求解模糊柔性作业车间调度的多目标进化算法

王 春[1],田 娜[2],纪志成[1],王 艳[1]
(1. 江南大学教育部物联网技术应用工程中心,江苏无锡 214122;2. 江南大学人文学院,江苏无锡 214122)

摘 要: 针对实际制造车间中工序加工时间具有不确定性,将加工时间采用模糊数表示,建立一种多目标模糊柔性作业车间调度模型,并提出了有效求解该模型的多目标进化算法. 算法采用混合机器分配和工序排序策略的方法产生初始种群,并采用插入空隙法对染色体进行解码. 定义一种新的基于可能度的个体支配关系和一种基于决策空间的拥挤算子,并将所提支配关系和拥挤算子运用于快速非支配排序. 接着,提出一种基于移动模糊关键工序的局部搜索策略. 实验部分首先通过田口试验方法来研究关键参数对算法性能的影响;其次,将所提算法与三种不同的优化算法作对比. 实验结果验证了所提算法的有效性.

关键词: 模糊柔性作业车间调度; 局部搜索; 多目标进化算法; 可能度; 模糊关键工序
中图分类号: TP302 **文献标识码:** A **文章编号:** 0372-2112 (2017)12-2909-08
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2017.12.012

Multi-objective Evolutionary Algorithm to Solve Fuzzy Flexible Job Shop Scheduling Problem

WANG Chun[1], TIAN Na[2], JI Zhi-cheng[1], WANG Yan[1]
(1. *Engineering Research Center of Internet of Things Technology Applications Ministry of Education, Jiangnan University, Wuxi, Jiangsu* 214122, *China*; 2. *School of Humanities, Jiangnan University, Wuxi, Jiangsu* 214122, *China*)

Abstract: Seeing that the processing time is uncertain in the actual manufacturing workshop, a multi-objective fuzzy flexible job shop scheduling model is established, and then an effective multi-objective evolutionary algorithm (MOEA) is proposed to solve this model. First, a method of mixing different machine allocation and operation sequencing strategies is adopted to generate initial population and a well-designed greedy inserting algorithm is adopted for chromosome decoding. Second, a Pareto dominant relation based on possibility degree and a modified crowding distance measure in decision space are defined and further employed to improve the fast nondominated sorting. Moreover, a problem-specific local search based on fuzzy critical path theory is incorporated into MOEA. Afterwards, the influence of key parameters is investigated based on the Taguchi method of experiment. Finally, extensive comparison with three existing algorithms is carried out, and the results demonstrate the effectiveness of the proposed algorithm.

Key words: fuzzy flexible job shop scheduling; local search; multi-objective evolutionary algorithm; possibility degree; fuzzy critical operation

1 引言

在实际的柔性作业车间调度问题中,由于受机器故障、环境参数的影响,工序在每台机器上的加工时间并不一定能够精确给出. 同时,伴随着模糊理论的不断发展和完善,运用模糊数表示和处理不确定参数优化问题已经得到广泛的关注和运用. 采用模糊数来表示柔性作业车间中的不确定加工时间和交货期的调度优化问题称为模糊柔性作业车间调度问题(Fuzzy Flexible Job Shop Scheduling Problem, FFJSSP).

近年来针对 FFJSSP,国内外学者做了很多研究工作. 雷[1]描述了一类有效的基于分解集成的遗传算法,

收稿日期:2016-05-10;修回日期:2016-08-24;责任编辑:覃怀银
基金项目:国家自然科学基金(No. 61572238);江苏省杰出青年基金(No. BK20160001)

在该算法中主种群被分解成两个子种群,两个子种群分别独立进化寻优后再合并成一个种群. 雷[2]同时提出了一类协同进化算法,设计了新的交叉算子和改进的锦标赛选择策略. 通过采用两个种群分别进化机器选择和工序排序来优化最大模糊完工时间. 王[3]设计了一类混合蜂群优化算法,该算法采用变邻域搜索作为局部搜索算子,具有很好的收敛性. Palacios[4]提出了一类混合遗传禁忌搜索算法. 其它最新的有效求解算法还包括高[5]提出的离散和声算法和王[6]提出的分布估计优化算法. 上述研究工作主要用来解决单目标模糊柔性作业车间调度问题. 对于多目标模糊柔性作业车间调度问题(Multi-Objective Fuzzy Flexible Job Shop Scheduling Problem,MOFFJSSP),王[7,8]提出了两类基于免疫和熵原理的多目标遗传算法(Multi-Objective Genetic Algorithm, MOGA). 采用偶极字符串和三维矩阵分别用来表示机器选择和工序排序部分对应的解,郑[9]提出了一类有效解决该问题的基于群的邻域搜索算法. 从当前国内外取得的研究成果看,有效解决MOFFJSSP的成果相对较少. 本文针对MOFFJSSP,以最小化模糊最大完工时间、模糊机器总负荷和模糊瓶颈机器负荷作为三个优化指标,提出一种有效求解该问题的多目标进化算法(Multi-Objective Evolutionary Algorithm,MOEA).

2 问题描述

MOFFJSSP问题可以描述为:工件集 $\mathcal{J}$ 里的 n 个工件 $\{J_1,J_2,\cdots,J_n\}$ 在机器集 $\mathcal{M}$ 里的 m 台机器 $\{M_1,M_2,\cdots,M_m\}$ 上进行加工,每个工件 J_i 包含一道或者多道工序 $O_{ij},j\in\{1,2,\cdots,n_i\}$,$n_i$ 为加工工件 J_i 的工序总数,所有工序按照给定的工艺路线进行加工,工序 O_{ij} 在可选机器集中选择一台机器 $M_k\in\mathcal{M}$ 上进行加工. 工件 J_i 的第 j 道工序在机器 M_k 上的加工时间表示为三角模糊数 $p_{ijk}=(p_{ijk}^1,p_{ijk}^2,p_{ijk}^3)$. 其中,$p_{ijk}^1$ 为最小加工时间,p_{ijk}^2 为最大可能加工时间,p_{ijk}^3 为最大加工时间. 调度目标为:为每道工序确定一台合适的加工机器,并对每台机器上分配的所有工序进行排列以确定其开始加工时间,以使所设定的优化目标达到最优. 工件工序在加工过程中要满足约束有:所有机器都相互独立且所有机器在零时刻均可用;同一时刻同一台机器只能加工一道工序;工序在加工过程中不能中断;同一工件的工序之间有先后约束,不同工件的工序之间没有先后约束;所有工件的优先级相同.

本文求解MOFFJSSP,考虑三个优化目标,即最小化模糊最大完工时间、模糊机器总负荷和模糊瓶颈机器负荷,分别定义如下:

$$f_1:C_{\max}=\max\{C_i\mid i=1,2,\cdots,n\} \tag{1}$$

$$f_2:W_T=\sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n_i}p_{ijk}u_{ijk} \tag{2}$$

$$f_3:W_{\max}=\max_{1\leqslant k\leqslant m}\left\{\sum_{i=1}^{n}\sum_{j=1}^{n_i}p_{ijk}u_{ijk}\right\} \tag{3}$$

$$u_{ijk}=\begin{cases}1, & \text{if machine } k \text{ is selected for operation } O_{ij}\\ 0, & \text{otherwise}\end{cases} \tag{4}$$

其中,C_i 为工件 J_i 的模糊完工时间. 式(1)~(3)对应本文优化的三个目标.

3 相关知识介绍

3.1 模糊数操作

在求解MOFFJSSP中,为了获得合理的调度方案需要定义一些模糊数操作. 本文首先采用文献[6]定义的三种模糊数操作,即模糊数求和操作、模糊数取大操作和模糊数比较操作。模糊数求和操作用于确定工序的模糊完工时间,模糊数取大操作用于确定每一道工序的模糊开始时间,模糊数比较操作用于比较各工件的模糊完工时间以获得整个调度的最大模糊完工时间. 其次,为了便于求取在局部搜索中每道工序的最晚开工时间,本文定义一种模糊数求差操作. 设 $A=(a_1,a_2,a_3)$ 和 $B=(b_1,b_2,b_3)$ 为两个三角模糊数,则求差操作定义为:$A-B=(a_1-b_1,a_2-b_2,a_3-b_3)$.

3.2 基于可能度的支配关系

对于MOFFJSSP问题,所优化的三个目标值都是模糊数,传统的基于精确目标值的Pareto支配关系已经不再适用. 因此,我们首先定义一种比较三角模糊数大小的可能度.

假设 $f_k(x_i)=(f_k^1(x_i),f_k^2(x_i),f_k^3(x_i))$ 和 $f_k(x_j)=(f_k^1(x_j),f_k^2(x_j),f_k^3(x_j))$ 分别为个体 x_i 和 x_j 对应的第 k 个模糊目标值,$l(f_k(x_i))=f_k^3(x_i)-f_k^1(x_i)$,$l(f_k(x_j))=f_k^3(x_j)-f_k^1(x_j)$,则 $f_l(x_i)$ 小于 $f_l(x_j)$ 的可能度 $P_k(x_i\leqslant x_j)$ 定义如式(5)且满足 $P_k(x_i\leqslant x_j)+P_k(x_j\leqslant x_i)=1$.

$$P_k(x_i\leqslant x_j)=\min\left\{\max\left\{\frac{(f_k^2(x_j)+f_k^3(x_j))-(f_k^1(x_i)+f_k^2(x_i))}{l(f_k(x_i))+l(f_k(x_j))},0\right\},1\right\} \tag{5}$$

接着,我们根据模糊目标函数值比较的可能度来定义一种新的个体支配关系比较方法. 设 M 为优化目标个数,个体 x_i 支配个体 x_j 记作 $x_i\prec x_j$,即:

$$x_i\prec x_j\Leftrightarrow\begin{cases}\forall\, q\in\{1,2,\cdots,M\}, & P_q(x_i\leqslant x_j)\geqslant P_q(x_j\leqslant x_i)\\ \exists\, k\in\{1,2,\cdots,M\}, & P_k(x_i\leqslant x_j)>P_k(x_j\leqslant x_i)\end{cases} \tag{6}$$

3.3 模糊析取图模型

模糊析取图模型采用0和*两个虚拟节点分别代表起始工序和终止工序. 每个节点代表一个加工工序,

上面的权值代表该工序对应的模糊加工时间 p_{ijk},且有 $p_0 = p_* = (0,0,0)$. 在图1 所示的模糊析取图中,节点0 到节点 * 的最长路径定义为模糊关键路径,模糊关键路径的长度 $L(0,*)$ 即为该调度方案对应的模糊最大完工时间,模糊关键路径上的工序为模糊关键工序. 图1 中的一条模糊关键路径为 $0\rightarrow O_{21}\rightarrow O_{22}\rightarrow O_{32}\rightarrow *$,对应的模糊完工时间为(31,42,51). 粗线连接的三道工序 O_{21}、O_{22}、O_{32} 为模糊关键工序. 假设析取图 G 上的一个节点 h 代表加工工序 O_h, $S^E(G,h)$, $C^E(G,h)$, $S^L(G,h,C_{\max}(G))$, $C^L(G,h,C_{\max}(G))$ 分别代表工序 O_h 的模糊最早开始和结束时间、模糊最晚开始和结束时间,$C_{\max}(G)$ 为有向图 G 对应的模糊 makespan. PM(G, h)和 SM(G,h)为工序 O_h 的机器前任工序和机器后任工序,PJ(G,h)和 SJ(G,h)分别代表 O_h 的工件前任工序和工件后任工序.

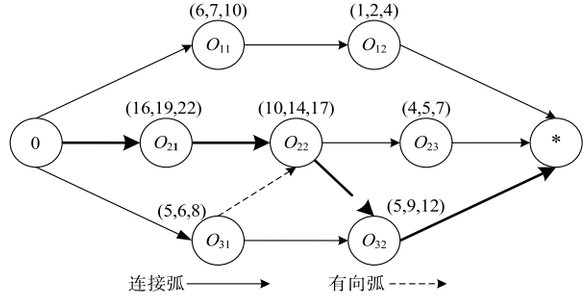

图1　模糊析取图示例

4　MOEA 算法描述

4.1　MOEA 算法的步骤

本文所提 MOEA 算法采用基于 NSGA-II[10] 算法的基本框架,具体步骤描述如算法 1.

算法 1　MOEA 算法

步骤 1:按初始化机制产生初始化种群 P,N 为种群规模;

步骤 2:运用插入法解码评价每一个个体的适应度;

步骤 3:判断算法是否达到终止准则(终止准则为算法进化代数达到最大进化代数). 若满足则转步骤 8,否则转步骤 4;

步骤 4:对父代种群 $P(t)$ 执行进化操作产生子代种群 $Q(t)$;

步骤 5:对子代种群中的优势个体进行基于析取图的局部搜索产生改善种群 $Q'(t)$;

步骤 6:合并种群 $P(t)$、$Q(t)$ 和 $Q'(t)$ 得到种群 $R(t)$,对 $R(t)$ 中目标函数值相同的冗余个体进行变异操作,得到种群 $R'(t)$;

步骤 7:对种群 $R'(t)$ 中的个体进行快速非支配排序产生下一代种群 $P(t+1)$,随后返回步骤 3;

步骤 8:输出 Pareto 最优解;

4.2　染色体编码

染色体编码由两部分组成即机器选择部分(Machine Selection,MS)和工序排序部分(Opeation Sequencing,OS),两部分的染色体长度都等于加工工序的总和 D. MS 的每一位代表每道工序选择的加工机器,OS 部分每一个数字代表加工工件,每个数字在染色体中出现的次数代表工件的第几道加工工序. 例如在染色体编码示例图 2 中,工序 O_{21} 在机器 3 上加工,所有工序的加工顺序为:$O_{31}\rightarrow O_{11}\rightarrow O_{21}\rightarrow O_{12}\rightarrow O_{22}\rightarrow O_{23}\rightarrow O_{32}$.

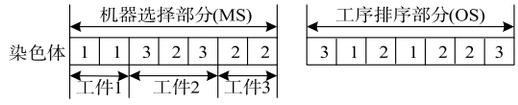

图2　染色体编码示例图

4.3　种群初始化

本文采用混合 4 种不同的机器选择策略和 4 种不同的工序排序规则的方法产生初始种群. 前三种机器选择策略分别为文献[11]提出的全局选择策略、局部选择策略和随机选择策略. 最后一种为最小加工时间策略,该策略为每道工序选择模糊加工时间最小的机器. 4 种工序排序规则分别为(Most Work Remaining, MWR)[12]、(Most Operation Remaining, MOR)[12]、随机排序和(Shortest Processing Time,SPT)[12]规则. 4 种机器选择策略选择概率为 0.5、0.2、0.2 和 0.1,4 种工序排序分配策略选择概率为 0.3、0.2、0.3 和 0.2.

4.4　染色体解码

本文采用插入法对染色体进行解码,以产生活动调度. 具体的解码步骤如算法 2:

算法 2　插入法解码

步骤 1:依次顺序从染色体的 OS 读取基因及对应的加工工序 O_{ij}.

步骤 2:从 MS 读取工序 O_{ij} 对应的加工机器 k,令 $p_{ijk} = (p^1_{ijk}, p^2_{ijk}, p^3_{ijk})$ 为模糊加工时间. 在机器 k 上依次查找空闲时间段 $[t^S_k, t^E_k]$. 其中,

t^S_k 和 t^E_k 分别为空闲时间段的模糊开始时间和模糊结束加工时间,在满足工件加工约束的前提下,工序 O_{ij} 的模糊开始加工时间为:

$$S_{ij} = \begin{cases} \max\{t^S_k, C_{ij-1}\}, & \text{if } j \geqslant 2 \\ t^S_k, & \text{if } j = 1 \end{cases} \tag{7}$$

步骤 3:按照式(8)判断空闲时间段是否满足插入条件,如满足则插入当前空闲时间段,否则将 O_{ij} 插入机器 k 已加工的最后一道工序之后,其模糊开始加工时间为 $\max\{C_{ij-1}, \mathrm{FM}_k\}$. 其中,$\mathrm{FM}_k$ 为机器 k 已加工的最后一道工序的模糊完工时间.

$$\begin{cases} \max\{t^S_k, C_{ij-1}\} + p_{ijk} \leqslant t^E_k, & \text{if } j \geqslant 2 \\ t^S_k + p_{ijk} \leqslant t^E_k, & \text{if } j = 1 \end{cases} \tag{8}$$

4.5　进化算子

所提 MOEA 算法的进化算子包括交叉算子和变异算子. 由于染色体编码由 MS 和 OS 两部分组成,所以交

叉算子和变异算子都分别包括基于 MS 和 OS 两部分的交叉和变异. 对于交叉操作,MS 部分采用文献[13]所提的均匀交叉,OS 部分采用文献[13]所提的 POX 交叉. 对于 MS 部分的变异操作,随机选择两道工序,分别从这两道工序的可加工机器集合中随机选择两台机器替换当前的加工机器;对于 OS 部分的变异操作,随机生成两个不同的位置,然后将此两个位置对应的基因进行交换.

4.6 改进基于决策空间的拥挤距离

对于 MOFFJSSP,由于优化目标值是模糊数,传统 NSGA-II 算法用于保持种群分布性和多样性的拥挤距离已经不能够求出. 同时,考虑在 MOFFJSSP 中,决策空间到目标空间存在多对一的映射,模糊目标函数值相同的个体在决策空间的编码可能不同,而编码重复的个体会造成种群多样性的丢失. 因此,本文提出一种改进的基于决策空间的拥挤距离.

设 p_i、p_j 为种群中的第 i 和第 j 个个体,我们定义 $H(p_i,p_j)$ 代表个体 p_i 和个体 p_j 机器编码部分的海明距离,其值大小为两个个体机器编码部分具有不同基因值的位置的个数. 接着,个体 p_i 的谐和海明距离 $\mathrm{HD}(p_i)$ 定义为该个体与种群中其它个体的海明距离最小的两个值 $H(p_i,p_j)$、$H(p_i,p_k)$ 的平均值,表述为等式(9). 对于非支配排序中相同序值层的个体,谐和海明距离值大的个体优先被选入到下一代种群.

$$\mathrm{HD}(p_i)=\left\{\frac{H(p_i,p_j)+H(p_i,p_k)}{2}\right\} \tag{9}$$

4.7 局部搜索

本算法主要从两个方面考虑,设计有效的局部搜索策略. 一是怎样从种群中选择出局部搜索的个体对象;二是对于选择出的个体,如何设计有效的局部搜索策略来搜索得到局部最优个体.

4.7.1 选择局部搜索个体对象

我们设置局部搜索的概率为 P_l,那么种群中局部搜索的个体数为$\lfloor N\times P_l\rfloor$,即局部搜索进行$\lfloor N\times P_l\rfloor$次. 这里,我们将文献[14]提出的锦标赛选择策略扩展用于模糊数操作来选取待局部搜索个体. 文献[15]提出的均匀设计策略用于生成一组规模为 N 的均匀分布的权向量集,集合里的每一个权向量 $\boldsymbol{\lambda}=[\lambda_1,\lambda_2,\lambda_3]$ 满足 $\lambda_1+\lambda_2+\lambda_3=1$. 通过式(10)对优化的三个模糊目标值进行加权:

$$F=\lambda_1\times f_1+\lambda_2\times f_2+\lambda_3\times f_3 \tag{10}$$

对于每一次的局部搜索,首先从权向量集中随机选择出一个权向量 $\boldsymbol{\lambda}$. 接着,从种群中随机选择出 s 个个体并求取 s 个个体对应的根据式(10)计算的模糊目标加权值,选择加权值最小的个体作为优势个体,最后对该个体进行局部搜索得到一个改进解. 所有的改进解构成改进种群.

4.7.2 邻域构造

本文通过移动一道模糊关键工序来构造局部搜索的邻域. 设析取图 G 上移除一个关键工序 O_l 后得到析取图 G_i^-. 移动关键工序的目的就是在 G_i^- 上将 O_l 插入新的机器得到 G_i' 且满足式 $C_{\max}(G_i')\leqslant C_{\max}(G)$. 假设插入位置在机器 M_k 上的工序 v 前且加工时间为 $p_{O_i,k}$,则工序 v 满足:

$$\begin{aligned}&\max\{C^E(G_i^-,\mathrm{PM}(G_i^-,v)),C^E(G_i^-,\mathrm{PJ}(O_l))\}+p_{O_i,k}\\&\leqslant\min\{S^L(G_i^-,v,C_{\max}(G)),S^L(G_i^-,\mathrm{SJ}(O_l),C_{\max}(G))\}\end{aligned} \tag{11}$$

设 Θ_k 是 G_i^- 上机器 M_k(M_k 为工序 O_l 的可加工机器集)上所有加工工序的工序集($O_l\notin M_k$),且按在机器 M_k 上的最早模糊开工时间升序排列. 我们定义如下两个工序子集集合如式(12),文献[16]证明 O_l 的最优插入位置 Y_k 在所有节点 $R_k\backslash L_k$ 之前和所有节点 $L_k\backslash R_k$ 之后.

$$\begin{aligned}R_k&=\{v\in\Theta_k\mid S^E(G,v)+p_{v,k}>S^E(G_i^-,O_l)\}\\L_k&=\{v\in\Theta_k\mid S^L(G,v,C_{\max}(G))<S^L(G_i^-,O_l,C_{\max}(G))\}\end{aligned} \tag{12}$$

目前还有待考虑的另一个重要问题是如何选择关键工序 O_l 插入的机器 M_k. 假设 M_{O_i} 为关键工序 O_l 可加工的机器集合. 我们首先令 $O_l\rightarrow M_k$ 表示上文所述的在析取图 G_i^- 中,对于某一道关键工序在某一台可加工机器 M_k 上寻找插入位置 Y_k 的一次操作,n_l 代表所有操作的总和. 令 $\Psi(G)=\{O_l\rightarrow M_k\mid l=1,2,\cdots,n_c,M_k\in M_{O_i}\}$ 包含所有的 $O_l\rightarrow M_k$. 这里我们将文献[15]提出的分层策略引入模糊数并按照式(13)求取 $\Psi(G)$ 的每次操作后,模糊机器总负荷的变化 Δt 和模糊关键机器负荷的变化 Δc,并将所有操作按照 Δt 升序排列,Δt 相同的操作按照 Δc 升序排列.

$$\begin{cases}\Delta t(O_l\rightarrow M_k)=p_{O_i,k}-p_{O_i,M(G,O_l)}\\\Delta c(O_l\rightarrow M_k)=W_k(G)+p_{O_i,k}\end{cases} \tag{13}$$

在析取图 G 上,我们按照如上所述的分层排序法依次执行 $O_l\rightarrow M_k$ 操作,对于任意一次操作,一旦在机器 M_k 上找到一个插入位置 Y_k,考虑计算复杂性,我们不再考虑其它插入位置. 同样,一旦在 $\Psi(G)$ 中执行某次操作找到了合适的插入位置,则为了考虑计算复杂性,$\Psi(G)$ 中剩下的操作都不再考虑.

4.7.3 局部搜索步骤

在每一次的局部搜索迭代过程中,如果局部搜索得到的最优解通过式(10)计算能够得到比原始解更小的模糊加权值,则接受局部搜索得到的最优解. 本文求解 MOFFJSSP 所采用的局部搜索是针对染色体解码后

对应的 G 而不是染色体编码本身,令 $\boldsymbol{U},\boldsymbol{V}$ 分别代表优势个体的机器选择部分向量和工序排序部分向量,则上述步骤可归纳为算法 3.

算法 3　个体局部搜索

```
1:j←0
2:G←染色体解码(U,V)
3:G_best←G,flag←0;
4:while G≠∅且j<T
5:    G←求取邻域解(G)
6:    if G≠∅且f(G,λ)<f(G_best)
7:        G_best←G,flag←1
8:    end if
9:    j←j+1;
10:end while
11:if flag==1
12:    {U',V'}←染色体编码(G_best)
13:else
14:    {U',V'}←∅
15:end
16:返回{U',V'}
```

4.8　算法复杂度分析

本文所提 MOEA 算法是在 NSGA-II 算法的框架下融入局部搜索,由于局部搜索无法准确求取平均计算复杂度,所以我们求取算法在最坏情况下的时间复杂度. 根据算法 1,在 MOEA 算法的每一代,算法的计算复杂性主要消耗在步骤 5 到步骤 7. 对于步骤 5 中一个个体的局部搜索,其最坏情况下的计算复杂度为 $O(MT)$. 由于种群中一共有$\lfloor N\times P_l\rfloor$个个体经历局部搜索,所以整个步骤5 的算法复杂度为 $O(MNP_lT)$. 对于步骤6,算法最坏情况下的计算复杂度为 $O(M((2+P_l)N))$. 步骤 7 主要包括非被占优解排序、计算拥挤度和基于拥挤度排序 3 个基本操作,其最坏情况下的时间复杂度分别为 $O(M((2+P_l)N^2))$、$O(D((2+P_l)N)\log((2+P_l)N))$、$O((2+P_l)N\log((2+P_l)N))$. 考虑 $O(M((2+P_l)N))$、$O(D((2+P_l)N)\log((2+P_l)N))$、$O((2+P_l)N\log((2+P_l)N))$相对于 $O(MNP_lT)$和 $O(M((2+P_l)N^2))$可忽略不计. 所以 MOEA 算法的复杂度为 $O(MNP_lT)+O(M((2+P_l)N^2))$,当 $P_lT<N$时,算法复杂度为 $O(MN^2)$.

5　仿真实验与分析

为了考察 MOEA 算法的性能,选取两组不同的测试集,第一组测试集包含 5 个 LeiData 测试函数[1,2]. 第二组测试集包含 4 个 WangData 测试函数(Wang 4×6, Wang 6×10, Wang 8×8, Wang 10×10)[8]. 实验平台为: Windows 7, Intel Core i3-4170 3.70GHz Processor, 4GB RAM,算法采用 Matlab 编程语言实现.

5.1　参数设置探讨

MOEA 算法包含如下 4 个关键参数:种群规模 N,局部搜索概率 P_l,局部搜索最大迭代次数 T,局部搜索单次随机联赛个体数 s. 为了考察这些参数对算法性能的影响,采用试验设计方法(Design of Experiment, DOE)基于 Lei 测试集的 LeiData1 进行讨论. 根据参数和因子水平的数量,选择规模为 $L_{16}(4^4)$的正交试验,即试验次数为 16,参数个数为 4,因子水平数为 4. 每个参数在不同水平下的取值及所有的参数组合见表 1.

算法在每种参数组合下均独立运行 10 次,10 次运行所得平均值作为评价指标. 16 组参数、每组 10 次运行所得所有非支配解组成的并集 RS^* 作为参考集,其规模为$|\mathrm{RS}^*|$. 第 i 组参数第 j 次运行所得解集为 RS_{ij}. 10 次运行所得非支配解与 RS^* 的交集的平均值 $N_{\mathrm{avg}}(i)$作为考核每种组合性能的指标,如式(14)所示,正交表和所得的平均值如表 1 所示,各参数的极差和重要程度如表 2 所示.

$$N_{\mathrm{avg}}(i)=\frac{1}{10}\left|\sum_{j=1}^{10}(\mathrm{RS}^*\cap\mathrm{RS}_{ij})\right| \tag{10}$$

表 1　正交表和 $N_{avg}(i)$统计

参数组合	水平(取值)				$N_{\mathrm{avg}}(i)$
	N	P_l	T	s	
1	1(50)	1(0.1)	1(1)	1(5)	16.2000
2	1(50)	2(0.2)	2(5)	2(10)	24.9000
3	1(50)	3(0.3)	3(15)	3(15)	21.1000
4	1(50)	4(0.4)	4(30)	4(20)	27.4000
5	2(100)	1(0.1)	2(5)	3(15)	28.3000
6	2(100)	2(0.2)	1(1)	4(20)	34.1000
7	2(100)	3(0.3)	4(30)	1(5)	42.3000
8	2(100)	4(0.4)	3(15)	2(10)	47.2000
9	3(150)	1(0.1)	3(15)	4(20)	34.6000
10	3(150)	2(0.2)	4(30)	3(15)	42.6000
11	3(150)	3(0.3)	1(1)	2(10)	51.3000
12	3(150)	4(0.4)	2(5)	1(5)	52.5000
13	4(200)	1(0.1)	4(30)	2(10)	49.6000
14	4(200)	2(0.2)	3(15)	1(5)	50.6000
15	4(200)	3(0.3)	2(5)	4(20)	53.8000
16	4(200)	4(0.4)	1(1)	3(15)	62.0000

从表 2 可以清楚地看出,四个参数中种群规模 N 对算法性能影响最大. 在迭代次数不变时,越大的种群规模能够得到越多的非支配解,且解的质量可能越好. 影响程度其次的是局部搜索概率 P_l. 越大的 P_l 意味着种群中局部搜索的个体数越多,这也反映出本文所提局部搜索策略是有效的. 参数 T 和 s 对算法性能影响较

表 2 各参数响应值

水平	N		P_l		T		s	
	参数取值	响应值	参数取值	响应值	参数取值	响应值	参数取值	响应值
1	50	22.4000	0.1	32.1750	20	40.9000	5	40.4000
2	100	37.9750	0.2	38.0500	30	39.8750	10	43.2500
3	150	45.2500	0.3	42.1250	40	38.3750	15	38.5000
4	200	54.0000	0.4	47.2750	50	40.4750	20	37.4750
极差	31.6000		15.1000		2.5250		5.7750	
序	1		2		4		3	

小且较小的 s 有益于算法获得较好性能的解. 基于上述分析,将算法参数设置为:N 为 100,P_l 为 0.1,T 为 50,s 为 10,交叉概率 0.9,变异概率 0.1,进化代数为 100.

5.2 算法性能指标

本文实验采用文献[17]里的超体积测度和集覆盖测度比较不同算法的性能.

(1)超体积测度,简称 HV 测度. HV 能同时反映出算法获得 PF 的收敛和分布性能的好坏,HV 值越大,算法的收敛性越好. 本文采用文献[18]提出的蒙特卡罗模拟法近似求取超体积,采样点的个数为 100000. 由于 HV 针对的是确定目标值,因此,对于每一个模糊目标值 $f=(f_1,f_2,f_3)$,我们采用 $Z_1(f)=(f_1+2\times f_2+f_3)/4$ 代替 f.

(2)集覆盖测度,简称 C 测度. C 指标能直接反映出两组 PF 之间的占优关系. 设 A 和 B 是不同的算法得到的两组 PF,当 $C(A,B)>C(B,A)$ 时,表明算法 A 获得的 PF 要优于算法 B.

5.3 局部搜索对算法性能的影响

我们首先将所提 MOEA 算法与不加局部搜索的 MOEA 算法(Multi-objective Evolutionary Algorithm with No Local Search,MOEANLS)算法作对比来验证局部搜索对算法性能的影响,结果如表 3 所示. 可以看出,除了 WangData2,8 个算例,MOEA 所得 PF 的 HV 都大于 MOEANLS. 对于 6 个算例(LeiData1 ~ LeiData5,WangData4),MOEA 所得 PF 支配 MOEANLS 所得 PF 的概率都要大于 MOEANLS 所得 PF 支配 MOEA 所得 PF 的概率,其余 3 个算例,两种算法互不支配. 数值试验结果表明本文所提局部搜索策略能够有效提高 MOEA 算法的收敛性能.

5.4 与其它算法的比较

为了验证本文所提算法的有效性我们将 MOEA 算法与文献[7,8]提出的两种 MOGA 算法和文献[10]的 NSGA-II 算法做对比,结果如表 4 和表 5 所示. 从表 4 和表 5 可以看出,针对所有测试算例,MOEA 算法获得 PF 都比其它三种算法获得的 PF 的 HV 和 C 测度好. 表 6 为 4 种对比算法的 CPU 运行时间,从表中可以看出 MOEA 的运行时间要短于 MOGA2 和 NSGA-II,MOEA 算法在 7 个测试函数中运行时间比 MOGA1 略长,但在 WangData2 和 WangData3 上 MOEA 算法的运行时间要比 MOGA1 短. 综上所述,本文所提的 MOEA 算法能够在保证求解效率的同时得到质量更高的解,比其它三种算法更适合解决 MOFFJSSP 问题.

表 3 局部搜索对算法性能的影响

算例	HV		MOEA vs MOEANLS	
	MOEA	MOEANLS	C(MOEA, MOEANLS)	C(MOEANLS, MOEA)
LeiData1	**5.9665e+4**	5.4157e+4	**0.2609**	0.0220
LeiData2	**1.8690e+5**	1.6984e+5	**0.1746**	0.0000
LeiData3	**1.5722e+5**	1.4682e+5	**0.9647**	0.0000
LeiData4	**1.3349e+5**	1.2747e+5	**0.5610**	0.0185
LeiData5	**4.7959e+5**	4.6036e+5	**0.7500**	0.0000
WangData1	**1.2144e+2**	1.2019e+2	0.0000	0.0000
WangData2	2.8958e+3	**2.9092e+3**	0.0000	0.0000
WangData3	**1.0011e+4**	9.9974e+3	0.0000	0.0000
WangData4	**1.2147e+4**	1.2043e+4	**0.0882**	0.0000

表 4 MOEA 算法与其它算法的 HV 测度比较

算法	算例								
	Lei Data1	Lei Data2	Lei Data3	Lei Data4	Lei Data5	Wang Data1	Wang Data2	Wang Data3	Wang Data4
MOEA	**5.9665e+4**	**1.8690e+5**	**1.5722e+5**	**1.3349e+5**	**4.7959e+5**	**1.2144e+2**	**2.8958e+3**	**1.0011e+4**	**1.2147e+4**
MOGA1	4.1432e+3	2.3967e+4	9.4500e+3	5.2290e+3	2.1860e+4	0.7200e+2	1.9149e+3	3.4908e+2	1.0583e+3
MOGA2	1.9806e+4	8.7549e+4	5.1416e+4	4.0382e+4	5.0406e+4	1.2070e+2	2.7221e+3	4.4151e+3	4.3019e+3
NSGA-II	1.3879e+4	6.5110e+4	3.2201e+4	2.8769e+4	6.6879e+4	1.2091e+2	2.8046e+3	2.0104e+3	4.7669e+3

表 5　MOEA 算法与其它算法的 C 测度比较

算例	MOEA(A)vs MOGA1(B)		MOEA(A)vs MOGA2(C)		MOEA(A)vs NSGA-II(D)	
	C(A,B)	C(B,A)	C(A,C)	C(C,A)	C(A,D)	C(D,A)
LeiData1	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
LeiData2	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
LeiData3	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
LeiData4	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
LeiData5	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
WangData1	**1.0000**	0.0000	**0.1667**	0.0000	**0.1667**	0.0000
WangData2	**1.0000**	0.0000	**0.6250**	0.0000	**0.9333**	0.0000
WangData3	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
WangData4	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000

表 6　MOEA 算法与其它算法的 CPU 运行时间比较

算例	MOEA	MOGA1	MOGA2	NSGA-II
LeiData1	1036.9	**762.3**	2401.8	2944.8
LeiData2	920.8	**766.3**	2450.9	3040.9
LeiData3	1557.1	**1051.6**	3278.9	4126.0
LeiData4	1865.7	**1050.1**	4312.5	4133.7
LeiData5	3790.8	**2025.4**	6326.9	7781.6
WangData1	166.7	**157.4**	457.6	580.3
WangData2	**414.0**	683.2	2085.8	2685.2
WangData3	**421.6**	421.7	1234.2	1627.1
WangData4	753.5	**535.0**	1776.7	2184.9

6　结论与未来展望

针对模糊多目标柔性作业车间调度问题,本文提出一种有效求解该类问题的多目标进化算法. 算法采用一种混合不同机器分配和工序排序策略的方法产生初始种群并通过采用插入法对染色体进行解码. 算法定义了一种新的基于可能度的个体支配关系和一种新的基于决策空间的拥挤算子,并将新的支配关系和拥挤算子运用到快速非支配排序中. 通过采用一种新的基于移动模糊关键工序的局部搜索策略来改善种群中的部分解. 通过仿真实验表明,所提算法能够比其它算法更有效地解决多目标模糊柔性作业车间调度优化问题.

参考文献

[1] Lei D M. A genetic algorithm for flexible job shop scheduling with fuzzy processing time[J]. International Journal of Production Research,2010,48(10):2995 – 3013.

[2] Lei D M. Co-evolutionary genetic algorithm for fuzzy flexible job shop scheduling[J]. Applied Soft Computing, 2012,12(8):2237 – 2245.

[3] Wang L,Zhou G,Xu Y,et al. A hybrid artificial bee colony algorithm for the fuzzy flexible job-shop scheduling problem[J]. International Journal of Production Research, 2013,51(12):3593 – 3608.

[4] Palacio J J,González M A,Vela C R,et al. Genetic tabu search for the fuzzy flexible job shop problem[J]. Computers & Operations Research,2015,54:74 – 89.

[5] Gao K Z,Suganthan P N,Pan Q K,et al. An effective discrete harmony search algorithm for flexible job shop scheduling problem with fuzzy processing time[J]. International Journal of Production Research, 2015, 53(19): 5896 – 5911.

[6] Wang S Y,Wang L,Xu Y,et al. An effective estimation of distribution for the flexible job-shop scheduling problem with fuzzy processing time[J]. International Journal of Production Research,2013,51(12):3778 – 3793.

[7] Wang X J,Gao L,Zhang C Y,et al. A multi-objective genetic algorithm for fuzzy flexible job-shop scheduling problem[J]. International Journal of Computer Applications in Technology,2012,45(45):115 – 125.

[8] Wang X J,Li W F,Zhang Y. An improved multi-objective genetic algorithm for fuzzy flexible job-shop scheduling problem[J]. Internatinal Journal of Computer Applications in Technology,2013,47(2/3):280 – 288.

[9] Zheng Y L,Li Y X,Lei D M. Multi-objective swarm-based neighborhood search for fuzzy flexible job shop scheduling [J]. International Journal of Advanced Manufacturing Technology,2012,60(9):1063 – 1069.

[10] Deb K,Pratap A,Agarwal S,et al. A fast and elitist multiobjective genetic algorithm:NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182 – 197.

[11] 赵诗奎,方水良,顾新建. 基于极限调度完工时间最小化的机器选择及 FJSP 求解[J]. 计算机集成制造系统,

2014,20(4):854 – 865.
Zhao S K,Fang S L,Gu X J. Machine selection and FJSP solution based on limit scheduling completion time minimization[J]. Computer Integrated Manufacturing Systems,2014,20(4):854 – 865.
[12] Pezzella F,Morganti G,Cia G. A genetic algorithm for the flexible job-shop scheduling problem[J]. Computers & Operations Research,2008,35(10):3202 – 3212.
[13] Zhang G H,Gao L,Shi Y. An effective genetic algorithm for the flexible job-shop scheduling problem[J]. Expert Systems with Applications,2011,38(4):3563 – 3573.
[14] Yuan Y,Xu H. Multiobjective flexible job shop scheduling using memetic algorithm[J]. IEEE Transactions on Automation Science and Enginering,2015,12(1):336 – 353.
[15] Tan Y Y,Jiao Y C,Li H,et al. MOEA/D + uniform design:a new version of MOEA/D for optimization problems with many objectives[J]. Computer & Operations Research,2013,40(6):1648 – 1660.
[16] Mastrolilli M,Gambardella L M. Effective neighborhood functions for the flexible job shop problem[J]. Journal of Scheduling,2000,3(1):3 – 20.
[17] Ziter E,Thiele L. Multi-objective evolutionary algorithms:A comparative case study and the strength Pareto approach[J]. IEEE Transaction on Evolutionary Computation,1999,3(4):257 – 271.
[18] Bader J,Ziter E. Hype:An algorithm for fast hypervolume-based many-objective optimization[J]. Evolutionary Computation,2011,19(1):45 – 76.

作者简介

王 春 男,1988 年 2 月出生于安徽省淮北市. 现为江南大学教育部物联网技术应用工程中心博士研究生. 主要研究方向为进化算法及其在生产调度中的应用.
E-mail:huaibeifwangchun@163.com

田 娜 女,1983 年 5 月出生于河北省石家庄市. 博士,现为江南大学人文学院副教授,硕士生导师. 主要研究方向为人工智能和数据挖掘.
E-mail:tianna@jiangnan.edu.cn

纪志成 男,1959 年 11 月出生于浙江省杭州市. 博士,现为江南大学教育部物联网技术应用工程中心教授,博士生导师. 主要研究方向为群智能优化算法、智能制造.
E-mail:zcji@jiangnan.edu.cn

王 艳(通信作者) 女,1978 年 9 月出生于江苏省盐城市. 博士,现为江南大学教育部物联网技术应用工程中心教授,博士生导师. 主要研究方向为网络化控制、智能制造系统能效优化.
E-mail:wangyan@jiangnan.edu.cn