基于遗传算法的作业车间调度优化求解方法

周辉仁[1]，郑丕谔[1]，宗 蕴[2]，张 扬[2]
(1. 天津大学 系统工程研究所，天津 300072；2. 山东大学 能源与动力学院，济南 250100)

摘 要：针对 job shop 调度问题，提出了一种遗传算法编码方法和解码方法。该方法根据问题的特点，采用一种按工序用不同编号进行的染色体编码方案，并采用矩阵解码方法。此编码与调度方案一一对应，并且该编码方案有多种交叉操作算子可用，无须专门设计算子。算例计算结果表明，该算法是有效的，适用于解决 job shop 调度问题，通过比较，该遗传算法优化 job shop 调度操作简单并且收敛速度快。

关键词：作业车间调度；遗传算法；编码方法；矩阵解码；优化
中图分类号：TP301.6；TP18 **文献标志码**：A **文章编号**：1001-3695(2008)10-2991-04

Method for GA-based solution to job shop scheduling optimization

ZHOU Hui-ren[1], ZHENG Pi-e[1], ZONG Yun[2], ZHANG Yang[2]
(1. *Institute of Systems Engineering, Tianjin University, Tianjin* 300072, *China*; 2. *School of Energy & Power Engineering, Shandong University, Jinan* 250100, *China*)

Abstract: This paper proposed a new encoding method and decoding method with the matrix form for a solution to genetic algorithm-based job shop scheduling problem. Based on a specific problem, designed a job activities' number-dependent coding of chromosomes and adopted the matrix decoding. As a result, codes by the new encoding method accord with the job scheduling schemed one-to-one were able to match multiple cross operators without a special design of operators. Result from a case study show that the genetic algorithm with the help of new encoding method presented a powerful ability and was able to effectively solve job shop scheduling problems. To show merits of the new encoding and decoding method, a comparison of different sized job shop scheduling problems in terms of job activity duration, sequence, and scheduling schemes, showed that with the help of the proposed method the genetic algorithm is encouraging, with solutions found through simple operations and fast convergence.

Key words: job shop scheduling; genetic algorithm(GA); encoding method; decoding with the matrix form; optimization

Job shop 调度问题(JSP)是一类典型的生产调度问题，是许多实际生产调度问题的简化模型，具有很强的工程背景，许多实际工程问题均可与之相转换。Job shop 调度问题是非常难解的 NP 问题[1]，因此开发求解 job shop 调度问题的有效算法一直是调度和优化领域的重要课题。目前，研究 job shop 调度问题的方法包括传统运筹学方法、启发式算法、神经网络、模糊理论、Lagrangian 松弛法、禁忌搜索、模拟退火、遗传算法和混合算法等[2~6]。针对 job shop 调度问题用遗传算法求解已有不少研究[7~9]，其编码方案采用比较多的是一种基于工序的表达方法，即将调度编码为工序的序列，给所有同一工件的工序指定相同的符号，然后根据它们在给定染色体中出现的顺序加以解释。为了有效地解决 job shop 调度问题，本文提出了一种按工序用不同编号进行的染色体编码方法，每一个编号表示一个工件的工序，将这些互不相等的编号编码成染色体进行进化计算，并提出矩阵的解码方法，即找到一个最优调度，使完工时间最小。

1 问题描述

典型的 job shop 调度问题可描述为 n 个工件在 m 台机器上加工，事先规定各工件在各台设备上的加工顺序，各工件在各设备上的操作时间已知，要求确定各设备上所有工件的加工次序，使某些加工性能指标最优。已知：a) 工件 $p_i(i=1,2,\cdots,n)$；b) 机器 $m_j(i=1,2,\cdots,m)$；c) 工序 $op_{i,j}(i=1,2,\cdots,n;\ j=1,2,\cdots,m)$ 表示第 i 个工件的第 j 道工序使用的机器号，$op_{i,j}=0$ 表示工件 p_i 在第 j 道工序不加工；d) 时间 $t_{i,j}$ 为第 i 个工件 p_i 使用第 j 台机器的时间，$t_{i,j}=0$ 表示工件 p_i 不使用机器 j。Job shop 调度满足下列约束条件：a) 每个工序利用每台机器不多于一次；b) 每个工件利用每台机器的顺序可以不同；c) 每个工件的工序必须依次加工，后工序不能先于前工序；d) 任何工件没有抢先加工的优先权，应服从生产顺序；e) 工件加工过程中没有新工件加入，也不临时取消工件的加工。

调度目标通常是最小化最大完成时间，即 $T=\min\{\max(C_i)\}$。其中：C_i 为工件 i 的完工时间；$i=1,2,\cdots,n$。

2 遗传算法设计

将遗传算法应用于 job shop 调度问题中的关键是采用有效的编码和解码方式以及适当的交叉、变异操作。遗传算法对种群重复地进行选择、交叉、变异等基本遗传操作，不断产生出比父代更适应环境的新一代种群，直到满足要求条件为止。

收稿日期：2007-11-23；修回日期：2008-03-06

作者简介：周辉仁(1972-)，男，山东高密人，博士，博士后在站，主要研究方向为工业工程、智能优化理论与方法(huirenzhou@126.com)；郑丕谔(1942-)，男，福建莆田人，教授，博导，主要研究方向为系统工程理论及应用；宗蕴(1969-)，男，山东济南人，硕士，主要研究方向为工业工程；张扬(1975-)，男，山东济南人，硕士，主要研究方向为工业工程.

2.1 个体编码

用遗传算法求解 job shop 调度问题时,实际上是将 job shop 调度问题用遗传算法编码加以表示并对各工序的优化排序进行研究,编码问题是设计遗传算法的首要和关键问题。Job shop 调度的遗传算法编码可归纳为两种[1]: a) 直接编码。将各调度作为状态,通过状态演化达到寻优目的,主要包括基于操作(工序)的编码、基于工件的编码、基于工件对关系的编码、基于完成时间的编码、随机键编码等。b) 间接编码。将一组工件的分配处理规则作为状态,算法优化的结果是一组最佳的分配规则序列,再由分配规则的序列构造调度,主要包括基于优先权规则的编码、基于先后表的编码、基于析图的编码和基于机器的编码等。这些编码方法各有特点,有些表征解空间可能存在非法解,有些不能保证全局最优解的存在,有些难以用遗传算法操作实行进化。

在用遗传算法求解 job shop 调度问题时,应用比较多的编码方法是基于操作(工序)的编码。例如文献[7 ~9]采用基于工序的编码方式,个体由所有工序的排序组成,每个基因代表一个工序,同一工件的所有工序采用同一工件序号表示,然后根据它们在个体排序中的顺序决定它们在不同机器上的加工顺序。用这种编码方式进行遗传算法设计对于解决 job shop 调度问题是有效的,但为保证后代的可行性,遗传操作需特别设计。

针对 job shop 调度问题的特点,本文提出一种改进的编码方案,对所有工件的工序用自 1 开始的连续互不相等的整数进行编号,每一个编号代表工件的工序,将这些编号编码成染色体。在同一机器上加工的工序,其编号在同一条染色体中的先后顺序为在该机器上加工的先后顺序;为了避免同一工件的不同工序在操作运算中产生死锁现象,对染色体中的基因在初始种群和操作运算后,对同一工件的不同工序对应的编号采用按整数从小到大重排序进行修正其在染色体中的位置,这样保证了自遗传算法产生的初始种群至交叉、变异后的新种群都为合法解。采用此种编码方案,有效地处理了 job shop 调度问题的约束条件。对于 3 ×3 job shop 调度问题的编码信息和方案如表 1、2 所示。

将以上 3 ×3 job shop 调度问题根据加工顺序进行编号如表 3 所示。

表 1 加工时间

J	M_1	M_2	M_3
J_1	3	3	2
J_2	1	5	3
J_3	3	2	3

表 2 加工顺序

J	M_1	M_2	M_3
J_1	1	2	3
J_2	1	3	2
J_3	2	1	3

表 3 工序编号

J	M_1	M_2	M_3
J_1	(1)	(2)	(3)
J_2	(4)	(6)	(5)
J_3	(8)	(7)	(9)

按编号生成染色体编码,如

1	2	4	3	5	7	6	8	9

为了避免在操作运算后出现非法编码,采用局部排序(sort 函数)修正的方法来重排序,若所得的一条染色体为

2	1	4	6	3	7	5	8	9

编号 1、2、3 分别代表的是工件 1 的第 1、2、3 道工序,从以上染色体看到对于工件 1,先加工第 2 道工序,再加工第 1 道工序,最后加工第 3 道工序,这与工艺约束是不允许的,可对[2 1 3]基因按从小到大重新排序为[1 2 3];同理,对于工件 2 的工序[4 6 5]重新排序为[4 5 6]。通过重排序处理后所得如下染色体:

1	2	4	5	3	7	6	8	9

即为合法的染色体,从而避免了非法染色体的出现。

2.2 群体规模选择

合适的群体规模对遗传算法的收敛具有重要意义。群体太小难以求得满意的结果,群体太大则计算复杂。根据经验,群体规模一般取 10 ~160。

2.3 适值函数

遗传算法在进行选择操作时会出现欺骗问题[9]:在遗传进化初期,通常会产生一些超常个体,若按照比例选择法,这些异常个体因竞争力太突出而控制了选择过程,影响算法的全局优化性能;在遗传进化后期,即算法接近收敛时,由于种群中个体适应度差异较小时,继续优化的潜能降低,可能获得某个局部最优解。适值函数设计不当有可能造成这种问题的出现。由于优化目标为最小化最大完成时间,令目标函数作指数变换得到适值函数:

$$f = \alpha \exp(-\beta \times \max(C_i)) \tag{1}$$

其中: α、β为正实数。

2.4 选择

选择是用来确定重组或交叉个体,以及备选个体将产生多少个子代个体。选择的第一步是计算适值,采用按比例的适应度分配,是利用比例于各个个体适应度的概率决定其子孙的遗留可能性。若有 M 个个体,其中某个个体 i,其适值为 f_i,则其被选择的概率表示为

$$P_i = f_i / \sum_{k=1}^{M} f_k \tag{2}$$

然后对各个染色体计算其累积概率,如第 k 个个体的累积概率为:

$$q_k = \sum_{i=1}^{k} p_i \tag{3}$$

第二步用轮盘赌选择法进行选择。为了选择交配个体,需要进行多轮选择,每一轮产生一个[0,1]均匀随机数,将该随机数作为选择指针来确定备选个体。

2.5 交叉与变异

交叉在遗传操作中起核心作用,交叉概率较大可增强遗传算法开辟新搜索空间的能力,但性能好的基因串遭到破坏的可能性较大,算法收敛速度降低且不稳定;若交叉概率较小,则遗传算法搜索可能陷入迟钝状态。对于基因串交叉可以采用基于路径表示部分匹配(PMX)交叉操作、顺序交叉(ordered crossover, OX)、循环交叉(cycle crossover, CX)等[10,11]。

部分匹配交叉是 Goldberg 等人于 1985 年针对 TSP 提出的基于路径表示的交叉操作,部分匹配交叉操作要求随机选取两个交叉点,以便确定一个匹配段,根据两个父个体中两个交叉点之间的中间段给出的影射关系生成两个子个体。

顺序交叉是 Davis 等人于 1985 年针对 TSP 提出的基于路径表示的交叉操作,该操作能保留排列并融合不同排列的有序结构单元。两个父体交叉时,通过选择父个体 1 的一部分,保存父体 2 中代码的相对顺序生成子个体。

循环交叉是 Oliver 等人针对 TSP 提出的交叉操作,循环交叉操作中子个体中的代码顺序根据任一父个体产生。

变异在遗传操作中属于辅助性的搜索操作,主要目的是维持群体的多样性。较低的变异概率可以防止群体中重要的单

一基因丢失,但降低了遗传算法开辟新搜索空间的能力;较高的变异概率将使遗传操作趋于纯粹的随机搜索,降低了算法的收敛速度和稳定性。这里对于基因串采用交换变异,即交换两个随机位置上的基因。

2.6　解码

在 job shop 调度问题中,解码是一项非常重要的工作。在车间调度中,针对解码方法研究得相对较少,现大都采用 AOV、AOE 网络图进行解码,相对较繁琐。这里提出一种矩阵形式的解码方法。根据以上 3 ×3 job shop 调度问题所生成的一条染色体如下:

1	2	4	5	3	7	6	8	9

根据表 1、2 生成一个矩阵如下:

$$TMJ=\begin{bmatrix}3&0&0&1&0&0&0&3&0\\0&3&0&0&0&5&2&0&0\\0&0&2&0&3&0&0&0&3\end{bmatrix}$$

其中:第 1 ~3 行分别表示机器 1 ~2 ~3;第 1 ~9 列分别表示表 3 中编号 1 ~9 对应的工序;矩阵中不为 0 的元素表示对应的工序在对应机器上的加工时间,元素为 0 表示对应的工序不在对应的机器上加工。

在矩阵 *TMJ* 中,只体现出工序的机器约束,体现不出同一工件的工序加工的先后顺序约束,这要结合染色体中的基因来完成。由于对染色体基因的修正处理不会出现非法调度,第一个加工的工序肯定是其中一个工件的第一个工序,那个工序对应矩阵中 *TMJ* 的数值即为该工序的最早完成时间,将其映射到一个行数和列数与矩阵 *TMJ* 相等的矩阵 *TT* 中,映射的位置相同。矩阵 *TT* 中元素不为 0 表示对应的工序在其对应机器上的最早完成时间。具体步骤如下:

a) 建立一个与矩阵 *TMJ* 行数和列数相等的 0 矩阵;

b) 将染色体中第一个加工的工序编号在矩阵 *TMJ* 中对应的加工时间以相同位置影射到矩阵 *TT* 中,本例为编号 1,所得矩阵 *TT* 如下所示:

$$TT=\begin{bmatrix}3&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\end{bmatrix}$$

c) 染色体中第二个加工的工序编号本例为 2,因编号 2 为工件 1 的第二个工序,有紧前工艺约束(编号 1),编号 2 在矩阵 *TMJ* 对应不为 0 的元素在第 2 行(即在机器 2 上加工),编号 1 位于编号 2 的前一列,由上式所示的矩阵 *TT* 中可以看出 J_{11}的最早完成时间为 3,矩阵 *TT* 第 2 行的最大数值为 0,考虑工艺约束和机器加工顺序,将矩阵 *TMJ* 第 2 行第 2 列的数值加上 max(0, 3)即为编号 2 的最早完成时间,将其映射到矩阵 *TT* 第 2 行第 2 列,便得到新的矩阵 *TT*:

$$TT=\begin{bmatrix}3&0&0&0&0&0&0&0&0\\0&6&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\end{bmatrix}$$

d) 染色体中第三个加工的工序本例为编号 4,由于编号 4 为工件 2 的第 1 个工序,没有紧前工艺约束,只考虑机器加工顺序,编号 4 在矩阵 *TMJ* 对应不为 0 的元素在第 1 行(即在机器 1 上加工),上式矩阵 *TT* 第 1 行的最大数值为 3,考虑机器加工顺序,将矩阵 *TMJ* 第 1 行第 4 列的数值加上 3 即为编号 4 的最早完成时间,将其映射到矩阵 *TT* 第 1 行第 4 列,便得到新的矩阵 *TT*:

$$TT=\begin{bmatrix}3&0&0&4&0&0&0&0&0\\0&6&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\end{bmatrix}$$

其余的依此类推,直到进行到第 9 个加工的工序,求得最终得矩阵 *TT*:

$$TT=\begin{bmatrix}3&0&0&4&0&0&0&11&0\\0&6&0&0&0&13&8&0&0\\0&0&9&0&7&0&0&0&14\end{bmatrix}$$

针对此调度和此条染色体,求矩阵 *TT* 的 MATLAB 程序如下:

```
TT = zeros(3,9);
O = [1,2,4,5,3,7,6,8,9]; % 染色体中基因
TT(:, O(1)) = TMJ(:, O(1)); % 第一个被加工的工序
for i = 2:9;
[a, b] = find(TMJ(:, O(i)) ~= 0);
if O(i) == (1|4|6); % 每个工件的第一个工序编号
TT(a, O(i)) = TMJ(a, O(i)) + max(TT(a,:));
else
TT(a, O(i)) = TMJ(a, O(i)) + max([max(TT(a,:)), … sum(TT(:, O(i) - 1))]);
end
end
```

由于矩阵中不为 0 的数值表示的是每个工序的最早完成时间,矩阵中最大的数即为该条染色体所表示的 job shop 调度问题的最大完成时间:$\max(C_i)=14$。

由于 job shop 调度问题是求解最小化最大完成时间,将所有染色体按上述步骤解码后所求得数值的最小数值为该次迭代的最小化的最大完成时间,染色体通过不断选择、交叉和变异操作,最终求得最优调度。

3　实例仿真和比较

3.1　实例 1

为了便于比较采用文献[8]中的实例,设有 8 个工件(J_1, J_2, …, J_8),5 台机器(M_1, M_2, …, M_5)的生产线作业调度问题,加工时间及加工顺序分别如表 4、5 所示。

表 4　8 ×5 job shop 调度问题加工时间表

M	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
M_1	10	18	7.6	17	13.6	0	9	6
M_2	9.8	15	1.9	15	10	5.4	9.4	9
M_3	11.2	18.5	13	5.6	8	12	34	0
M_4	13	24	20.5	16.4	14.9	14.8	13.6	19
M_5	20.8	22	11	30	0	24	20	57

表 5　8 ×5 job shop 调度问题加工顺序表

M	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
M_1	1	2	1	3	1	5	3	4
M_2	2	1	3	4	2	1	1	1
M_3	3	3	2	5	4	2	2	5
M_4	4	4	5	2	3	4	5	2
M_5	5	5	4	1	5	3	4	3

表 6　Job shop 调度方案

M	T	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
M_1	开始	13.6	63.8	23.6	81.8	0	173.2	52.4	98.8
	结束	23.6	81.8	31.2	98.8	13.6	173.2	61.4	104.8
M_2	开始	23.6	48.8	77.4	98.8	33.4	43.4	9	0
	结束	33.4	63.8	79.3	113.8	43.4	48.8	18.4	9
M_2	开始	52.4	81.8	64.4	113.8	100.3	52.4	18.4	104.8
	结束	63.6	100.3	77.4	119.4	108.3	64.4	52.4	104.8
M_2	开始	63.6	100.3	137.9	29	45.4	158.4	124.3	9
	结束	76.6	124.3	158.4	45.4	60.3	173.2	137.9	29
M_2	开始	164	142	107	0	108.3	118	87	30
	结束	184.8	164	118	30	108.3	142	107	87

根据本文提出的编码用遗传算法在计算机上进行仿真实验。这里交叉概率取 0.9,并分别采用部分匹配(PMX) 交叉操作、循环交叉(CX) 操作,变异概率取 0.5,种群规模均取 50,最大迭代次数均为 20,得出的调度方案如表 6 所示,进化值曲线如图 1 所示,10 次随机仿真的最终结果如表 7 所示。

表 7 调度计算结果与比较

指标(最小完成时间)	最优指标	平均指标	种群规模	最大进化次数
直接用工序编号编码并用拒绝策略	241.5	242.2	100	100
文献[8]的编码方案	241.5	241.7	100	100
本文算法(PMX)	184.8	184.8	50	20
本文算法(CX)	184.8	184.8	50	20

3.2 实例 2

在文献[9]分别用优先列表编码方式的遗传算法和基于工序编码方式的遗传算法,并采用自适应遗传算子对一标准的 job shop 调度问题进行了仿真,具体数据参见该文献。

文献[9]种群规模取 100,迭代次数取 1 000,所求得最优解为55。本文算法交叉概率取 0.9,采用部分匹配(PMX) 交叉操作、交换变异概率取 0.5,种群规模取 50,迭代次数为 30,得出的最优解为 55,进化值曲线如图 2 所示。

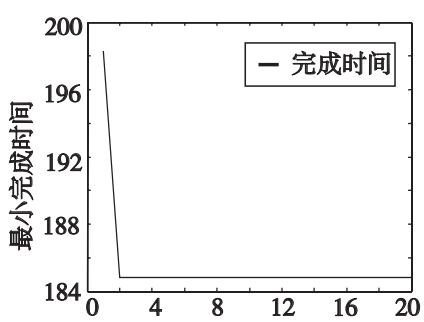

图1 进化值曲线

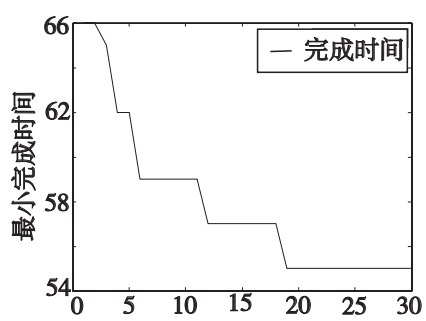

图2 进化值曲线

3.3 比较

在实例 1 中,本文方法以较小的种群规模和较少的迭代次数求得的最优调度明显优于文献[8]中所求得的结果;在实例 2 中,本文方法比文献[9]用较小的种群规模和较少的迭代次数求得最优解。通过比较,本文中的遗传算法在收敛速度和稳定性方面的效果是比较好的,明显优于文献[8,9]的算法。

4 结束语

从实例仿真可以看出,本文提出的遗传算法基因编码和解码方法简洁,能清楚地反映出调度方案关系,能应用多种交叉算子进行操作;收敛速度快,能有效地解决大规模 job shop 调度问题。通过比较,解的质量是比较满意的,有广泛实际应用的良好前景。

参考文献:

[1] 王凌. 车间调度及其遗传算法[M]. 北京: 清华大学出版社, 2003: 56-101.

[2] 王凌, 郑大钟. 基于遗传算法的 job shop 调度问题研究进展[J]. 控制与决策, 2001, 16(增): 641-645.

[3] FOO S Y, TAKEFUJI Y. Inter-linear programming neural networks for job shop scheduling[C] //Proc of IEEE IJCNN. San Diago: [s. n.], 1988: 341-348.

[4] WU C G, LI X P, ZHOU P. A job shop oriented virus genetic algorithm[C] // Proc of the 1st World Congress on Intelligent Control and Automation. 2004: 2132-2136.

[5] 赵良辉, 邓飞其. 解决 job shop 调度问题的模拟退火算法改进[J]. 计算机工程, 2006, 32(21): 38-40.

[6] 曹承煜, 李人厚, 樊健. 车间调度算法的研究和开发[J]. 控制理论与应用, 2000, 17(1): 31-34.

[7] 王万良, 吴启迪, 宋毅. 求解作业车间调度问题的改进自适应遗传算法[J]. 系统工程理论与实践, 2004, 30(2): 58-62.

[8] 柳林. 基于遗传算法的 job shop 调度问题求解[J]. 计算机应用, 2006, 26(7): 1694-1696.

[9] 庄新村, 卢宇灏, 李从心. 基于遗传算法的车间调度问题[J]. 计算机工程, 2006, 32(1): 193-194.

[10] 王小平, 曹立明. 遗传算法—理论、应用与软件实现[M]. 西安: 西安交通大学出版社, 2002: 28-38.

[11] 周明, 孙树栋. 遗传算法原理及应用[M]. 北京: 国防工业出版社, 1999: 146-254.

(上接第 2990 页)

[2] 周兵, 沈均毅, 彭勤科. 集群环境下的并行聚类算法[J]. 计算机工程, 2004, 30(4): 4-7.

[3] GANTI V, GEHRKE J, REMAKRISHNAN R. CACTUS-clustering categorical data using summaries[C] //Proc of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM, 1999: 73-83.

[4] WANG W, YANG J, MUNTZ R. STING: a statistical information grid approach to spatial data mining[C] //Proc of the 23th International Confe-rence on Very Large Data Bases. Athens: Morgan Kaufmann Publishers, 1997: 186-195.

[5] CHENG C, FU A, ZHANG Y. Entropy-based subspace clustering for mining numerical data[C] // Proc of the 5th ACM SIGKDD Conference. New Work: ACM, 1999: 84-93.

[6] WARSCHKO T M, BLUM J M, TICHY W F. ParaStation: efficient parallel computing by clustering workstations: design and evaluation [J]. Journal of Systems Architecture, 1998, 44(3): 241-260.

[7] BOUTSINAS B, GNARDELLIS. On distributing the clustering process[J]. Pattern Recognition, 2002, 23(8): 999-1008.

[8] LAMEHAMEDI H, BENSAID A D, KEBBAL E G, *et al*. Adaptive programming: application to a semi-supervised point prototype clustering algorithm[C] //Proc of International Conference on Parallel and Distributed Processing Techiques. Nevada: CSREA Press, 1999: 2753-2759.

[9] 陆建江, 徐宝文. 区间数据的并行模糊聚类算法[J]. 东南大学学报: 自然科学版, 2003, 33(4): 406-409.

[10] SHI Y, KENNEDY J. Swarm intelligence[C] //Proc of Swarm Intelligence Symposium. San Mateo: Morgan Kaufman Publishers, 2001: 50-60.

[11] CLERC M, KENNEDY J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space[J]. IEEE Trans Evol Comput, 2002, 6(1): 58-73.

[12] Particle Swarm Central [EB/OL]. (2006-11-20) [2007-10-12]. http://www.particleswarm.info/papers.html.

[13] TANDON V, EL-MOUNAYRI H, KISHAWY H. NC and milling optimization using evolutionary computation[J]. International Journal of Mach Tools Manuf, 2002, 42(5): 595-605.

[14] COCKSHOTT A R, HARTMAN B E. Improving the fermentation medium for echinocandin B production part II: particle swarm optimization[J]. Process Biochem, 2001, 36(9): 661-669.

[15] ABIDO M A. Optimal power flow using particle swarm optimization [J]. International Journal of Electrical Power and Energy Systems, 2002, 24(7): 563-571.