双目标无等待流水线调度的加权混合算法*)

谈 超 李小平
(东南大学计算机科学与工程学院 南京 210096)

摘 要 针对最小化"总完工时间"和"最大完工时间"的双目标无等待流水线作业调度问题提出了一种粒子群加权混合优化算法,通过随机加权的方式将其转换成单目标问题,并应用基于升序排列的 ROV (ranked-order-value)编码规则,将粒子群优化算法应用于无等待流水线作业调度问题。为了提高算法的性能,增强算法的搜索能力,提出的混合算法应用了 NEH 方法构造初始种群,在一个较好的初始值上进行粒子群优化,为防止种群陷入局部最优造成早熟,在粒子群每次迭代之后对全局最优解加入扰动并进行变邻域搜索。仿真实验结果表明该混合调度算法具有良好的性能。

关键词 双目标无等待流水线作业调度,粒子群优化,ROV 编码,目标加权,变邻域搜索

Weighed Hybrid Heuristics for Bi-objective No-wait Flow Shops Problem

TAN Chao LI Xiao-ping
(School of Computer Science and Engineering, Southeast University, Nanjing 210096, China)

Abstract A weighed hybrid particle swarm optimization was introduced for bi-objective no-wait flow shops to minimize makespan and total flow-time problem, which makes the problem into single objective by random weighted and apply PSO on no-wait flow shops problem with the rule based on ranked-order-value. In order to improve the optimization of PSO and increase the algorithm's exploring ability, we can improve the performances through constructing the initial swarm by using NEH and executing PSO on a better initial value. After each step of PSO, we interrupted the global best value and execute the Variable Neighborhood Search for avoiding the swarm fall into the local best value. Passing through the testing experiments, this hybrid scheduling algorithms precedes other scheduling algorithms.

Keywords Bi-objective no-wait flow shops, Discrete particle swarm optimization, Objective weighted, Variable neighborhood search

无等待调度是一类重要的约束流水线调度问题,它要求任务的加工从开始到结束必须连续进行,任务在机器上加工和传送都不能出现等待,即任务在给定机器上的开始时间必须延迟以满足该工序的完成时间与下一个工序在下一台机器上的开始时间相符合。

这类调度广泛存在于冶金、化工和食品加工等行业,通常被模型化为无等待调度。

文献[1]的研究表明,离散粒子群优化(Discrete PSO, DPSO)算法求解以生产周期为目标的流水线调度问题更具优越性。本文进一步研究了文献[2]提出的 DPSO 算法,并结合双目标无等待流水作业调度问题特性,提出了一种混合调度算法来提高其有效性。

1 双目标无等待流水调度问题的描述

无等待流水线调度问题(no-wait flow shop, NWFS)可描述为:给定数量的机床和工件,每个工件在各个机床上的加工顺序相同,并且在某一时刻只能够在一台机床上加工,另外在某一时刻每台机床上只能加工一个工件,同一工件相邻的两工序之间没有等待时间。如何安排生产,使生产周期最优就构成了无等待流水线调度问题。

所有工件的总完工时间(total flow-time, TFT)和最大完工时间(makespan, C_{max})是流水线生产调度问题中两个常用的评价指标,TFT 最小可促使资源稳定有效的利用及任务的快速周转,C_{max}最小能减小总的生产时间,二者同时最小化可以极大地减少生产成本,提高效益。当优化目标为最小化C_{max}和 TFT 的加权组合时,该双目标函数可以等价表示为:

$$F(x)=\alpha \cdot C_{max}+\beta \cdot TFT \tag{1}$$

其中,C_{max}代表一个调度的最大完工时间,TFT 代表总完工时间,α,β代表随机权重。

2 PSO 算法原理

粒子群优化算法(particle swarm optimization, PSO)是 Kennedy 和 Eberhart 于 1995 年发明的一种优化算法[3],其源于对鸟群捕食行为的模拟。PSO 是一种演化算法,同遗传算法(genetic algorithm, GA)类似,是基于迭代的优化工具,但不具有选择和变异等操作。粒子群优化算法容易理解、易于实现、鲁棒性好,有研究表明 PSO 比 GA 具有更高的效率。所以近些年来成为求解优化问题一个很有意义的研究方向。

PSO 求解优化问题时,每个粒子初始化为解空间的一个候选解,求解的过程就是不断更新自己的位置和速度。和其它进化算法类似,每次迭代的过程是基于上一代基础在解空间中搜索。每个粒子通过学习两个极值来更新其速度和位

*)基金项目:国家自然科学基金项目(60504029)。**谈 超** 硕士研究生,研究方向为数据库;**李小平** 博士生导师,研究方向为调度理论与算法研究。

置，第一个是每个粒子从初始到当前迭代次数搜索产生的最优解，即个体极值 $pbest=(pbest_1, pbest_2, \cdots, pbest_i)$；第二个是粒子种群目前的最优解 gbest，又称为全局极值。

PSO 算法描述如下：

设：$X_i(X_{i1}, X_{i2}, \cdots, X_{in})$ 为微粒 i 的当前位置，$V_i(V_{i1}, V_{i2}, \cdots, V_{in})$ 为微粒 i 的当前飞行速度，$P_i(P_{i1}, P_{i2}, \cdots, P_{in})$ 为微粒 i 所经历的最好位置。

粒子飞行（迭代）中根据以下公式来更新其速度和位置：

$$v_i^{k+1}=w\cdot v_i^k+c_1.\,rand_1(pB_i^k-x_i^k)+c_2.\,rand_2(gB^k-x_i^k) \tag{2}$$

$$X_i^{k+1}=X_i^k+V_i^k \tag{3}$$

其中，w 表示惯性系数，c_1 表示认知系数，c_2 表示社会系数，$rand_1$ 和 $rand_2$ 为随机数。

式(2)分为 3 部分，决定了粒子的移动方向。首先是原来速度 v；第 2 个与自身最优值($pbest, pB$)比较，学习自身经验；第 3 个是向群体目前最优值($gbest, gB$)学习，属于社会行为，是群体合作行为的体现。

3 粒子群离散化

由于 PSO 算法中微粒的位置为连续值矢量，无法实现流水线调度中工件排序的更新，因此构造从微粒位置矢量到流水线工件排序的映射是应用 PSO 算法解决 NWFS 问题的首要问题，为解决该问题，引入了一种基于升序排列的 ROV[4] (ranked-order-value)编码规则，将粒子群优化算法应用于 NWFS 问题。

ROV 规则：对于一个微粒的位置矢量，首先将值最小的分量位置赋予 ROV 值 1，其次将值次小的分量位置赋予 ROV 值 2，依此类推，直到所有分量位置都赋予唯一的 ROV 值。

由于微粒位置矢量的各个位置分量的值有大小次序关系，ROV 规则就是利用这种次序关系结合随机编码，将微粒的连续位置转换成离散的排序即机器上各工件的加工顺序，从而计算工件（微粒）所对应调度的目标函数值。

3.1 位置矢量编码

对于无等待流水线调度问题，最直接的编码方法就是采用基于工件序列的自然数编码，用位置矢量的一维表示一个工件，粒子本身表示所有工件的排列，即一个调度。

对于 n 个工件 m 台机床的调度问题，若位置矢量的分量用工序表示，则从第 1 维分量到第 n 维分量就形成了一个工序序列。位置矢量中，同一工件的所有工序采用相同的符号，而同一工件的不同工序可根据其在工序序列中的出现次序区分。假定有 6 个工件的调度问题，设其位置矢量为[1.8, 1.7, 0.6, 3.7, 2.8, 0.9]，根据粒子各个位置矢量大小分别给予 ROV 赋值：4,3,1,6,5,2，从而得到工件的加工次序即调度顺序。表 1 为粒子位置矢量与调度之间的对应关系。

表 1 位置矢量与对应的 ROV 值和工件序列之间的对应关系

粒子维数	1	2	3	4	5	6
位置矢量	1.8	1.7	0.6	3.7	2.8	0.9
ROV 值	4	3	1	6	5	2
工件序列	4	3	1	6	5	2

为了用粒子群算法解决调度问题，调度顺序通常采用位置矢量的模型，每个粒子的矢量代表任务的排列顺序，该元素位置的排列就是一个调度顺序。具体实现中，兼顾任务间的偏序关系不断调整。

3.2 离散粒子群优化

粒子群算法的实质在于每个粒子根据自己和同伴的飞行经验不断调整位置和速度，从而向最优位置飞行。粒子的新位置是粒子的速度、个体极值和全局极值相互作用的结果。结合编码方法，定义位置和速度的更新操作，将粒子群算法离散化以适用于无等待流水线调度问题的求解，过程如下：

1) 微粒的位置：

将所有工件(n 个)的排列作为粒子的位置矢量，即 $X=(X_1, X_2, \cdots, X_n)$；

2) 微粒的速度：

速度将用于改变微粒的位置，因此可定义为对微粒位置的一种变换；

3) 微粒速度和位置的更新公式：

$$V=w.\,v+r.\,c_1(Xpbest-X)+r.\,c_2(Xgbest-X) \tag{4}$$

$$X=X+V \tag{5}$$

其中，$Xpbest$ 为微粒当前的个体最佳位置，$Xgbest$ 为群体的最佳位置，r 为区间[0,1]上的随机数，w 为惯性权重，c_1 是认知系数，调节向 $pbest$ 的飞行；c_2 是社会系数，调节向 $gbest$ 的飞行。

粒子位置的移动方式如下：

①若 $r<w$，粒子进行插入变异，改变工件的排列顺序；

②若 $r<c1$，粒子根据自身的 $pbest$ 进行交叉变异，实现自身学习；

③若 $r<c2$，粒子根据全局的 $gbest$ 进行交叉变异，实现群体学习。

3.3 离散离子群优化算法框架

算法基本步骤为：

1) 确定 w, c_1, c_2 及种群个数 N；

2) 初始化粒子群，得到 $pbest, gbest$ 的初始值；

3) 粒子的更新：根据式(4)和(5)对每一个粒子的速度和位置进行更新；

4) 计算适应度值（目标函数），更新 $pbest, gbest$；

5) 判断终止条件是否满足，满足停止；否则转 3)。

4 双目标优化

在多于一个目标的优化问题中，目标之间一般都是互相冲突的，双目标优化问题中的两个目标几乎不可能在同一解上同时达到最优，因此双目标优化问题的解往往不是单一的，而是一组解的集合。本文解决的双目标 NWFS 问题中考虑的是两个基于加工完成时间的性能指标，即总完成时间 TFT 和最大完工时间 C_{max}，为了解决这个问题，采用了一种随机加权累加函数的算法。

4.1 随机加权累加函数

假定某一决策过程中，需要同时考察多个目标，且要求所有目标函数在满足约束的条件下越小越好，优化问题表示如下：

$$\min_{k\in R} F(X)=(f_1(x), f_2(x), \cdots, f_k(x))^T \tag{6}$$

$$X=(x_1, x_2, \cdots x_n)^T, X\subset R$$

其中 $F(X)$ 为优化目标向量，X 为解空间。

将多个目标整合为单一目标，是处理多目标优化问题的有效策略，即将原问题转化为单目标优化问题来求解。加权线性累加函数的数学描述如下：

$$F(x)=\sum_{i=1}^{k}w_i f_i(x) \tag{7}$$

其中：$w_i(i=1,\cdots,k)$是目标 $f_i(x)$的非负权重系数，且满足：$\sum_{i=1}^{k}w_i=1$

显然，在算法的优化过程中，每一组权向量将决定目标空间中的一个搜索方向。为了获得尽可能多的非劣解，权值按如下方式随机确定：

$$\alpha_i=random_i/\sum_{j=1}^{N}random_j \tag{8}$$

其中，$random$ 为 0 到 1 之间均匀分布的一个随机数。

4.2 双目标优化处理

4.2.1 目标函数的计算

任务 J_i 与 J_j 相邻时在第 m 台机器上的完工时间距离为 D_{ij}，任务之间的距离是独立的，即与其他任务的时间参数无关，只与这两个任务的加工时间有关，故可以用一个非对称矩阵 D 表示所有任务之间的距离。因为一个调度就是任务集的一个全排列，所以一个调度的最大完工时间可以等价转换为 $C_{\max}=\sum_{i=0}^{n-1}D_{[i],[i+1]}$；设一个调度中的第 i 个任务 J_i 的完工时间为 C_i，其总完工时间 TFT 可表示为相邻任务间距离的加权和，$TFT=\sum_{i=0}^{n-1}(n-i)D_{[i],[i+1]}$；对于二者线性组合仍然为相邻任务间距离的一个加权和(式 10)。

任务 J_i 与 J_j 相邻时在第 m 台机器上的完工时间距离 D_{ij}，可用如下公式[5]得出：

$$D_{i,j}=E_{j,m}-E_{i,m}=\max_{k=1,\cdots,m}\{t_{i,k}+\sum_{p=k}^{m}(t_{j,p}-t_{i,p})\} \tag{9}$$

则 $C_{\max}$和 TFT 的双目标函数为：

$$F(x)=\alpha\cdot C_{\max}+\beta\cdot TFT=\sum_{i=0}^{n-1}[\alpha+(n-i)\beta]D_{[i],[i+1]} \tag{10}$$

4.2.2 随机加权函数

针对双目标(总完成时间 TFT 和最大完工时间 $C_{\max}$)优化问题，在构造双目标函数 F(x)时采用了随机加权累加函数。故有：

$$\begin{aligned}F(x)&=\frac{random_1}{\sum_{j=1}^{2}random_j}C_{\max}+\frac{random_2}{\sum_{j=1}^{2}random_j}F\\&=\sum_{i=0}^{n-1}\left[\frac{random_1}{\sum_{j=1}^{2}random_j}+(n-i)\frac{random_2}{\sum_{j=1}^{2}random_j}\right]D_{[i],[i+1]}\end{aligned} \tag{11}$$

5 双目标无等待流水线调度的混合算法

5.1 构造初始粒子群

为了使初始种群具备一定的质量和分散度，本混合调度算法中采用了基于 NEH 插入[6]的初始化方法，构造有效的初始粒子群。

NEH 插入算法流程：

步骤 1 给定所有工件的一个 ROV 排序 S；

步骤 2 令 $k=1$，取 S 中的前 2 个工件，对 2 种可能的加工次序进行评价，选最大完成时间较小的序列作为当前序列；

步骤 3 令 $k=k+1$，依次将第 k 个工件插入到当前序列的 k 个可能位置，从中选最大完成时间小的序列作为当前序列；

步骤 4 重复步骤 3，直到 S 中所有工件均得到新的排序。

5.2 对全局最优值 gbest 加扰动

PSO 中信息传递是单向的。全局最优值 *gbest* 将信息传给其他粒子，带动其他粒子在较好的区域搜索，整个粒子群向最优解的方向进化。因此 *gbest* 对 PSO 的优化性能有重大影响。为提高 PSO 的优化性，增强算法对 *gbest* 的探索能力，可通过对 *gbest* 执行变邻域搜索(variable neighborhood search，VNS)[7]算法，即在粒子群每次迭代之后对 *gbest* 执行 VNS 搜索来提高性能。为了加强算法跳出局部最优的能力，本文算法还采取了对 *gbest* 加随机扰动的措施。

5.3 变邻域搜索(VNS)

考察 PSO 的进化过程可知，PSO 初期收敛速度非常快，经若干次迭代后，粒子失去了多样性，趋向同一化，收敛速度明显变慢。因大部分粒子的个体极值等于全局极值，算法长期徘徊在若干旧状态上，造成早熟。研究表明，PSO 算法尽管全局搜索能力较好，但其局部搜索能力较弱，而 VNS 变邻域搜索具备较强的局部探索能力。VNS 算法通过系统性地改变邻域搜索中的邻域结构，使算法获得全局最优解的概率高于采用单一邻域结构的搜索。

通过搜索不同的邻域来提高求解质量，采用交换邻域结构(SWAP)和插入邻域结构(INSERT)使算法具有跳出局部最优的能力。

VNS 算法基本流程如下：

1) 确定邻域结构 $N_k(k=1,2,\cdots k_{\max})$和初始解 S_0，令最优解 gbest$=S_0$；

2) 满足收敛准则输出 gbest，否则转 3)；

3) 对 S_0 进行扰动，随机交换 S_0 的两个不同工序得到 span；

4) k=0；

5) do {

```
    S1=S0;
    span1=span;
    if (k==0) swap;
    if (k==1) insert;
    if (span1<span){
      S1=S0; span=span1; k=0;}
    else k++;
} (until k>1)
```

其中，swap 操作为：在工件序列中随机选择不同的两个位置 x，y，把 x 位置上与位置 y 上的工序进行交换，记为 $swap(x,y)$。

insert 操作为：在工件序列中随机选择不同的两个位置 x，y，把 x 位置上的工序插入位置 y，其它工序依次移动，记为 $insert(x,y)$。*Insert* 操作后，每次调度完毕要重新调整位置元素的顺序关系相当于一次变异，根据优先级对粒子的位置和速度进行变异。

5.4 双目标无等待流水线加权混合算法流程

综上所述，本文提出一种新的混合调度算法，解决双目标无等待流水线调度问题，首先用 ROV 规则将 PSO 算法实行工序编码，使其离散化应用于流水线生产调度；再应用随机加权函数实现双目标向单目标的转换；接下来在算法过程中从所有 $pbest_i^k$ 中选出最好的个体，令 $gbest^k=pbest_i^k$ 中最好解，并对 $gbest^k$ 加上随机扰动后执行变邻域搜索(VNS)。

混合调度算法基本流程如下：

1) 工序 ROV 编码离散化；

2) 双目标加权，权重计算；

3)初始化粒子群(每个粒子的解向量及其目标函数值)，

应用 NEH 插入算法，得到初始解；

4) Do{

飞行（更新粒子的位置和速度）：

①通过自身学习和社会学习更新工件排序；

②计算目标函数值。

更新 $pbest_i$；

找出并更新 gbest；

对 gbest 加扰动；

变邻域搜索(VNS)；

} Until N 次迭代，或迭代 L 次后 gbest 不再改变；

5) 输出工件调度及目标函数值。

5.5 参数设计

1) 群体规模 N

实验表明，随着种群规模的增加，搜索质量有改善的趋势，但仅在较小的范围内变化。这表明种群规模对算法的搜索造成的影响不太大。鉴于粒子群算法对种群大小不十分敏感，仿真中设种群规模 N 等于 job 个数。

2) 惯性系数 w

较大的 w 可以加强 PSO 的全局搜索能力，而较小的 w 则有利于局部搜索。因此，可在算法开始搜索早期，w 取较大的值，使得 PSO 探索较大的区域，较快定位最优解的大致位置。随着搜索过程的深入，w 逐渐减小，开始精细搜索。在本文提出的混合算法仿真中取 $w=0.2$，效果较好。

3) 认知系数 c_1 和社会系数 c_2

c_1 反映了微粒飞行过程中所记忆的最好位置对微粒飞行速度的影响，称为认知系数；c_2 则反映了整个微粒群所记忆的最好位置对微粒飞行速度的影响，称为社会系数。c_1 和 c_2 分别调节粒子向个体极值和全局极值的飞行，合适的 c_1 和 c_2 可以加快收敛且不易陷入局部极值。本文算法在仿真实验中取 $c_1=0.2$，$c_2=0.8$。

4) 算法终止条件

为了兼顾算法的搜索效率和优化质量，若搜索到的最优值连续 L 次不变或迭代到一定次数 N 后则停止搜索。本算法在仿真实验中取 $N=1000$。

6 仿真结果及分析

Taillard Benchmark 包含 120 个标准流水线调度问题实例，为了评价算法的性能，本算法的仿真实验在求 TFT 时使用了前 60 个实例，每个实例计算 10 次，分别求出总完成时间的最优值 TFT；在求 C_{max} 时使用了由 Carlier 设计的 8 个测试实例 Car1－Car8[8]。粒子种群规模为作业数目，$c_1=0.2$，$c_2=0.8$，$w=0.2$，对 gbest 执行 5 次随机扰动，最大迭代次数 N 为 1000。采用 Java 语言编码，在处理器为 PIV2.4G、内存 512M 的 PC 机上运行，运行环境为 WindowsXP 及 MyEclips6.0。

本文提出的算法仿真结果如表 2－表 4 所示，其中平均值偏差(AVG%)是指平均值相对问题最优值的偏差。

表 2 为本文算法所得 TFT 值（每种规模 $J*m$ 选出一个结果）与 PAN 等[2] 提出的 DPSO 混合调度算法以及当前已知最好解 BES(LR)[8] 的比较，T 为平均 CPU 时间(ms)。本文作者将 DPSO 算法扩展到双目标 NWFS 问题，采用与文献[2]完全相同的参数设置，并使用相同的总迭代次数（算法终止条件）。

表 2　以 TFT 值为目标函数的各算法的仿真结果比较

问题	J,m	BES(LR)	DPSO	本文所提算法		AVG(%)
				TFT	T_{CPU}	
Ta001	20,5	14226	15674	13980	9265	−1.73
Ta016	20,10	19608	22011	19129	9297	−2.44
Ta025	20,20	34990	39012	34275	8781	−2.04
Ta034	50,5	69795	82467	75895	19000	8.74
Ta049	50,10	88230	110367	99118	21156	12.34
Ta060	50,20	127823	168386	155024	92875	21.28

由表 2 可知，本文提出的混合离散离子群调度算法所得结果与当前已知的最优解非常接近，并在小规模范围内超过了已知最优解；本文算法大大改进了 DPSO 算法的结果，同时 CPU 耗时也很小，表明了该算法具有很好的搜索质量以及在使用了 D 矩阵计算总完成时间的优越性。

表 3 列出了本文提出的混合算法与 RAJ 算法对标准 PFSP 问题测试算例 Car1－Car8 运行的结果，其中 C 表示问题的最优 makespan(C_{max})值，AVG 为平均相对百分偏差。

表 3　本文提出的混合算法与 RAJ 算法的结果比较

问题	J,m	C	本文算法结果	AVG(%)
Car1	11,5	8142	8369	2.79
Car2	13,4	8242	8242	0.00
Car3	12,5	8866	8904	0.43
Car4	14,4	9195	9679	5.26
Car5	10,6	9159	8757	−4.39
Car6	8,9	9690	9386	−3.14
Car7	7,7	7705	7599	−1.38
Car8	8,8	9372	8530	−8.98

由表 3 可知，本文提出的混合离散粒子群算法在解决流水线调度问题中随着种群规模的增大在平均偏差上比 RAJ 算法有了一定的提高，表明对粒子群算法进行领域搜索能有效提高算法性能。

表 4 列出本文的双目标 NWFS 调度的加权混合算法在经过随机加权累加函数加权后的测试结果，使用测试用例来源于 Taillard Benchmark 标准流水线调度问题实例。由表 4 可见，在双目标 NWFS 调度算法中嵌入局部搜索是有效的，并且随机加权累加函数这种处理多目标优化问题的策略展示了良好的有效性。

表 4　混合算法的双目标加权仿真结果

问题	J,m	本文所提算法		加权结果($TFT*0.8+C_{max}*0.2$)
		TFT	C_{max}	
Ta001	20,5	13980	1506	11485.20
Ta016	20,10	19129	1924	15688.00
Ta025	20,20	34275	3046	28029.20
Ta034	50,5	75895	3471	61410.20
Ta049	50,10	99118	4268	80148.00
Ta060	50,20	155024	6098	125238.8

结束语　本文提出了一种解决双目标无等待流水线调度问题的粒子群混合优化算法，采用 ROV 规则和位置矢量编码，建立了位置矢量与工件调度之间的映射关系，使用了 NEH 算法构造初始粒子群、加随机扰动、变邻域搜索等方法来提高算法的搜索效率，并使用了随机加权累加函数来处理双目标优化问题，将 TFT 和 C_{max} 这两个评价调度问题的目标函数转化为单目标优化，有效地解决了双目标问题。为了在

（下转第 213 页）

证明：若 $x\in \underline{apc_\beta}''(X)$，则对于任意 $y\in U, x\in N(y)$ 时，有 $P(N(y),X)\leqslant\beta$。因 $x\in N(x)$，故 $P(N(x),X)\leqslant\beta$，即 $x\in \underline{C_\beta}(X)$。故 $\underline{apc_\beta}''(X)\subseteq\underline{C_\beta}(X)$。再由 $x\in N(x)$ 即得 $\underline{C_\beta}(X)\subseteq\underline{apc_\beta}'(X)$。另一式子类似可证。

推论 设 (U,C) 是覆盖近似空间，$0\leqslant\beta<0.5$，若集合类 $\{N(x)\mid x\in U\}$ 构成了 U 的一个划分，则变精度覆盖粗糙集模型Ⅰ中的近似算子、模型Ⅱ中的近似算子、推广式①和推广式②等价，即覆盖近似空间 (U,C) 中近似算子满足：

$$\underline{apc_\beta}''(X)=\underline{C_\beta}(X)=\underline{apc_\beta}(X)=\underline{apc_\beta}'(X),$$

$$\overline{apc_\beta}'(X)=\overline{C_\beta}(X)=\overline{apc_\beta}(X)=\overline{apc_\beta}''(X)。$$

结束语 粗糙集模型的推广研究主要包括两方面的内容：一是推广近似空间中的等价关系，另一种方法是拓广被近似的对象。本文把论域上的等价关系扩展为论域上的覆盖，定义了两种类型的变精度覆盖粗糙集模型，并讨论了它们的基本性质。由于变精度覆盖粗糙集模型Ⅱ的上、下近似算子不满足对偶性，所以从对偶性的角度出发又对变精度覆盖粗糙集模型Ⅱ进一步进行推广，得到了两对对偶的上、下近似算子，讨论了它们之间的关系。如何将相关理论研究结果应用于信息系统知识发现是我们下一步要解决的问题。

参考文献

[1] Pawlak Z, Skowron A. Rudiments of rough sets[J]. Information Sciences, 2007, 177:3-27

[2] Pawlak Z, Skowron A. Rough sets: Some extensions[J]. Information Sciences, 2007, 177:28-40

[3] Pawlak Z, Skowron A. Rough Sets and Boolean Reasoning[J]. Information Sciences, 2007, 177:41-73

[4] Ziarko W. Variable precision rough set model[J]. Journal of Computer System Science, 1993, 46(1): 39-59

[5] 陶志，许宝栋，汪定伟，等. 基于变精度粗糙集理论的粗糙规则挖掘算法[J]. 信息与控制，2004，33(1)：18-22

[6] Beynon M. Reducts within the variable precision rough sets model: A further investigation[J]. European Journal of Operational Research, 2001, 134: 592-605

[7] 张文修，梁怡，吴伟志，等. 信息系统与知识发现[M]. 北京：科学出版社，2003：56-67

[8] Inuiguchi M. Structure-Based Approaches to Attribute Reduction in Variable Precision Rough Set Models[A]//Proceeding of 2005 IEEE International Conference on Granular Computing [C]. Beijing, China, IEEE GrC2005, 2005:34-39

[9] Mieszkowicz - Rolka A, Rolka L. Remarks on approximation quality in variable precision fuzzy rough sets model[A]//Rough Sets and Current Trends in Computing. 4th International Conference, RSCTC 2004[C]. Heidelberg: Springer-Verlag, 2004: 402-411

[10] Mieszkowicz-Rolka A, Rolka L. Fuzzy implication operators in variable precision fuzzy rough sets model[A]//Artifical Intelligence and Soft Computing- ICAISC 2004 (volume 3070)[C]. Heidelberg: Springer-Verlag, 2004: 498-503

[11] 巩增泰，孙秉珍，邵亚斌，等. 一般关系下的变精度粗糙集模型[J]. 兰州大学学报(自然科学版)，2005，41(6)：110-114

[12] 张亚军，王艳平. 基于覆盖的变精度粗糙集模型[J]. 辽宁工学院学报，2006，26(4)：274-276

[13] Zhu W, Wang F Y. Reduction and axiomization of covering generalized rough sets[J]. Information Sciences, 2003, 152: 217-230

[14] 张文修，吴伟志，梁吉业，等. 粗糙集理论与方法[M]. 北京：科学出版社，2003：41-131

(上接第 202 页)

大规模种群范围内进一步改善搜索结果，可借助于量子行为的研究成果来发展粒子群算法，例如，在位置更新中可采用矢量法[16]来实现两个或多个目标的分别更新，这将是该算法下一步的改进目标。

参考文献

[1] Zhang Hong, Li Xiaodong, Li heng, et al. particle swarm optimization-based schemes for resource-constrained project scheduling [J]. Automation in Construction, 2005, 14:393-404

[2] Pan Q K, Tasgetiren M F, Liang Y C. A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem with makespan criterion[c]//Proceedings of the International Workshop on UK Planning and Scheduling Special Interest Group. London, UK: City University, 2005:31-41

[3] Kennedy J, Eberhard R C, Particle swarm optimization//Proceedings of IEEE International Conference on Neural Networks. Piscataway, NJ, USA, 1995:1942-1948

[4] 王凌，刘波. 微粒群优化与调度算法[M]. 北京：清华大学出版社，2008

[5] Li Xiaoping, Wang Qian, Wu Cheng. An Efficient Method for No-Wait Flow Shop Scheduling to Minimize Makespan[J]//Proceedings of the 10th International Conference on Computer Supported Cooperative Work in Design. 2006

[6] Nawaz M, Enscore E, Ham I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. Omega, 1983, 11:91-95

[7] Hansen P, Mladenovic N. Variable neighborhood search [J]. Computers in Operations Research, 1997, 24: 1097-1100

[8] Carlier J. Ordonnancements a contraints disjonctives. R. A. I. R. O. RechercheOperationelle/Oper. Res., 1978, 12:333-351

[9] Rajendran C. A no-wait flowshop scheduling heuristic to minimize makespan. Oper Res soc., 1994, 45:472-478

[10] 窦全胜，周春光，马铭，等. 群核进化粒子群优化方法. 计算机科学[J]，2005，32(8)：134-137

[11] 童鋆，吴智铭，童争雄. 一种自适应的微粒群算法. 计算机科学[J]，2007，34(9)：198-199，252

[12] 曾立平，黄文奇. 一种求解车间作业调度问题的混合邻域结构搜索算法. 计算机科学[J]，2005，32(5)：177-180，189

[13] 张居阳，孙吉贵. 组合优化调度问题求解方法. 计算机科学[J]，2003，30(2)：9-16

[14] Grabowski J, Pempera J. Some local search algorithms for no-wait flow-shop problem with makespan criterion [J]. Computers& Operations Research, 2005, 32(8): 2197-2212

[15] Aldowaisan T, Allahverdi A. New heuristic for no - wait flow - shops to minimize makespan [J]. Computer& Operation research, 2003, 30(8):1219-1231

[16] Sun J, Feng B, Xu W B. Particle swarm optimization with particles having quantum behavior//Proc. Of the IEEE CEC. 2004: 325-331