

基于遗传算法的粗糙集属性约简算法

李伟生, 易 哲

(重庆邮电大学 计算机科学与技术学院, 重庆 400065)

摘 要: 针对目前粗糙集遗传约简算法不能确保得到约简的不足, 分别提出基于二进制编码基因和符号编码基因的遗传约简算法. 基于二进制编码基因的遗传算法加入修正算子以确保遗传算法在可行解的空间搜索. 基于符号编码基因的遗传算法融合交叉算子和变异算子以降低遗传算法的复杂度. 实验结果表明, 两种编码方式的遗传约简算法都能确保得到约简.

关键词: 粗糙集; 约简; 遗传算法; 二进制编码基因; 符号编码基因

中图分类号: TP18

文献标识码: A

文章编号: 1000-7180(2010)03-0071-04

Rough Set Attribute Reduction Algorithm Based on GA

LI Wei-sheng, YI Zhe

(College of Computer Science and Technology, Chongqing University of Posts
and Telecommunications, Chongqing 400065, China)

Abstract: To remove the drawbacks that the rough set genetic reduction algorithm can not ensure to gain reductions, this paper presents two novel genetic reduction algorithms which are based on binary genes and character genes, respectively. The binary genes-based genetic algorithm is joined a modified operator to insure the algorithm to search in the possible solution space. The character genes-based genetic algorithm integrates the crossover operator and mutation operator for reducing the complexity of GA. The experiments results show that both of the genetic reduction algorithms can ensure reduction.

Key words: rough set; reduction; genetic algorithm; binary genes; character genes

1 引言

粗糙集理论^[1]是由波兰华沙理工大学 Pawlak 教授于 1982 年提出的一种研究不完整、不确定知识和数据的表达、学习归纳的理论方法. 该方法在人工智能、信息安全、数据挖掘得到了广泛的应用^[2].

属性约简是粗糙集理论的核心内容, 求解粗糙集最小约简已经被证明是 NP-hard 问题^[3]. 这些算法在数据量较小的情况下性能非常优异, 但对于海量数据的处理能力还不足.

遗传算法是一种非常有效的全局搜索方法, 因其在求解 NP-hard 问题所表现出来的能力, 有一些学者将遗传算法用于解决粗糙集中属性约简问题.

这种方法最先是由 Wroblewski 在文献[4]中提出, 国内学者做了不少研究改进这种算法^[5-6], 这些研究主要集中在改进算法的复杂度和加入启发式信息方面. 这些方法只能尽量逼近最优解, 并不能确保得到的是约简. 文中针对该不足, 从约简的定义出发, 对遗传算法进行局部优化, 提出了分别基于二进制编码和符号编码的遗传约简算法. 实验结果表明, 两种编码的算法都能确保得到约简.

2 粗糙集基本理论

定义 1(决策表) 决策表^[2] S 是一个四元组 $S = \langle U, R, V, f \rangle$, 其中 U 是一个非空的有限对象集, 称为论域; $R = C \cup D$ 属性集合, 子集 C 和 D 分

收稿日期: 2009-03-18; 修回日期: 2009-05-18

基金项目: 国家自然科学基金项目(60842003)

别称为条件属性集和结果属性集, V 是属性值的集合; $f: U \times R \rightarrow V$ 是一个信息函数, 它指定 U 中每一个对象 x 的属性值。

定义 2(不可分辨关系) 在决策表 S 中, 对于每个属性子集 $B \subseteq R$, 定义一个不可分辨关系 $IND(B)$, 即

$$IND(B) = \{(x, y) \mid (x, y) \in U^2, \forall b \in B(b(x) = b(y))\}$$

定义 3(下近似集) 对每个概念 X (样例子集) 和不可分辨关系 B , 包含于 X 中的最大可定义集都是根据 B 能够确定的, 称为 X 的下近似集, 记为 $B_-(X)$ 。

定义 4(正域) 设 U 为一个论域, P 和 Q 为定义在 U 上的两个等价关系簇, Q 的 P 正域记为 $POS_P(Q)$, 定义为

$$POS_P(Q) = \bigcup_{r \in U/Q} P_-(X)$$

定义 5(约简) 设 U 为一个论域, P 和 Q 为定义在 U 上的两个等价关系簇, 若 P 的 Q 独立子集 $S \subseteq P$ 有 $POS_S(Q) = POS_P(Q)$, 则称 S 为 P 的 Q 约简。

3 二进制编码的遗传约简算法

3.1 算法描述

本算法采用染色体长度为条件属性个数的二进制编码, 这样很容易描述条件属性中属性的取舍情况: 1 表示对应的条件属性被选取了, 0 表示对应的条件属性没有被选取。例如: 假设有 5 个条件属性 $\{a_1, a_2, a_3, a_4, a_5\}$, 有一个可能的约简为 $\{a_1, a_2, a_4\}$, 则它应表示为 11010。算法描述如下:

步骤 1: 随机产生初始种群 $pop(t)$;

步骤 2: 对 $pop(t)$ 适应值进行评估; 判断是否符合中止条件, 符合则退出, 不符合则转入步骤 3;

步骤 3: 从 $pop(t)$ 中选择下一代 $pop(t+1)$;

步骤 4: 对 $pop(t+1)$ 进行交叉操作;

步骤 5: 对 $pop(t+1)$ 进行变异操作;

步骤 6: 对 $pop(t+1)$ 进行修正操作;

步骤 7: $t = t + 1$, 转步骤 2。

3.2 适应度函数

适应度函数是评价染色体适应性的唯一确定性指标, 它的选取决定了遗传算法的进化方向。属性约简要求得到的属性个数尽可能少, 并且约简后属性集的正域要求和原始属性集的正域相等。在本算法中, 变异算子后加了修正算子, 可以确保最终得到的属性集的正域等于原始正域, 因此适应度函数可以

只考虑属性的个数, 定义为: $F(\tau) = 1/L_\tau$, 其中 L_τ 为染色体中编码为 1 的基因个数。

3.3 选择算子

对父代种群中的个体, 按照各自的适应度在整个种群的个体适应度的总和中所占的比例, 采用轮盘赌方式选择。并辅以最优化保存策略。最优化保存策略在得到新一代个体之后, 如果其中最坏个体(适应值最小)的适应值小于上一代最好个体(适应值最大)的适应值, 则用上一代最好的个体代替新一代最坏的个体。

3.4 交叉算子和变异算子

交叉采用单点交叉, 以一定的概率选择个体参与交叉。对于参与交叉的两个父辈个体, 随机选取交叉点, 然后对交叉后的部分子串进行交换, 即产生下一个代个体。变异操作中, 以一定的概率选择个体进行变异, 随机地选择参与变异的个体的位, 对该位取反。

3.5 修正算子

对每条染色体做如下操作:

步骤 1: 从第一位开始, 将基因位为 0 的基因置为 1, 直到得到的属性集的正域等于原始属性集的正域。

步骤 2: 从第一位到最后一位, 将基因位为 1 的基因置为 0, 如果得到的属性集的正域等于原始属性集的正域, 则继续下一位的操作, 否则, 重新将此基因置为 1。

修正算子确保每一个染色体都是一个可能的约简, 使遗传算法的搜索始终在可行解空间进行。

4 符号编码的遗传约简算法

本算法采用符号编码方式, 将变异操作融入交叉算子中, 3.1 节中的算法去掉步骤 5 和步骤 6 即为本算法的流程。用遗传算法求解约简的操作可以演变为寻找合适的属性排序的操作。例如: 决策表有 5 个条件属性 $\{a_1, a_2, a_3, a_4, a_5\}$, 那么一个简单的编码可以是 12345。假设这个决策表的约简是 $\{a_1, a_2, a_4\}$, 那么用遗传算法的目标是找到形如 12435 或者 21453 这样的序列。然后配合贪心算法, 就可以求得约简。依靠遗传算法全局寻优, 避免一般属性约简算法中的组合爆炸搜索。贪心算法描述如下:

步骤 1: R 为属性的集合, $(b_1 \cdots b_N) = \tau(a_1 \cdots a_N)$ 为属性的一个有序队列, 以数列 τ 表示顺序;

步骤 2: 对 $i = 1$ 到 N , 执行步骤 3 和步骤 4;

步骤 3: $R \leftarrow R - b_i$;

步骤4:如果不是约简, $R = R \cup b_i$.

在步骤4中,判断是否为约简采用的计算属性集的正域是否和原始属性集正域相等的方法. N 为条件属性个数. 在初始属性集里,依次减去每个属性,如果减去该属性后的得到属性集不是约简,则重新加入刚才减去的属性. 贪心算法得到的是一个可能的约简结果,但不一定是最小约简. 有了上述算法之后,剩下的工作就是找到一个合适的属性排序,这个工作就由遗传算法来完成. 一个长度为 N 的排序 τ 就对应于遗传算法中的一个染色体.

4.1 适应度函数和选择算子

本算法中定义每个排序 τ 的适应度函数为: $F(\tau) = 1/L_\tau$, 其中 L_τ 为贪心算法得到的结果中属性的个数. 选择算子和3.2节中的选择算子相同.

4.2 融合变异操作的交叉算子

为了解决在二进制编码中的单点或多点交叉策略在符号编码中出现非法染色体的问题,在文献[6]中,Goldberg提出了一种部分匹配交叉(Partially Matched Crossover, PMX)算子,PMX操作先依据均匀随机分布产生两个交叉点,定义这两个交叉点之间的距离为一匹配区域,适应位置交换策略交换两个父串的匹配区域. 比如:

$A = 3\ 1\ 2\ |0\ 4\ 5\ |6$

$B = 4\ 0\ 5\ |3\ 2\ 1\ |6$

首先交换匹配区域:

$A_1 = 3\ 1\ 2\ |3\ 2\ 1\ |6$

$B_1 = 4\ 0\ 5\ |0\ 4\ 5\ |6$

对于 A_1 和 B_1 匹配区域外出现的遍历重复,以及匹配区域内的位置映射关系,逐一进行交换. 例如 A_1 前面的3被换成0,因为在匹配区域内, A_1 的3对应与 B_1 的0.

$A_1 = 0\ 5\ 4\ |3\ 2\ 1\ |6$

$B_1 = 2\ 3\ 1\ |0\ 4\ 5\ |6$

另外可能出现下面这种特殊情况:

$A = 5\ 1\ 2\ |0\ 4\ 3\ |6$

$B = 4\ 0\ 5\ |3\ 2\ 1\ |6$

交换匹配区:

$A_1 = 5\ 1\ 2\ |3\ 2\ 1\ |6$

$B_1 = 4\ 0\ 5\ |0\ 4\ 3\ |6$

当交换重复数字时,比如 A_1 第2个位置的1重复,而匹配区中1对应于 B_1 的3,如果把1和3交换,仍然重复,因为3也在 A_1 的匹配区中. 当出现这种情况时,继续寻找3在 B_1 中的对应值,此处为0,然后把0和1交换. 如果0仍然重复,就继续寻找直到

找到不重复的数为止.

$A_1 = 5\ 0\ 4\ |3\ 2\ 1\ |6$

$B_1 = 2\ 1\ 5\ |0\ 4\ 3\ |6$

然而这种算子的复杂度比较高. 当出现后面描述的特殊情况时,每交换一次,需要多遍历中间的匹配区域一次^[7].

文中提出了一种能融合变异算子的部分匹配交叉算子,在一般情况下能够使部分匹配,在特殊情况下,只保证其中的一条染色体部分匹配,而另一条染色体相当于施加了一定概率的变异. 算子描述如下:

步骤1:随机选取两个基因座 i 和 j 紧后的位置为交叉点,即将第 $i+1$ 个基因座和第 j 个基因座之间的各个基因座定义为交叉区域.

步骤2:对交叉区域中的每个基因座 p ($p = i+1, i+2, \dots, j$),在个体 T_x 中求出 $t_q^x = t_p^x$ 的基因座 q ,在个体 T_y 中求出 $t_r^y = t_p^y$ 的基因座 r ,然后依次交换基因座值 t_q^x 和 t_r^y , t_p^x 和 t_p^y 和 t_r^y . 所得结果即为 T'_x 和 T'_y .

示例如下:

步骤1:随机找两点做为匹配区域,本例中 $i = 3, j = 6$;数字下面的短横线代表即将进行交换的基因座.

$T_x = 3\ 1\ 2\ |\underline{0}\ 4\ 5\ |6$

$T_y = 4\ \underline{0}\ 5\ |\underline{3}\ 2\ 1\ |6$

步骤2:将 T_x 中的0和3对换; T_y 中同样也是0和3对换. 交换后结果如下:

$T_x = 0\ 1\ 2\ |\underline{3}\ 4\ 5\ |6$

$T_y = 4\ 3\ 5\ |\underline{0}\ 2\ 1\ |6$

步骤3:将 T_x 中的2和4对换; T_y 中同样也是2和4对换. 交换后结果如下:

$T_x = 0\ \underline{1}\ 4\ |3\ 2\ \underline{5}\ |6$

$T_y = 2\ 3\ \underline{5}\ |0\ 4\ \underline{1}\ |6$

步骤4:将 T_x 中的1和5对换; T_y 中同样也是1和5对换. 交换后结果如下:

$T'_x = 0\ 5\ 4\ |3\ 2\ 1\ |6$

$T'_y = 2\ 3\ 1\ |0\ 4\ 5\ |6$

交换完成后,可以看到两个染色体中间的部分是匹配的,然而对于前面提到的特殊情况,不能保证中间的交叉部分完全匹配. 示例如下:

$T_x = 5\ 1\ 2\ |\underline{0}\ 4\ \underline{3}\ |6$

$T_y = 4\ \underline{0}\ 5\ |\underline{3}\ 2\ 1\ |6$

$T_x = 5\ 1\ \underline{2}\ |3\ 4\ \underline{0}\ |6$

$T_y = \underline{4}\ 3\ 5\ |0\ \underline{2}\ 1\ |6$

$T_x = 5 \underline{1} 4 \mid 3 \underline{2} 0 \mid 6$

$T_y = 2 \underline{3} 5 \mid 0 \underline{4} 1 \mid 6$

$T'_x = 5 \underline{0} 4 \mid 3 \underline{2} 1 \mid 6$

$T'_y = 2 \underline{3} 5 \mid 1 \underline{4} 0 \mid 6$

对于这种特殊情况,只保证了其中一条染色体的部分匹配了,而对于另外一条染色体相当于施加了一定概率的变异.由于做交叉操作的染色体和匹配区域都是随机选择的,因此变异的产生也是随机的.

5 实验测试

实验分别采用 KDDCup99 的入侵检测数据集和 UCI 机器学习数据集中的多组数据^[8].

5.1 实验测试一

KDDCup99 数据集中的每条网络连接记录包含 41 个条件属性值.从数据集中随机抽取了 7 208 条数据试验.首先对该数据集用基于属性重要性的离散化算法^[2]进行离散化.另外为了实验证明文中提出的算法能确保得到约简,采用了基于可辨识矩阵和逻辑运算的属性约简算法^[2](算法 1)做对比,该算法可以找到所有可能的约简,当有多个约简结果时,选择属性个数最少的作为最终的约简结果.用该算法将实验数据进行属性约简后得到的条件属性集(14 个)为: {2 3 4 8 10 12 16 22 24 25 30 35 37 39}.

遗传约简算法用 VC++ 6.0 编程实现.实验在 P4 3.0GHz,512MB 内存的电脑上运行.在实验中,二进制编码的遗传算法(算法 2)的参数选定如下:初始种群数为 60,交叉概率 0.5,变异概率 0.1.符号编码的遗传算法(算法 3)的交叉算子融合了变异操作,没有进行变异运算.其他参数选定和算法 2 相同.将遗传算法进化代数分别设置为 20,100,300 并重复试验 10 次,取平均值作为结果.对比试验如表 1 所示.

表 1 对比实验(一)

	算法 2			算法 3			文献[4]		
进化代数	20	100	300	20	100	300	20	100	300
约简结果	14	14	14	14	14	14	7	6	5

可以发现,算法 2 和算法 3 的约简结果一直稳定在 14,而文献[4]的算法约简结果随进化代数变化而变化,却始终无法得到正确的约简结果,这正说明了文献[4]中的算法不能确保得到约简.

5.2 实验测试二

采用 UCI 数据集的实验方法与前面描述的完全相同,遗传算法进化代数为 20.数据集大小用记录数目条件属性数目表示.实验结果如表 2 所示.

表 2 对比实验(二)

UCI 数据集	数据集大小	算法 1	算法 2	算法 3	文献[4]
agaricus-lepiota	846×22	12	12	12	6
chess	1 598×36	24	24	24	5
iris	150×4	3	3	3	2

5.3 实验结果分析

(1) 在二进制编码的遗传算法中加入文中提出的修正算子能确保得到约简.

(2) 文中提出的在基于符号编码的遗传算法中的融入变异的交叉算子有效.并且该算法也能确保得到约简.

6 结束语

文中分别提出了基于二进制编码和符号编码的遗传约简算法.在基于二进制编码的遗传算法中,加入了修正算子,修正算子按照约简的定义确保每个染色体都在可行解空间搜索.在基于符号编码的遗传算法中,融合了交叉算子和变异算子,降低了遗传算法复杂度.这两种编码的算法都能确保得到约简,有很大的实际意义.下一步工作为改进二进制编码遗传算法中的修正算子和符号编码中的贪心算法.

参考文献:

- [1] Pawlak Z. Rough sets[J]. International Journal of Information and Computer Science, 1982, 11(5):341-356.
- [2] 李抒音,姚崇怀.基于增量式更新带权差别集合的属性约简算法[J].微电子学与计算机,2007,25(11):68-71.
- [3] 胡峰,王国胤.属性序下的快速约简算法[J].计算机学报,2007,30(6):1429-1435.
- [4] Jakub W. Finding minimal reducts using genetic algorithms [R]. Poland: Warsaw university of Technology, 1995.
- [5] 吕军,冯博琴,李波.基于遗传算法的属性约简[J].微电子学与计算机,2006,23(7):150-153.
- [6] 吕跃进,刘南星,陈磊.一种基于并行遗传算法的粗糙集属性约简[J].计算机科学,2008,35(3):219-221.
- [7] 刘洋,冯博琴,周江卫.一种改进的基于差别矩阵的属性约简算法[J].微电子学与计算机,2007,24(5):133-137.
- [8] 徐章艳,刘作鹏,杨炳儒,等.一个复杂度为 $\max\{|O|C|(|U|,O(|C|2(|U/C|)))\}$ 的快速属性约简算法[J].计算机学报,2006,29(3):391-399.

作者简介:

李伟生 男,(1975-),博士,教授,硕士生导师.研究方向为智能信息处理、模式识别.

易 哲 男,(1984-),硕士研究生.研究方向为智能信息处理.