

$\text{int}[\frac{\text{ROV}(x_i^{(n)})-1}{n}]+1$ 、工件号  $j = \text{mod}[\frac{\text{ROV}(x_i^{(n)})-1}{n}]+1 \rightarrow O_{j,s}$  进行解码,其中,  $\text{ROV}(x_i^{(n)})$  为元素值,  $\text{int}[\cdot]$  表示元素取整,  $\text{mod}[\cdot]$  为求余函数。ROV 映像转换后的离散空间域编码 (3, 12, 14, 16, 10, 6, 20, 4, 11, 15, 17, 2, 18, 9, 5, 13, 1, 7, 19, 8) 在解码后变成 ( $O_{31}, O_{23}, O_{43}, O_{14}, O_{52}, O_{12}, O_{54}, O_{41}, O_{13}, O_{53}, O_{24}, O_{21}, O_{34}, O_{42}, O_{51}, O_{33}, O_{11}, O_{22}, O_{44}, O_{32}$ ), 然后依照此顺序一一予以指派工件。

#### 4.3 算法运行机制

MPSO 算法运行机制的伪代码如下。

```

Initialize parameters of  $N_p, \delta, c_1, c_2, \text{reduce\_rate}$ , and  $\text{time\_limit}$ 
Generate initial position  $X_i(0)$  and velocity  $V_i(0)$  for particle  $i, \forall i=1, 2, \dots, N_p$ 
Set  $P_i(0)=X_i(0), G_b(0)=X_1(0), G_w(0)=X_1(0), \text{timer}=0, t=0$ 
Do
  For  $i=1$  to  $N_p$  // 计算粒子群内每一个粒子的位置  $P_i(t)$ 
    If  $Z(X_i(t)) \leq Z(P_i(t))$  then
       $P_i(t) = X_i(t)$ 
    End if
  Next  $i$ 
   $\text{best} = 1, \text{worst} = 1$  // 找出当前粒子群中最好与最差的位置  $G_b(t), G_w(t)$ 
  For  $i=1$  to  $N_p$ 
    If  $Z(X_i(t)) \leq Z(P_{\text{best}}(t))$  then
       $\text{best} = i$ 
    End if
    If  $Z(X_i(t)) \geq Z(P_{\text{worst}}(t))$  then
       $\text{worst} = i$ 
    End if
  Next  $i$ 
  If  $Z(P_{\text{best}}(t)) < Z(G_b(t))$  then // 局部搜寻
     $G_b(t) = P_{\text{best}}(t)$ 
    Local_search( $G_b(t)$ ) // 两两交换局部搜寻
  End if
  If  $Z(P_{\text{worst}}(t)) > Z(G_w(t))$  then // 淘汰群内最差的粒子
     $G_w(t) = P_{\text{worst}}(t)$ 
    If random number  $\leq \text{threshold}$  then
      Generate a new particle randomly
      Replace  $P_{\text{worst}}(t), X_{\text{worst}}(t), V_{\text{worst}}(t)$ 
    End if
  End if
  For  $i=1$  to  $N_p$  // 更新每一个粒子的速度与位置
     $V_i(t+1) = \delta \cdot V_i(t) + c_1 \cdot r_1 \cdot (P_i(t) - X_i(t)) + c_2 \cdot r_2 \cdot (G_b(t) - X_i(t)), V_i(t+1) \in [V_{\min}, V_{\max}]$ ,
     $X_i(t+1) = X_i(t) + V_i(t+1), X_i(t+1) \in [X_{\min}, X_{\max}]$ 
  Next  $i$ 
  Similarity check for a couple of particles

```



If similarity check is true, then

Generate a new particle randomly and replace one of the particles by the new one

End if

$t = t + 1, \delta = \delta \times \text{reduce\_rate}, \delta \in [\delta_{\min}, \delta_{\max}], \text{update timer}$

While ( $\text{timer} \leq \text{time\_limit}$ )

除编码与解码之外, MPSO 算法与传统 PSO 算法的不同之处如下:

(1) 用惯性权重来控制粒子群优化搜索过程中整体探索与局部探索的比重。初期着重于整体性探索, 惯性权重比较大; 随着运行时间的增加, 搜索偏重于局部性探索, 则惯性权重逐步减小。

(2) 执行一次迭代后, 为了增加群体多样性, 将当前目标函数值最差的粒子淘汰, 然后随机产生一个新的粒子来替代。

(3) 为了增加搜索的有效性, 假使当前目标函数值最好的粒子改变时, 借由两两对调局部搜索 (pairwise interchange local search) 机制来加以辅助运行, 改善搜索的有效性。

## 5 仿真实验与分析

### 5.1 实验数据生成

所有的实验仿真测试均在台式计算机 (Intel Xeon E5-1630 3.7GHz CPU、内存 12 GB RAM) 上完成。MIP 法使用 IBM ILOG CPLEX Optimization Studio Ver.12.7.1 优化软件包求解, 每组算例求解执行的最大可接受时间设定为 2 h。启发式算法、MPSO 算法以及传统 PSO 算法运用 GCC C++ 编程, 程序操作系统为 LINUX (Ubuntu 14.04 LTS)。实验测试的数据参照现实作业环境随机生成, 每一个数值均为独立且服从均匀分布。工件的权重  $w_j$  为  $U[1, 5]$ 、释放时间  $r_j$  为  $U[0, 25]$  与  $U[0, 75]$ , 加工处理作业时间  $p_{jk}$  为  $U[1, 10]$  与  $U[1, 20]$ ,  $k$  道道工序所配备的机器数量  $m_k$  为  $U[1, 3]$  与  $U[1, 5]$ 。工件数量为 3、5、7、9、10、11、13、15、20、25、30、4 种实验参数组合成的 8 个组合情境均随机产生 10 组仿真测试数据, 因此每种工件数量共有 80 组算例数据。MPSO 算法的参数包括粒子群数 (工件数量  $n$ )、惯性权重 1、自我学习权重 2 以及社会学习权重 2, 这些参数均经过实验设计测试进行设置。算法停止的条件设定为  $0.1n$  s, 每组算例数据均执行 5 次, 从中选择最好的结果。

### 5.2 实验结果分析

采用求解质量偏差百分比  $P_D = (A - O) / O$  来评估不同算法的求解质量, 其中,  $A$  为算法求

得的解,  $O$  为评估基准值。当工件数量较小时,  $O$  为 MIP 获得的最优解。当工件数量较大时, MIP 无法在 2 h 内获得最优解,  $O$  为 2 h 内取得的上界值  $B_U$  与下界值  $B_L$ , 实际的  $P_D$  在根据  $B_U$  与  $B_L$  计算得到的  $P_D$  之间。

#### 5.2.1 各种实验参数之下四种算法的绩效比较

通过初步分析实验结果发现, 当工件数量  $n$  处于一定范围时, 算法的绩效相似, 因此将其分成 3 个区间进行分析。MIP 法、启发式算法、MPSO 算法与传统 PSO 算法在各种实验参数之下的绩效如表 1 所示。绩效用解目标值  $R$  与运行时间  $T$  表示。工件数量  $n$  增大时, MIP 法无法在最大可接受时间内获得最优解, 其解目标值细分为  $B_U$  与  $B_L$ 。

由表 1 可知, 对于  $p_{jk}$ 、 $m_k$ 、 $r_j$  的不同组合, 当求解问题的规模较小 ( $n \leq 5$ ) 时, MIP 法、MPSO 算法、传统 PSO 算法与启发式算法均能获得问题的最优解或近优解。问题规模较大 ( $5 < n \leq 10$ ) 时, MIP 法在可接受时间内获得大部分算例的最优解, MPSO 算法和传统 PSO 算法能快速获得近优解, 且优于启发式算法的解。当问题规模继续增大 ( $n \geq 11$ ) 时, MIP 法在可接受时间内只能获得少部分问题的最优解, 其  $B_U$  远大于 MPSO 算法和传统 PSO 算法获得的解目标值, 且劣于启发式算法的解目标值。这验证了 MIP 法的求解运行时间与能力和问题的规模或复杂度强相关, 表明 MIP 法只适用于求解小规模的问题, 启发式算法几乎都不需要运行时间, 但其获得的解不如 MPSO 算法和传统 PSO 算法的解。因此, MPSO 算法与传统 PSO 算法更适合快速求解各种规模的晶粒分类拣选调度问题, 特别是大规模问题 ( $n \geq 011$ ), 并且 MPSO 算法的运行机制使得其获得的解优于传统 PSO 算法。

#### 5.2.2 求解质量分析

不同加工处理作业时间  $p_{jk}$ 、 $k$  次工序所配备的机器数量  $m_k$ 、工件数量  $n$  以及工件的释放时间  $r_j$  等参数组合之下, 启发式算法、MPSO 算法与传统 PSO 算法的求解质量偏差  $P_D$  如表 2 所示。

表 1 4 种算法在不同参数组合下的绩效汇总表  
Tab.1 Performance summary of four algorithms under the different parameters combination

$p_{jk}$	$r_j$	$m_k$	$n$	MIP 法			启发式算法		MPSO 算法		传统 PSO 算法	
				$B_U$	$B_L$	$T(s)$	$R$	$T(s)$	$R$	$T(s)$	$R$	$T(s)$
U[1,10]	U[0,25]	U[1,3]	$n \leq 5$	414	414	57.575	417	0	414	0.400	414	0.400
			$5 < n \leq 10$	955	921	3 868.676	1 004		954	0.867	956	0.867
			$n \geq 11$	3 199	1 751	7 202.976	3 150		2 804	1.900	2 815	1.900
		U[1,5]	$n \leq 5$	387	387	36.895	387		387	0.400	387	0.400
			$5 < n \leq 10$	950	923	4 585.549	971		950	0.867	950	0.867
			$n \geq 11$	2 882	1 708	6 865.067	2 804		2 564	1.900	2 569	1.900
	U[0,75]	U[1,3]	$n \leq 5$	714	714	30.780	714		714	0.400	714	0.400
			$5 < n \leq 10$	1 472	1 455	3 700.886	1 486		1 472	0.867	1 472	0.867
			$n \geq 11$	4 094	3 138	6 997.783	3 925		3 747	1.900	3 750	1.900
		U[1,5]	$n \leq 5$	729	729	57.570	729		729	0.400	729	0.400
			$5 < n \leq 10$	1 625	1 589	6 620.550	1 633		1 625	0.867	1 625	0.867
			$n \geq 11$	3 789	3 145	7 107.378	3 644		3 556	1.900	3 559	1.900
U[1,20]	U[0,25]	U[1,3]	$n \leq 5$	675	675	55.392	682		675	0.400	675	0.400
			$5 < n \leq 10$	1 578	1 536	3 367.686	1 677		1 578	0.867	1 579	0.867
			$n \geq 11$	5 038	2 693	7 106.458	4 883		4 455	1.900	4 462	1.900
		U[1,5]	$n \leq 5$	689	689	93.871	694		689	0.400	689	0.400
			$5 < n \leq 10$	1 492	1 399	4 897.421	1 550		1 489	0.867	1 490	0.867
			$n \geq 11$	5 038	2 693	7 106.458	4 883		4 455	1.900	4 462	1.900
	U[0,75]	U[1,3]	$n \leq 5$	954	954	12.790	964		954	0.400	954	0.400
			$5 < n \leq 10$	2 091	2 060	3 645.243	2 171		2 092	0.867	2 093	0.867
			$n \geq 11$	7 182	4 148	7 157.131	6 887		6 255	1.900	6 273	1.900
		U[1,5]	$n \leq 5$	910	910	80.342	910		910	0.400	910	0.400
			$5 < n \leq 10$	2 009	1 959	4 966.592	2 053		2 009	0.867	2 009	0.867
			$n \geq 11$	5 897	4 216	7 094.846	5 814		5 496	1.900	5 505	1.900

表 2 不同参数组合下 3 种算法的求解质量偏差  $P_D$   
Tab.2 The solution quality deviation percentage  $P_D$  of the three algorithms under the different parameters combination %

$p_{jk}$	$m_k$	$n$	$r_j$											
			U[0,25]						U[0,75]					
			$B_U$			$B_L$			$B_U$			$B_L$		
			启发式 算法	MPSO 算法	传统 PSO 算法	启发式 算法	MPSO 算法	传统 PSO 算法	启发式 算法	MPSO 算法	传统 PSO 算法	启发式 算法	MPSO 算法	传统 PSO 算法
U[1,10]	U[1,3]	$n \leq 5$	0.79	0	0	0.79	0	0	0.01	0	0	0.01	0	0
		$5 < n \leq 10$	5.08	(0.04)	0.11	8.61	3.35	3.50	0.95	0.02	0.02	1.94	1.00	1.00
		$n \geq 11$	1.72	(8.00)	(7.68)	67.83	50.35	50.92	(1.16)	(4.61)	(4.54)	20.21	15.48	15.57
	U[1,5]	$n \leq 5$	0	0	0	0	0	0	0	0	0	0	0	0
		$5 < n \leq 10$	2.03	0	0.04	4.98	2.92	2.95	0.40	0	0	2.63	2.22	2.22
		$n \geq 11$	(0.24)	(6.86)	(6.69)	52.23	40.60	40.88	(1.13)	(3.05)	(2.99)	13.28	10.85	10.93
U[1,20]	U[1,3]	$n \leq 5$	1.11	0	0	1.11	0	0	0.63	0	0	0.63	0	0
		$5 < n \leq 10$	5.87	0	0.09	8.76	2.77	2.86	3.73	0.06	0.09	5.21	1.50	1.53
		$n \geq 11$	(0.54)	(7.46)	(7.31)	66.16	52.70	52.95	(0.38)	(7.84)	(7.65)	54.49	41.47	41.82
	U[1,5]	$n \leq 5$	0.77	0	0	0.77	0	0	0	0	0	0	0	0
		$5 < n \leq 10$	3.63	(0.12)	(0.07)	10.07	6.09	6.14	2.14	0	0.02	4.57	2.41	2.43
		$n \geq 11$	(0.54)	(7.46)	(7.31)	66.16	52.70	52.95	0.28	(4.23)	(4.07)	31.83	25.27	25.49

注:括号里面的值代表  $P_D$  为负值,表示该算法的解比 MIP 法在 2 h 可接受时间内获得的最好解更接近最优解。

由表 2 所示的结果可知, MPSO 算法、传统 PSO 算法与启发式算法的求解质量偏差  $P_D$  不仅与工件数量有关, 也受  $p_{jk}$ 、 $m_k$  与  $r_j$  的影响。对于  $p_{jk}$ 、 $m_k$ 、 $r_j$  的不同组合, 随着求解问题规模的变大, 3 种算法  $B_U$  的求解质量偏差  $P_D$  基本呈现由小( $P_D=0$ )到大再到更小( $P_D<0$ )的趋势,  $B_L$  的  $P_D$  呈现由小到大的变化趋势。这表明当求解问题的规模足够大( $n\geq 11$ )时, 3 种算法获得的解优于或接近在可接受时间内 MIP 法获得的最优解, 且求解问题的规模越大, 求解质量偏差  $P_D$  的优势越明显。其中, MPSO 算法的求解质量优于传统 PSO 算法, 传统 PSO 算法的求解质量优于启发式算法。原因在于, 求解问题规模的变大导致可行解空间的增大, 进而造成 MIP 法在最大可接受时间获得的最好解与最优解的偏差变大。

5.2.3 实验参数的影响分析

为比较实验参数对启发式算法、MPSO 算法与传统 PSO 算法的影响, 统一将加工处理作业时间  $p_{jk}$  ( $U[1, 10]$  与  $U[1, 20]$ )、 $k$  道道工序所配备的机器数量  $m_k$  ( $U[1, 3]$  与  $U[1, 5]$ ) 以及工件的释放时间  $r_j$  ( $U[0, 25]$  与  $U[0, 75]$ ) 的两个水平取值分别标记为 -1 与 -2。3 种算法在  $p_{jk}$ 、 $m_k$  与  $r_j$  等参数不同取值之下的  $P_D$  如图 2~图 4 所示。

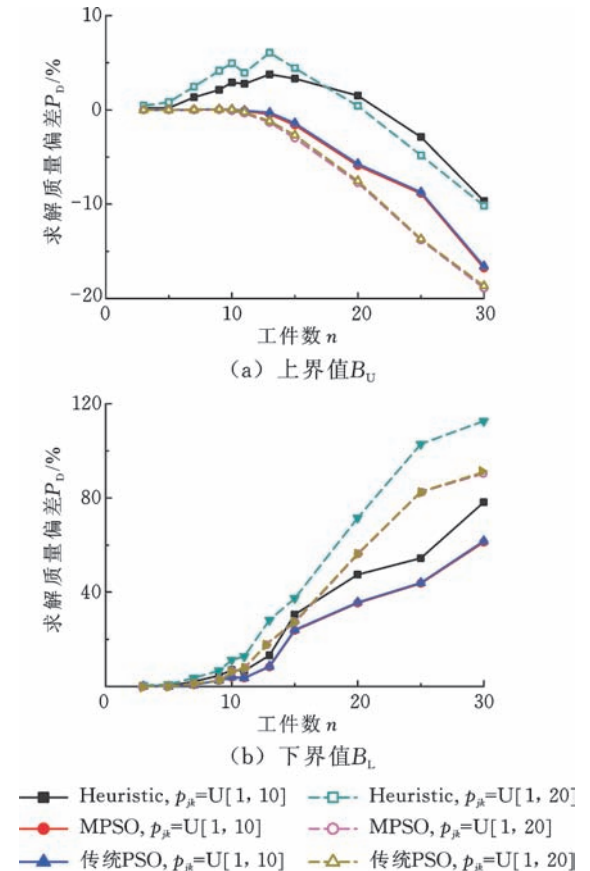


图 2  $p_{jk}$  对 3 种算法  $P_D$  影响

Fig.2 Influence of  $p_{jk}$  on  $P_D$  of three algorithms

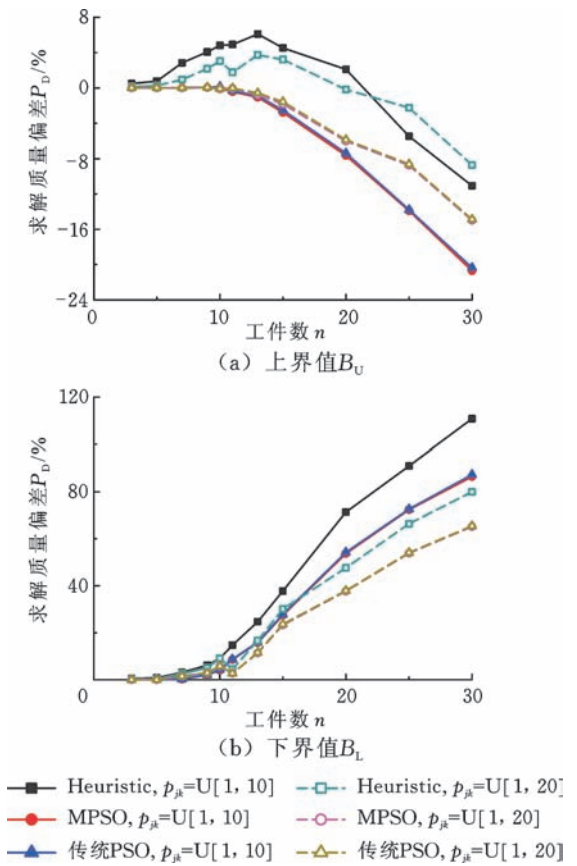


图 3  $m_k$  对 3 种算法  $P_D$  影响

Fig.3 Influence of  $m_k$  on  $P_D$  of three algorithms

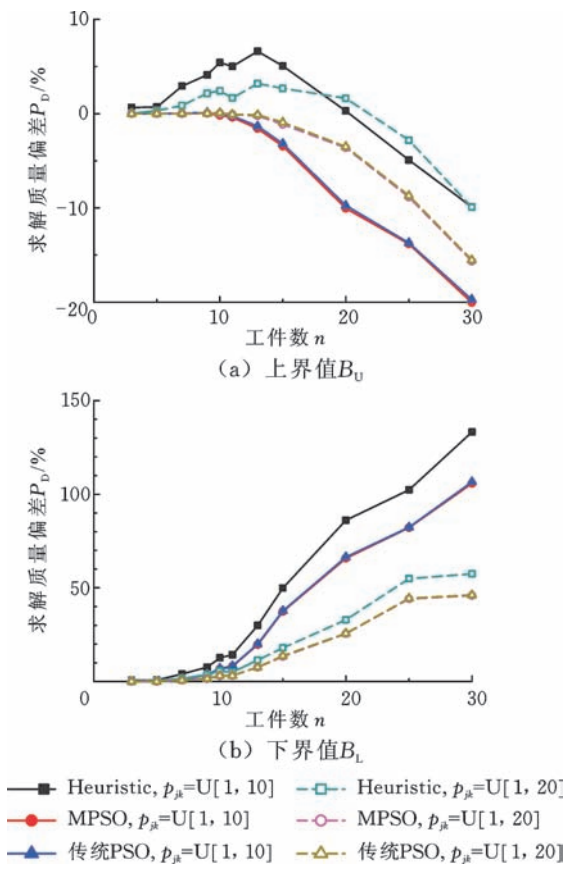


图 4  $r_j$  对 3 种算法  $P_D$  影响

Fig.4 Influence of  $r_j$  on  $P_D$  of three algorithms

由图 2~图 4 可知,实验参数  $p_{jk}$ 、 $m_k$  与  $r_j$  对启发式算法  $P_D$  的影响大于 MPSO 算法和传统 PSO 算法。问题的规模  $n=11$  时,实验参数的取值对 MPSO 算法和传统 PSO 算法的  $P_D$  几乎没影响,图 2~图 4 中显示为几乎重合的数据点。根据  $P_D$  的计算方法可知,若调度问题的复杂度或难度变高,MIP 法很难在可接受的时间内获得最优解,其他 3 种算法的  $B_U$  的  $P_D$  会变小, $B_L$  的  $P_D$  会变大。因此,总体上实验参数对算法的影响如下: $p_{jk}=U[1,20]$  的  $B_U$  的  $P_D$  优于  $p_{jk}=U[1,10]$  的  $B_U$  的  $P_D$ ,而  $p_{jk}=U[1,20]$  的  $B_L$  的  $P_D$  劣于  $p_{jk}=U[1,10]$  的  $B_L$  的  $P_D$ ;  $m_k=U[1,3]$  的  $B_U$  的  $P_D$  优于  $m_k=U[1,5]$  的  $B_U$  的  $P_D$ ,而  $m_k=U[1,3]$  的  $B_L$  的  $P_D$  劣于  $m_k=U[1,5]$  的  $B_L$  的  $P_D$ ;  $r_j=U[0,25]$  的  $B_U$  的  $P_D$  优于  $r_j=U[0,75]$  的  $B_U$  的  $P_D$ ,而  $r_j=U[0,25]$  的  $B_L$  的  $P_D$  劣于  $r_j=U[0,75]$  的  $B_L$  的  $P_D$ 。需要注意的是,实验参数对启发式算法  $B_U$  的  $P_D$  的影响与 MPSO 算法和传统 PSO 算法有所不同,在问题规模变大( $n \geq 15$ )时发生了转变。原因是实验参数取值对启发式算法求解质量的影响大于问题规模的影响。总之,当问题规模较大( $n > 11$ )时,3 种实验参数变化之下:MPSO 算法的  $P_D$  变化范围小于传统 PSO 算法,传统 PSO 算法的  $P_D$  变化范围小于启发式算法,这表明 MPSO 算法的稳健性最高。

## 6 结语

本文针对 LED 制造工厂的晶粒分类拣选调度问题的并行多机生产特性,建立了求解此类调度问题的混合整数规划模型,兼顾求解效率与质量,设计了一种简单易行的启发式算法和一种改进粒子群优化算法。根据企业的生产实践设计了实验算例,通过比较 4 种算法的调度绩效,发现启发式算法虽能快速求解,但是求解的质量仍然有改善的空间,改进粒子群优化算法在求解速度与质量的要求下均能满足企业晶粒分类拣选调度的实务要求。同时,对于加工处理时间、释放时间以及每一道次工序所配备的机器数量等参数的变化,改进粒子群优化算法的变化范围最小、最稳健。未来研究方向将侧重于考虑将改进粒子群优化算法用于其他的调度问题,以及探讨晶粒分类拣选工序的生产调度问题的多目标优化求解方法。

## 参考文献:

- [1] ANAND E,PANNEERSELVAM R.Literature Review of Open Shop Scheduling Problems[J].Intelligent Information Management,2015,7(1):33-52.
- [2] NADERI B,GHOMI S M T F,AMINNAYERI M,et al.Scheduling Open Shops with Parallel Machines to Minimize Total Completion Time[J].Journal of Computational and Applied Mathematics,2011,235(5): 1275-1287.
- [3] ABDELMAGUID T F,SHALABY M A,AWWAD M A.A Tabu Search Approach for Proportionate Multi-processor Open Shop Scheduling [J]. Computational Optimization & Applications,2014,58(1): 187-203.
- [4] SHIANG W J,LIN Y H,RAU H.Application of Simulation to the Scheduling Problem for a LED Sorting System[C]//IEEE International Conference on Machine Learning and Cybernetics,Baoding,China: ICMLC,2009: 2875-2879.
- [5] ROEMER T A.A Note on the Complexity of the Concurrent Open Shop Problem [J]. Journal of Scheduling,2006,9(4): 389-396.
- [6] BRÄSEL H,HERMS A,MÖRIG M,et al.Heuristic Constructive Algorithms for Open Shop Scheduling to Minimize Mean Flow Time[J].European Journal of Operational Research,2008,189(3): 856-870.
- [7] TANG L,BAI D.A New Heuristic for Open Shop Total Completion Time Problem[J].Applied Mathematical Modelling,2010,34(3):735-743.
- [8] AHMADIZAR F,FARAHANI M H.A Novel Hybrid Genetic Algorithm for the Open Shop Scheduling Problem[J].International Journal of Advanced Manufacturing Technology,2012,62(5/8):775-787.
- [9] NADERI B,FATEMI GHOMI S M T,AMINNAYERI M,et al.A Study on Open Shop Scheduling to Minimize Total Tardiness [J].International Journal of Production Research,2011,49(15): 4657-4678.
- [10] 王军强,郭银洲,崔福东,等.基于多样性增强的自适应遗传算法的开放式车间调度优化[J].计算机集成制造系统,2014,20(10):2479-2493.  
WANG Junqiang, GUO Yinzhou, CUI Fudong, et al.Diversity Enhancement-based Adaptive Genetic Algorithm for Open-shop Scheduling Problem[J]. Computer Integrated Manufacturing System,2014, 20(10): 2479-2493
- [11] HUANG Y M,LIN J C.A New Bee Colony Optimization Algorithm with Idle-time-based Filtering Scheme for Open Shop-scheduling Problems[J].Expert Systems with Applications,2011,38(5):5438-5447.