DOI:10.11918/j.issn.0367-6234.201603051

带多重加工前约束的单机 MOPJ 调度方法

周炳海, 王 科

(同济大学 机械与能源工程学院, 上海 201804)

摘 要: 为有效解决晶圆加工过程中带换模时间、品种间晶舟分配的不确定性以及参数调整等多重加工前约束的单机单作业多订单 MOPJ(multi-order-per-job)调度问题,对问题域进行描述,以订单总完成时间最小为优化目标,建立数学规划模型. 给出求解较优调度解的定理,并提出具有双层嵌套编码机制的混合差分进化的入侵杂草调度算法,该算法引入具有学习机制的算子以改善解的质量. 为有效提高算法的收敛性,在变异及邻域操作中考虑自适应过程. 仿真实验结果表明,该算法是有效且可行的,优化晶舟分配的调度较未优化的调度可提高至少 10%的性能.

关键词: 单作业多订单调度;差分进化;入侵杂草;自适应;晶舟分配

中图分类号: TP391 **文献标志码**: A **文章编号**: 0367-6234(2017)07-0158-07

Scheduling method of multi-order-per-job for a single machine with multiple preprocess constraints

ZHOU Binghai, WANG Ke

(School of Mechanical Engineering, Tongji University, Shanghai 201804, China)

Abstract: To efficiently address the multi-order-per-job (MOPJ) scheduling problem of a single machine with multiple preprocess constraints in wafer fabrications, including setup time, uncertain allocation of front opening unified pods (FOUPs), machine parameter adjustment, a scheduling problem domain was described and a mathematical programming model was set up with an objective of minimizing total completion time, and several theorems were established to obtain superior feasible solutions, in addition, a hybrid invasive weed optimization algorithm combined with differential evolution and adopted a two-level encoding mechanism was developed, in which the learning mechanism was introduced to enhance the quality of the solution. Moreover, adaptive process was applied to the mutation and neighborhood search to effectively improve the algorithm convergence. Finally, simulation results verify the validness and feasibility of the proposed algorithm and show that a 10% improvement is made on the performance by the scheduling approach.

Keywords: MOPJ scheduling; differential evolution; invasive weed; adaptive strategy; allocation of FOUPs

目前,已经有很多学者对单一产品、不考虑换模时间等因素的单机单作业多订单(multi-order-per-job,MOPJ)问题提出了多种调度方法. 文献[1]对单一产品且不考虑其他因素的理想化单机 MOPJ 调度模型采用禁忌搜索算法进行了简单的研究. 文献[2]对理想化单机 MOPJ 问题,提出了分组遗传算法. 文献[3]在理想化单机 MOPJ 调度模型的基础上,增加了订单准备时间因素,提出了列生成启发式算法并对其进行有效求解. 文献[4]针对理想化的单机 MOPJ 调度模型提出了启发式算法,可以求解超大规模订单的调度问题. 在单机 MOPJ 调度研究领域,文献[5]和[6]在批调度(batch scheduling)模型中考虑了多个品种. 但批调度的研究重点是对批次的调度,与本文研究的调度问题有较大区别. 文献[7]对多品种、考虑换模时间以及衰退效应约束的单机 MOPJ 调度问题进行了研究,但每个品种所使用的晶舟数量是确定的. 优化晶舟在每个品种间的分配可以明显提高 MOPJ 调度的效率,具有实用性和研究价值,目前鲜有文献对其进行研究.

参数调整是为保证某产品的加工质量,若设备连续加工其他产品超过一定阈值,则加工该产品前需要调整参数. 晶圆制造系统中,已不乏研究引入参数调整约束后的调度. 文献[8]在多作业族的单机调度模型中考虑了参数调整约束,提出了简单启发式算法. 文献[9]在多目标的并行机批调度问题中考虑了参数调整,并提出了蚁群优化算法. 在 MOPJ 调度问题中考虑参数调整以保证加工过程中的质量,具有实用性和普遍性,而目前鲜有文献探讨参数调整背景下的 MOPJ 调度问题.

本文研究的单机 MOPJ 调度问题以总加权完成

收稿日期: 2016-03-10
基金项目: 国家自然科学基金(71471135,61273035)
作者简介: 周炳海(1965—),男,教授,博士生导师
通信作者: 周炳海,bhzhou@tongji.edu.cn

时间最小为目标函数，同时考虑换模时间、参数调整、订单品种多样性等约束，优化订单的分配，晶舟的加工顺序以及晶舟在品种间的分配.

1　问题描述

以往研究单机 MOPJ 调度问题，主要包含订单成组形成作业和作业排序两个子问题. 如图 1 所示，本文所研究的问题还需确定如何在每个品种间分配晶舟(为与调度术语一致，本文将用作业指代晶舟)，同时在作业排序时，考虑了参数调整.

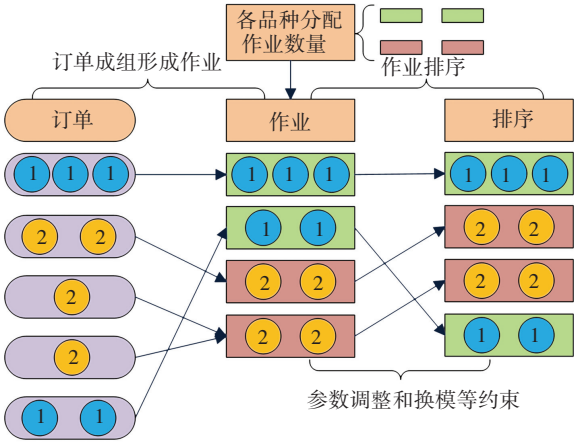

图 1　单机 MOPJ 调度问题实例

Fig.1　The instance of a single machine MOPJ scheduling problem

为有效地描述调度问题，在文献[7]的基础上，做如下假设：1)订单需保持完整性；2)每个订单只含有一种产品，不同订单的产品种类可以不同；3)每个作业只含有同个品种的多个订单；4)每一个作业中载有的晶圆数不能超过其承载上限；5)作业的数量能够满足调度要求；6)不同品种的订单形成的作业，属于不同的作业族，即一个品种对应一个作业族；7)设备连续加工不同作业族的两个作业时考虑换模时间；8)设备若超过一定次数未加工过某作业族，当加工该作业族的作业时需考虑参数调整时间；9)设备一旦开始工作，除换模外，不会中断工作；10)同一个作业中的所有订单有相同的完工时间；11)订单大小为 0 的订单，称为虚拟订单，虚拟订单在实际中不参与调度.

为方便描述，现定义符号如下：I 为订单数；I_f 为订单数最多的品种的订单数量；i_f 为订单索引，表示品种 f 的第 i 个订单；F 为总的品种数量；f 为品种索引，表示第 f 个品种；J 为总作业数；j_f 为作业族 f 的作业数量；j 为作业索引，表示第 j 个作业；N 为总的作业数量；Q 为晶舟承载的上限；$C_{i_f}^f$ 为品种 f 的第 i_f 个订单的完成时间；C_j 为作业 j 的完成时间；p_f 为品种 f 单位晶圆的加工时间；s_f 为品种 f 的换模时间；q_f 为品种 f 的参数调整时间；n_f 为品种 f 的参数调整的阈值；$\sigma_{i_f}^f$ 为品种 f 的第 i_f 个订单的大小；$Y_{i_f}^f$ 为品种 f 的第 i_f 个订单，若不为虚拟订单，则其值为 1，反之为 0.

决策变量：$X_{i_f j}^f$ 为若品种 f 的第 i_f 个订单被分配到 j 上，则其值为 1，反之为 0；Z_j^f 为若第 j 个作业属于作业族 f，则其值为 1，反之为 0；U_j 为若第 j 个作业需要换模，则其值为 1，反之为 0；V_j 为若第 j 个作业需要参数调整，则其值为 1，反之为 0.

由假设 1)和 11)可知，一个客户订单只能分配到一个作业中，虚拟订单不对应任何作业，即需满足

$$\sum_{j=1}^{J} X_{i_f j}^f Y_{i_f}^f = 1, \quad f = 1,2,\cdots,F,$$
$$i_f = 1,2,\cdots,I_f, \quad Y_{i_f}^f \neq 0.$$

由假设 2)~4)可知，一个作业至少包含一个订单，且作业的大小不得超出上限 Q，即需同时满足

$$\sum_{i_f=1}^{I_f} X_{i_f j}^f Y_{i_f}^f \sigma_{i_f}^f Z_j^f \leqslant Q, f = 1,2,\cdots,F,$$
$$j = 1,2,\cdots,J;$$
$$\sum_{f=1}^{F} \sum_{i_f=1}^{I_f} X_{i_f j}^f Y_{i_f}^f \geqslant 1, j = 1,2,\cdots,J.$$

由假设 2)、3)和 6)可知，一个作业只能属于一个作业族，订单的品种与作业族相对应，即满足

$$\sum_{f=1}^{F} Z_j^f = 1, j = 1,2,\cdots,J;$$
$$X_{i_f j}^f - Z_j^f \leqslant 0, j = 1,2,\cdots,J,$$
$$f = 1,2,\cdots,F, \quad i_f = 1,2,\cdots,I_f.$$

由假设 5)可知，所有的订单都应该被分配到作业中，当作业总数为一定值时，各个品种的作业的数量需满足调度要求，且参与调度的作业能被有效利用，即不同品种订单形成的作业的数量需满足

$$j_f \geqslant \begin{cases} \lfloor M \rfloor, & M - \lfloor M \rfloor = 0, \\ \lfloor M \rfloor + 1, & M - \lfloor M \rfloor \neq 0, \end{cases}$$
$$f = 1,2,\cdots,F;$$
$$j_f \leqslant \sum_{i_f=1}^{I_f} Y_{i_f}^f, f = 1,2,\cdots,F;$$
$$\sum_{f=1}^{F} j_f = N.$$

式中：M 为一正实数，$M = \sum_{i_f=1}^{I_f} Y_{i_f}^f \sigma_{i_f}^f / Q$，$\lfloor M \rfloor$ 表示不大于 M 的最大整数.

在假设 7)和文献[8]的基础上可知，设备前后连续加工不同作业族的两个作业时需考虑换模，即需满足

$$Z_j^f - Z_{j-1}^f \leqslant U_j, j = 1,2,\cdots,J,$$
$$f = 1,2,\cdots,F.$$

在假设 8)和文献[8]的基础上可知,设备若超过一定次数未加工过某作业族,当加工该作业族的作业时,需考虑参数调整时间,即需满足

$$Z_j^f - \sum_{j'=j-n_f}^{j-1} Z_j^f \leqslant V_j,\ j = 1,2,\cdots,J,$$
$$f = 1,2,\cdots,F.$$

由假设 9)可知,只有当前一个作业完成加工后,后一个作业才能进行加工,即需满足

$$C_{j-1} + p_f \sum_{f=1}^{F} \sum_{i_f=1}^{I_f} X_{i_f j}^f Y_{i_f}^f \sigma_{i_f}^f Z_j^f + \sum_{f=1}^{F} U_j s_f Z_j^f + \sum_{f=1}^{F} V_j q_f Z_j^f \leqslant C_j,\quad j = 1,2,\cdots,J.$$

由假设 10)可知,每一个订单的完成时间与其所分配作业的完成时间相等,即需满足

$$C_{i_f}^f = \sum_{j=1}^{J} C_j X_{i_f j}^f,\ f = 1,2,\cdots,F,$$
$$i_f = 1,2,\cdots,I_f.$$

根据文献[4],调度目标为最小化每个订单的完成时间之和,即最小化订单的总完成时间

$$\min \sum_{f=1}^{F} \sum_{i_f=1}^{I_f} C_{i_f}^f Y_{i_f}^f.$$

为了进一步深入分析问题,针对构建的模型,给出了相关的引理、定理.

定理 1 若有 m 个从小到大排序的相同品种的订单需要分配到连续两个作业 j 和 $j-1$ 中,x 表示 m 中的第 x 个订单,σ_i 表示第 i 个订单的大小,若 x 及之前的订单中最大订单小于 x 之后所有订单大小的 $1/(x-1)$,且 x 之后的订单不满足上述关系,同时 $\sum_{i=x+1}^{m} \sigma_i < Q$,$\sum_{i=1}^{x} \sigma_i < Q$,此时将 x 及之前的订单分配到 $j-1$ 作业,x 之后的订单分配到 j 作业中,所获得的方案具有较优的解.

证明 假设调度 S_1 为将 x 订单及其之前的订单分配到作业 $j-1$ 中,之后的订单分配到作业 j 中;调度 S_2 为将 x 订单之前的订单分配到 $j-1$ 中,x 订单及之后的订单分配到 j 中;调度 S_3 为将 $x+1$ 订单及其之前的订单分配到 $j-1$ 中,之后的订单分配到 j 中. 3 种调度的总完成时间分别为 $TC(S_1)$、$TC(S_2)$、$TC(S_3)$,且

$$TC(S_1) = x(\sigma_1 + \sigma_2 + \cdots + \sigma_x) + (m-x)(\sigma_1 + \sigma_2 + \cdots + \sigma_m),$$
$$TC(S_2) = (x-1)(\sigma_1 + \sigma_2 + \cdots + \sigma_{x-1}) + (m-x+1)(\sigma_1 + \sigma_2 + \cdots + \sigma_m),$$
$$TC(S_3) = (x+1)(\sigma_1 + \sigma_2 + \cdots + \sigma_{x+1}) + (m-x-1)(\sigma_1 + \sigma_2 + \cdots + \sigma_m).$$

则 $TC(S_2) - TC(S_1) = \sum_{i=x+1}^{m} \sigma_i - (x-1)\sigma_x$.

从条件可知,$\sigma_x \leqslant \sum_{i=x+1}^{m} \sigma_i/(x-1)$,因此 $TC(S_2) - TC(S_1) \geqslant 0$.

同理可证 $TC(S_3) - TC(S_1) = x\sigma_{x+1} - \sum_{i=x+2}^{m} \sigma_i$,且其值不为负. 上述定理得证.

引理 1 若一个作业安排在 j 位置进行加工,可获得较优的解,不影响参数调整和换模的条件下,具有相同订单的作业安排的位置应与其相邻.

证明 如图 2 所示,假设位置 j 之前的所有作业包含的订单数为 n_1,时间为 t_1,完成时间为 TC_1,j_n 为 j 位置后任意一个位置,位置 j 及 j_n 间所有作业包含的订单数为 n_2,时间为 t_2,总完成时间为 TC_2,作业 a 及作业 b 包含的订单数量都为 n,作业的处理时间为 p.

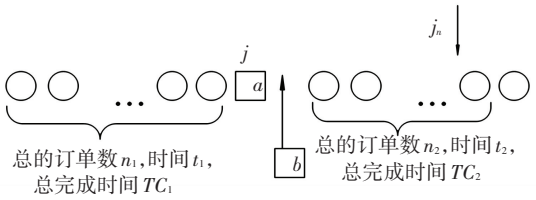

图 2 作业位置的确定

Fig.2 Location of jobs

若作业 a 在 j 位置上获得的解较优,则满足关系

$$[n(t_1 + p) + (TC_2 + n_2 p)] - [TC_2 + n(t_1 + t_2 + p)] = n_2 p - n t_2 < 0.$$

同样作业 b 在作业 $j+1$ 位置上较其他位置亦满足关系

$$[n(t_1 + p + p) + (TC_2 + 2n_2 p)] - [TC_2 + n_2 p + n(t_1 + t_2 + 2p)] = n_2 p - n t_2 < 0,$$

此时的解较优.

定理 2 参考文献[4],若作业 a 与作业 b 为相同作业族 f 中的作业,且满足

$$\sum_{i_f=1}^{I_f} X_{i_f j_a}^f Y_{i_f}^f \sigma_{i_f}^f Z_{j_a}^f / \sum_{i_f=1}^{I_f} X_{i_f j_a}^f Y_{i_f}^f < \sum_{i_f=1}^{I_f} X_{i_f j_b}^f Y_{i_f}^f \sigma_{i_f}^f Z_{j_b}^f / \sum_{i_f=1}^{I_f} X_{i_f j_b}^f Y_{i_f}^f,$$

将作业 a 的位置安排在作业 b 的位置前,即 $j_a < j_b$,此时获得的方案较优.

证明 令调度 S_1 为作业 a 在作业 b 之前,调度 S_2 为作业 b 在作业 a 之前. 假设作业 a 和作业 b 相邻,且已经加工的时间为 t,令

$$l_a = \sum_{i_f=1}^{I_f} X_{i_f j_a}^f Y_{i_f}^f \sigma_{i_f}^f Z_{j_a}^f,\quad k_a = \sum_{i_f=1}^{I_f} X_{i_f j_a}^f Y_{i_f}^f,$$
$$l_b = \sum_{i_f=1}^{I_f} X_{i_f j_b}^f Y_{i_f}^f \sigma_{i_f}^f Z_{j_b}^f,\quad k_b = \sum_{i_f=1}^{I_f} X_{i_f j_b}^f Y_{i_f}^f.$$

则

$$TC(S_1) = k_a(t + p_f l_a) + k_b[t + p_f(l_a + l_b)],$$
$$TC(S_2) = k_b(t + p_f l_b) + k_a[t + p_f(l_a + l_b)],$$

那么,$TC(S_2) - TC(S_1) = p_f(k_a l_b - k_b l_a)$,由条件可

知，$k_a l_b > k_b l_a$，且 p_f 为品种 f 单位晶圆的加工时间，$p_f > 0$，因此，$TC(S_2) - TC(S_1) > 0$.

假设作业 a 和作业 b 不相邻，上述条件相同，则将作业 a 及作业 b 分别拆分成 l_a 和 l_b 个大小为 1 的作业，并结合引理 1，可证得，作业 a 位置较作业 b 位置靠前的方案较优.

2　算法构建

文中将所要解决的问题分为 3 个层级，分别为 1)作业在品种间的分配；2)订单成组形成作业；3)作业的排序. 虽然不少文献提供了解决包含 2)、3)子问题的 MOPJ 调度问题的方法，但依然缺少解决 3 个层级结构问题的理论方法，原有的方法也很难扩展解决类似的问题. 同时由于考虑了参数调整，因此文中 2)、3)子问题涉及了订单、作业以及作业族 3 个层次的调度，与传统的订单、作业两个层次的调度不同. 因此在上述模型和定理的基础上，提出了订单成组算法以及混合差分进化的入侵杂草算法.

2.1　订单成组算法

订单成组算法以定理 1、2 为基础，可以在极短时间内解决子问题 2). 假设品种 f 的订单数量为 m，大小分别为 $\sigma_1, \sigma_2, \cdots, \sigma_m$，分配的作业数量为 n. 定义作业容量为作业中所包含的订单的数量.

其具体步骤如下：

步骤 1　将订单按照从大到小的顺序排列，即 $\sigma_1 \geqslant \sigma_2 \geqslant \sigma_3 \cdots \geqslant \sigma_m$.

步骤 2　计算 $a = \lfloor m/n \rfloor$，$b = m - an$.

式中 $\lfloor m/n \rfloor$ 表示不大于 m/n 的最大整数.

若 $b \geqslant 1$，则作业 1 至作业 b 的作业容量为 $a + 1$，作业 $b + 1$ 至作业 n 的作业容量为 a；

若 $b = 0$，则所有作业的作业容量为 a.

步骤 3　从最后一个作业开始将订单按照顺序分配到作业中，每一次分配先检查当前订单是否可以分配到已分配订单的作业中，且保证作业容量不超过上述分配的容量，否则分配到最新的作业中. 越靠后的作业优先分配.

步骤 4　若步骤 3 中没有作业可以分配，且已无未分配订单的作业. 根据剩余订单的数量 m_r，将作业 1 至作业 m_r 的作业容量加 1.

步骤 5　根据定理 2，将所有作业进行排序.

2.2　混合差分进化的入侵杂草算法

混合差分进化的入侵杂草算法(hybrid invasive weed optimization-differential evolution，HIWO-DE)以双层嵌套的编码机制为基础，将差分进化及订单成组算法融合到入侵杂草算法的每一次迭代中. 同时在变异操作中，提出了具有学习机制(learning mechanism)的算子，学习作业在前几代中的排序，调节变异算子的大小，以使作业向理想的排序靠近. 在变异操作和邻域操作算子中插入了自适应算子，使得算法在前期具有较大的搜索空间，后期具有较强的局部搜索能力.

步骤 1　编码. 采用双层嵌套的编码方式，如图 3所示. 第一层采用正整数编码，表示为杂草个体，每一个正整数表示每一个品种分配的作业数量. 第二层采用的编码中，每一个作业的第一个位置为作业族编号；第二个位置为随机实数，所有作业的第二个位置的随机实数组成目标向量，随机实数从小到大排序，决定了作业的加工顺序；其他位置表示该作业所包含的订单. 第二层编码中每一个作业族作业数对应第一层编码中的正整数.

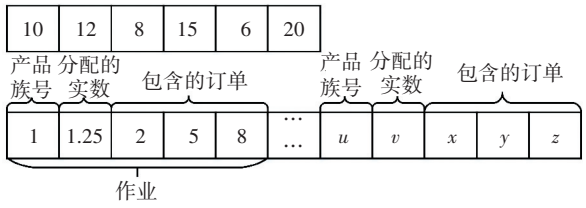

图 3　编码方式

Fig.3　Encoding presentation

步骤 2　初始化.

1)初始化杂草个体. 利用约束(6~8)为每一品种限定作业数量的范围，采用随机生成的方法，为每一品种产生正整数，即为一个杂草个体，正整数表示作业数量. λ 个杂草个体构成一个分散性较好的初始群体.

2)初始化目标向量. 为每一个作业族中的作业产生一个随机数，并从小到大排序. 所有作业族中的随机数都从小到大进行排序，产生一个初始化目标向量. 每一个杂草个体产生 μ 个目标向量，组成次群体.

步骤 3　每一个杂草个体进行订单成组(2.1 订单成组算法). 将成组排序后的作业与步骤 2.2 产生的随机数对应.

步骤 4　变异. 以当前次种群中目标函数值最优的个体作为扰动向量，再从次种群中随机选择两个不同个体作为差分向量，通过下面公式得到变异向量：

$$\boldsymbol{v}_{p,r_f,G+1} = \boldsymbol{x}_{p,r_f,G} + F_1(\boldsymbol{x}_{\text{best},r_f,G} - \boldsymbol{x}_{p,r_f,G}) + F_2 \mid \boldsymbol{x}_{p_1,r_f,G} - \boldsymbol{x}_{p_2,r_f,G} \mid ,$$

式中：p、p_1、p_2、best $\in \{1,2,\cdots,\mu\}$，且 $p_1 \neq p_2$，best 表示次群体中目标函数值最优的个体；$\boldsymbol{x}_{p,r_f,G}$、$\boldsymbol{x}_{p_1,r_f,G}$、$\boldsymbol{x}_{p_2,r_f,G}$、$\boldsymbol{x}_{\text{best},r_f,G}$ 分别表示为第 G 代的 p、p_1、p_2、best 个体中的 f 品种的第 r_f 个作业的向量；F_1 为自

适应缩放算子，F_2 为学习算子.

F_1 较大时，变异的随机性较大，不利于找到精确的最优解；F_1 较小时，收敛速度较快，且易收敛于非最优解. 为避免早熟，F_1 值在算法初期应较大，保持个体的多样性，易找到全局的最优解；在算法后期，F_1 的值应较小，易于最优解的搜索和稳定. F_2 则根据历代的状态调节数值大小，若向量有增大的趋势，则取正值；若向量有减小的趋势，则取负值. 根据文献[10-13]的设计规则，得

$$F_1 = 2^r \times F_0,\ r = \mathrm{e}^{1-\frac{G_\mathrm{m}}{(G_\mathrm{m}+1-G)}},$$

$$F_2 = \sin\left(\frac{\Delta k_{r_f}}{|\Delta K_{r_f}^{\max}|}\pi\right),$$

式中：F_0 为缩放因子，G_m 为最大内层进化代数，G 为当前内层代数，Δk_{r_f} 表示作业 r_f 在 G 代与 $G-1$ 代中的位置差，$\Delta K_{r_f}^{\max}$ 表示历代中作业 r_f 的最大位置差.

步骤 5 交叉. 交叉操作是在变异产生的第 p 个个体 $v_{p,G}$ 和种群中的第 p 个体 $x_{p,G}$ 之间进行，交叉得到试验向量

$$\boldsymbol{u}_{p,r_f,G+1} = \begin{cases} \boldsymbol{v}_{p,r_f,G+1}, & \text{if } (CR < rand(r_f)) \text{ or } (r_f = rnb(r_f)) \\ x_{p,r_f,G}, & \text{else.} \end{cases},$$

式中：CR 为交叉因子，$rand(r_f)$ 为评价作业 r_f 时产生均匀分布的随机数，$rnb(r_f)$ 为随机选择的整数.

每一作业族中，将交叉后得到的数从小到大重新对应到每个作业中.

步骤 6 目标向量选择. 选择操作是在试验向量 $\boldsymbol{u}_{p,G+1}$ 与原种群的个体 $\boldsymbol{x}_{p,G}$ 之间进行，选择的原则是目标函数值更优的个体进入到下一代，用公式表示为

$$\boldsymbol{x}_{p,G+1} = \begin{cases} u_{p,G+1}, & \text{if } (f(u_{p,G+1}) \leqslant f(x_{p,G})) \\ x_{p,G}, & \text{else.} \end{cases},$$

式中 f 为目标函数.

步骤 7 种子数量确定. 在文献[14-15]的基础上，采用基于个体目标函数值产生种子，种子数量计算方式为

$$\varphi_\lambda = \lfloor \varphi_{\max} - \frac{\varphi_{\max} - \varphi_{\min}}{f_{\max} - f_{\min}} \times (f_\lambda - f_{\min}) \rfloor,$$

式中：$\varphi_{\min}$ 和 $\varphi_{\max}$ 分别为最小和最大的种子数量，$f_{\min}$ 和 $f_{\max}$ 分别为最小和最大的目标函数值，φ_λ 表示 λ 个体的种子数量，f_λ 表示 λ 个体的目标函数值.

步骤 8 邻域操作算子. 针对每个作业族的作业，设计两种操作算子，±1 变异和自适应随机变异.

±1 变异，随机选择两个作业族，对其中一个作业族的作业数量+1，另一个作业族的作业数量-1.

自适应随机变异采用如下公式确定变异的最大范围值

$$c = (rang_{\max} - rang_{\min})(g_\mathrm{m} - g)/g_\mathrm{m} + rang_{\min},$$

式中：$rang_{\max}$ 和 $rang_{\min}$ 分别为最大和最小允许变异的值，g_m 为最大外层进化代数，g 为当前外层代数.

在 $[1,c]$ 中随机确定一个值 d，选择两个作业族，其中一个作业族加上 d，另一个作业族减去 d.

步骤 9 产生种子. 随机选择一种邻域操作算子，对杂草个体执行操作，新个体即为当前个体的种子.

步骤 10 竞争排除. 将种群中的父代个体及子代个体按照目标函数值排序，选择较好且互不相同的 λ 个个体作为下一代的种群.

2.3 算法流程图

算法的总体流程图如图 4 所示.

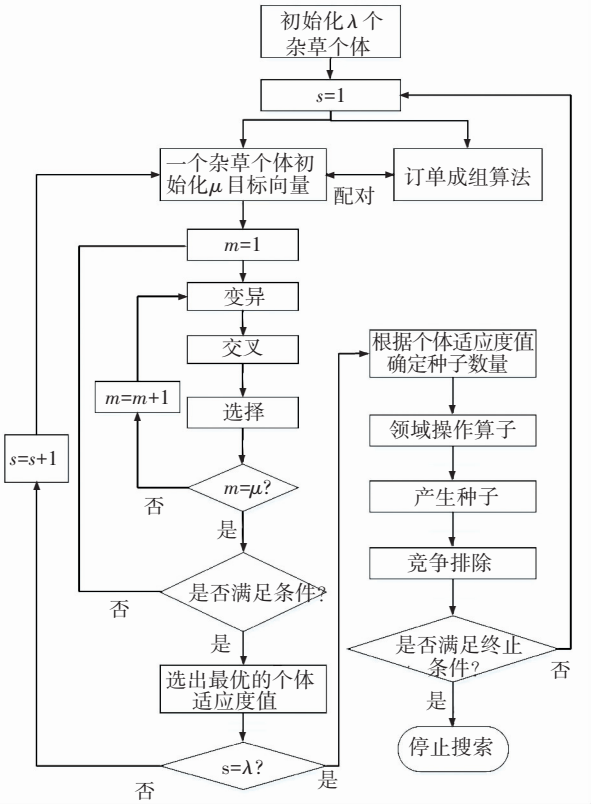

图 4 算法总体流程

Fig.4 Framework of the algorithm

3 仿真实验分析

为有效评价算法的性能，本文以求解时间和文献[16]中提出的解的优度 PR 作为算法评价指标，$PR = V(H,T,Y)/\mathrm{Best}(Y)$，其中 $V(H,T,Y)$ 表示算法 H 在迭代次数为 T 时对实例 Y 的求解结果，$\mathrm{Best}(Y)$ 表示对实例 Y 求解所能得到的最优值. 解的优度能够较好地反映算法的收敛性能，其值越小，表明算法的收敛性能越好.

3.1　参数分析

实验参数参照文献[2,16]中的参数设计规则设定. 为确定较佳的内层迭代次数,根据约束(6~8)的作业分配约束,产生 10 个实例作为杂草个体进行实验,并与未使用定理 2(Theorem2)、未使用学习算子的差分进化进行对比,得到迭代次数对解的优度值和求解时间的影响,结果分别见图 5 和图 6.

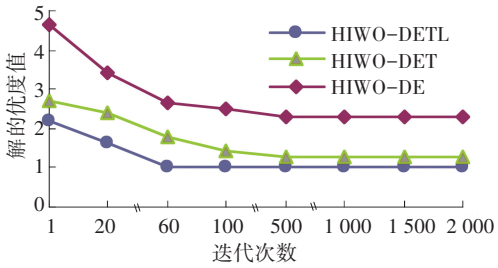

图 5　迭代次数对解的优度值的影响

Fig.5　Impact on the performance ratio by the iteration times

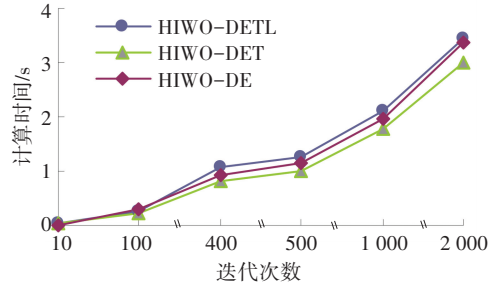

图 6　迭代次数对计算时间的影响

Fig.6　Impact on computation time by the iteration times

由图 5 可知,结合了学习算子和定理 2 的算法在对作业进行排序时明显优于其他两种算法的收敛性,且可以看出当迭代次数为 300~2 000 时,3 种算法的解都趋于稳定. 同时可以看出结合了学习算子以及定理 2 后,算法的收敛速度更快,当迭代次数为 60 时已经找到了较稳定的解. 综合考虑时间成本和解的优度,本文认为后续实验中,HIWO-DETL(HIWO-DE-Theorem2-Learning)算法的内层迭代次数定为 100,HIWO-DET(HIWO-DE-Theorem2)算法的内层迭代次数定为 250,HIWO-DE 算法的内层迭代次数定为 300 是合理的.

用类似的方法,按照不同的参数组合,进行大量实验. 通过分析实验结果得到:将算法的外层初始种群 λ 设为 10,内层初始种群 μ 设为 20,缩放因子 F_0 设为 0.6,交叉因子 CR 设为 0.5,最大与最小允许变异的值 $rang_{\max}$、$rang_{\min}$ 分别设为 5 和 1,最大与最小种子数量 $\varphi_{\max}$、$\varphi_{\min}$ 设为 6 和 1,外层迭代次数 g_m 定为 10,可以以较高的效率对问题进行求解.

3.2　算例分析

利用 2.1 所提出的订单成组算法(记 H1)对单品种背景下的单机 MOPJ 调度问题进行订单成组和作业排序,对比文献[4]所提出的启发式算法(记 H2),为说明 H1 优于 H2,以解的比值及时间作为评价指标,如表 1 所示.

表 1　H1 与 H2 的对比

Tab.1　Comparison between H1 and H2

N	H1		H2	
	比值	时间/ms	比值	时间/ms
100	0.83	10	1	16
500	0.79	12	1	25
1 000	0.78	16	1	31
5 000	0.77	281	1	374
10 000	0.78	983	1	1 139

从表 1 可知,H1 算法所求的解显著优于 H2 所求的解,且时间较 H2 更短. 可见本文提出的方法在解决单品种问题上具有明显的优势. 由于本算法的优势,也使得本文在解决多品种问题及品种间的晶舟分配问题具有良好的性能.

对比 4 种算法在不同的作业数量及订单数量的情况下解的优度值,实验结果如表 2 所示.

表 2　相关解的优度值

Tab.2　Performance ratio of the related solutions

I	J	m	f	HIWO-DETL	HIWO-DET	HIWO-DE	DE
50	30	[3,18]	7	1.013 3	1.363 6	1.589 0	1.613 6
	20	[3,8]	7	1.067 5	1.110 4	1.274 4	1.755 8
50	35	[3,18]	7	1.011 2	1.154 6	1.387 2	1.474 6
	25	[3,8]	7	1.077 4	1.084 5	1.121 3	1.666 9
50	40	[3,18]	7	1.002 0	1.207 2	1.298 8	1.367 9
	30	[3,8]	7	1.002 2	1.107 5	1.159 9	1.479 9
100	60	[3,18]	7	1.017 9	1.229 9	1.378 3	1.368 5
	30	[3,8]	7	1.060 6	1.391 1	1.636 6	3.459 1
100	70	[3,18]	7	1.008 8	1.054 2	1.141 0	1.228 5
	35	[3,8]	7	1.064 7	1.292 6	1.382 0	1.334 0
100	80	[3,18]	7	1.024 6	1.033 8	1.173 7	1.152 8
	40	[3,8]	7	1.092 7	1.356 6	1.469 8	1.561 3

从表 2 可以看出,HIWO-DETL 算法的优度值明显优于其他 3 种算法. 由于 DE 算法在对问题进行求解时,无法合理分配各个品种的作业数量,因此在求解开始随机给定了每个品种的作业数量,只能求解在该作业分配的情况下较优的调度方案,而无法解决作业在各个品种间的分配问题. 因此其他几个算法所求得的解要优于 DE 算法. 故本文所提的算法可以在解决单机 MOPJ 问题的同时,有效解决晶舟在品种间的分配,优化调度方案.

3.3　应用分析

对产品种类数为 7、9、11、13、15、17,对应的订

单数为 50、100、150 的调度问题进行仿真实验,实验结果见图 7.

OS 表示优化空间(optimization space),且

$$OS = \frac{V(\mathrm{DE}) - V(\mathrm{IWO\text{-}DE})}{V(\mathrm{IWO\text{-}DE})}.$$

式中:V(IWO-DE) 表示优化了作业分配的求解结果,V(DE) 表示未优化作业分配的求解结果.

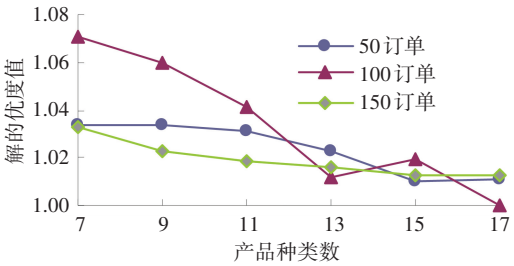

图 7 产品种类数对解优度值的影响

Fig.7 Impact on the performance ratio by the number of product types

由图 7 易知,当订单数相同时,随着产品种类数的增加,解的优度值有下降的趋势,这与当种类数增加时,订单及作业分配的限制增加,调度组合减少,算法更容易求得较优调度结果相符. 从对不同种类与订单下 OS 值测算的实验中发现,优化作业在品种间分配的调度较未优化的调度的性能至少提高了 10%,从而说明优化作业分配在 MOPJ 调度中的重要性.

4 结 论

1)采用双层嵌套的编码机制,以晶舟的数量为纽带将上、下层间的编码映射统一,为解决在传统单机 MOPJ 问题上增加晶舟分配问题的调度提供了研究思路.

2)将差分进化融合到自适应入侵杂草算法的迭代中,同时构造具有学习机制的算子,根据历代位置,调整参数大小,改善了算法的性能,丰富了解决此类调度问题的理论方法.

3)仿真实验表明,HIWO-DETL 具有较好的求解性能,在 MOPJ 调度问题中考虑晶舟的优化具有一定的意义.

参考文献

[1] QU P, MASON S J.Using tabu search on the single machine multi-orders per job scheduling problem[C]//IIE Annual Conference and Exhibition 2004. Houston: Institute of Industrial Engineers, 2004: 1831-1835.

[2] SOBEYKO O, MONCH L. Grouping genetic algorithms for solving single machine multiple orders per job scheduling problems[J]. Annals of Operations Research, 2015, 235(1): 709-739.

[3] JAMPANI J, MASON S J. Column generation heuristics for multiple machine, multiple orders per job scheduling problems[J]. Annals of Operations Research, 2008, 159(1): 261-273.

[4] MASON S J, CHEN J S. Scheduling multiple orders per job in a single machine to minimize total completion time[J]. European Journal of Operational Research, 2010, 207(1): 70-77.

[5] ERRAMILLI V, MASON S J. Multiple orders per job compatible batch scheduling[J]. IEEE Transactions on Electronics Packaging Manufacturing, 2006, 29(4): 285-296.

[6] ERRAMILLI V, MASON S J. Multiple orders per job batch scheduling with incompatible jobs[J]. Annals of Operations Research, 2008, 159(1): 245-260.

[7] WANG Teng, ZHOU Binghai. Scheduling multiple orders per job with multiple constraints on identical parallel machines[J]. Journal of Donghua University (English Edition), 2013(6): 466-471.

[8] CAI Y W, KUTANOGLU E, HASENBEIN J, et al. Single-machine scheduling with advanced process control constraints[J]. Journal of Scheduling, 2012, 15(2): 165-179.

[9] LI L, QIAO F, WU Q D. ACO-based multi-objective scheduling of parallel batch processing machines with advanced process control constraints[J]. The International Journal of Advanced Manufacturing Technology, 2009, 44(9/10): 985-994.

[10] LIN Q, ZHU Q, HUANG P, et al. A novel hybrid multi-objective immune algorithm with adaptive differential evolution[J]. Computers & Operations Research, 2015, 62: 95-111.

[11] 陈华, 范宜仁, 邓少贵. 基于 logistic 模型的自适应差分进化算法[J]. 控制与决策, 2011, 26(7): 1105-1108.
CHEN Hua, FAN Yiren, DENG Shaogui. Adaptive differential evolution algorithm based on logistic model[J]. Control and Decision, 2011, 26(7): 1105-1108.

[12] LU C L, CHIU S Y, HSU C H, et al. Enhanced differential evolution based on adaptive mutation and wrapper local search strategies for global optimization problems[J]. Journal of Applied Research and Technology, 2014, 12(6): 1131-1143.

[13] PIOTROWSKI A P. Adaptive memetic differential evolution with global and local neighborhood-based mutation operators[J]. Information Sciences, 2013, 241: 164-194.

[14] RANI D S, SUBRAHMANYAM N, SYDULU M. Multi-objective invasive weed optimization: an application to optimal network reconfiguration in radial distribution systems[J]. International Journal of Electrical Power & Energy Systems, 2015, 73: 932-942.

[15] 桑红燕, 潘全科. 求解流水车间批量流集成调度的离散入侵杂草优化算法[J]. 控制理论与应用, 2015(2): 246-250.
SANG Hongyan, PAN Quanke. A discrete invasive weed optimization algorithm for the intergrated lot-streaming flow shop scheduling problem[J]. Control Theory & Applications, 2015(2): 246-250.

[16] QU P, MASON S J. Metaheuristic scheduling of 300 mm jobs containing multiple orders[J]. IEEE Transactions on Semiconductor Manufacturing, 2005, 18(4): 633-643.

(编辑 杨 波)