文章编号: 1001-0920(2011)11-1675-05

模糊环境下多目标差异作业单机批调度问题研究

卢冰原[1], 黄传峰[1], 贾兆红[2]

(1. 南京工程学院 经济管理学院, 南京 211167; 2. 安徽大学 智能计算与信号处理教育部重点实验室, 合肥 230039)

摘 要: 针对现实生产制造系统中存在的时间参数模糊化问题, 采用梯形模糊数表征时间参数, 给出一种具有模糊交货期和模糊加工时间, 以最小化提前/拖期惩罚、制造跨度以及加工费用为目标的多目标差异作业单机批调度问题模型. 在对该问题进行求解方面, 针对基本粒子群算法容易陷入局部最优的问题, 引入混沌局部搜索策略, 给出了一种基于混沌优化技术的混合粒子群算法. 仿真实验验证了所提出算法的可行性和有效性.

关键词: 差异作业单机批调度; 多目标优化; 模糊环境; 粒子群算法

中图分类号: TP278 **文献标识码:** A

Research on multi-objective scheduling a single batch-processing machine with non-identical job sizes under fuzzy environment

LU Bing-yuan[1], *HUANG Chuan-feng*[1], *JIA Zhao-hong*[2]

(1. Economy and Management School, Nanjing Institute of Technology, Nanjing 211167, China; 2. The Key Lab of Intelligente Computing and Signal Processing of Ministry of Education, Anhui University, Hefei 230039, China. Correspondent: LU Bing-yuan, E-mail: bingyuanlu@126.com)

Abstract: To solve the problems correlated with fuzzy temporal parameter in real manufacture system, based on trapezoidal fuzzy number, a fuzzy single batch-processing machine with non-identical job sizes(NSBM) model for minimized makespan, earliness/tardiness penalties and processing cost which has fuzzy processing time and fuzzy due time is introduced in this paper. Then, aiming at the problems of easily getting into the local optimum of basic particle swarm optimization(PSO) algorithm, a hybrid PSO algorithm based on chaotic local optimizer is proposed for the fuzzy multi-objective NSBM problem above. Finally, simulation results show the feasibility and effectiveness of the algorithm.

Key words: single batch-processing machine with non-identical job sizes; multi-objective optimization; fuzzy environment; particle swarm optimization

1 引 言

差异作业单机批调度(NSBM)问题是由Uzsoy[1]于1994年提出的, 它兼顾了古典调度和批调度的特征, 加工采取批处理方式, 作业尺寸存在差异, 每批作业总尺寸不能超过机器容量. NSBM问题属于典型的NP-hard问题[1], 在现实的企业生产过程中有着广泛的应用. 在NSBM问题的求解方面, 近年来学者们大多采用智能优化算法. Sevaux等人[2-3]采用遗传算法(GA)求解NSBM问题, 但遗传算法在解决较大规模问题时收敛性能较差. Melouk等人[4]则运用模拟退火算法(SA)来解决NSBM问题, 但模拟退火算法所产生的近似解的质量低于遗传算法. 此外, 粒子群算法(PSO)在多种工程优化问题中的应用也正在展开[5-6], 但基本PSO算法易陷入局部最优. 引入其他启发式算法构建混合粒子群算法以提高算法的性能是PSO的一种发展趋势.

目前, 针对NSBM问题的研究主要集中在理想的确定性环境, 有关参数为可预知的准确值, 优化目标单一(以最小化制造跨度为主). 然而, 在现实生产系统中, 受人力、机器的影响, 包括作业交货期、作业加工时间在内的时间参数存在着模糊不确定性, 同时也希望对包括制造跨度在内的多个目标进行优化.

本文将NSBM问题拓展到更加接近现实的模糊环境中, 用梯形模糊数表征时间参数, 同时对基本

收稿日期: 2010-06-30; **修回日期:** 2011-01-03.

基金项目: 江苏省教育厅高校哲学社会科学基金项目(09SJD630036); 南京工程学院科研基金重点项目(QKJA2009015).

作者简介: 卢冰原(1977−), 男, 副教授, 博士, 从事商务智能等研究; 黄传峰(1973−), 男, 副教授, 博士, 从事优化算法等研究.

PSO算法进行改进, 给出了一种基于混沌优化的混合粒子群算法, 利用混沌的特性来改善PSO的性能, 对具有模糊时间参数的多目标NSBM问题进行求解. 实验表明, 该方法能够提高PSO的求解精度和搜索性能.

2 模糊环境下的多目标NSBM问题模型

2.1 模糊环境下的NSBM问题

NSBM问题可以描述如下: 作业集合为$j=1,2,\cdots,n$, 其中作业j的加工时间为t_j, 完工时间为C_j, 交货期为E_j, 尺寸为s_j; 机器容量为B_0, 各作业分批进行加工, 在第k批作业序列b_k中, $k=1,2,\cdots,m$, 所有作业的总尺寸不大于B_0, 每批加工过程不允许中断, b_k的加工时间为T_k, T_k等于b_k中最后完工的作业所用的时间; 批次之间存在时间间隔d_h, $h=1,2,\cdots,m-1$; 机器每批次加工费用为$H(t)$.

本文针对具有模糊交货期和模糊加工时间的情况进行研究, 采用梯形模糊数(TrFN)表征模糊时间参数, 交货期表示为四元组$E_j(E_j^o,E_j^l,E_j^r,E_j^p)$, 如图1所示. 隶属函数$\mu_{E_j}(C_j)$表示客户对作业$j$的满意度. 如果作业在区间$[E_j^l,E_j^r]$内完工, 则满意度为1; 如果在$[E_j^o,E_j^p]$外完工, 则满意度为0. 如果作业$j$的完工时间$C_j$与对应的交货期$E_j$不相符, 将会受到提前或拖期惩罚.

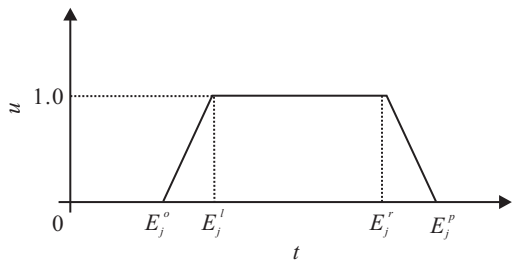

图 1 模糊交货期(TrFN)

类似地, 作业加工时间t_j也用TrFN表征, 表示为四元组$(t_j^o,t_j^{ml},t_j^{mu},t_j^p)$. 加工时间分布函数$\mu_{t_j}(t)$表示$t_j$取值为$t$的可能性程度, 位于区间$[t_j^{ml},t_j^{mu}](t|\mu_{t_j}(t)=1)$内的$t$被认为是$t_j$最可能的取值; t_j在区间$[t_j^o,t_j^p](t|\mu_{t_j}(t)=0)$以外的区域内取值的可能性最小. 同样$C_j$和$T_k$等也用TrFN表征, 相应数值可通过TrFN代数运算获得.

本文研究的NSBM问题以最小化提前/拖期惩罚、最小化制造跨度、最小化加工费用为目标. 在模糊交货期提前/拖期惩罚函数构造方面, 采用有符号距离[7]判别作业完成时间和交货期的先后次序. 针对模糊数有符号距离测量精度偏低的问题, 文中同时引入了区间数距离[8], 当模糊数T和E的有符号距离绝对值$|d(T,E)|\leqslant\xi$时, T与E的间距用区间数距离$D(T,E)$表示, 否则用有符号距离表示, 阈值$\xi\in(0,1]$. 根据文献[9], 同时具有提前和滞后限制的作业j的惩罚函数$P_j^*=P_j(C_j,E_j)$定义为

$$P_j^*=\begin{cases}B_jD(C_j,E_j),\ d(C_j,E_j)<0,\ |d(C_j,E_j)|\leqslant\xi;\\ A_jD^2(C_j,E_j),\ d(C_j,E_j)>0,\ |d(C_j,E_j)|\leqslant\xi;\\ B_jd(C_j,E_j),\ d(C_j,E_j)\leqslant 0,\ |d(C_j,E_j)|>\xi;\\ A_jd^2(C_j,E_j),\ d(C_j,E_j)>0,\ |d(C_j,E_j)|>\xi.\end{cases}\tag{1}$$

考虑到现实生产过程中, 每个作业的重要性程度可能不同, 根据重要程度分配不同的权值ω_j, 则n个作业的NSBM问题的提前/拖期惩罚函数表示为

$$f_1=\sum_{j=1}^{n}P_j(C_j,E_j)\omega_j/n,\ \omega_j\in[0,1],\ \sum\omega_j=1.\tag{2}$$

本文第2个优化目标制造跨度表示为

$$C_{\max}=\sum_{k=1}^{m}T_k+\sum_{h=1}^{m-1}d_h,$$

其结果也可表述为TrFN四元组$(C_{\max}^o,C_{\max}^{ml},C_{\max}^{mu},C_{\max}^p)$, 相应数值可通过TrFN加法操作和取大操作完成. 根据可能性理论[7], $C_{\max}$期望函数$E[C_{\max}]$表示为

$$f_2=(C_{\max}^o+C_{\max}^{ml}+C_{\max}^{mu}+C_{\max}^p)/4.\tag{3}$$

本文第3个优化目标总加工费用表示为

$$f_3=\sum_{k=1}^{m}H(T_k).\tag{4}$$

单次机器加工费用是该批次加工时间的函数, 如果减少单个批次中的作业数量, 则提前/拖期惩罚会减少, 但批次的增加将导致总加工费用的增加.

2.2 多目标优化策略

本文研究的NSBM问题, 要求同时最小化上述3个目标. 最小化多目标优化问题通常可表示为

$$\mathrm{Min}\,y=F(x)=\{f_1(x),f_2(x),\cdots,f_n(x)\}.\tag{5}$$

其中: 决策向量$x\in R^m$, 目标向量$y\in R^n$, $f_p(x)$为第p个目标函数$(1\leqslant p\leqslant n)$. 通常, 多目标问题不存在唯一的全局最优解, 但存在一类解使得对一个或多个目标函数不能进一步优化, 而对于其他目标函数不至于劣化, 这种解称为非劣解.

定义 1[11] 对于问题的2个目标向量$\boldsymbol{U}=\{u_1,u_2,\cdots,u_n\}$和$\boldsymbol{V}=\{v_1,v_2,\cdots,v_n\}$, 称$\boldsymbol{U}$非劣于$\boldsymbol{V}$, 记为$\boldsymbol{U}\prec\boldsymbol{V}$, 当且仅当$\forall i=1,2,\cdots,n$, $u_i\leqslant v_i$且$\exists i,u_i<v_i$.

定义 2[11] 可行解x_u称为多目标问题的非劣最优解, 当且仅当x_u非劣于$\forall x\in R^m$, 即不存在可行解x_v, 使得相应的目标向量$\boldsymbol{V}=F(x_v)$优于$\boldsymbol{U}=F(x_u)$. 多目标问题的所有非劣最优解组成的集合称为Pareto最优解集, 记为PS*; 与Pareto最优解集对应

的非劣最优目标向量集合 $\mathrm{PF}^* = \{F(x)|x \in \mathrm{PS}^*\}$ 称为 Pareto 前沿.

对于多目标优化问题, 有如下定理:

定理 1[11]　给定一权向量

$$\omega = W = \Big\{\omega|\omega \in R^n, \sum_{i=1}^{n}\omega_i = 1, \omega \geqslant 0\Big\},$$

形成如下单目标优化问题:

$$\mathrm{Min}\sum_{i=1}^{n}\omega_i f_i(x), \tag{6}$$

则问题 (6) 的最优解一定是问题 (5) 的非劣解. 当权向量的值取遍空间 W 时, 便可以得到问题 (5) 的所有非劣解.

多目标问题的不同目标函数通常属于不同的量纲, 容易出现部分目标函数总是处于支配地位的现象, 所以有必要对各目标函数进行归一化处理. 本文采用文献 [10] 中基于模糊逻辑的方法处理目标函数的归一化问题. 本文研究的 NSBM 问题存在 3 个目标函数 $f_1(x)$, $f_2(x)$ 和 $f_3(x)$, 对于任一可行解 x, 定义向量 $\boldsymbol{f}(x) = (f_1(x), f_2(x), f_3(x))^{\mathrm{T}}$, 根据文献 [10] 中的定理分别计算出 3 个目标函数的下界 f_1^*, f_2^*, f_3^*, 则 $\boldsymbol{f}(x) \in \prod_{p=1}^{3}[f_p^*, +\infty)$. 选择混沌优化作为启发式, f_p^C 为第 p 个目标经启发式后得到的最优值, 对于 $\boldsymbol{f}(x)$ 的每个分量, 根据其在区间 $[f_p^*, f_p^C + \alpha_p]$ 中的位置, 利用模糊逻辑进行归一化处理, 其中 $f_p^* = f_p^C$ 时, $\alpha_p = 0.01 f_p^*$, 否则 $\alpha_p = 0$; 然后按下式计算各目标函数的隶属函数:

$$u_p(f_p(x)) = \begin{cases} 1,\ f_p(x) = f_p^*; \\ \dfrac{f_p^C - f_p(x) + \alpha_p}{f_p^C - f_p^* + \alpha_p},\ f_p(x) \in [f_p^*, f_p^c + \alpha]; \\ 0,\ f_p(x) \geqslant f_p^C + \alpha_p. \end{cases} \tag{7}$$

其中 $f_p(x) \in [f_p^*, f_p^C + \alpha_p]$. 这样, 由 $\boldsymbol{f}(x)$ 可以生成 $f'(x) = (u_1(f_1(x)), u_2(f_2(x)), u_3(f_3(x)))^{\mathrm{T}}$. 由于向量 $f'(x)$ 的每一个分量都是经过归一化处理后属于区间 $[0, 1]$ 上的值, 消除了目标函数量纲不同带来的问题, 因而可用 $f'(x)$ 表示解 x 的质量.

针对 NSBM 问题的特点, 同时考虑到求解复杂度, 根据文献 [11] 中定理 1, 可将多目标优化问题转化为单目标优化问题进行求解. 此时, 权系数的取值对搜索方向具有重要的指导作用, 直接影响到搜索结果的优劣. 权系数一般很难人为确定, 而且固定的权值必将限制种群的多样性. 本文采用权系数随机生成的方式, 即

$$w_p^t = r_p^t \Big/ \sum_{p=1}^{3} r_p^t, \tag{8}$$

其中 r_p^t 是迭代中第 t 代产生的一个随机数. 同一代种群的不同粒子使用不同的权系数来计算各自的适应度值, 使得有的非劣解虽然采用某组权系数得到的适应度值较大而被淘汰, 但在另一组权系数下得到的适应度值则较小而被保留. 每一代都要根据式 (8) 重新计算权系数, 当权系数取值不同时, 各非劣解都有机会被选中, 从而保证算法有能力搜索到问题解空间的更多区域.

3　基于混沌优化的混合粒子群求解算法

在粒子群算法中, 每个优化问题的解称为粒子, 每个粒子都有自己的位置、速度 (决定飞行的方向和距离) 和适应度值. 在每一次迭代中, 粒子通过跟踪个体极值 P_{best} 和全局极值 G_{best} 来更新自己. 在找到这 2 个最优值时, 每个粒子根据下式更新自己的速度和位置[5]:

$$v_{id}(t+1) = wv_{id}(t) + c_1 r_1[P_{\mathrm{best}_{id}}(t) - x_{id}(t)] + c_2 r_2[(G_{\mathrm{best}_d}(t) - x_{id}(t)], \tag{9}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1). \tag{10}$$

其中: 粒子 i 的信息可用 D 维向量表示, 位置表示为 $X_i = (x_{i1}, x_{i2}, \cdots, x_{id})$, 速度为 $V_i = (v_{i1}, v_{i2}, \cdots, v_{id})$; c_1 和 c_2 称为加速因子, $c_1 > 0, c_2 > 0$, 分别调节向 G_{best} 和 P_{best} 方向飞行的最大步长, 合适的 c_1, c_2 可以加快收敛且不易陷入局部最优, 通常取 $c_1 = c_2 = 2$; r_1 和 r_2 为 [0,1] 区间的随机数; w 称为惯性因子, 可以根据需要调整其大小, 以便对解空间进行不同范围的探察. 对于 PSO 算法, 其性能对参数的依赖性也较大, 容易出现导致问题陷入局优解的早熟收敛问题. 本文引入混沌优化技术, 利用混沌的特性改善算法的性能, 给出一种基于混沌优化技术的混合 PSO 算法. 利用混沌对 PSO 参数进行自适应调整, 同时为了避免陷入局部最优, 并寻求全局最优解, 对 PSO 算法得到的 G_{best} 进行混沌优化, 本文选用 Logisic 混沌映射来产生混沌变量[12]. 给定变量 ω 在第 s 代的值, 则可根据下式得到第 $s+1$ 代的值:

$$\omega_i^{(s+1)} = \mu\omega_i^{(s)}(1 - \omega_i^{(s)}),\ 0 < \omega_i < 1.0. \tag{11}$$

3.1　粒子群编码与初始化

对于工件数为 n 的问题, 粒子用 n 维向量表示, 第 i 个粒子的位置向量 $X_i = [x_{i1}, x_{i2}, \cdots, x_{in}]$, 表示待加工作业的一个排列, 其中 $x_{ij}(j = 1, 2, \cdots, n)$ 对应工件 j 的优先值, 加工时按照 j 递增的顺序进行; 速度向量 $V_i = [v_{i1}, v_{i2}, \cdots, v_{in}]$, 每一组的速度 v 用来更新 x_i 中的对应维度 x.

令粒子的总数为 p_{size} (种群规模), 粒子群初始化工作按以下步骤进行:

Step 1: 按作业的加工时间递增的顺序获得粒子 X_i, 加工时间相同的作业按作业编号的递增顺序排列.

Step 2: 随机产生自然数 $\mathrm{rand}_1, \mathrm{rand}_2 \in [1, n]$, 交换 X_i 中的第 rand_1 和 rand_2 位, 作为 X_{i+1}.

Step 3: 循环进行 Step 2, 直至个体数达到种群规模 p_{size}.

3.2 粒子适应度计算

根据式 (8), 同一代种群中的不同粒子使用不同的权系数来计算各自的适应度, 从而保证算法有能力搜索到问题解空间的更多区域. 本文所求的多目标优化问题的适应度函数可表示为

$$\mathrm{Fit}(x_i) = \sum_{p=1}^{3} w_p u_p(f_p(X_i)). \tag{12}$$

其中: w_p 表示第 p 个目标的权重, X_i 表示第 i 个粒子.

迭代中产生的粒子 X_i 仅仅是一个待加工作业序列, 只有对 X_i 进行分批后, 才能计算粒子个体的适应度. 本文采用以下分批规则[13]求解 X_i 的适应度:

Step 1: 若已经产生了 k 个待加工批次 $b_1, b_2, \cdots, b_k$, $k \in [1, m]$, 其中尚有空余机器的批次为 $\{b_1^*, \cdots, b_h^*, \cdots, b_g^*\}$, $g \in [1, k]$. 当前待分批的作业为 X_{ij}, 则将 X_{ij} 按升序纳入 b_k^*. 若纳入之后, 该批的总尺寸仍然不超过 B_0, 则 $X_{ij} \in b_k^*$; 若所有尚有空余机器的批次均不能容纳 X_{ij}, 则新建批 b_{k+1}^*, $X_{ij} \in b_{K+1}^*$.

Step 2: 对 $X_i = [X_{i1}, X_{i2}, \cdots, X_{in}]$ 中所有的 X_{ij}, $j \in [1, n]$, 执行 Step 1, 得到最终的分批方案.

Step 3: 完成分批后, 得到粒子 X_i 适应度 $\mathrm{Fit}(X_i)$.

3.3 混沌局部优化

为了保持种群的持续进化能力, 本文对 PSO 算法产生的每一代的全局极值 G_{best} 进行混沌局部优化, 其过程描述如下:

Step 1: 对 PSO 搜索到的 G_{best} 根据式 (13) 计算混沌变量的初值 $\omega_i^{(0)}$, 即

$$\omega_i^{(s)} = [x_i^{(s)} - a_i]/d_i. \tag{13}$$

其中: $s = 0$; a_i, d_i 为常数.

$$a_i = \mathrm{Min\_}x_i,\ d_i = \mathrm{Max\_}x_i \mathrm{Min\_}x_i,$$
$$x_i^{(s)} \in (\mathrm{Max\_}x_i, \mathrm{Min\_}x_i).$$

Step 2: 根据式 (11) 计算出下一代的混沌变量 $\omega_i^{(s+1)}$.

Step 3: 根据式 (14) 将混沌变量 $\omega_i^{(s+1)}$ 转换成

$$x_i^{(s+1)} = a_i + d_i \omega_i^{(s+1)},\ i = 1, 2, \cdots, n. \tag{14}$$

Step 4: 计算 $x_i^{(s+1)}$ 的适应度. 若 $x_i^{(s+1)}$ 优于 G_{best} 或者达到最大迭代次数, 则将 $x_i^{(s+1)}$ 作为局部混沌搜索的解输出, 算法结束; 否则, 转 Step 2.

3.4 自适应参数策略

PSO 的性能受参数 w, r_1, r_2 影响较大, 在基本 PSO 算法中, 将 r_1, r_2 设置为随机数的方法并不能保证整个问题空间被完全遍历. 本文利用混沌对 PSO 的相关参数进行自适应调整, 即

$$r_i(t+1) = 4r_i(t)(1 - r_i(t)),\ r_i(t) \in (0, 1), \tag{15}$$

以提高算法的局部和全局搜索能力. 此外, w 也影响到 PSO 的收敛性能, 较大的 w 有利于搜索较大的空间, 但搜索精度较低; 较小的 w 能提高搜索精度, 但容易陷入局部极值. 本文采用如下线性函数动态调节 w:

$$w(t) = w_{\max} - (w_{\max} - w_{\min})t/N_{\max}. \tag{16}$$

其中: $w_{\max}$ 和 $w_{\min}$ 分别表示 w 的最大值和最小值, t 为当前代数, $N_{\max}$ 为最大进化代数. 在搜索初期, w 取值较大, 算法具有良好的全局搜索能力. 随着迭代代数的增加, w 逐渐减小, 使得在局部区域粒子速度逐渐放慢, 以提高算法的搜索精度.

3.5 混合粒子群算法

基于混沌优化的混合粒子群求解算法 (HPSO) 步骤如下:

Step 1: 设定 HPOS 的有关参数, 如最大进化代数 $N_{\max}$, 种群数 p_{size}, 迭代次数 $N = 0$.

Step 2: 根据 3.1 节中的方法, 生成 p_{size} 个粒子, 完成粒子群初始化.

Step 3: 根据 3.2 节中的适应度计算方法, 对种群中全体粒子计算粒子 X_i^n 的当前适应度, 令粒子个体最优解 $P_{\mathrm{best}_i}^n = X_i^n$, 并记录种群内最优解 G_{best}^n.

Step 4: 判断算法终止条件. 如果迭代次数 N 等于最大迭代次数 $N_{\max}$, 则输出最终解, 转 Step 6; 否则, 转 Step 5.

Step 5: 对种群中每个粒子 X_i^n, 执行以下操作:

Step 5.1: 利用 $P_{\mathrm{best}_i}^n$ 和 G_{best}^n 进行交叉, 根据式 (9) 和 (10) 更新 X_i^n 的速度和位置;

Step 5.2: 计算粒子 X_i^n 的当前适应度, 若 X_i^n 的当前适应度优于其历史最优适应度, 则将历史最优解 $P_{\mathrm{best}_i}^n$ 更新为 X_i^n;

Step 5.3: 寻找当前种群内最优解 G_{best}^n, 若优于历史最优解, 则更新 G_{best}^n;

Step 5.4: 根据 3.3 节中的方法对 G_{best}^n 进行混沌局部搜索, 若搜索出更加优化的解, 则更新 G_{best}^n;

Step 5.5: $n = n + 1$.

Step 6: 对最终解进行解码, 输出调度安排方案和目标函数.

4 实验分析

目前对于具有模糊时间参数的多目标 NSBM 问题的研究文献较少, 尚缺乏经典算例. 本文给出一个有 12 个作业的 NSBM 算例, 机器容量为 130, 各作业的交货期、加工时间和作业尺寸如表 1 所示. 批次间隔时间固定为 0.5 个时间单位, 加工费用 $H(T_k) =$

$100\,T_k + 45$, 以最小化制造跨度、加工费用和提前/拖期惩罚为目标. 采用HPSO算法进行处理, 设定粒子群初始种群规模 p_{size} 为 100, 最大进化代数 $N_{\max}$ 为 400, $w_{\max}=1.2$, $w_{\min}=0.2$, $c_1=c_2=2$, 混沌搜索最大代数为 $0.6\,N_{\max}$. 针对表 1 所示算例, 采用HPSO算法连续运行 30 次, 其中 17 次得到非劣最优解. 获得非劣解时的平均收敛代数为 96, 其中一种近似最优的作业调度安排为: 作业 8 和 11 为第 1 批次; 作业 4, 6 和 9 为第 2 批次; 作业 3 和 12 为第 3 批次; 作业 5, 7 和 10为第第 4 批次; 作业 1 和 2 为第 5 批次.

表 1　作业信息

作业	模糊加工时间	尺寸	模糊交货期
1	(8.4, 9.6, 14.4, 15.6)	36	(31.5, 36.2, 53.5, 57.5)
2	(8.4, 9.6, 14.4, 15.6)	62	(32.1, 35.8, 54.1, 58.3)
3	(4.2, 4.8, 7.2, 7.8)	56	(14.1, 16.2, 24.2, 26.0)
4	(4.2, 4.8, 7.2, 7.8)	41	(6.9, 8.3, 11.3, 12.6)
5	(6.3, 7.2, 10.8, 11.7)	37	(22.8, 26.2, 39.6, 42.9)
6	(2.1, 2.4, 3.6, 3.9)	49	(7.3, 8.4, 11.5, 12.0)
7	(8.4, 9.6, 14.4, 15.6)	55	(23.1, 26.4, 39.5, 41.9)
8	(2.1, 2.4, 3.6, 3.9)	43	(2.3, 2.5, 3.0, 3.2)
9	(4.2, 4.8, 7.2, 7.8)	38	(7.1, 8.0, 11.2, 12.5)
10	(6.3, 7.2, 10.8, 11.7)	32	(22.9, 26.7, 38.9, 42.1)
11	(2.1, 2.4, 3.6, 3.9)	45	(2.4, 2.5, 3.1, 3.3)
12	(6.3, 7.2, 10.8, 11.7)	52	(13.9, 16.2, 23.9, 26.1)

本文分别从非劣解的数量和分布情况考察算法在解决多目标NSBM问题上的有效性. 对于多目标优化算法, 通常可用分布度 Δ 表示所求解集的分布性, 用分布度的变化率 Δ_v 表示算法运行的稳定性. 分布度越小表示所求解集的分布性越好, 分布度变化率的值越小表明算法运行的稳定性越好. 计算方法分别为

$$\Delta = \sum_{i=1}^{\text{ParetiSet}} \frac{|d_i - \bar{d}\,|}{\text{ParetoSet}}, \tag{17}$$

$$\Delta_v = \sum_{i=1}^{\beta} \frac{|\Delta_i - \bar{\Delta}\,|}{\beta}. \tag{18}$$

其中: d_i 表示非劣解集中相邻 2 个个体之间的欧几里德距离, $\bar{d}$ 为 d_i 的平均值, Δ_i 为一次运行结束后所求得的 Δ 的值, $\bar{\Delta}$ 为 Δ_i 的平均值, ParetoSet 表示非劣解集, β 为算法运行的次数.

为体现HPSO算法的性能, 针对该算例同时采用HPSO, 基本PSO和GA分别进行处理. 3 种算法的初始种群规模与最大进化代数设置相同, 基本PSO算法参数设置如下: $w=0.7, c_1=c_2=2$; GA 中交叉概率为 0.85, 变异概率为 0.05, $\beta=30$. 3 种算法得到的非劣解数量、非劣解集的分布情况以及平均收敛代数等数据的对比结果如表 2 所示.

表 2　算法性能分析

算法	非劣解数量	分布度	分布度变化率	平均收敛代数
HPSO	17	0.583	0.043	96
GA	11	0.691	0.057	145
PSO	9	0.733	0.052	88

由表 2 可以看出, 3 种算法都可以找到NSBM问题的非劣解, 但GA的平均收敛代数最高, 基本PSO算法收敛代数与HPSO接近, 但求解的质量较差, 获得非劣解的数量明显低于HPSO. 与基本PSO和GA相比, HPSO不仅能产生更多的在已知非劣解前沿上的解, 而且所得到的非劣解具有很好的分布性和稳定性.

5　结　论

本文针对具有模糊交货期和模糊加工时间的多目标NSBM问题, 利用可能性理论、有符号距离、区间数距离以及TrFN代数构建了包括提前/拖期惩罚、制造跨度等目标函数. 针对多目标问题的特征, 采用了基于模糊逻辑和随机权重的适应度函数, 以获得更多优良的非劣解. 在NSBM问题求解方面, 针对粒子群算法易陷入局部最优问题, 引入混沌局部搜索策略, 构造出了一种基于混沌优化的混合粒子群算法, 以提高算法的局部搜索能力. 实验表明, 混合粒子群算法具有较好的搜索效率和搜索质量, 而且具有较快的收敛速度, 改善了基本PSO算法易陷入局部极值的问题. 本文讨论了差异作业批调度的单机问题, 然而在现实生产系统中, 多机问题也是大量存在的, 作者将对此做进一步的研究.

参考文献(References)

[1] Uzsoy R. Scheduling a single batch processing machine with non-identical job sizes[J]. Int J of Production Research, 1994, 32(7): 1615-1635.

[2] Sevaux M, Peres S D. Genetic algorithms to minimize the weighted number of late jobs on a single machine[J]. European J of Operational Research, 2003, 151(2): 296-306.

[3] Purushothaman Damodaran, Paraveen Kumar Manjeshwar, Krishnaswami Srihari. Minimizing makespan on a batch-processing machine with non-identical job sizes using genetic algorithms[J]. Int J of Production Economics, 2006, 103(2): 882-891.

[4] Melouk S, Damodaran P, Chang P-Y. Minimizing make span for single machine batch processing with nonidentical job sizes using simulated annealing[J]. Int J of Production Economics, 2004, 87(2): 141-147.

[5] Eberhart R C, Shi Y. Partical swarm optimization: Developments, applicationsand resources[C]. Congress on Evolutionary Computation 2001. Piscataway: IEEE Press, 2001: 81-86.

[6] Kennedy J, Eberhart R C. Particle swarm optimization[C]. IEEE Int Conf. Piscataway: IEEE Service Center, 1995: 1942-1948.

(下转第1684页)