doi:10.3969/j.issn.1673-4785.2011.02.007

一种带禁忌搜索的粒子并行子群最小约简算法

马胜蓝,叶东毅
(福州大学 数学与计算机科学学院, 福建 福州 350108)

摘 要:为了提高基于群体智能的粗糙集最小属性约简算法的求解质量和计算效率,提出一个结合长期记忆禁忌搜索方法的粒子群并行子群优化算法.并行的各子群不仅具有禁忌约束,而且包含多样性和增强性策略.由于并行的子群共同陷入局部最优的概率小于一个粒子群陷入局部最优的概率,该算法可提高获得全局最优的可能性,并减少受初始粒子群体的影响.多个UCI数据集的实验计算表明,提出的算法相对于其他的属性约简算法具有更高的概率搜索到最小粗糙集约简.因此所提出的算法用于求解最小属性约简问题是可行和较为有效的.

关键词:属性约简;粗糙集;禁忌搜索;粒子群优化算法;并行子群
中图分类号: TP18 **文献标识码**:A **文章编号**:1673-4785(2011)02-0132-09

A minimum reduction algorithm based on parallel particle sub-swarm optimization with tabu search capability

MA Shenglan, YE Dongyi
(College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China)

Abstract: In order to improve the solution quality and computing efficiency of rough set minimum attribute reduction algorithms based on swarm intelligence, a parallel particle sub-swarm optimization algorithm with long-memory Tabu search capability was proposed. In addition to the taboo restriction, some diversification and intensification schemes were employed. Since parallel sub-swarms have a lower probability of simultaneously getting trapped in a local optimum than a single particle swarm, the proposed algorithm enhances the probability of finding a global optimum and decreases the influence of initial particles. Experimental results on a number of UCI datasets show that the proposed algorithm has a higher probability of finding a minimum attribute reduction in rough sets compared with some existing swarm intelligence based attribute reduction algorithms. Therefore, the proposed algorithm is feasible and relatively effective for the minimum attribute reduction problem.

Keywords: attribute reduction; rough set; tabu search; particle swarm optimization; parallel particle sub-swarm

作为特征选择的一个有效方法[1-2],粗糙集属性约简及其算法近年来得到了广泛的研究[3-4].其中,对于NP难的最小属性约简问题[3]的研究也取得了新的进展,提出了若干基于进化计算或群体智能的最小约简算法.例如,Wroblewski和Bazan等利用遗传算法寻找最小约简[5];R. Jensen等提出的基于蚁群算法的最小约简算法[6];最近,Abdel-Rahman Hedar等人提出将禁忌算法用于最小约简的搜索[7];而文献[8]和[9]则探讨了粒子群方法在求解最小约简问题中的应用.相对启发式属性约简算法而言,这些最小约简算法提高了求得最小约简的可能性,但是在求解质量和计算效率方面仍有待进一步改进.

针对上述情况,提出了一个带长期记忆的禁忌搜索和并行子群粒子群优化方法相结合的最小属性约简算法.PSO算法简单、易于实现,通常比遗传算法收敛速度快;另一方面,应用蚁群算法求解最小属性约简问题(如AntRSAR[6])需要采取和属性个数

收稿日期:2010-03-07.
基金项目:国家自然科学基金资助项目(60805042);福建省自然科学基金资助项目(2010J01329).
通信作者:叶东毅. E-mail:yiedy@fzu.edu.cn.

一致的种群规模,当规模较大时,计算复杂度明显增加. 而应用 PSO 算法则没有这方面的限制. 本文中并行的各子群不仅具有禁忌约束,而且包含多样性和增强性策略,采用这 2 种策略主要是为了降低算法陷入局部极小的可能性. 实验计算结果表明,与基于粒子群优化的属性约简算法、基于禁忌搜索的属性约简算法和基于蚁群优化的最小属性约简算法相比,本文算法取得了较好的改进效果.

1　基本概念和记号

为叙述方便起见,首先简单回顾粗糙集理论的一些基本概念[10-11]以及禁忌搜索和粒子群优化的基本思想.

1.1　粗糙集与属性约简

四元组 $S=(U,A,V,f)$ 是一个信息系统,其中 U 为对象的非空有限集合,即论域;A 为属性的非空有限集合,信息系统中 A 常分为条件属性 C 和决策属性 D;$V=\bigcup\limits_{a\in A}V_a$,$V_a$ 是属性的值域;f 为 $U\times A\to V$ 是一个信息函数,它为每个对象的每个属性赋予一个信息值,即 $\forall a\in A, x\in U, f(x,a)\in V_a$. 设 $P\subseteq A$ 且 $P\neq\varnothing$,定义由属性子集 P 导出的二元关系如下:

$$\mathrm{IND}(P)=\{(x,y)\mid \forall a\in P, f(x,a)=f(y,a)\}.$$

若 $(x,y)\in\mathrm{IND}(P)$,则称 x 和 y 是 P 不可区分的,即依据 P 中所含各属性无法区分 x 和 y.

设集合 $X\subseteq U$,P 是一个等价关系,称 $\underline{P}X$ 为集合 X 的 P 下近似集;$\overline{P}X$ 为集合 X 的 P 上近似集;集合 $\mathrm{BN}_P(X)$ 为 X 的 P 边界域.

$$\underline{P}X=\{x\mid x\in U, [x]_P\subseteq X\},$$

$$\overline{P}X=\{x\mid x\in U, [x]_P\cap X\neq\varnothing\},$$

$$\mathrm{BN}_P(X)=\overline{P}X-\underline{P}X.$$

设 P 和 Q 为论域上的等价关系,Q 的 P 正域记作 $\mathrm{POS}_P(Q)$:

$$\mathrm{POS}_P(Q)=\bigcup_{X\in U/Q}\underline{P}X,$$

$$k=\gamma_P(Q)=\frac{|\mathrm{POS}_P(Q)|}{|U|}.$$

称知识 Q 是 k 度依赖于知识 P 的,记作 $P\Rightarrow_k Q$. 当 $k=1$ 时,称 Q 完全依赖于 P;当 $0<k<1$ 时,称 Q 粗糙依赖于 P;当 $k=0$ 时,称 Q 完全独立于 P.

属性约简是去除一些冗余的属性而不减少原来集合的分类质量. 一个约简可以定义如下:

$$\mathrm{Red}=\{R\subseteq C\mid \gamma_R(D)=\gamma_C(C),$$

$$\forall B\subset R, \gamma_B(D)\neq\lambda_C(D)\}.$$

一个数据集可能有多个属性约简,最小约简是指包含属性个数最少的一个属性约简.

所有属性约简的交集为核,记为

$$\mathrm{Core}(C)=\cap\ \mathrm{Red}.$$

1.2　禁忌搜索

禁忌搜索是一种启发式算法[12],其基本过程如下.

算法 1　基本的禁忌搜索算法

1) 选择问题的一个初始解 x_0,设置禁忌表(T_L)为空并且设迭代次数 $k:=0$;

2) 根据禁忌限制生成一个当前解 x_k 的邻居解空间 $M(x_k)$;

3) 计算邻居解空间 $M(x_k)$ 的最优解 x_{k+1} 并且更新禁忌表 T_L;

4) 如果满足停止条件则停止该算法,否则转向 2).

1.3　标准粒子群优化算法 PSO

最早由 Kennedy 等人[13]提出的 PSO 算法是一种群体搜索算法,它根据对环境的适应度将群体中的个体移动到好的区域. 每个个体看作是 D 维搜索空间中的一个没有体积的微粒(点),在搜索空间中以一定的速度飞行,这个速度根据它本身的飞行经验和同伴的飞行经验来动态调整. 第 i 个微粒表示为 $\boldsymbol{X}_i=(x_{i1},x_{i2},\cdots,x_{iD})$,它经历过的最好位置(有最好的适应值)记为 $\boldsymbol{P}_i=(p_{i1},p_{i2},\cdots,p_{iD})$,也称为 $\boldsymbol{p}_{\mathrm{best}}$. 在群体所有微粒经历过的最好位置的索引号用符号 g 表示,即 P_g,也称为 $\boldsymbol{g}_{\mathrm{best}}$. 微粒 i 的速度用 $\boldsymbol{V}_i=(v_{i1},v_{i2},\cdots,v_{iD})$ 表示. 对每一代,它的第 d 维($1\leqslant d\leqslant D$)根据如下方程进行变化:

$$v_{id}=w\times v_{id}+c_1\times\mathrm{rand}()*(p_{id}-x_{id})+c_2\times\mathrm{Rand}()*(p_{gd}-x_{id}),$$

$$x_{id}=x_{id}+v_{id}.$$

式中:w 为惯性权重;c_1 和 c_2 为加速常数;rand() 和 Rand() 为 2 个在[0,1]范围里变化的随机值.

2　基于禁忌搜索和粒子并行子群的属性约简算法 TSPPSOAR

禁忌搜索算法通过搜索当前解的邻居来获得解,其中禁忌表有助于避免搜索陷入局部最优,但是该方法的搜索能力并不强;而粒子群优化算法可以通过粒子群来搜索解,具有较强的搜索能力,但是容易陷入局部最优. 本文通过结合这 2 种算法的思想来提高搜索能力并避免陷入局部最优,由此提出了一个改进的算法,记为 TSPPSOAR. 该算法将原始粒子群分解成具有较少个体数目的并行子群来提高

获得最优解的概率,同时利用了带有长期记忆的禁忌策略和 2 个优化方案来提高小群体数目的子群的搜索能力.

一般地,基本的禁忌搜索算法仅利用了短期记忆,即禁忌表. 为了取得更好的性能,需要长期记忆来保存搜索过程中获得的重要特征. 因此,在本文算法的禁忌搜索过程中加入了长期记忆"属性的使用频率",并通过如下的 2 个方案来自动调节解的状态.

多样性:通过属性的使用频率来调节解,使算法在解空间搜索到新的区域.

增强性:提高一些有较好前景(有可能达到全局最优点)的解的优先权,以搜索该类较优解周围可能存在的更优解.

2.1 粒子的解表示

TSPPSOAR 利用二进制串来表示一个解(粒子的位置). 如果对于一个解 x 中的某个属性 x_i($i=1,2,\cdots,|C|$, $|C|$ 为所有条件属性的个数)的值为 1,则 x 具有第 i 个属性;同样地,如果 $x_i=0$,则 x 不具有第 i 个属性.

2.2 禁忌表

禁忌表 T_L 的作用是用来避免陷入局部最优和避免重复生成不合适的粒子的位置解. 全部为 1(所有属性都被考虑)和全部为 0(所有属性都不考虑)的解将长期存放在 T_L 表中. T_L 剩余中的位置($|TL|$-2)被用于保存最近粒子到达的位置,其中 $|T_L|$ 表示禁忌表的长度.

禁忌表的更新策略为:如果禁忌表还未存满,则将新的粒子的位置解插入禁忌表中;否则将($|T_L|-2$)中最先插入的解替换为新的解.

2.3 粒子速度的表示

采用文献[14]的方法,每个粒子的速度被表示为$[1,V_{\max}]$区间内的 1 个正数,表示每次粒子有多少个位要被改变为与全局所经历最优位置相对应的位置一致. 2 个粒子之间的差跟 2 个粒子的不一致的位的个数有关[14].

需要注意的是,算法中必须限制粒子的速度而避免飞过全局最优解而只能找到次优解. 本文中设置 $V_{\max}=(1/3)*|C|$,如果 $V<1$,则 $V=1$.

2.4 粒子位置更新策略

设更新后的速度为 V,当前粒子位置和全局所经历的最优位置 $p_{g\text{best}}$ 之间的不一致的位个数为 x_g,则更新策略如下所示.

1)如果 $V<=x_g$,则在当前粒子和全局所经历最优位置不一致的位上随机取 V 位置为相同. 更新后该粒子仍然保持着自身到全局最优位置的搜索能力.

2) 如果 $V>x_g$,先将当前粒子位置更新为与全局所经历最优位置相同,再在更新前粒子的位置和全局所经历最优位置相同的位上随机抽取置(V-x_g)位置为不同,这样粒子在到达当前全局最优位置之前还可以继续保持朝其他方向飞行搜索的能力.

3)如果改变位置后粒子的位置在 T_L 表中出现,则重新执行 1)、2)来生成新的位置.

2.5 粒子的位置质量衡量标准

算法中使用适应值来衡量粒子的位置解的质量优劣. 适应度函数如下所示[14]:

$$\text{Fitness}=\partial\times\gamma_R(D)+\beta\times\frac{|C|-|R|}{|C|}. \quad (1)$$

式中:$\gamma_R(D)$ 是属性 R 相对于决策属性 D 的依赖度;$|R|$ 是被选择的子集的属性个数;∂ 和 β 分别控制着依赖度的值和子集的长度,其中 ∂ 取 0 ~ 1 的数,$\beta=1-\partial$. 本文中设置 ∂ 为 0.9,$\beta=0.1$. 设置较高的 ∂ 值可以尽量保证最优解本身是一个约简,而不降低原集合分类质量. 算法的目标是最大化适应值.

2.6 并行子群

根据概率论可以得出:

1)如果一个粒子群 A 得到最优解的概率为 p,则 A 陷入局部最优解的概率则为$(1-p)$.

2)将 A 分解成 n 个并行子群 A_i,则 n 个 A_i 构成一个新的群体 A';在求解同一个问题时,每个子群 A'得到最优解的概率为 p',总群体 A'陷入局部最优解的概率为$(1-p')^n$.

3)如果 p'不会远小于 p,则粒子群 A'陷入局部最优的概率小于粒子群 A 进入局部最优的概率;因此具有并行子群的总群体 A'获得最优解的概率可以满足如下条件:$1-(1-p')^n>p$.

群体个数为 N 的粒子群有 2 种分解策略:一种是分解为子群个数为 n_1,则每个子群内粒子个数为 N/n_1;另一种是选择分解的群体个数 $n_2>n_1$,则每个子群体内粒子个数 $N/n_2<N/n_1$. 2 种策略分别通过增大 p'和 n 的值来提高获得最优解的概率. 对于这 2 种策略章节 3.2 都做了实验比较.

经过分解后的粒子群具有类似于如图 1 的拓扑结构[15],子群内部独立地搜索解空间,而子群之间通过全网络连接,互相交流着该子群经历过的最好的位置的信息.

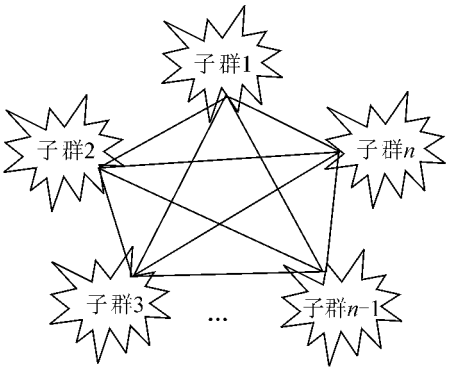

图 1　TSPPSOAR 拓扑结构

Fig. 1　The topology of TSPPSOAR

2.7　多样性-增强性方案

TSPPSOAR 算法将粒子群分解为并行的子群,但是由于每个子群的群体个数的下降,必然会导致每个子群的搜索能力的下降. 如 2.6 中所述,要想降低陷入局部最优的概率,必须让子群获得最优解的概率不会远小于原群体获得最优解的概率,所以在子群内部采用额外的方案来确保每个子群的搜索能力.

因此,TSPPSOAR 应用 3 种方案[7]来提高搜索的多样性和增强性. 为描述完整起见,以下介绍这 3 个方案. 第一个方案是在一定迭代时间内无法搜索出可以改进现有的最优解时所执行的多样性方案. TSPPSOAR 中定义了一个具有 $|C|$ 维的向量 $\boldsymbol{v}^F$,其中向量中第 i 维代表群体中每个粒子经历过的位置中选择属性 i 的次数. 因此一个多样性的新解 x^{div} 可以通过 $\boldsymbol{v}^F$ 中每个属性的被选择的概率来获得. 以下给出 x^{div} 的生成过程:

过程 1　$[x^{\mathrm{div}}]=\mathrm{Diverse}(\boldsymbol{v}^F)$

1)随机生成数字 $r_1,r_2,\cdots,r_{|C|}\in(0,1)$.

2)重复执行如下步骤从 $i=1,2,\cdots,|C|$.

3)如果 $r_i>v_i^F/\sum_{j=1}^{|C|}v_j^F$ 则设置 $x^{\mathrm{div}}=1$,否则设置 $x^{\mathrm{div}}=0$.

上述的多样性方案主要选择过去较少被选择到的属性,将粒子导向一个新的解空间,从而加大了逃离出局部最小点的可能性.

在获得 x^{div} 后,如果全局经历的最优位置 p_{gbest} 在经过一段时间后仍然没有任何改进,则 TSPPSOAR 应用增强性方案来改进每个粒子经历过的最优位置 p_{best}. 在过程 2.2 中通过对 p_{best} 进行约简而不减少 $\gamma_{p\mathrm{best}}(D)$,将原解从代数协调集调整为代数约简.

过程 2　$\mathrm{Shaking}(p_{\mathrm{best}})$

1)对每个粒子经历的最优位置 p_{best} 重复执行如下步骤;

2)将 p_{best} 中的为 1 的属性构造成一个新的集合 $W=\{w_1,w_2,\cdots,w_{|W|}\}$;

3)通过设置 $w_i=0$ 来删除属性 w_i,计算 γ 值;

4)如果 γ 增加或者 γ 不变(但是由于属性个数减少)则更新 p_{best};

5)更新全局最优解 $p_{g\mathrm{best}}$.

最后的方案是约简激励. 在 TSPPSOAR 迭代搜索的过程中,得到的约简都被保存在一个叫做 RedSet 的集合中. 利用核的概念,在 RedSet 中取 n_R 个最优约简的交集构造一个新的解 x^{ERJ}. 如果 x^{ERI} 的属性个数比粒子全局经历的最优位置 $p_{g\mathrm{best}}$ 的属性个数至少少 2 个,则 x^{ERJ} 中根据启发性信息将一个 0 的位置改为 1. 该方案一直执行到 X^{ERI} 中的属性个数比 $p_{g\mathrm{best}}$ 的属性少一个.

过程 3　$[x^{\mathrm{ERJ}}]=\mathrm{EliteReducts}(\mathrm{RedSet},n_R)$

1)如果 RedSet 为空则过程结束,否则到 2);

2)设 n_F 为 RedSet 中最优约简的属性个数,设置 x^{ERI} 为 RedSet 中 n_R 个最优约简的交集;

3)如果 x^{ERI} 的属性个数小于 (n_F-1),则执行 4),否则过程结束;

4)如果 $\mathrm{POS}_{x^{\mathrm{ERI}}}(D)$ 等于原论域的正域个数,则过程结束;

5)在 x^{ERI} 的所有 0 位置中选择一个,在置为 1 的情况下可以得到最大的 r 值的位置,将其置为 1,转到 3).

激励过程相当于在 RedSet 的交集"新核"基础上,寻找比搜索到的全局最小约简 $p_{g\mathrm{best}}$ 更优的解.

2.8　TSPPSOAR 算法

基于禁忌搜索和粒子并行子群的属性约简算法 TSPPSOAR,将原群体分解成 n 个并行的子群,每个子群同时进行属性约简,以提高获得最优解的概率.

在该算法中,惯性权重 W 随着迭代次数的增加而不断减小,公式如下:

$$W=W_{\max}=\frac{W_{\max}-W_{\min}}{\mathrm{iter}_{max}}\times\mathrm{iter}. \tag{2}$$

式中:$W_{\max}$ 和 $W_{\min}$ 为初始设定的值,iter 为当前迭代的次数. 具体算法如下.

算法 2　TSPPSOAR

1)初始化一群粒子,包括随机的位置和速度;初始化禁忌表 T_L,禁忌表中包含 2 个位置 $(0,\cdots,0)$ 和 $(1,\cdots,1)$,设置 $\boldsymbol{v}^F$ 为空向量,RedSet 为空集合,迭代次数 iter 为 0,选择 $I_{\max}$、I_{imp}、I_{shak} 和 I_{div},令 $I_{\max}>I_{\mathrm{imp}}>I_{\mathrm{shak}}>I_{\mathrm{div}}$. 将粒子群分割成 n 个含有相同粒子数的子群 $(\mathrm{child}_1,\mathrm{child}_2,\cdots,\mathrm{child}_n)$,每个子群维护着各自的禁忌表 T_{L_i}、V_i^F.

2)对于每个子群 child_i 执行 3) ~8).

3)利用式(1)求出每个粒子的适应值.

4)对每个粒子,将它和它经历过的最优位置 p_{best} 作比较,如果较优,则将其作为该粒子的最优位置 p_{best}.

5)对每个粒子,将它和该子群全局所经历最优位置 $g_{\text{best}i}$ 作比较,如果较优,则重新设置 $g_{\text{best}i}$ 的索引.

6)比较 $g_{\text{best}i}$ 和总群体的最优位置 g_{bestall},如果较优,则重新设置 g_{bestall} 索引.

7)更新粒子的速度,利用禁忌表 T_{L_i} 更新粒子位置.

8)根据章节 2.2 中提到的更新策略更新子群的禁忌表 T_{L_i};根据粒子的位置更新 V_i^F,如果该粒子位置已经是一个约简,则存入 RedSet,最后设置 iter = iter + 1.

9)如果迭代次数大于 $I_{\max}$ 或者总群体最优位置 g_{bestall} 未改进的次数超过 I_{imp},则转向 12).

10)如果总群体最优位置未改进的次数超过 I_{shak},应用过程 2 对每个子群的每个粒子经历的最优位置 p_{best} 进行处理,转向 2).

11)如果总群体最优位置未改进的次数超过 I_{div},则应用过程 1 来重新获得每个子群的每个粒子的当前位置,转向 2).

12)应用过程 3 来获得 x^{ERI},如果 x^{ERI} 比总群体最优位置 p_{gbestall} 更优,则设置 $p_{\text{gbestall}} = x^{\text{ERI}}$. 图 2 为算法 2 的流程图.

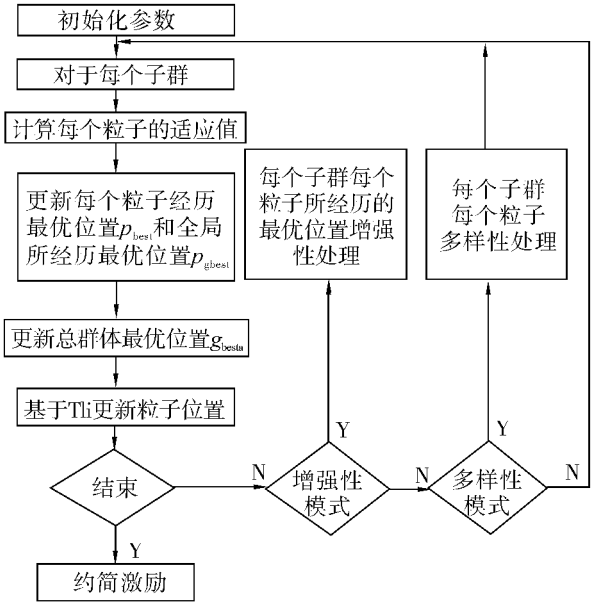

图 2 TSPPSOAR 算法流程

Fig. 2 The flowchart of TSPPSOAR

3 实验与讨论

3.1 数据集介绍

实验测试所用的数据为 11 个 UCI 数据集,具体的数据信息如表 1 所示.

表 1 数据集

Table 1 Datasets

数据集	实例个数	条件个数
Hayes	132	5
Bupa	345	6
Corral	32	6
Breast	191	9
Led24	200	24
Mushroom	8 124	22
Soybean1	47	35
Soybean2	307	35
Splice	2 126	60
Sponge	76	45
Vote	300	16

3.2 实验参数

首先对算法的符号做简要说明.

TSAR 为基于禁忌搜索的最小属性约简算法[7];PSORSFS 为基于粒子群优化算法的最小数型属性约简算法[14];AntRSAR[16] 为改进的基于蚁群优化[6]的最小属性约简算法.

TSPPSOAR(10,10)和 TSPPSOAR(5,5,5)为本文所提的属性约简算法,括号内的数字 $(n_1, n_2, \cdots, n_N)$ 表示的是分解方法:粒子群总群体个数为 N,第 i 个并行子群个数为 n_i,且 $N = \sum n_i$. 实验过程中对每个数据集均测试 100 次,实验结果用数字"a^b"表示:a 代表算法在该数据集上属性约简后的属性长度,b 代表 100 次试验中有 b 次达到长度为 a 的属性约简.

实验中的参数设置见表 2.

表 2 TSPPSOAR、PSORSFS 和 TSAR 参数设置

Table 2 The parameter settings of TSPPSOAR、PSORSFS and TSAR

	population	$I_{\max}$	$V_{\max}$	n_R	$\lvert T_L \rvert$
TSPPSOAR	20、15	100	$(1/3) \times N$	10	7
PSORSFS	20	100	$(1/3) \times N$	—	—
TSAR	—	100	—	10	7
AntRSAR	$\lvert C \rvert$	250	—	—	—

其中 pupulation 为总群体个数,$V_{\max}$ 为粒子的最大速度. 禁忌表长度 $\lvert T_L \rvert$ 被设置为 7,除了保存全 1 和全 0 的位置外,剩余 5 个用于保存最近访问的位置. 实验中发现,该长度的禁忌表以一定概率地让粒子跳出局部最小点. I_{div} 和 I_{shak} 分别设置为 10 和 20,这种设置可以避免粒子花费过多的迭代时间在没有任何改进的搜索中,从而让粒子进入一个新的解空

间.在约简激励过程中,利用了 $n_R(n_R = 10)$ 个 RedSet 中最优的约简.表格中的"—"符号表示该算法没有使用该列所示的参数.

3.3　2 种 TSPPSOAR 分解策略比较

首先比较 TSPPSOAR 在 2 种不同分解策略下的实验结果,TSPPSOAR(10,10)代表第 1 种分解策略,TSPPSOAR(5,5,5)代表第 2 种策略.实验结果如表 3、4 所示;其中表 4 为约简的属性平均长度值.

表 3　2 种 TSPPSOAR 分解策略实验数据比较 1

Table 3　The comparisons of two partition strategies in TSPPSOAR (1)

数据集	TSPPSOAR(5,5,5)	TSPPSOAR(10,10)
Hayes	1^{100}	1^{100}
Bupa	3^{100}	3^{100}
Corral	4^{100}	4^{100}
Breast	$8^{98}9^{2}$	$8^{98}9^{2}$
Led24	$11^{26}12^{73}13^{1}$	$11^{19}12^{79}13^{2}$
Mushroom	$4^{68}5^{32}$	$4^{63}5^{37}$
Soybean1	$2^{89}3^{10}4^{1}$	$2^{80}3^{14}4^{6}$
Soybean2	$9^{75}10^{25}$	$9^{62}10^{29}11^{9}$
Splice	$9^{81}10^{18}11^{1}$	$9^{67}10^{32}11^{1}$
Sponge	$8^{80}9^{19}10^{1}$	$8^{70}9^{28}10^{2}$
Vote	$8^{90}9^{8}10^{2}$	$8^{97}9^{1}10^{2}$

表 4　2 种 TSPPSOAR 分解策略实验数据比较 2

Table 4　The comparisons of two partition strategies in TSPPSOAR (2)

数据集	TSPPSOAR (10,10)	TSPPSOAR (5,5,5)
Hayes	1	1
Bupa	3	3
Corral	4	4
Breast	8.02	8.02
Led24	11.83	11.75
Mushroom	4.37	4.32
Soybean1	2.66	2.12
Soybean2	9.47	9.25
Splice	9.34	9.20
Sponge	8.32	8.21
Vote	8.05	8.12

从表 3 中可以看出在每个数据集的 100 次计算试验中,除了 Vote 数据集外,TSPPSOAR(5,5,5)得到最小约简的个数均多于 TSPPSOAR(10,10).而从表 4 可以看出在大部分数据集上,3 个子群的 TSPPSOAR(5,5,5)的结果比 2 个子群的 TSPPSOAR(10,10)的结果更优,虽然在 Vote 数据集上比 TSPPSOAR(10,10)的结果差,但是约简结果显示已经具有 90% 的高概率达到最优约简,并且从图 3 中可以看出 TSPPSOAR(5,5,5)比 TSPPSOAR(10,10)具有更小的计算消耗(计算消耗在 3.4 中介绍).因此在子群中引入带有长期记忆的禁忌策略和多样性增强性策略保证了群体在减少粒子个数的同时不降低搜索解的能力.

下面的实验比较所提及的 TSPPSOAR 主要是指第 2 种策略下的算法,即指 TSPPSOAR(5,5,5).

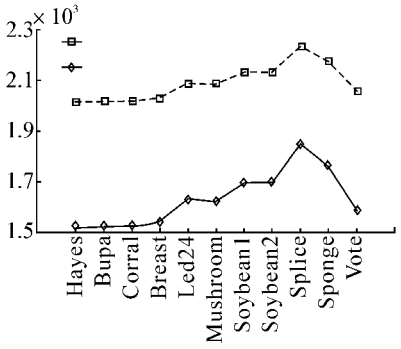

图 3　2 种 TSPPSOAR 分解策略的计算开销比较

Fig. 3　Computational costs on two partition strategies in TSPPSOAR

3.4　TSPPSOAR 与 TSAR、PSORSFS 和 AntRSAR 的比较

现将 TSPPSOAR 与 PSORSFS、TSAR 和 AntRSAR 分别进行比较,实验结果如表 5 所示.为了给出表 5 更详细的结果比较,表 6 中列出 TSPPSOAR、PSORSFS、TSAR 和 AntRSAR 的约简平均值以及这 4 种算法约简的结果的 Z 检验[17](置信度参数为 0.05);另外,在置信度 0.05 差异显著情况下,用括号标出 TSPPSOAR 与其他 3 种属性约简算法相比较具有较优结果的算法.

从表 5 ~ 6 中可以看出:在 11 个数据集上,TSPPSOAR 的约简结果优于 PSORSFS;在 Vote 数据集上,TSPPSOAR 约简结果不如 TSAR,但在剩余的 10 个数据集上,均比 TSAR 的约简结果好;TSPPSOAR 在数据集 Hayes、Bupa、Corral、Breast、Led24 和 Soybean1 上与 AntRSAR 无显著差异,在数据集 Mushroom、Splice 和 Vote 上约简结果略逊于 AntRSAR,在数据集 Soybean2 和 Sponge 上约简结果优于 AntRSAR.

表5 TSPPSOAR、PSORSFS、TSAR 和 AntRSAR 比较1

Table 5 The comparisons of TSPSOAR, PSORSFS, TSAR and AntRSAR (1)

数据集	TSPPSOAR (5,5,5)	TSAR	PSORSFS	AntRSAR
Hayes	1^{100}	1^{100}	1^{100}	1^{100}
Bupa	3^{100}	$3^{98}4^{2}$	3^{100}	3^{100}
Corral	4^{100}	$4^{85}5^{15}$	$4^{88}5^{12}$	4^{100}
Breast	$8^{98}9^{2}$	8^{100}	8^{100}	8^{100}
Led24	$11^{26}12^{73}13^{1}$	$11^{4}12^{2}13^{19}14^{46}15^{29}$	$11^{5}12^{64}13^{30}14^{1}$	$11^{26}12^{74}$
Mushroom	$4^{68}5^{32}$	$4^{32}5^{16}6^{35}7^{16}8^{1}$	$4^{32}5^{23}6^{7}7^{37}8^{1}$	$4^{82}5^{18}$
Soybean1	$2^{89}3^{10}4^{1}$	$2^{34}3^{17}4^{12}5^{31}6^{4}7^{2}$	$2^{31}3^{19}4^{36}5^{13}6^{1}$	$2^{88}3^{12}$
Soybean2	$9^{75}10^{25}$	$9^{15}10^{11}11^{19}12^{26}13^{12}14^{6}15^{11}$	$9^{34}10^{24}11^{28}12^{13}13^{1}$	$9^{55}10^{45}$
Splice	$9^{81}10^{18}11^{1}$	$9^{7}10^{87}11^{6}$	$9^{6}10^{78}11^{6}$	9^{100}
Sponge	$8^{80}9^{19}10^{1}$	$8^{29}9^{39}10^{16}11^{13}12^{3}$	$8^{38}9^{45}10^{17}$	$8^{69}9^{27}10^{4}$
Vote	$8^{90}9^{8}10^{2}$	8^{100}	$8^{71}9^{10}10^{13}11^{6}$	8^{100}

表6 TSPPSOAR, PSORSFS, TSAR 和 AntRSAR 比较2

Table 6 The comparisons of TSPSOAR、PSORSFS、TSAR and AntRSAR (2)

数据集	约简平均值				Z 检验		
	TSPPSOAR	TSAR	PSORSFS	AntRSAR	TSAR - TSPPSOAR	PSORSFS - TSPPSOAR	AntRSAR - TSPPSOAR
Hayes	1	1	1	1	—	—	—
Bupa	3	3.02	3	3	1.421	—	—
Corra	4	4.15	4.12	4	4.181 (TSPPSOAR)	3.674 (TSPPSOAR)	—
Breast	8.02	8	8	8	-1.421	-1.421	-1.421
Led24	11.75	13.94	12.27	11.74	20.905 (TSPPSOAR)	7.141 (TSPPSOAR)	-0.157
Mushroom	4.32	5.38	5.52	4.18	8.689 (TSPPSOAR)	8.646 (TSPPSOAR)	-2.306 (AntRSAR)
Soybean1	2.12	3.6	3.34	2.12	10.591 (TSPPSOAR)	10.326 (TSPPSOAR)	0
Soybean2	9.25	10.21	10.23	9.45	13.209 (TSPPSOAR)	8.347 (TSPPSOAR)	3.017 (TSPPSOAR)
Splice	9.2	9.99	10.10	9	14.120 (TSPPSOAR)	14.34 (TSPPSOAR)	-4.677 (AntRSAR)
Sponge	8.21	9.22	8.79	8.35	8.385 (TSPPSOAR)	6.938 (TSPPSOAR)	1.983 (TSPPSOAR)
Vote	8.12	8	8.54	8	-3.119 (TSAR)	4.150 (TSPPSOAR)	-3.119 (AntRSAR)

本文通过比较基于依赖度值 r 的计算次数来评价计算开销，显然如果数据集的属性个数较多，依赖度值 r 的计算将会占用大量的计算空间. TSAR 的计算开销为最大迭代过程中所耗费的 $50|C|$ 次、增强模式下最多执行的 $|C|-2$ 次和激励过程中的 $3|C|$ 次；PSORSFS 计算开销最多为2 000次；AntRSAR 最少需要的开销为 $250|C|\times Rmin$，Rmin 为约简的最小长度；TSPPSOAR 计算开销为最大迭代过

程中所耗费的 100population 次、增强模式下的 $2n(|C|-2)$ 次(n 为并行子群的群体个数)和激励过程中的 $3|C|$. 则各算法的计算开销为:

TSARmax:$54|C|-2$;

PSORSFSmax:2 000;

AntRSARmin:$250|C|\times$Rmin;

TSPPSOAR(10,10)max:$20\ 00+4|(C|-2)$;

TSPPSOAR(5,5,5)max:$15\ 00+6|(C|-2)$.

如下图 4 ~5 给出各算法之间的计算开销比较.

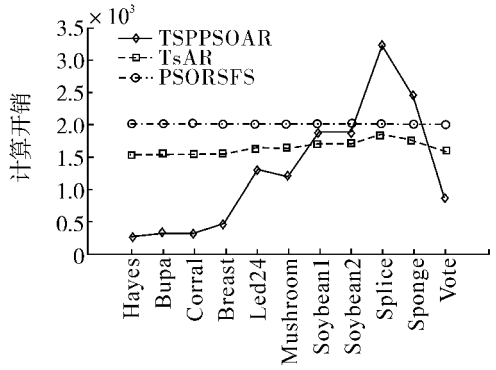

图 4　TSPPSOAR、TSAR 和 PSORSFS 的计算开销比较

Fig. 4　Computational costs on TSPPSOAR, TSAR and PSORSFS

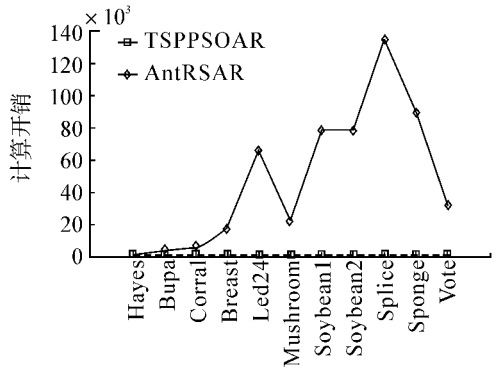

图 5　TSPPSOAR 和 AntRSAR 的计算开销比较

ig. 5　Computational costs on TSPPSOAR and AntRSAR

从图 4 中可以看出 TSPPSOAR 的计算开销介于 TSAR 和 PSORSFS 之间,使得计算开销保持在一个较低的状态,并且在 splice 数据集上拥有最低的计算开销. 从图 5 中可以看出 AntRSAR 虽然在数据集 Mushroom、Splice 和 Vote 上约简结果比 TSPPSOAR 优,但是蚁群算法计算开销高于 TSPPSOAR,通过综合考虑计算开销和约简结果,TSPPSOAR 优于 AntRSAR. 综上可以看出,TSPPSOAR 相对于其他 3 种 CI 属性约简算法(PSORSFS、TSAR 和 AntRSAR),有较低的时间和空间开销,并且有较好的约简结果.

4　结束语

针对粗糙集最小属性约简问题,提出了结合基于记忆的禁忌搜索思想的粒子群并行子群算法(TSPPSOAR). 实验的结果表明了该算法的有效性:不仅具有粒子群在问题空间上很强的搜索能力,而且通过禁忌搜索的长、短期记忆能有效地逃离局部极小点;并行的子群提高了搜索到最小约简的概率;同时计算开销小是其优于基于蚁群的属性约简算法的优势. TSPPSOAR 较高概率达到最小约简的特性可以有效地减少受粒子群初始粒子选择的影响,提高了算法的适应性.

参考文献:

[1] SWINIARSKI R W, SKOWRON A. Rough set methods in feature selection and recognition[J]. Pattern Recognition Letters, 2003, 24: 833-849.

[2] CHOUCHOULAS A, SHEN Q. Rough set-aided keyword reduction for text categorization[J]. Applied Artificial Intelligence, 2001, 15(9): 843-873.

[3] SKOWRON A, RAUSZER C. The discernibility matrices and functions in information systems[C]//Dordrech: Kluwer Academic Publishers, 1992: 311-362.

[4] HU X. Knowledge discovery in databases: an attribute-oriented rough set approach[D]. Regina, Saskatchewan: Canada, Computer Science Faculty of Graduate Studies, University of Regina, 1995: 1-152.

[5] WROBLEWSKI, J. Finding minimal reducts using genetic algorithms[C]//Proc of the Second Annual Joint Conference on Information Sciences. Wrightsville Beach, USA, 1995: 186-189.

[6] JENSEN R, SHEN Q. Finding rough set reducts with ant colony optimization [C]//Proceedings of the 2003 UK Workshop on Computational Intelligence. Bristol, UK, 2003: 15-22.

[7] HEDAR A R, WANG J, FUKUSHIMA M. Tabu search for attribute reduction in rough set theory[J]. Soft Computing, 2008, 12(9): 909-918.

[8] 叶东毅, 廖建坤. 基于二进制粒子群优化的最小属性约简算法[J]. 模式识别与人工智能, 2007, 20(3): 295-300.

YE Dongyi, LIAO Jiankun. Minimum attribute reduction algorithm based on binary particle swarm optimization[J]. Pattern Recognition and Artificial Intelligence, 2007, 20(3): 295-300.

[9] 叶东毅, 廖建坤. 最小约简问题的一个离散免疫粒子群算法[J]. 小型微型计算机系统, 2008, 29(6): 550-

555.
YE Dongyi, LIAO Jiankun. Immune discrete particle swarm optimization algorithm for minimum attribute reduction problems[J]. Journal of Chinese Computer Systems, 2008, 29(6): 550-555.
[10]王国胤. Rough 集理论与知识获取[M]. 西安:西安交通大学出版社, 2001: 23-145.
[11]张文修,梁怡,吴伟志. 信息系统与知识发现[M]. 北京: 科学出版社, 2003: 56-68.
[12]GLOVER F. Tabu search-part I, ORSA[J]. Journal on Computing 1989, 1(3): 190-206.
[13]KENNEDY J, EBERHART R C. Particle swarm optimization[C]//Proc of the IEEE International Conference on Neural Networks. Perth, Australia, 1995: 1942-1948.
[14]WANG Xiangyang, YANG Jie, PENG Ningsong, et al. Finding minimal rough set reducts with particle swarm optimization[J]. Lecture Notes in Artificial Intelligence, 2005, 3641: 451-460.
[15]MENDES R, KENNEDY J, NEVES J. The fully informed particle swarm:simpler, maybe better[J]. IEEE Transactions on Evolutionary Computation, 2004, 7(8): 204-210.
[16]DENG Tingquan, YANG Chengdong. An improved ant colony optimization applied to attributes reduction[J]. Fuzzy Information and Engineering, 2009, 54: 1-6.
[17]MONTGOMERY D C, RUNGER G C. Applied statistics and probability for engineers[M]. 3rd ed. New York, USA:John Wiley & Sons Inc, 2003: 278-326.

作者简介:

马胜蓝,男,1986 年生,硕士研究生,主要研究方向为计算智能.

叶东毅,男,1964 年生,教授,博士生导师,主要研究方向为计算智能、数据挖掘. 曾获得国家科技进步二等奖(主要成员 1 项)、福建省科学技术二等奖 1 项和福建省科学技术三等奖 2 项. 出版著作和教材 6 部, 发表学术论文 70 余篇.

2011 国际信息技术与应用论坛
2011 International Forum on Information Technology and Applications(IFITA 2011)

所有 IFITA 2011 投稿论文,要反映在信息技术与应用及相关领域中未发表的有新意或创见的技术、工程、理论、方法、应用及其它相关领域研究成果,其工作语言为“英文/中文”. IFITA 2011 的相关主题包括(但不限于):

1. 网格计算,分布计算,并行计算,服务计算,移动计算,普适计算,对等计算,金融计算,经管计算,生物信息计算,生物灵感计算,情感计算, 科学计算,高性能计算,计算几何,数学建模,计算方法,算法研究.

2. 通讯工程,软件工程,人工智能,模式识别,知识发现,数据挖掘,数据库技术,智能代理,逆向工程,信息隐藏,数据仓库,中文处理,虚拟现实,互联网络,移动商务,电子商务,电子政务,电子医疗,工程管理,服务管理,知识管理.

3. 物联网技术,机器人技术,计算机视觉,多媒体技术,传感器网络,嵌入式系统,理论计算机,粗糙集技术,Web 智能,自然语言处理,自然语言理解,语音处理,计算机辅助设计,计算机辅助制造,计算机辅助管理,计算机辅助制造,产业应用、政策与发展,企业信息化.

4. 管理信息系统,决策支持系统,地理信息系统,信息安全技术,数字水印,搜索引擎技术,图形图象处理,商务智能,遥测遥控遥感,环境科学信息技术,管理科学信息技术,数学建模与模型计算,计算机应用技术,通讯工程应用技术,自动化技术与工程.

5. 基于或利用信息技术与应用的教学思想、教学理论、教学方法、教学研究、教学改革、教学创新、教学探索、教学经验等.

会议网站:http://www.iitaa.com/ifita/english/index.asp.