文章编号:1001 - 9081(2005)11 - 2621 - 03

基于粗糙集理论的增量式规则获取

郭 森,王知衍,吴志成,严和平
(华南理工大学 计算机科学与工程学院,广东 广州 510640)
(Ybbsss1210@126.com)

摘 要:基于粗糙集理论提出了一种新的规则提取算法:基于粗糙集和搜索树的规则提取算法。该算法是以现有规则集中的信息为启发信息,通过对解空间进行宽度优先启发式搜索,产生新规则。以该算法为基础,产生关于新增对象的规则,并对现有规则进行更新。

关键词:粗糙集;搜索树;宽度优先启发式搜索;规则提取
中图分类号: TP182;TP311.131 **文献标识码**:A

Incremental rules extraction based on rough set theory

GUO Sen, WANG Zhi-yan, WU Zhi-cheng, YANG He-ping
(School of Computer Science and Engineering, South China Unviersity of Technology, Guangzhou, Guangdong 510640, *China)*

Abstract: The incremental rules extraction is a focus problem of KDD. A novel rules extraction algorithm which is called "RDBRST" (Rule Derivation Based on Rough set and Search Tree) was proposed. It was one kind of width first heuristic search algorithms. The incremental rules were extracted and the current rule set was updated based on this algorithm. An example was given to illustrate the characteristics of this new incremental algorithm.

Key words: rough set; search tree; width first heuristic search algorithm; rule derivation

0 引言

粗糙集理论[1]从 1982 年诞生以来,已成功应用于机器学习、数据挖掘、模式识别等领域,现有的基于粗糙集理论的规则提取算法可以很有效地从知识库提取规则。但它需要一次性地处理全部知识。而在实际应用中人们获取知识的过程往往是渐进的,规则集中的规则需要进行不断被更新,可以想像,如果对所有新增记录都重新进行一遍操作的话,对资源的占用是极其巨大的。现在已有很多人在做这方面的研究,并取得了一些成果[2~5]。但现有算法都是基于区分矩阵的,或者是基于改进的区分矩阵,这些算法的共同特点是在中间过程需要 $n \times n$ 的空间来存储区分矩阵(n 为决策表中对象总数),因此该类算法的空间复杂度极高。本文提出了一种新的算法,它不需要经过创建区分矩阵这个中间过程,因而相对现有算法而言具有较低的空间复杂度。另外,在规则产生及更新过程中,它充分利用现有规则集中的信息,可以减少规则产生过程中的搜索空间范围,从而进一步降低复杂度。

1 基本概念

1.1 粗糙集理论的基本概念

粗糙集理论[1,6]中的信息存储在一个二维决策表中,$S = (U, A, V, f)$。其中 U 为一有限非空的对象集合;A 为属性集,$A = C \cup D$,其中 C 为条件属性,D 为决策属性;V 为 A 的值域;f 为一信息函数:$f: U \times A \rightarrow V$。

对于 $P \subseteq A$,我们可以定义一个 P 上的不可区分关系:

$ind(P) = \{(x, y) \in U \times U \mid \forall a \in P, f(x, a) = f(y, a)\}$

对于 $x \in U$,我们用集合 $[x]_P = \{y \in U \mid (x, y) \in ind(P)\}$ 来表示包含元素 x 的等价类,$[x]_P = \bigcap_{p \in P} [x]_P$。

对于一个等价类集 $[x]_P$,我们用 $Des([x]_P) = (a = v)$ 来对它的特性进行描绘,其中 $a \in P$, $v \in V$。

我们用 rx 来表示由该决策表提取出的规则:

$rx: Des([x]_C) \Rightarrow Des([x]_D)$。

相容性的概念:用 $rx \mid C$ 来表示规则 rx 的前项,用 $rx \mid D$ 来表示后项,对于 x, $y \in U$, $x \neq y$, 如果由 $rx \mid C = ry \mid C$ 可以得出 $rx \mid D = ry \mid D$ 的结论,则称决策规则 rx 是相容的。

属性约简的概念可定义为:如果 $[x]_{\hat{C}} \subseteq [x]_D$, $\hat{C} \subseteq A$,存在 $c \in \hat{C}$,并且 $[x]_{\hat{C}-\{c\}} \subseteq [x]_D$,则我们称属性 c 为可约简属性。

1.2 搜索树的概念

搜索树是在问题解空间的搜索过程中产生的节点和指针构成的一棵隐式定义的状态空间树的子树[7]。它是一棵有向的无环树,树中的根节点没有父节点,所有其他节点有且只有 1 个父节点,每个节点可以有 1 个或多个子节点,或者没有子节点。如果节点没有子节点,称其为叶节点;其他的称为内部节点。

本文采用的搜索算法是宽度优先启发式算法,其基本原理是:以接近起始节点的程度依次逐层扩展节点,按启发信息对节点进行重排和剪枝,直至得到问题解或空间搜索穷尽。

2 基于粗糙集和搜索树的规则提取算法

本文根据粗糙集理论的基本原理和宽度优先搜索算法,

收稿日期:2005 - 05 - 16 **基金项目**:国家科技型中小企业技术创新基金项目(02C26214400224);广东省计划项目(2002A1020104)
作者简介:郭森(1972 -),男,江西人,博士研究生,主要研究方向:模式识别、数据挖掘、粗糙集; 王知衍(1955 -),男,浙江人,教授,博士生导师,主要研究方向:虚拟现实技术、模式识别、图像处理.

基于决策表中对象 x 的相关矩阵,以现有规则集为启发信息,对解空间进行启发式搜索,从而得到关于对象 x 的规则。

2.1 相关性及相关矩阵的定义

定义 1 $S=(U, A, V, f)$ 为一决策表,x 和 y 均为 U 中的对象。如果 y 的存在与否不会对关于对象 x 的规则造成影响,则称 y 与 x 无关。否则,则称 y 与 x 相关。

定义 2 由所有与 x 相关的对象及对象 x 本身组成的矩阵被称为对象 x 的相关矩阵。

定理 1 满足下面两个条件之一,则 x 为与 y 无关对象,反之则为相关对象。

1) $\forall c \in C, c(x) \neq c(y)$;

2) $\forall d \in D, d(x) \neq d(y)$;

证明:设 $S=(U, A, V, f)$,由决策表 S 中的关于对象 x 的规则:rx:$Des([x]_{\hat{C}}) \Rightarrow Des([x]_D)$,即 $[x]_{\hat{C}} \subseteq [x]_D$。$U$ 中不包含对象 y。另有一个新的决策表 $S'=(U', A, V, f)$,其中 $U'=U \cup \{y\}$,在新的决策表 S' 中对应于上面 rx 的等价类的定义为 $[x]'_{\hat{C}}$ 和 $[x]'_D$。

1) 因为 $\forall c \in C, c(x) \neq c(y)$,根据等价类的定义,$y \notin [x]'_{\hat{C}}$,即 S' 中增加对象 y 后,包含 x 的等价类保持不变,$[x]'_{\hat{C}}=[x]_{\hat{C}}$;

如果 $d(x)=d(y)$,则 $[x]'_D=[x]_D \cup \{y\}$,即 $[x]_D \subset [x]'_D$;

如果 $d(x) \neq d(y)$,则 $[x]_D=[x]'_D$;

由以上分析我们可以得出 $[x]_D \subseteq [x]'_D$。

$\because [x]_{\hat{C}} \subset [x]'_D$

$\therefore [x]'_{\hat{C}} \subseteq [x]_D \subseteq [x]'_D$

$\therefore [x]'_{\hat{C}} \subseteq [x]'_D$。因此,由不论对象 y 存在与否,关于对象 x 的规则不变,即 x 与 y 无关。

(2)$\because \forall d \in D, d(x)=d(y) \therefore [x]'_D=[x]_D \cup \{y\}$。

如果对象 y 满足 $\hat{c}(x)=\hat{c}(y)$ 的条件,则 $[x]'_{\hat{C}}=[x]_{\hat{C}} \cup \{y\}$;如果不满足,则 $[x]'_{\hat{C}}=[x]_{\hat{C}}$。因此 $[x]'_{\hat{C}} \subseteq ([x]_{\hat{C}} \cup \{y\})$。

$\because [x]_{\hat{C}} \subseteq [x]_D$

$\therefore [x]'_{\hat{C}} \subseteq ([x]'_{\hat{C}} \cup \{y\}) \subseteq ([x]_D \cup \{y\})$

又 $\because [x]'_D=[x]_D \cup \{y\} \therefore [x]'_{\hat{C}} \subseteq ([x]'_D$。因此,由不论对象 y 存在与否,关于对象 x 的规则不变,即 x 与 y 无关。

2.2 RDBRST 算法实现

RDBRST 算法如下:

输入:1) $S=(U, A, V, f)$,$A=C \cup D$,

$C=(c1, c2, \cdots, cm\}$, $D=\{d\}$

2) 现有规则集 R;

3) 新增对象:$u=\{u1, u2, \cdots, un, du\}$

4) 决策表 S 中关于 u 的相关矩阵;

5) 初始化 *open* 集合:

$open=\{Dex([u]_{C1}), Des([u]_{C2}, \cdots, Des([u]_{Cm})\}$

输出:由 u 生成的新规则 R'。

开始:

1) 从规则集 R 中提取规则后件为 $Des([u]_D)$ 的规则,放入 R_1 中。

2) 如果集合 *open* 为空,则结束,否则继续。

3) 把 *open* 中的第一个节点移出,放入 *closed* 的扩展节点集合中。

4) 对 *closed* 中的节点产生后继节点,设该节点为 $Des([u]_{\hat{C}})$,则产生后续节点的方法为:对所有 $c' \in C$ 并且 $c' \notin \hat{C}$,则 $C'=\{c'\} \cup \hat{C}$,$closed=closed \cup \{Des([u])_{C'}\}$。

5) 对 *closed* 集合中的节点进行筛选。设 *closed* 中的节点为 $Dex([u]_{C'})$,只要满足下面条件中的一个,将其从 *closed* 集合中删除。

a) $Dex([u]_{C'})$ 满足规则集 R_1 中某项规则的前件,即在 R_1 集中有一个规则前件 $Dex([u]_{\bar{C}})$,存在 $Dex([u]_{\bar{C}}) \subseteq Dex([u]_{C'})$ 的关系。

b) 与该节点相同的节点已存在于 *open* 集合或者 *closed* 集合中。

c) 基于相关矩阵,存在 $Dex([u]_{C'}) \subseteq Dex([u]_D)$。

对于符合条件 C 的节点,

$R'=R' \cup \{Dex([u]_{C'}) \Rightarrow Dex([u]_D)\}$,

$R_1=R_1 \cup \{Dex([u]_{C'}) \Rightarrow Dex([u]_D)\}$。

6) 将 *closed* 中的所有节点放入 *open* 集合末端。

7) 转 2)。

结束。

现以算例 1 为例来说明。表 1 为决策表,$C1$ 至 $C4$ 为条件属性,D 为决策属性。表 2 为基于该表提取的已有规则集。表 3 为关于对象 u 的相关矩阵。新增对象 $u=(2, 2, 3, 1, 3)$;*Open* 集合的初始值为 $\{C1=2, C2=3, C3=3, C4=1\}$;输出结果为 $\{C1_2C3_3 \Rightarrow D_3, C2_2C3_3 \Rightarrow D_3$。图 1 为算例 1 的解空间搜索示意图。

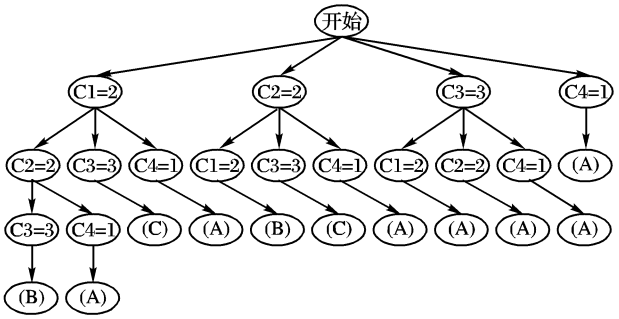

图 1 RDBRST 算法搜索示意图

注:(A),(B),(C)分别为满足算法中的 A,B,C 三个条件

表 1 决策表(S)

$C1$	$C2$	$C3$	$C4$	D
2	1	2	3	1
2	2	1	2	1
1	3	1	2	2
3	3	1	3	2
1	1	3	1	3

表 2 规则集(R)

rx	rx | $C \Rightarrow rx$ | D	rx	rx | $C \Rightarrow rx$ | D
1	$C1_2 \Rightarrow D_1$	8	$C1_1C4_2 \Rightarrow D_2$
2	$C3_2 \Rightarrow D_1$	9	$C1_3 \Rightarrow D_2$
3	$C2_1C4_3 \Rightarrow D_1$	10	$C3_1C4_3 \Rightarrow D_2$
4	$C3_1C4_2 \Rightarrow D_1$	11	$C3_3 \Rightarrow D_3$
5	$C2_2 \Rightarrow D_1$	12	$C1_1C2_1 \Rightarrow D_3$
6	$C2_3 \Rightarrow D_2$	13	$C4_1 \Rightarrow D_3$
7	$C1_1C3_1 \Rightarrow D_2$	14	

表 3　相关矩阵(S')

$C1$	$C2$	$C3$	$C4$	D
2	1	2	3	1
2	2	1	2	1
2	2	3	1	3

2.3　RDBRST 算法复杂度分析

设决策表 $S = (U, A, V, f)$, $A = C \cup D$, $C = \{c1, c2, \cdots, cm\}$, $D = \{d\}$, 新增知识:$u = \{u1, u2, \cdots, um, du\}$。RDBRST 算法的空间复杂度其最好结果为 m,即搜索仅进行到第一层;最坏结果是完成整个搜索空间的遍历,其空间复杂度为搜索树所扩展的各层节点之和。

由 RDBRST 算法构造的搜索树第 k 层的节点数为:$closed(k) = \prod_{i=1}^{k}(m+1-i)$,整个搜索树空间的总节点,即该算法空间复杂度最差结果为:$O\left(\sum_{k=1}^{m} closed(k)\right)$。由于在实际应用中,条件属性的个数要远少于决策表中的对象数,因此 RDBRST 算法的空间复杂度要优于现有的基于区分矩阵的增量规则提取算法。

该算法的时间复杂度主要由三个方面决定,即搜索树的节点数目、相关矩阵大小、规则表中规则后件为 $Des([u]_D)$ 的规则数。

设规则集中符合条件的规则数为 p,相关矩阵包含 q 个对象,搜索进行到第 k 层时 $closed$ 和 $open$ 集合中的节点数分别为 $closed(k)$ 和 $open(k)$,算法的时间复杂最好的情况为 $m \times (p+q)$,最坏的情况为对整个解空间进行穷尽搜索,时间复杂度为:$O\left(\sum_{k=1}^{m}(open(k-1)+p+q) \times closed(k)\right)$。由此可见,在这种情况下的时间复杂度依然很大。但在算法中我们采用现有规则集中的相关规则作为启发信息,可以减少 $closed$ 和 $open$ 集合中的节点数,因而可以大大降低时间复杂度。

3　基于粗糙集理论的增量式规则提取算法

本文所提出的增量式规则提取算法分为两个步骤:1) 对现有规则的更新; 2) 增加新规则。

决策表中加入新的对象,现有规则集中的部分规则需要进行更新。

$S = (U, A, V, f)$ 为一决策表,其中 $A = C \cup D$,现有规则集为 R,x 为 U 中的对象,u 为新增对象,由定理 1 我们可得如下推论:$\exists c \in C$, $c(x) = c(u)$ 并且 $d(x) \neq d(u)$,则由对象 x 产生的规则需要被更新。(证明方法同定理 1,略)

规则更新算法如下:

输入:1) $S = (U, A, V, f)$, $A = C \cup D$, $C = \{c1, c2, \cdots, cm\}$, $D = \{d\}$;

2) 现有规则集 R;

3) 新增对象:$u = \{u1, u2, \cdots, um, du\}$;

4) 关于对象 u 的相关矩阵 S'。

输出:更新后的规则集 R。

开始:

1) 从对象 u 的相关矩阵 S' 中读取一个对象 x (对象 u 除外)。

2) 对相关规则进行更新。更新方法如下:

a) $\hat{C} = \{c \mid c \in C \wedge c(x) = c(u) \wedge d(x) \neq d(u)\}$。

b) 现有规则集如果存在某项规则的前件 $Des([u]_{\hat{C}})$ 满足 $Des([u]_{\hat{C}}) \subseteq Des([u]_{\hat{C}})$,则删除该规则。

c) 生成决策表 S 中关于对象 x 的相关矩阵,$open$ 集合的初始值为 $Des([u]_{\hat{C}})$,调用 RDBRST 算法产生新规则并将新规则加入到规则集 R 中。

3) 如果 S' 扫描完毕则结束。否则读取下一个对象(对象 u 除外),转 2)。

结束。

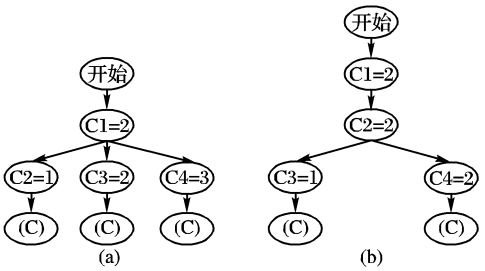

图 2　规则更新搜索图

注:(C) 为满足 RDBRST 算法中的 C 条件

按照规则更新算法,规则集中规则 $C1_2 \Rightarrow D_1$ 被更新为三条规则:$C1_2C2_1 \Rightarrow D_1$, $C1_2C3_2 \Rightarrow D_1$ 和 $C1_2C4_3 \Rightarrow D_1$(见图 2(a)); 规则 $C2_2 \Rightarrow D_1$ 被更新为 $C1_2C2_2C3_1 \Rightarrow D_1$ 和 $C1_2C2_2C4_2 \Rightarrow D_1$(见图 2(b))。新规则的提取是在更新后的规则集的基础上,直接调用 RDBRST 算法生成的。

表 4　加入新对象后的规则集合表

rx	$rx \mid C \Rightarrow rx \mid D$	rx	$rx \mid C \Rightarrow rx \mid D$
1	$C1_2C2_1 \Rightarrow D_1$	10	$C1_1C3_1 \Rightarrow D_2$
2	$C1_2C3_2 \Rightarrow D_1$	11	$C1_1C4_2 \Rightarrow D_2$
3	$C1_2C4_3 \Rightarrow D_1$	12	$C1_3 \Rightarrow D_2$
4	$C3_2 \Rightarrow D_1$	13	$C3_1C4_3 \Rightarrow D_2$
5	$C2_1C4_3 \Rightarrow D_1$	14	$C3_3 \Rightarrow D_3$
6	$C3_1C4_2 \Rightarrow D_1$	15	$C1_1C2_1 \Rightarrow D_3$
7	$C1_2C2_2C3_1 \Rightarrow D_1$	16	$C4_1 \Rightarrow D_3$
8	$C1_2C2_2C4_2 \Rightarrow D_1$	17	$C1_2C3_3 \Rightarrow D_3$
9	$C2_3 \Rightarrow D_2$	18	$C2_2C3_3 \Rightarrow D_3$

4　结语

相对于现有的基于粗糙集的增量规则提取算法而言,本文提出的 RDBRST 算法省去了创建处理区分矩阵的中间过程,具有较低的空间复杂度,特别是对于大规模样本的规则提取中,它的优势更加明显。该算法的时间复杂度受到决策表结构、现有规则集及相关矩阵的影响。

参考文献:

[1] PAWLAK Z. Rough set[J]. International Journal of Computer and information science, 1982, 11(5): 341 - 356.

[2] BUSSE G, LERS JW. A system of knowledge discovery based on Rough sets[A]. Proceeding of 5th international workshop RSFD96 [C], 1996. 443 - 444.

[3] 於东军, 王士同. 一种增量式规则提取算法[J]. 小型微型计算机, 2004, (1): 79 - 81.

[4] 安利平, 吴育华. 增量式获取规则的粗糙集方法[J]. 南开大学学报(自然科学版), 2003, (6): 98 - 103.

[5] 李滔, 王俊普. 一种基于粗糙集的网页分类方法[J]. 小型微型计算机系统, 2003, (3): 520 - 522.

[6] 刘清. 粗糙集理论及粗糙集推理[M]. 北京: 科学出版社, 2001.

[7] 蔡自兴, 徐光右. 人工智能及其应用[M]. 北京: 清华大学出版社, 1996.