文章编号: 1000−8152(2012)06−0715−08

# 混合粒子群算法求解多目标柔性作业车间调度问题

张 静[1,2], 王万良[1], 徐新黎[1], 介 婧[1]

(1. 浙江工业大学 计算机科学与技术学院, 浙江 杭州 310023; 2. 浙江工业大学 信息工程学院, 浙江 杭州 310023)

**摘要**: 柔性作业车间调度问题是生产管理领域和组合优化领域的重要分支. 本文提出一种基于Pareto支配的混合粒子群优化算法求解多目标柔性作业车间调度问题. 首先采用基于工序排序和机器分配的粒子表达方式, 并直接在离散域进行位置更新. 其次, 提出基于Baldwinian学习策略和模拟退火技术相结合的多目标局部搜索策略, 以平衡算法的全局探索能力和局部开发能力. 然后引入Pareto支配的概念来比较粒子的优劣性, 并采用外部档案保存进化过程中的非支配解. 最后用于求解该类问题的经典算例, 并与已有算法进行比较, 所提算法在收敛性和分布均匀性方面均具有明显优势.

**关键词**: 粒子群; 多目标优化; 柔性作业车间调度问题; Baldwinian学习策略

**中图分类号**: TP301.6 **文献标识码**: A

# Hybrid particle-swarm optimization for multi-objective flexible job-shop scheduling problem

ZHANG Jing[1,2], WANG Wan-liang[1], XU Xin-li[1], JIE Jing[1]

(1. College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou Zhejiang, 310023, China;
2. College of Information Engineering, Zhejiang University of Technology, Hangzhou Zhejiang, 310023, China)

**Abstract:** Flexible job-shop scheduling is a very important branch in both fields of production management and combinatorial optimization. A hybrid particle-swarm optimization algorithm is proposed to study the mutli-objective flexible job-shop scheduling problem based on Pareto-dominance. First, particles are represented based on job operation and machine assignment, and are updated directly in the discrete domain. Then, a multi-objective local search strategy including Baldwinian learning mechanism and simulated annealing technology is introduced to balance global exploration and local exploitation. Third, Pareto-dominance is applied to compare different solutions, and an external archive is employed to hold and update the obtained non-dominated solutions. Finally, the proposed algorithm is simulated on numerical classical benchmark examples and compared with existing methods. It is shown that the proposed method achieves better performance in both convergence and diversity.

**Key words:** particle swarm optimization; multi-objective optimization; flexible job-shop scheduling problem; Baldwinian learning mechanism

## 1 引言(Introduction)

调度的主要任务是合理分配现有的人力和物力资源, 以满足生产过程中经济上或性能上的目标[1]. 柔性作业车间调度问题(flexible job-shop scheduling problem, FJSP)突破机器唯一性的限制, 工序可由不同机器加工, 更贴近实际, 在柔性制造系统、化工制造行业、交通运输系统等领域都得到了广泛的应用.

实际生产中调度问题往往包含着多个相互冲突的目标, 但已有文献对多目标调度的问题研究相比单目标的少很多, 柔性多目标调度问题将是一个值得研究的方向[2]. 目前求解多目标FJSP的方法主要有进化算法和群智能优化算法. 文献[3–4]设计了混合遗传算法, 采用集成权重系数法以最小化多个求解目标的加权和. 文献[5]设计了进化算法和局部优化的混合方法. 文献[6]将进化算法和模糊逻辑相结合, 均采用Pareto方法以均衡FJSP的完工时间和机器负载. 文献[7]设计了一种局部搜索算法. 文献[8]将蚁群算法和局部搜索算法相结合, 并针对FJSP开发了一种仿真模型, 均根据目标值的重要程度为目标值设置不同的权重. 文献[9]提出基于 Pareto 的离散蜂群算法, 将其与多种局部搜索方法相结合, 并基于Pareto概念对种群进行支配快速非支配排序. 从以上文献可以看出, 不管是进化算法还是群智能优化算法, 与局部搜索相结合的混合算法的研究得到了很大发展, 且这样的混合算法能有效防止算法过早收敛, 因此采用混合算法求解多目标FJSP是可行的途

收稿日期: 2011−06−29; 收修改稿日期: 2012−01−05.

基金项目: 国家自然科学基金资助项目(60874074, 61070043); 浙江省自然科学基金资助项目(Y1090592); 中国博士后科学基金资助项目(20090451486).

径之一.

粒子群(particle swarm optimization, PSO)[10]是基于群体智能理论的一种新兴演化计算技术, 近年来得到了学术界和工程界的广泛重视[11–12]. 文献[13]将粒子表达为可用机器的优先水平, 并将PSO和模拟退火技术(simulated annealing, SA)结合求解多目标FJSP, 文献[14]采用基于随机键编码的ROV规则实现粒子连续位置到工件排序离散值的转化, 使PSO适用于求解流水车间调度问题, 但以上两篇文献皆采用加权系数法将FJSP的3个目标转化为一个目标进行求解, 一次求解只能得到一个最优解, 难以很好的反应实际多目标问题, 基于Pareto的方法可解决上述问题. 文献[15]采用优先规则将作业车间调度问题(job-shop scheduling problem, JSP)转化为连续优化问题, 并设计了具有Pareto外部档案的PSO. 文献[16]采用基于优先权的设备分配规则和工序排序策略, 并引入$\delta$方法精英解策略以使PSO获得最好的局部引导. 文献[17]采用具有离散操作的基于Pareto外部档案的PSO, 并将其与变邻域搜索策略相结合, 达到最小化JSP总流经时间和拖期/惩罚时间的目的. 尽管将PSO应用于调度问题都需要一些调整和改变, 但具有离散操作的PSO无需寻求将调度问题转化为连续问题的恰当方法, 因此在调度方面更具有优势, 且基于Pareto支配关系的离散PSO求解多目标FJSP的文献还相对较少, 如何使PSO具有较强的探索能力和开发能力, 从而获得具有良好分布性的解集仍是值得研究的问题.

结合已有文献的优势与不足, 提出基于Pareto支配关系的混合粒子群算法(hybrid particle swarm optimization, HPSO)用于解决多目标FJSP. HPSO采用两级染色体编码方式表示粒子, 并直接在离散域进行粒子位置更新操作, 减少了计算时间. 首先采用混合方法初始化粒子群, 加快算法收敛速度. 其次针对FJSP的特点, 提出一种新的Baldwinian学习策略(baldwinian learning, BL), 并将其与SA结合构造一种新的多目标局部搜索策略(multi-objective local search, MLS), 有效避免算法早熟. 然后针对多目标FJSP解的分布性问题, 建立一个外部档案存放算法全局搜索过程中的非支配解, 并根据Pareto支配关系判断进化过程中的解与外部档案中解的优劣性. 最后将HPSO用于解决多目标FJSP经典Benchmark算例, 实验仿真说明所提算法能求得更多分布均匀的接近Pareto前沿的非支配解.

## 2 问题描述(Problem formulation)

多目标FJSP是一类满足任务配置和顺序约束要求的资源分配问题, 可以描述为: 有$n$个相互独立的工件需要在$m$台机器上加工, 每个工件有若干个工序, 能加工某一个工序的机床有多台, 工序的加工时间根据机器的性能不同而变化. 本调度需要解决的问题是: 为每道工序分配加工机器, 并确定机器上各个工序的加工顺序, 在满足约束的条件下最小化完工时间、机器总负载和单台机器最大负载. 数学模型[17]建立如下:

### 2.1 变量定义(Variables definition)

1) 标号: $p$为工件号, $q$为工序号, $h$为机器号, $O_{pq}$表示工件$p$的第$q$个工序, $J$为工件集合, $M$为机器集合, $M_{pq}$为工序$O_{pq}$的可选机器集合($M_{pq}\in M$), $W_h$为机器$h$的负载.

2) 给定参数: $n$为总工件数, $m$为总机器数, $n_p$为工件$p$所含的工序数, $t_{pqh}$为工序$O_{pq}$上在机器$h$上的加工时间, $U$为一个很大的数.

3) 决策变量: $S_{pq}$为工序$O_{pq}$的开始加工时间, $C_{pq}$为工序$O_{pq}$的结束加工时间, $C_p$为工件$p$的完工时间, $C_{\max}$为所有工件的最大完工时间.

$$\sigma_{pqh}=\begin{cases}1, \text{ 工件}p\text{的第}q\text{道工序分配的机器为}h,\\ 0, \text{ 其他},\end{cases}$$

$$\zeta_{pqh-p'q'h'}=\begin{cases}1, \text{ 工件}p\text{的第}q\text{道工序先于工件}p'\\ \quad \text{的第}q'\text{道工序在机器}h\text{上加工},\\ 0, \text{ 其他}.\end{cases}$$

### 2.2 数学模型(Mathematical model)

为便于求解, 加工过程需满足以下假设和约束: 所有的机器在时间$t=0$时刻都是可以使用的. 每个工件都可以在$t=0$时刻开始加工. 在给定的时间内, 一台机器只能加工一道工序. 对于每个工件的各道工序只能按照事先给定的顺序加工.

数学模型和约束条件如下:

$$f_1=\min C_{\max}=\min\{\max_{p=1}^{n}\{C_p\}\}, \tag{1}$$

$$f_2=\sum_{h=1}^{m}W_h, \tag{2}$$

$$f_3=\max_{h=1}^{m}\{W_h\}, \tag{3}$$

s.t.

$$\begin{cases}S_{p(q+1)}\geqslant S_{pq}+t_{pqh}\times\sigma_{pqh},\\ P\in J,\ h\in M_{pq},\ q=1,2,\cdots,n_p-1,\end{cases} \tag{4}$$

$$\begin{cases}S_{p'q'}+(1-\varsigma_{pqh-p'q'h})U\geqslant S_{pq}+t_{pqh},\\ p,p'\in J,\ h\in M_{pq},\ q,q'=1,2,\cdots,n_p,\end{cases} \tag{5}$$

$$\begin{cases}S_{p(q+1)}+(1-\varsigma_{p(q+1)h-p'q'h})U\geqslant C_{pq},\\ p,p'\in J,\ h\in M_{pq},\ q,q'=1,2,\cdots,n_p-1,\end{cases} \tag{6}$$

$$\sum_{h=1}^{M_{pq}}\sigma_{pqh}=1,\ p\in J,\ q=1,2,\cdots n_p, \tag{7}$$

$$C_p\leqslant C_{\max},\ p\in J, \tag{8}$$

$$\sum_{p=1}^{n}\sum_{q=1}^{n_p}t_{pqh}\times\sigma_{pqh}\leqslant W_h,\ h\in M_{pq}, \tag{9}$$

$$\begin{cases} S_{pq},\ t_{pqh} \geqslant 0,\ p \in J, \\ q = 1, 2, \cdots, n_p,\ h = 1, 2, \cdots, m, \end{cases} \tag{10}$$

$$\begin{cases} \sigma_{pqh},\ \varsigma_{pqh-p'q'h} \in \{0, 1\}, \\ p,\ p' \in J,\ h \in M_{pq},\ q, q' = 1, 2, \cdots, n_p. \end{cases} \tag{11}$$

式(1)−(3)依次表示目标函数完工时间、机器总负载和单台机器最大负载, 式(4)表示每个工件的工序必须按着给定的顺序加工, 式(5)和式(6)表示一个工序只能在所选机器空闲并且该工件的前道工序加工完成后才能加工, 式(7)表示每个工序只能从其候选机器集合中选择一台机器, 式(8)表示最大完工时间, 式(9)表示机器负载, 式(10)表示工序的开始时间和加工时间, 式(11)表示决策变量的取值范围.

## 3 多目标混合粒子群算法(Multi-objective hybrid particle swarm optimization)

为有效求解多目标FJSP, 本文采用两级染色体编码方式, 提出基于BL和SA技术的MLS策略, 构造了新的多目标HPSO. 具体介绍如下:

### 3.1 编码(Encoding)

求解FJSP首先需对其进行编码, 采用两级染色体表示一个粒子$X_i$的位置, $i = 1, 2, \cdots, P$, $P$为种群规模. 第1条染色体$A$向量表示基于工序的编码, 第2条染色体$B$向量表示基于机器的编码. 两条染色体的长度一致, 均为$g = \sum n_p$. $A$的各个元素为工件号, 并表示了工序加工的先后顺序, $B$的各元素为机器号, 表示加工相应工序的机器.

本文基于上述编码方式, 对粒子群进行初始化. $A$向量采用随机生成的方式和最多工序剩余的工件先加工的原则. $B$向量采用随机分配机器的方式和初始种群定位法[5]. 同时为避免HPSO早熟收敛, 采用部分粒子重新初始化的方式打破之前的种群平衡: 若外部档案中的非支配解连续gen代未有改进, 将种群中10%粒子随机初始化, 并随机选择外部档案中的粒子替代种群的一个粒子, 其中gen为外部档案进化停滞周期.

### 3.2 多目标全局搜索策略(Multi-objective global search structure)

为了存放HPSO在全局搜索过程中的获得的非支配解, 建立一个外部档案Achive, 并基于Pareto支配关系对Achive更新, 更新策略简单易行: 对于一个新的粒子$Q$, 将其与Achive中的成员进行比较, 如它被Achive中的任一个成员支配, 则拒绝$Q$加入. 如果$Q$和Achive中的所有成员互不支配, 则直接加入到Achive中. 如果$Q$支配了档案中的部分成员, 则移除受支配的成员, 同时将$Q$加入Achive中.

采用一种适用于求解FJSP的PSO位置更新方式作为HPSO算法的全局搜索策略[18], 该更新公式可直接在离散域操作, 有效加快算法的收敛速度. 而在多目标优化条件下, 粒子的全局最优位置不再唯一, 本文采用基于Pareto外部档案的方法, 从外部档案中随机选取, 能够使粒子在多个非支配解中随机选择全局的学习信息, 从而避免粒子因学习信息单一而快速趋同, 进而避免PSO过早收敛, 有效增强其探索能力. 位置更新公式如下:

$$\begin{aligned} {X_i}^{k+1} = c_2 \otimes f_4(c_1 \otimes f_3(c_1 \otimes f_2(w \otimes \\ f_1({X_i}^k), {pB_i}^k), {pB_i}^k), gB^k), \end{aligned} \tag{12}$$

其中: $X_i^k (i = 1, 2, \cdots, P)$,表示粒子$i$第$k$代的位置, $w$为惯性权重, $c_1$为认知系数, $c_2$为社会系数, $w$, $c_1$, $c_2 \in [0, 1]$, $pB_i^k$是粒子$i$第$k$代自身最优位置, $gB^k$是第$k$代的全局最优位置, $f_1$, $f_2$, $f_3$和$f_4$是操作算子. 更新公式(12)由a), b), c)和d)4部分组成:

a)
$${E_i}^k = w \otimes f_1({X_i}^k) = \begin{cases} f_1({X_i}^k),\ r < w, \\ {X_i}^k, \quad \text{其他}, \end{cases} \tag{13}$$

其中$r$为(0,1)间的随机数, 式(13)表示粒子对先前状态的思考. 实现方法为: $A$向量随机交换两个不同基因的位置, $B$向量随机选择一个不同的机器替换.

b)
$${F_i}^k = c_1 \otimes f_2({E_i}^k, {pB_i}^k) = \begin{cases} f_2({E_i}^k, {pB_i}^k),\ r < c_1, \\ {E_i}^k, \quad \text{其他}. \end{cases} \tag{14}$$

c)
$${G_i}^k = c_1 \otimes f_3({F_i}^k, {pB_i}^k) = \begin{cases} f_3({F_i}^k, {pB_i}^k),\ r < c_1, \\ {F_i}^k, \quad \text{其他}. \end{cases} \tag{15}$$

式(14)和式(15)表示粒子根据自身最优位置${pB_i}^k$进行两次调整. $f_2(E_i^k, {pB_i}^k)$和$f_3(F_i^k, {pB_i}^k)$分别表示POX(precedence preserving order based crossover)和MPX(rand-point preservation crossover)交叉算子, 具体操作见文献[18].

d)
$${X_i}^k = c_2 \otimes f_4({G_i}^k, gB^k) = \begin{cases} f_4({G_i}^k, gB^k),\ r < c_2, \\ {G_i}^k; \quad \text{其他}. \end{cases} \tag{16}$$

公式(16)表示粒子根据全局最优位置$gB^k$调整. $f_4(G_i^k, gB^k)$的操作过程同$f_3(F_i^k, gB^k)$.

### 3.3 多目标局部搜索策略(Multi-objective local search structure)

采用上述位置编码和更新公式, 粒子可直接在离散域中进行更新, 这种PSO算法具有收敛快的优点, 但易陷入局部最优, 因此本文加入多目标局部搜索策略(MLS), 以增强PSO的探索能力. MLS包括多目标BL策略和SA技术, 改进的BL策略用于调整$A$向量, 对工序进行最佳排序. SA技术用于调整$B$向量, 为工序搜索最佳的机器分配.

Baldwinian学习策略[19]被认为是一种生物学习与进化不断迭代交替的过程, 它能够改变搜索空间的形状, 且能够提供尽可能好的路径以寻找到最优解[20], 它的主要学习过程如图1所示. 图1为粒子$i$某一维的示意图, 粒子$X_i$通过向粒子$X_k$和粒子$X_l$学习, 得到新的粒子$Y_i$, $i=1,2,\cdots,P$, $P$为种群中粒子的个数. $Y_i$比$X_i$更靠近最优区域, 经过多次迭代, 粒子将不断的向最优值方向收敛. BL学习策略能有效提高算法的探索能力, 但BL公式[19]仅适用于连续优化问题, 不能直接应用于调度问题, 因此本文对BL进化公式进行改进, 使其适用于FJSP的离散操作. 文献[21]针对FJSP特点将PSO速度更新公式中的变量表示工件工序在编码的中的位置, 本文借鉴其表达方式, 得到新的Baldwinian学习公式

$$Y_{ipq}(k)=X_{ipq}(k)+s'\times|X_{lpq}(k)-X_{kpq}(k)|, \quad (17)$$

$$s'=\begin{cases}1, & r<s,\\ 0, & \text{其他}.\end{cases} \quad (18)$$

式(17)中$X_{ipq}$表示粒子$i$的工件$p$的第$q$个工序在粒子编码中的位置索引, 其中: $i\neq l\neq k$, $i,l,k\in\{1,2,\cdots,P\}$. $s$是Baldwinian学习强度, $r$为$(0,1)$之间的随机数. 根据式(17)和式(18)的更新, 得到工序的新位置号, 将其按顺序排列获得新的工序编码$A$向量. 同时保持工序的机器分配不变, 得到对应的机器编码$B$向量.

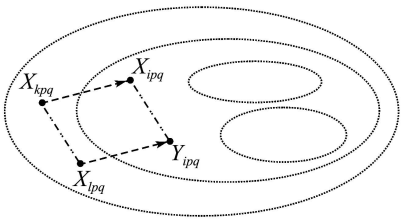


图 1 Baldwinian学习过程

Fig. 1 Baldwinian learning process

为使BL策略适合于求解多目标优化问题, 引入Pareto支配概念判断解的优劣性, 改进后的搜索过程为: 粒子群$X=\{X_i|i=1,2,\cdots,P\}$按照复制尺度$p_s$产生新粒子群$Y=\{Y_i|i=1,2,\cdots,P\}$, $Y_i=\{Y_{iz}|\,z=1,2,\cdots,p_s\}$, 分别计算粒子群$Y_i$和$X_i$的3个目标函数值, 并根据Pareto支配关系判断解的优劣, 然后在$p_s+1$个粒子中随机选择其中一个非支配解$Z_i$组成新的粒子群$Z=\{Z_i|i=1,2,\cdots,P\}$并赋值给之前的粒子群$X$.

然后采用SA技术对$B$向量进行邻域搜索, 以完工时间$f_1$为评价准则, 从负载最大的机器上随机选择一个工序, 将具有最小负载的机器分配给它.

### 3.4 混合粒子群算法求解多目标FJSP(HPSO for multi-objective FJSP)

HPSO算法求解多目标FJSP的具体步骤如下:

**Step 1** 确定参数. 设定种群规模$P$, 最大迭代次数$G$, Archive进化停滞周期gen. PSO更新公式中的常量(惯性常量$w$、学习因子$c_1$和$c_2$、自适应调整参数$s_{\max}$和$s_{\min}$)、BL策略中的常量(复制尺度$p_s$、学习强度$s$)、SA算法中的常量(初始温度$T_0$、终止温度$T_{\rm end}$、退火率$B$).

**Step 2** 种群初始化. 采用3.1节的方法初始化粒子群的位置为$X_1$, $X_2$, $\cdots$, $X_P$, 计算粒子适应值, 根据Pareto支配关系, 将非支配解放入外部档案Archive中, 初始化局部最优值$pB_i=X_i$, $i=1,2,\cdots,P$, 全局最优值$gB=$rand(Archive).

**Step 3** 根据式(12)更新粒子位置. 计算新种群的适应度值, 同时根据Pareto支配关系更新外部档案Archive、局部最优值$pB_i$和全局最优值$gB$.

**Step 4** MLS策略. 根据3.3节的步骤进行多目标BL搜索, 然后随机选择$r\in[1,P/2]$个粒子进行SA局部搜索, 并更新Archive, $pB_i$和$gB$.

**Step 5** 若Archive连续gen代未有改进, 将种群中10%粒子随机初始化, 并随机选择Archive中的一个粒子替代种群的一个粒子.

**Step 6** 判断是否达到终止条件(如迭代次数$<G$), 若满足, 则输出最优值. 否则转Step 3.

### 3.5 算法复杂度分析(Computational complexity of the HPSO algorithm)

假设目标函数个数为$M$, 粒子种群规模为$N$, 迭代次数为$TM$. 从3.4节算法流程可以看出, HPSO每次迭代主要有4部分组成: 第1部分计算目标值, 其时间复杂度约为$\mathrm{O}(MN)$; 第2部分种群内与外部档案中非支配解的比较, 一个粒子是否放入外部档案中, 最多需要与$N$个粒子比较, 则最坏情况下其时间复杂度约为$\mathrm{O}(MN^2)$; 第3部分粒子位置更新, 其中$A$向量根据自身位置调整和交叉调整各一次, $B$向量分别调整一次和两次, 其时间复杂度分别为$\mathrm{O}(3N)$和$\mathrm{O}(5N)$, 则第3部分的时间复杂度最坏约为$\mathrm{O}(8N)$; 第4部分多目标局部搜索, 其时间复杂度最坏约为$(p_s+1)\mathrm{O}(MN^2)$.

因此本文所提算法HPSO的时间复杂度为

$$\begin{aligned}\mathrm{O}(M,N,TM)=&\ [\mathrm{O}(MN)+\mathrm{O}(MN^2)+\mathrm{O}(8N)+\\ &(p_s+1)\times\mathrm{O}(MN^2)]\times TM\approx\\ &(p_s+1)\times\mathrm{O}(MN^2)\times TM.\end{aligned}$$

可以看出, HPSO的计算量主要与种群规模、目标值个数、迭代次数和MLS的复制尺度有关. 基于Pareto档案的PSO[22]进行一次迭代的时间复杂度为$\mathrm{O}(MN^2)$, 因此HPSO并没明显增加基于Pareto方法的PSO的时间复杂度.

## 4 实验结果与分析(Simulation and analysis)

### 4.1 测试问题(Test problems)

为了测试算法性能, 本节采用具有代表性的14个不同规模的FJSP测试算例进行算法比较, 并采

用前10个算例进行参数分析. 其中Case 1−Case 4是由Kacem[5–6]设计的问题, Case 5−Case 14是由Brandimarte[23]设计的MK1-MK10问题, 14个算例中Case 2−Case 4为完全FJSP(每个工序都可以被机器集内的任一机器加工), 其余为部分FJSP(某些工序只能被机器集内的某几台机器加工), 对于相同数目的机器和工件, 部分FJSP比完全FJSP更难以解决. 采用VC6.0编程, 运行环境为P4CPU/2. 66 GHz/4 GB RAM. 测试算法性能的参数如下:

1) 误差比($ER$): 用来描述在算法产生的非支配解中, 不属于真实Pareto前端的解所占的百分比: $ER=\sum_{i=1}^{n} e_i/n$, $n$为$PF_{\text{true}}$中非支配解的个数, 若$i$属于$PF_{\text{known}}$, 则$e_i=0$, 否则$e_i=1$. $ER$值越小, 说明属于$PF_{\text{known}}$的非支配解比例越高. 文中Pareto最优解集$PF_{\text{known}}$指所有算法非支配解的集合, $PF_{\text{true}}$指各个算法对应的非支配解集合.

2) 当代距离指标($GD$): $GD=(\sum_{i=1}^{n} D_i^2)^{\frac{1}{2}}/n$用来描述算法所获得的非支配解与问题的真实Pareto前端之间的距离, $n$为$PF_{\text{known}}$非支配解的个数, $D_i$为目标空间$PF_{\text{true}}$中第$i$个解与$PF_{\text{known}}$之间的欧氏距离.

### 4.2 参数分析(Sensitivity analyses)

由于不同的参数对算法的结果的影响不同, 本节分别讨论了PSO算法中的惯性权重$w$, 学习因子$c_1$, $c_2$(PSO中通常设$c_1=c_2$, 文中具有离散操作的PSO根据$c_1$调整两次, 根据$c_2$调整一次, 因此令$2c_1=c_2$). BL搜索中的复制尺度$p_s$, 学习强度$s$. 采用性能比较参数$ER$对算例Case 1−Case 10分别独立运行20次过程中所获得的非支配解进行比较, 其他参数设置为$T_0=3$, $T_{\text{end}}=0.01$, $B=0.9$, $s_{\max}=0.9$, $s_{\min}=0.2$, gen $=10$, Case 1−Case 6中$P=50$, $G=1000$, 其余$P=50$, $G=2000$, 比较结果见表1 − 4, 表中Sum$ER$为10个测试算例的误差比之和.

表 1 不同惯性权重$w$下HPSO性能比较结果

Table 1 The computational results of HPSO with different inertia weights $w$

| 算例 | $w=0.1$ | $w=0.2$ | $w=0.3$ | $w=0.4$ | $w=0.5$ | $w=0.6$ | $w=0.7$ | $w=0.8$ | $w=0.9$ |
|---|---|---|---|---|---|---|---|---|---|
| Case 1 | 0 | 0.2000 | 0 | 0 | 0 | 0 | 0 | 0.4000 | 0.3333 |
| Case 2 | 0 | 0 | 0 | 0.5000 | 0 | 0 | 0 | 0.5000 | 0 |
| Case 3 | 0.3333 | 0.2500 | 0 | 0.5000 | 0.2500 | 0.2500 | 0.2500 | 0.5000 | 0.2500 |
| Case 4 | 0 | 0 | 0 | 0.6667 | 0 | 0 | 0.6667 | 0.6667 | 0.5000 |
| Case 5 | 0.0625 | 0.5000 | 0.8636 | 0.7222 | 0.7222 | 0.6667 | 0.8000 | 0.9048 | 0.9565 |
| Case 6 | 0.8182 | 0.7778 | 0.3333 | 0.5556 | 0.7692 | 0.6923 | 1 | 0.9333 | 1 |
| Case 7 | 1 | 0.7500 | 0.8750 | 0.6875 | 0.9744 | 0.8065 | 1 | 0.9211 | 1 |
| Case 8 | 0.8182 | 0.5758 | 0 .7955 | 0.6571 | 0.2727 | 0.9333 | 0.7949 | 0.9302 | 0.9500 |
| Case 9 | 0.4737 | 0.8750 | 0.5000 | 0.5333 | 0.7143 | 0.7619 | 0.6875 | 0.8000 | 0.9130 |
| Case 10 | 0.4348 | 0.8140 | 0.7805 | 0.9792 | 0.8431 | 0.5172 | 1 | 1 | 1 |
| Sum$ER$ | 3.9407 | 4.7426 | 4.1479 | 5.8016 | 4.5459 | 4.6279 | 6.1991 | 7.5561 | 6.9028 |

表 2 不同学习因子$c$下HPSO性能比较结果

Table 2 The computational results of HPSO with different learning factors $c$

| 算例 | $c$ | | | | | |
|---|---|---|---|---|---|---|
| | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| Case 1 | 0 | 0.3333 | 0 | 0 | 0.2000 | 0 |
| Case 2 | 0.5000 | 0 | 0.5000 | 0 | 0 | 0 |
| Case 3 | 0 | 0.2500 | 0.5000 | 0 | 0.5000 | 0.5000 |
| Case 4 | 0 | 0 | 0.6667 | 0 | 0 | 0.6667 |
| Case 5 | 0.7647 | 0.7333 | 0.6667 | 0.5000 | 0.7500 | 0.8125 |
| Case 6 | 0.6667 | 0.8462 | 0.6667 | 0.7273 | 0.8333 | 0.2308 |
| Case 7 | 1 | 0.8462 | 0 | 0.9500 | 0.9231 | 0.6190 |
| Case 8 | 1 | 0.8837 | 0 .5385 | 0.4483 | 0.9250 | 0.8286 |
| Case 9 | 0.9643 | 0.5000 | 0.7368 | 0.5909 | 0.6500 | 0.5625 |
| Case 10 | 0.8333 | 0.9259 | 0.6667 | 0.6667 | 0.6667 | 0.9697 |
| Sum$ER$ | 5.729 | 5.3186 | 4.9421 | 3.8832 | 5.4481 | 5.1898 |

表 3 不同复制尺度$p_s$下HPSO性能比较结果

Table 3 The computational results of HPSO with different clonal scale $p_s$

| 算例 | $p_s$ | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| Case 1 | 0 | 0 | 0 | 0 | 0 |
| Case 2 | 0.5000 | 0 | 0 | 0 | 0 |
| Case 3 | 1 | 0 | 0 | 0.2500 | 0.8000 |
| Case 4 | 0.6667 | 1 | 0.6667 | 0 | 0 |
| Case 5 | 0.9600 | 0.5714 | 0.5000 | 0.3333 | 0.6316 |
| Case 6 | 0.3636 | 0.8462 | 0.7692 | 0.5455 | 0.3333 |
| Case 7 | 0.8696 | 0.2222 | 0.3000 | 1 | 1 |
| Case 8 | 0.8667 | 0.2273 | 0 .9250 | 0.5172 | 0.9474 |
| Case 9 | 0.6500 | 0.4444 | 0.5263 | 0.5909 | 0.8000 |
| Case 10 | 0.7049 | 0.5600 | 1 | 0.5833 | 0.8519 |
| Sum$ER$ | 6.5815 | 3.8715 | 4.6872 | 3.8202 | 5.3642 |

表 4 不同学习强度$s$下HPSO运行结果

Table 4 The computational results of HPSO with different strength of baldwinian learning $s$

| 算例 | $s=0.1$ | $s=0.2$ | $s=0.3$ | $s=0.4$ | $s=0.5$ | $s=0.6$ | $s=0.7$ | $s=0.8$ | $s=0.9$ |
|---|---|---|---|---|---|---|---|---|---|
| Case 1 | 0.3333 | 0 | 0 | 0.4000 | 0 | 0 | 0.3333 | 0 | 0 |
| Case 2 | 0 | 0.5000 | 0 | 0 | 0 | 0 | 0.5000 | 0 | 0 |
| Case 3 | 0.3333 | 0.2500 | 0.3333 | 0 | 0.5000 | 0 | 0.2500 | 0.2500 | 0.5000 |
| Case 4 | 0.6667 | 0 | 0 | 0 | 1 | 0.6667 | 0 | 0.6667 | 0 |
| Case 5 | 1 | 1 | 1 | 0.7619 | 0.8235 | 0.7895 | 0.6154 | 0.3636 | 0.7333 |
| Case 6 | 0.7500 | 0.5714 | 0.9500 | 0.7500 | 1 | 0.8750 | 0.5385 | 1 | 0.7857 |
| Case 7 | 0.5769 | 1 | 1 | 1 | 0.4286 | 0.8462 | 0.8846 | 0.9286 | 1 |
| Case 8 | 0.7174 | 0.9149 | 0 .2381 | 0.9189 | 0.9688 | 0.8750 | 0.8444 | 0.7857 | 0.9355 |
| Case 9 | 0.7222 | 0.6000 | 0.5294 | 0.5714 | 0.5000 | 0.7647 | 0.6316 | 0.7143 | 0.7368 |
| Case 10 | 0.3548 | 0.5897 | 0.7736 | 0.8667 | 1 | 1 | 0.8514 | 0.9818 | 1 |
| Sum$ER$ | 5.4546 | 5.4260 | 4.8244 | 5.2689 | 6.2209 | 5.8171 | 5.4492 | 5.6907 | 5.6913 |

本文采用的具有离散操作的PSO算法结合了遗传操作和PSO的信息交换方法, 惯性权重$w$相当于GA中的变异概率, 学习因子$c$相当于交叉概率. 从表1中可以看出, $w=0.1$时误差比$ER$值最小, 获得的非支配解最好, $w=0.8,0.9$时$ER$值最大, 此时PSO算法更接近随机搜索, 性能较差. 对学习因子$c$的测试结果见表2, $c=0.6,0.7$时算法性能较好, 尤其在0.7处最好, 可以看出$c$太小PSO可能会陷入迟钝状态, 太大则高性能的模式被破坏的可能性增大, 因此取$w=0.1$, $c=0.7$. 表3为不同复制尺度$p_s$下10个算例的测试结果, $p_s=2,3,4$时$ER$较小, $p_s$增大则算法的邻域搜索范围越大, 若太大算法则更接近于随机搜索, 且随着$p_s$的加大, 算法的计算时间必然增加, 而搜索性能并无明显改善, 因此取$p_s=2$. 从表4中可以看出, 学习强度$s=0.3,0.4$的时候$ER$值较小, $s$太大或者太小会使算法陷入早熟, 本文取$s=0.3$.

### 4.3 多目标局部搜索环节的有效性测试(The validity test of MLS)

为了考察HPSO算法中各个环节的有效性, 考虑HPSO算法的如下变种:

1) $Y_1$算法. HPSO中不采用BL进行局部搜索.

2) $Y_2$算法. HPSO中不采用SA进行局部搜索.

3) $Y_3$算法. HPSO算法本身, 采用混合初始化方式. MNOR(most number of operation remaining)原则和随机初始化向量$A$, Kacem的方法和随机初始化向量$B$, 各占50%.

采用性能比较参数$ER$和$GD$对Case 1−Case 14分别运行20次过程中所获得的非支配解进行比较. CPUs为$Y_3$算法(本文所提算法HPSO)运行20次的平均运行时间, 单位为秒, 具体数据见表5. 可以看出, 由于BL策略和SA技术的加入, $Y_3$算法求得的非支配解更接近Pareto最优解集, 且几乎对于所有的测试算例, 在求解非支配解的多样性和稳定性等方面的性能得到明显改善, 其良好的分布性能表明MLS策略有助于产生更多的非支配解去填补Pareto前端的空白区域. 从运行时间CPUs来看, $Y_3$算法能在较合理的时间内获得令人满意的结果.

表 5 3种算法性能比较结果

Table 5 The comparative results of three algorithms

| 算例 | $ER$ | | | $GD$ | | | CPUs/s |
|---|---|---|---|---|---|---|---|
| | $Y_1$ | $Y_2$ | $Y_3$ | $Y_1$ | $Y_2$ | $Y_3$ | |
| Case 1 | 0 | 0 | 0 | 0 | 0 | 0 | 14.9190 |
| Case 2 | 1 | 0 | 0 | 1.4142 | 0 | 0 | 17.2725 |
| Case 3 | 1 | 0.7500 | 0 | 8.4853 | 0.4330 | 0 | 14.0485 |
| Case 4 | 1 | 1 | 0 | 15.5563 | 1.5635 | 0 | 35.4292 |
| Case 5 | 1 | 0.5385 | 0.3889 | 36.8782 | 0.5044 | 0.3333 | 39.2230 |
| Case 6 | 1 | 0.2857 | 0.1000 | 16.9706 | 0.2020 | 0.1000 | 38.9145 |
| Case 7 | 0.5 | 0.4194 | 0.0833 | 4.2426 | 0.7634 | 0.2700 | 325.0262 |
| Case 8 | 1 | 0.0769 | 0.0938 | 59.3970 | 4.5690 | 0.0988 | 169.1863 |
| Case 9 | 0 | 0.3571 | 0 | 0 | 0.8391 | 0 | 190.2452 |
| Case 10 | 1 | 0.1739 | 0.2785 | 7.2111 | 0.4900 | 0.1895 | 305.4140 |
| Case 11 | 0 | 0. 3077 | 0 | 0 | 0.5217 | 0 | 206.1891 |
| Case 12 | 0 | 0 | 0 | 0 | 0 | 0 | 544.2784 |
| Case 13 | 1 | 0.5000 | 0.2727 | 33.9779 | 1.3345 | 0.6212 | 628.1406 |
| Case 14 | 1 | 0.1875 | 0.5082 | 82.2375 | 0.4098 | 0.1779 | 615.2437 |

### 4.4 HPSO与其他类型算法的比较(Comparison HPSO with other algorithms)

为了进一步测试HPSO算法的有效性, 将HPSO与PSO+SA[13]和PSO+TS[24]、MOPSO[16]、ESM[7]和P−DABC[9]5种类型算法的求解结果进行比较, PSO+SA、PSO+TS与ESM分别采用加权系数法,

MOPSO和P−DABC采用基于Pareto的方法. 其中ESM给出了Case 1−Case 14的求解结果, P−DABC给出了Case 1−Case 4四种算例的求解结果, 其余3种算法给出了Case 1, Case 3和Case 4的求解结果. 详细结果见表6和表7. 表6−7中的数字依次为完工时间$f_1$、机器总负载$f_2$、单台机器最大负载$f_3$.

对问题Case 1−Case 4的求解结果如表6所示, 对于问题Case 1, HPSO获得了比PSO+SA, PSO+TA和ESM更多的非支配解, 而相比MOPSO和P−DABC, 求得的非支配解虽然不同, 但彼此不受对方所得解支配的非支配解数量相同, 说明HPSO在对Case 1的求解上优于PSO+SA, PSO+TA和ESM, 且不比MOPSO和P−DABC差. 对于Case 2, HPSO与ESM的计算结果相同, 支配了P−DABC求得的非支配解. 对于Case 3, HPSO与MOPSO的计算结果相同, 相比于其他4种算法获得了更多的非支配解. 而对于Case 4, HPSO求得了其他5种算法均未求得的一组非支配解. 可以看出, HPSO在问题Case 1−Case 4的求解质量上略优于其他5种算法. 对问题Case 5−Case 14的求解结果如表7所示, 对于后10个测试算例中的6个, HPSO求得的解能完全支配ESM所得解, 不仅求解质量优于ESM, 在非支配解的数量上也占有很大优势, 而对于其他4个算例, 两种算法获得的解互不支配, 说明HPSO不差于ESM算法. 综上所述, HPSO不仅优于采用加权系数方法的PSO+SA、PSO+TS与ESM算法, 且优于基于Pareto方法的MOPSO和P−DABC算法.

表 6　6种算法关于Case 1−Case 4的求解结果

Table 6　Experimental results of six algorithms about Case 1−Case 4

| 算例 | PSO+SA | PSO+TS | MOPSO | ESM | P−DABC | HPSO |
|---|---|---|---|---|---|---|
| Case 1 | 15,75,12 | 14,77,12 | 14,77,12 | 14,77,12 | 14,77,12 | 14,77,12 |
| | 16,73,13 | 15,75,12 | 15,75,12 | 15,75,12 | 15,75,12 | 15,75,12 |
| | — | — | 17,77,11 | 16,77,11 | 16,73,13 | 16,77,11 |
| | — | — | 16,73,13 | 17,73,13 | — | 17,73,13 |
| | — | — | 16,78,11 | — | — | 16,73,14 |
| Case 2 | — | — | — | 11,62,10 | 11,63,11 | 11,62,10 |
| | — | — | — | 11,61,11 | 12,61,11 | 11,61,11 |
| | — | — | — | 12,60,12 | 12,60,12 | 12,60,12 |
| Case 3 | 7,44,6 | 7,43,6 | 7,42,6 | 7,42,6 | 7,43,5 | 7,42,6 |
| | — | — | 7,43,5 | 8,42,5 | 8,42,5 | 7,43,5 |
| | — | — | 8,42,5 | 8,41,7 | 8,41,7 | 8,42,5 |
| | — | — | 8,41,7 | — | — | 8,41,7 |
| Case 4 | 12,91,11 | 11,93,11 | 11,91,11 | 11,91,11 | 11,93,11 | 11,91,11 |
| | — | — | 12,93,10 | — | 12,91,11 | 11,95,10 |
| | — | — | — | — | — | 12,93,10 |

表 7　HPSO与ESM关于Case 5−Case 14的求解结果

Table 7　Experimental results of HPSO and ESM about Case 5−Case 14

| 算例 | ESM | HPSO |
|---|---|---|
| Case 5 | 42,162,42 | 42,161,42 |
| | — | 42,162,38 |
| | — | 41,164,37 |
| | — | 41,163,38 |
| Case 6 | 28,155,28 | 27,153,27 |
| | — | 28,145,27 |
| | — | 28,151,26 |
| | — | 29,144,29 |
| Case 7 | 204,852,204 | 204,850,204 |
| Case 8 | 68,352,67 | 68,349,66 |
| | — | 68,346,67 |
| Case 9 | 177,702,177 | 173,686,173 |
| | — | 174,683,174 |
| | — | 175,684,173 |
| | — | 176,682,175 |
| Case 10 | 75,431,67 | 74,434,74 |
| | — | 75,418,72 |
| | — | 75,408,75 |
| | — | 75,412,73 |
| | — | 76,435,72 |
| | — | 76,439,71 |
| | — | 76,411,74 |
| Case 11 | 150,717,150 | 141,692,141 |
| | — | 142,691,142 |
| | — | 142,688,142 |
| | — | 144,673,144 |
| | — | 144,680,144 |
| | — | 144,683,143 |
| | — | 145,673,144 |
| | — | 145,683,143 |
| | — | 150,669,150 |
| Case 12 | 523,2524,523 | 523,2524,523 |
| | — | 524,2519,524 |
| Case 13 | 311,2374,299 | 383,2238,381 |
| Case 14 | 227,1989,221 | 293,1932,271 |
| | — | 293,1914,285 |

## 5　结论(Conclusion)

尽管对多目标FJSP的研究取得了一些进展, 但在较高维问题上现有算法还很难取得比较满意的结果, 本文提出的HPSO在求解多目标FJSP问题上具有较优的性能, 特别是多目标局部搜索算法的加入使HPSO算法的探索性能得到明显增强, 说明将PSO算法与局部搜索算法相混合是一种可行的改进途径. 最后以完工时间、机器总负载和单台

机器最大负载为求解目标, 将HPSO与近几年文献中的算法进行比较, 验证了HPSO的有效性和优越性.

## 参考文献(References):


[1] 王万良, 吴启迪. 生产智能算法及其应用 [M]. 北京: 科学出版社, 2007: 12 – 13.
(WANG Wanliang, WU Qidi. *Production Scheduling Intelligent Algorithm and Application* [M]. Beijing: Science Press, 2007: 12 – 13.)

[2] LEI D M. Multi-objective production scheduling: a survey [J]. *International Journal of Advanced Manufacturing Technology*, 2009, 43(9/10): 926 – 938.

[3] 吴秀丽, 孙树栋, 余建军, 等. 多目标柔性作业车间调度优化研究 [J]. 计算机集成制造系统, 2006, 12(5): 731 – 736.
(WU Xiuli, SUN Shudong, YU Jianjun, et al. Research on multi-objective optimization for flexible job shop scheduling [J]. *Computer Integrated Manufacturing Systems*, 2006, 12(5): 731 – 736.)

[4] GAO J, GEN M, SUN L, et al. A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems [J]. *Computers & Industrial Engineering*, 2007, 53(1): 149 – 162.

[5] KACEM I, HAMMADI S. Approach by localization and multi-objective evolutionary optimization for flexible job-shop scheduling problems [J]. *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, 2002, 32(1): 1 – 13.

[6] KACEM I, HAMMADI S, BORNE P. Pareto optimality approach for flexible job shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic [J]. *Mathematics and Computers in Simulation*, 2002, 30(3/5): 245 – 276.

[7] XING L N, CHEN Y W, YANG K W. An efficient search method for multi-objective flexible job shop scheduling problems [J]. *Journal of Intelligent Manufacturing*, 2009, 20(3): 283 – 293.

[8] XING L N, CHEN Y W, YANG K W. Multi-objective flexible job shop schedule: design and evaluation by simulation modeling [J]. *Applied Soft Computing*, 2009, 9(1): 362 – 376.

[9] LI J Q, PAN Q K, GAO K Z. Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems [J]. *International Journal of Advanced Manufacturing Technology*, 2011, 55(9/12): 1159 – 1169.

[10] KENNEDY J, EBERHART R. Particle swarm optimization [C] //*Proceedings of the 4th IEEE International Conference on Neural Networks*. Piscataway: IEEE, 1995: 1942 – 1948.

[11] 万春秋, 王峻, 杨耕, 等. 动态评价粒子群优化及风电场微观选址 [J]. 控制理论与应用, 2011, 28(4): 449 – 456.
(WAN Chunqiu, WANG Jun, YANG Geng, et al. Dynamic evaluation based particle swarm optimization and wind farm micrositing [J]. *Control Theory & Applications*, 2011, 28(4): 449 – 456.)

[12] 薛云灿, 郑东亮, 杨启文. 基于改进离散粒子群算法的炼钢连铸最优浇次计划 [J]. 控制理论与应用, 2010. 27(2): 273 – 277.
XUE Y C, ZHENG D L, YANG Q W, Optimum steel making cast plan with unknown cast number based on the modified discrete particle swarm optimization [J]. *Control Theory & Applications*, 2010, 27(2): 273 – 277.

[13] XIA W J, WU Z M. An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems [J]. *Computers & Industrial Engineering*, 2005, 48(2): 409 – 425.

[14] LI B B, WANG L, LIU B. An effective PSO-based hybrid algorithm for multiobjective permutation flow shop scheduling [J]. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 2008, 38(4): 818 – 831.

[15] LEI D M. A Pareto archive particle swarm optimization for multi-objective job shop scheduling [J]. *Computers & Industrial Engineering*, 2008, 54(4): 960 – 971.

[16] MOSLEHI G, MAHNAM M. A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search [J]. *International Journal of Production Economics*, 2011, 129(1): 14 – 22.

[17] TAVAKKOLI-MOGHADDAM R, AZARKISH M, SADEGHNEJAD-BARKOUSARAIE A. A new hybrid multi-objective Pareto archive PSO algorithm for a bi-objective job shop scheduling problem [J]. *Expert Systems with Applications*, 2011, 38(9): 10812 – 10821.

[18] 张静, 王万良, 徐新黎, 等. 改进粒子群算法求解柔性作业车间批量调度问题 [J]. 控制与决策, 2012, 27(4): 513 – 518.
(ZHANG Jing, WANG Wanliang, XU Xinli, et al. An improved particle swarm algorithm for batch scheduling problem in flexible job shop [J]. *Control and Decision*, 2012, 27(4): 513 – 518.)

[19] BRUCE H W, DAVID J D. *Evoluation and Learning: the Baldwin Effect Reconsidered* [M]. Cambridge MA: Massachusetts Institute of Technology, 2003.

[20] GONG M G, JIAO L C, ZHANG L N. Baldwinian learning in clonal selection algorithm for optimization [J]. *Information Sciences*, 2010, 180(8): 1218 – 1236.

[21] 白俊杰, 龚毅光, 王宁生, 等. 多目标柔性作业车间分批优化调度 [J]. 计算机集成制造系统, 2010, 16(2): 396 – 403.
(BAI Junjie, GONG Yiguang, WANG Ningsheng, et al. Multi-objective flexible job shop scheduling with lot-splitting [J]. *Computer Integrated Manufacturing Systems*, 2010, 16(2): 396 – 403.)

[22] 雷德明, 吴智铭. Pareto档案多目标粒子群优化 [J]. 模式识别与人工智能, 2006, 19(4): 475 – 480.
(LEI Deming, WU Zhiming. Pareto archive multi-objective particle swarm optimization [J]. *Pattern Recognization and Artificial Intelligence*, 2006, 19(4): 475 – 480.)

[23] BRANDIMARTE P. Routing and scheduling in a flexible job shop by tabu search [J]. *Annals of Operations Research*, 1993, 41(3): 157 – 183.

[24] ZHANG G H, SHAO X Y, LI P G, et al. An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem [J]. *Computers & Industrial Engineering*, 2009, 56(4): 1309 – 1318.


## 作者简介:


**张 静** (1986–), 女, 博士研究生, 从事生产调度、智能算法等的研究, E-mail: bay_229@163.com;

**王万良** (1957–), 男, 教授, 博士生导师, 从事CIMS、生产计划与调度、智能自动化等的研究, E-mail: wwl@zjut.edu.cn;

**徐新黎** (1977–), 女, 副教授, 博士, 从事智能计算、生产调度、多Agent系统的研究;

**介 婧** (1972–), 女, 教授, 博士, 从事智能计算、智能控制的研究.