分类号 TP391 密级 公开

UDC 编号

学 位 论 文

THESIS FOR DEGREE

基于经典粗糙集理论的规则生成算法研究

作者 刘 喆

申请学位级别 工学硕士 专业名称 计算机软件与理论

辽宁工程技术大学

LIAONING TECHNICAL UNIVERSITY

论 文 题 目：基于经典粗糙集理论的规则生成算法研究

论文英文题目：Research on Rules Generation based o Classic Rough Set Theory

姓名 刘 喆 （07 级）

专业 计算机软件与理论 研究方向 数据挖掘

导师 李义杰 职称 教授

论 文

完成日期 答辩日期

授予学位日期：

辽宁工程技术大学

学位论文原创性声明

本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果。其他同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

本人学位论文与资料若有不实，愿意承担一切的法律责任。

学位论文作者签名：_____

年　　月　　日

关于论文使用授权的说明

本人完全了解学校有关保留、使用学位论文的规定，即 :学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅，学校可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编本学位论文。

保密的学位论文在解密后应遵守此协议

学位论文作者签名：_____　　导师签名：_____

年　月　日　　年　月　日

摘要

关键词： 数据挖掘； 粗糙集； 约简； 覆盖

Abstract

hhhhhhhhhhhhhhhello

Keywords: Data Mining; Rough Set; Reduct; Loan

目 录

1 绪论 **1**
1.1 选题的目的和意义 1
1.2 国内外研究现状 1
1.3 研究的主要内容和方法 3
1.4 论文的组织结构 3

2 粗糙集基础理论 **4**
2.1 基本概念 4
2.2 知识约简 6
2.3 决策表模型 8
2.4 两类决策表 8
2.5 支持子集和支持度 9
2.6 多条件支持度 10
2.7 重要性和相对核 10
2.8 相对约简 10
2.9 规则生成 11
2.10 粗糙集应用中存在的辩证关系 12
2.10.1 键属性 12
2.10.2 规则的确定性和条数 13

3 对于经典粗糙集理论的几点思考 **14**
3.1 划分不是必须的 14
3.1.1 划分的实质是什么 14
3.1.2 必要性分析 14
3.2 计算全部下近似不是必须的 15
3.3 精确下近似（或者说完全下近似）是不是必须的 15
3.4 重要性，支持度等等概念是不是必须的 15
3.5 约简是不是必须的 15
3.6 算法的复杂性分析和证明 15
3.7 其他算法的复杂性分析及证明或引用证明 15
3.8 关于覆盖率的证明 15
3.9 关于平均规则长度的对比及证明 15

3.10 关于属性排序的问题 15
3.11 要解释的概念 16
3.12 对规则生成方法的再认识 16
3.13 对“下近似”概念的再认识 16
3.14 关于新的想法 17

4 新算法和相关的理论依据 18
4.1 覆盖率分析 20
4.2 时间复杂度分析 22
4.3 规则长度分析 26
4.4 实验环境 27
4.5 重要代码分析 28

5 结论和展望 30
5.1 实验结果 30
5.2 结论 30
5.3 展望 30

参考文献 33

致谢 37

攻读学位期间发表的学术论文及科研成果 38

1 绪论

1.1 选题的目的和意义

电子计算机的发展，尤其是互联网的发展，给人们提供了海量的信息。现在，即使是一个不太出名的网络结点，日访问量都需要以T来计算。面对如此巨大的数据量，虽然我们有大型数据库来存储信息，并且存储设备也一直在按照摩尔定律发展，但是我们的头脑却不能按这个速度发展，还是很快地被这海量的数据所淹没。出于从如此庞大的数据集中发现人们真正感兴趣的信息这一目的，就产生了“数据挖掘”这一新兴的学科。

数据挖掘的基本思想是简单的，也是有意义的：就是从海量的数据中发现其中存在的规律，可以用“沙里淘金”来形容。

数据挖掘理论和技术发展至今，已经有了很多成熟的模型，比如，分类、聚类、规则生成、趋势预测等等，这些理论和方法也已经成功地应用于金融、市场、风险评测等等商业领域，创造了很高的价值。

在现实生活中，人们往往希望通过已有的经验或数据记录来对未来的事件或者新的事例进行预测，在数据挖掘的相关研究中，这一方面的研究称为“决策支持”。它的典型形式就是通过已有的数据，形成“预测规则”，也就是发现已有数据中存在的规律或模式，从而实现用机器（通常是用电脑）来实现知识发现。可见，规则生成在辅助决策支持中起着举足轻重的作用。

传统的以统计学为基础的数据挖掘算法如关联规则算法，决策树算法等等，在处理确定性问题时，计算速度快，处理精度高。然而生活中更多的是不确定的问题，这一要求推动了数据挖掘中新的领域不断产生。

1.2 国内外研究现状

在上世纪70年代，波兰学者Z.Pawlak和一些波兰科学院、波兰华沙大学的逻辑学家们，一起从事关于信息系统逻辑特性的研究。在

这些研究的基础上，产生了一种新的处理模糊和不确定性知识的数学工具——粗糙集理论[43]。1982 年， Z.Pawlak 发表了经典论文《Rough Sets》[22]，宣告了粗糙集理论的诞生。此后，粗糙集理论引起了许多数学家，逻辑学家和计算机研究人员的兴趣，他们在粗糙集的理论和应用方面作了大量的研究工作。1991 年 Z.Pawlak 的专著 [23] 和 1992 年应用专集的出版，对这一段时期理论和实践工作的成果作了较好的总结，同时促进了粗糙集在各个领域的应用。此后召开的与粗糙集有关的国际会议进一步推动了粗糙集的发展。

和传统的统计相关的数据挖掘算法相比，粗糙集算法有一个显著的优点，即它不需要人为的指定一些先验阈值（比如 [44] 中提到的关联算法中的支持度，置信度），它所要用到的只是原始数据。这样就可以避免不同的阈值得到不同的结果，而且也省去了从经验中得到先验知识的步骤。其主要思想就是在保持分类能力不变的前提下，通过知识约简[5, 13, 14]，导出问题的决策或分类规则。目前，粗糙集理论已被成功地应用于机器产、决策分析、过程控制、模式识别与数据挖掘等领域[45, 55]。经典的粗糙集理论是以“等价关系”为基础来讨论的，后来又发展为其它几种粗糙集理论，如基于覆盖的粗糙集[1, 4, 6, 7, 15, 18, 49, 50, 53]，基于一般关系的粗糙集[55]、S-粗集[54]等等。

在处理不确定性方面，还有模糊集($Fuzzy\ set$)的理论[40–42]，由于它与粗糙集有理论很大的互补性[34]，于是出现了有关于粗糙集和糊糊集相结合的“模糊粗糙集”($fuzzy\ rough\ set$)[33, 39, 58]和“粗糙模糊集”($rougt\ fuzzy\ set$)[3, 17, 28]等等理论。这些理论都各有它们的特点和适用范围，关于相关的应用研究也在迅速开展[57]。

在粗糙集的应用中，通过定义“上近似”和“下近似”，进而可以计算“属性的依赖性”，这个指标可以显示出不同属性之间的依赖关系；还可以计算“属性的重要性”，这个指标可以看出某一属性在整个属性集中的价值和它提供的信息量；还可以计算与原属性集有相同分类能力的“约简”属性集，在不改变系统分类能力的前提下对属性集进行简化，进而可以生成决策规则，辅助决策。

属性约简是粗糙集理论的一大特色，在很多的规则生成算法中，这也是算法的第一步。然而，研究表明，找出所有约简是的的时间复杂性是指数的，而找出最小约简的复杂性也是 $NP-hard$ 的[30, 55]。特别是， Ron Kohavi 和 Brian Frasca 在 [11] 中用实验指出，属性约简得到的核

属性集甚至可能是无用的。

1.3 研究的主要内容和方法

本文在系统地研究了粗糙集的基本理论的基础上，主要研究了经典粗糙集方法中一些概念和操作的必要性，对其进行适当的取舍和提炼，提出了一种新的从大量数据中生成规则的算法。

新算法的预期目标如下：

1. 与经典粗糙集方法分类能力相同或基本相当
2. 与经典粗糙集方法执行效率基本相当或更快
3. 比经典粗糙集方法产生相当数量的或更少的规则数
4. 比经典粗糙集方法产生的规则相当或更短

如果算法实现了以上预期目标，显然我们在一定程度上抓住了数据更本质的特征，更有利于数据信息的使用。

算法的改进总要有参照对象，在本研究中，我们与由波兰华沙大学与挪威科技大学联合开发的 *Rosetta* 软件[19]提供的几个粗糙集的相关算法进行对比。实验的数据取自 *UCI* 机器学习研究所[35]，如 *Iris* 数据集[8](部分数据见表 5.2)， $New-thyroid$ 数据集(表 5.3)，$Yellow-small(balloons)$ 数据集等等。我们的算法实现则采用 *haskell* [10]来实现，利用它的 *lazy* 特性，可以自动实现运算量的最小化，而且它还提供了丰富易用的数据结构和库函数。

1.4 论文的组织结构

本文的接下来的组织结构如下：

第 2 章概述了经典粗糙集理论在规则生成算法方法需要用到的一些基本理论和术语。

第 3 章讨论了经典粗糙集理论和方法中用到的一些基本概念的合理性和可优化的地方，并基本阐述了本文的新算法的基本思想。

第 4 章详细阐述了新算法和它的理论依据，并给出了严格的证明。

第 5 章总结了本文，并对进一步的研究方向给出了建议和看法。

2 粗糙集基础理论

粗糙集理论是一种新的处理模糊和不确定性知识的数学工具。

在上世纪70年代，波兰学者Z.Pawlak和一些波兰科学院，波兰华沙大学的逻辑学家们，一起从事关于信息系统逻辑特性的研究。粗糙集理论就是在这些研究的基础上产生的。1982年，Z.Pawlak发表了经典论文《Rough Sets》，宣告了粗糙集理论的诞生。此后，粗糙集理论引起了许多数学家，逻辑学家和计算机研究人员的兴趣，他们在粗糙集的理论和应用方面作了大量的研究工作。1991年Z.Pawlak的专著[23]和1992年应用专集的出版，对这一段时期理论和实践工作的成果作了较好的总结，同时促进了粗糙集在各个领域的应用。此后召开的与粗糙集有关的国际会议进一步推动了粗糙集的发展。

粗糙集理论的主要思想就是在保持系统分类能力不变的前提下，通过知识约简，导出问题的决策或分类规则。目前，粗糙集理论已被成功地应用于机器学习、决策分析、过程控制、模式识别与数据挖掘等领域[45, 55]。经典的粗糙集理论是以“等价关系”为基础来讨论的，这里，我们也是采用的这种方法。

和传统的统计相关的数据挖掘算法相比，粗糙集算法还有一个显著的优点，即它不需要人为的指定一些先验阈值（比如[44]中提到的关联算法中的支持度，置信度），它所要用到的只是原始数据。这样就可以避免不同的阈值得到不同的结果，而且也省去了从经验中得到先验知识的步骤。

2.1 基本概念

设$U \neq \emptyset$是应用中感兴趣的对象组成的非空有限集合，称为“论域”。对于$\forall X \subseteq U$，称X为U中的一个概念或范畴。空集也认为是一个概念。U中的任何概念放称为“知识”[2, 12, 20, 21, 24–26, 32, 36, 38, 48, 51, 56]。

一个划分$\mathscr{L}$定义为：

定义 2.1. [55]

$$\mathscr{L} = \{X_1, X_2 \dots X_n\}; X_i \subseteq U, X_i \cap X_j = \emptyset, i \neq j; \bigcup_{i=1}^{n} = U. \tag{2.1}$$

U 上的一族划分称为 U 的一个知识库。

设 R 是 U 上的一个等价关系，U/R 表示 U 的所有等价类构成的集合，$[x]_R$ 表示包含元素 $x \in U$ 的 R 等价类。一个知识库就是一个关系系统 $K = (U, \boldsymbol{R})$，其中 U 为非空有限集，称为论域，$\boldsymbol{R}$ 是 U 上的一族等价关系。

若 $P \subseteq \boldsymbol{P}$，且 $P \neq \emptyset$，则 $\bigcap P$ 也是一个等价关系，称为 $\boldsymbol{P}$ 上的不可区分关系，记为 $ind(\boldsymbol{P})$，且有

$$[x]_{ind(\boldsymbol{P})} = \bigcap_{R \in \boldsymbol{P}} [x]_R \tag{2.2}$$

这样，$U/ind(\boldsymbol{P})$ 表示与等价关系族 $\boldsymbol{P}$ 相关的知识，称为 K 中关于 U 的 $\boldsymbol{P}$ 基本知识。有时用 $U/\boldsymbol{P}$ 来代替 $U/ind(\boldsymbol{P})$，$ind(\boldsymbol{P})$ 的等价类称为知识 $\boldsymbol{P}$ 的基本概念或基本范畴。

令 $X \subseteq U$，R 为 U 上的一个等价关系，当 X 能表达为某些 R 基本范畴的并时，称 X 为 R 可定义的；否则称 X 为 R 不可定义的。R 可定义集也称作 R 精确集，而 R 不可定义集也称为 R 非精确集或 R 粗糙集（rough set）。

对于粗糙集可以使用两个精确集，即粗糙集的上近似和下近似来刻画。

对于知识库 $K = (U, \boldsymbol{R})$，对于每个子集 $X \subseteq U$ 和一个等价关系 $R \in ind(K)$，我们定义如下两个子集：

定义 2.2. [55]

$$\underline{R}X = \bigcup \{Y \in U/R \mid Y \subseteq X\} \tag{2.3}$$

定义 2.3. [55]

$$\overline{R}(X) = \bigcup \{Y \in U/R \mid Y \cap X \neq \emptyset\} \tag{2.4}$$

分别称它们为 X 的 R 下近似集和 R 上近似集[27, 31, 46, 47]。

或者这样表示：

$$\underline{R}X = \{x \in U \mid [x]_R \subseteq X\} \tag{2.5}$$

$$\overline{R}X = \{x \in U \mid [x]_R \cap X \neq \emptyset\} \tag{2.6}$$

如图 2.1 所示即为粗糙集的形象表示：如果我们的知识最小粒度是图中的单位方框，其中不规则折线框出的部分就是粗糙集本身，而中间上色的区域则是它的下近似，外面的虚线框则表示它的上近似。

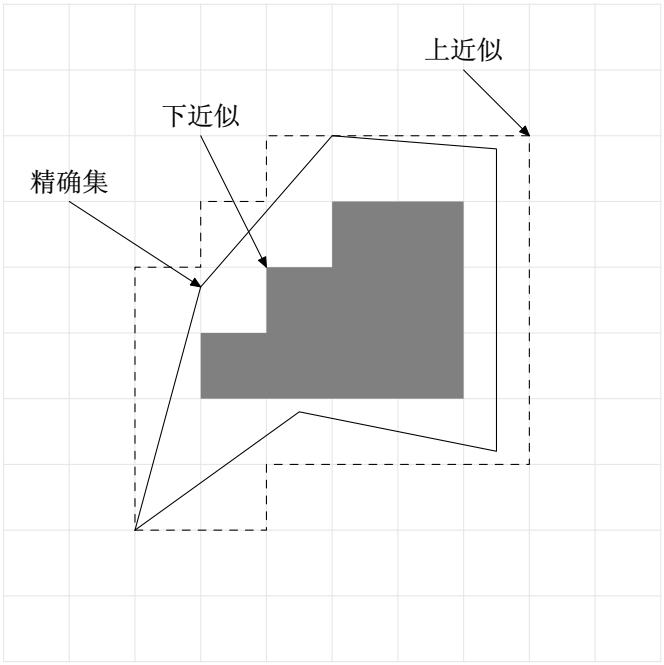

图 2.1 粗糙集

Fig. 2.1 Rough set

2.2 知识约简

知识约简是粗糙集理论的核心内容之一。知识库中的知识（属性）并不是同等重要的，甚至其中某些知识是冗余的，而知识约简就是在保持分类能力不变的前提下，删除知识库中不相关或不重要的知识（属性）。

知识约简中有两个基本的概念：约简（reduct）和核（core）。

定义 2.4. [55] *令 $\boldsymbol{R}$ 是一族等价关系，$R \in \boldsymbol{R}$，如果*

$$ind(\boldsymbol{R}) = ind(\boldsymbol{R} - \{R\})$$

则称 R 为 $\boldsymbol{R}$ 中不必要的；否则称 R 为 $\boldsymbol{R}$ 中必要的。

如果每一个 $R \in \boldsymbol{R}$ 都为 $\boldsymbol{R}$ 中必要的，则称 $\boldsymbol{R}$ 为独立的；否则称 $\boldsymbol{R}$ 为依赖的。

定义 2.5. [55] *设* $\boldsymbol{Q} \subseteq \boldsymbol{P}$，*如果* $\boldsymbol{Q}$ *是独立的，且* $ind(\boldsymbol{Q}) = ind(\boldsymbol{P})$，*则称* $\boldsymbol{Q}$ *为* $\boldsymbol{P}$ *的一个约简。显然，* $\boldsymbol{P}$ *可以有多种约简。* $\boldsymbol{P}$ *中所有必要关系组成的集合称为* $\boldsymbol{P}$ *的核，记作* $core(\boldsymbol{P})$ 。

而且我们知道：

定理 2.6. [55] $core(\boldsymbol{P}) = \bigcap red(\boldsymbol{P})$，*其中* $red(\boldsymbol{P})$ *表示* $\boldsymbol{P}$ *的所有约简。*

$core$ 这个概念，一方面可以作用所有约简的计算基础，另一方面可解释为不可消去的知识特性集合。

在应用中，更常用的是“相对约简”和“相对核”的概念。为此，我们先要定义一个分类相对另一分类的“正域”。

定义 2.7. *令* P *和* Q *为* U *中的等价关系，* Q *的* P *正域记为* $pos_P(Q)$，*即*

$$pos_p(Q) = \bigcup_{X \in U/Q} \underline{PX} \tag{2.7}$$

令 $\boldsymbol{P}$ 与 $\boldsymbol{Q}$ 为等价关系族，$R \in \boldsymbol{P}$，如果

$$pos_{ind(\boldsymbol{P})}(ind(\boldsymbol{Q})) = pos_{ind(\boldsymbol{P}-\{R\})}(ind(\boldsymbol{Q})) \tag{2.8}$$

则称 R 为 $\boldsymbol{P}$ 中 Q 不必要的；否则称 R 为 $\boldsymbol{P}$ 中 Q 必要的。在不至混淆的情况下，也用 $pos_{\boldsymbol{P}}(\boldsymbol{Q})$ 代替 $pos_{ind(\boldsymbol{P})}(ind(\boldsymbol{Q}))$ 。

如果 $\boldsymbol{P}$ 中的每个 R 都是 $\boldsymbol{Q}$ 必要的，称 $\boldsymbol{P}$ 为 $\boldsymbol{Q}$ 独立的。

设 $\boldsymbol{S} \subseteq \boldsymbol{P}$，$\boldsymbol{S}$ 为 $\boldsymbol{P}$ 的 $\boldsymbol{Q}$ 约简当且仅当 $\boldsymbol{S}$ 是 $\boldsymbol{P}$ 的 $\boldsymbol{Q}$ 独立子族且 $pos_{\boldsymbol{S}}(\boldsymbol{Q}) = pos_{\boldsymbol{P}}(\boldsymbol{Q})$ 。$\boldsymbol{P}$ 的 $\boldsymbol{Q}$ 约简称为相对约简。

$\boldsymbol{P}$ 中所有 $\boldsymbol{Q}$ 必要的原始关系构成的集合称为 $\boldsymbol{P}$ 的 $\boldsymbol{Q}$ 核，简称相对核，记为 $core_{\boldsymbol{Q}}(\boldsymbol{P})$ 。

类似的，我们有：

定理 2.8. [55] $core_{\boldsymbol{Q}}(\boldsymbol{P}) = \bigcap red_{(Q)}(\boldsymbol{P})$，*其中* $red_{\boldsymbol{Q}}(\boldsymbol{P})$ *是所有* $\boldsymbol{P}$ *的* $\boldsymbol{Q}$ *约简构成的集合。*

之所以用“相对”的概念，是因为在以“决策表”为表达形式的应用中来说，实际用的就是它。

2.3 决策表模型

一个信息系统 S 是一个系统 $S=(U,A)$，其中 $U=\{u_1,u_2\ldots u_{|U|}\}$ 是有限非空集，称为论域。U 中的元素称为对象，$A=\{a_1,a_2\ldots a_{|A|}\}$ 也是有限非空集，A 中的元素称为属性，对每个 $a\in A$，有一个映射 $a:U\rightarrow a(U)$，且 $a(U)=\{a(u)|u\in U\}$ 称为属性 a 的值域。如果 $A=C\cup D$,$C\cap D=\emptyset$，则称信息系统 (U,A) 为一个决策表，其中 C 中的属性称为条件属性，D 中的属性称为决策属性，如图 2.2 所示。

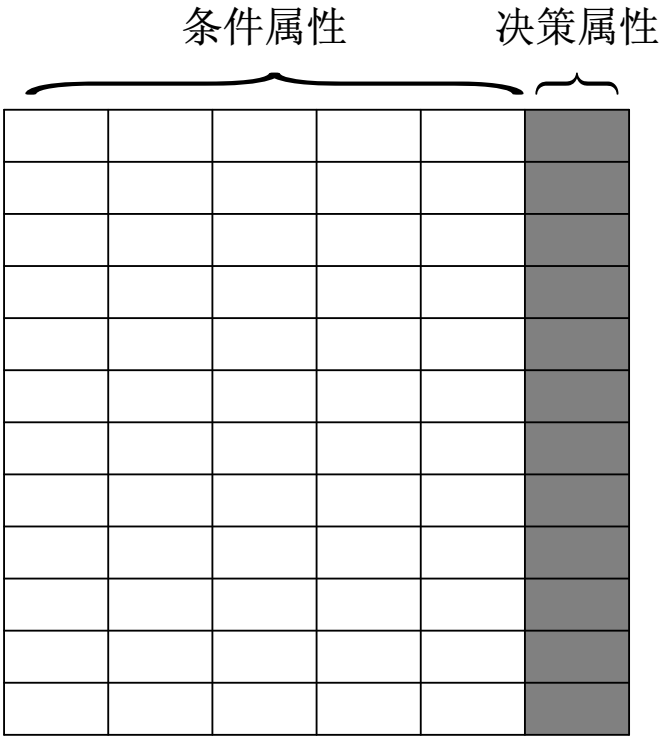

图 2.2 决策表模型

Fig. 2.2 Decision Table Mode

在应用粗糙集理论时，有一些概念上的改变和另一些新的概念的提出。

2.4 两类决策表

对于决策表模型 $K=(U,A)$，其中，$A=C\cup D,C\cap D=\emptyset$ 代表决策表中的属性集，而 C 叫做决策表 K 的条件属性，D 叫做 K 的决策属性，$U\neq\emptyset$ 是决策表中所有对象构成的论域。对于 $\forall x\in U,\forall P\in A$, 用 ${}^{x}P$ 记作元素 x 在属性集 P 下对应的值集。

如果这个决策表中$\forall x,y \in U, C_x = C_y \rightarrow D_x = D_y$，那么这个决策表就叫作“一致”决策表，否则就叫作不一致”决策表。这个定义实质上是在说：如果一个决策中，同样的条件属性必然有同样的决策属性，它就是一致的，否则就是不一致的。

这一定义的应用意义在于，我们能不能通过条件属性确定的判断决策属性的值，对于一致决策表，是可能的；对于不一致的决策表，则是不可能的。这样，在做规则生成时，就要作不同的考虑：对于一致决策表，规则的数据覆盖率应该是100%的；对于不一致决策表，规则的覆盖率应该是尽可能高的。这一点会在第4章有详细的论述。

2.5 支持子集和支持度

在决策表(U, A)中，对于每个属性$a \in A$，引入U中的一个划分U/a：两个对象$u, v \in U$在同一类中当且仅当$a(u) = a(v)$。设$W \subseteq U$，对于划分U/a，定义W的下近似为

$$W^{(U/a)^-} = \bigcup_{V \in U/a, V \subseteq W} V \tag{2.9}$$

也用$S_a(W)$来表示，称为W关于属性a的支持子集，$spt_a(W) = |S_a(W)|/|U|$称为W关于属性a的支持度；这个概念意味着规则“a蕴涵$y = y(W)$”有强度$spt_a(W)$。定义W的上近似为

$$W^{(U/a)^+} = \bigcup_{V \in U/a, V \cap W \neq \emptyset} V \tag{2.10}$$

$acc_a(W) = |W^{(U/a)^-}|/|W^{(U/a)^+}|$称为$W$关于属性$a$的近似精度。

对于决策属性D，我们考虑决策属性$y \in D$的整体决策，它关于属性$a \in C$的支持子集和支持度为：

$$S_a(y) = \bigcup_{W \in U/y} W^{(U/a)^-} = \bigcup_{W \in U/y} \left(\bigcup_{V \in U/a, V \subseteq W} V \right) \tag{2.11}$$

$$spt_a(y) = | \bigcup_{W \in U/y} W^{(U/a)^-}|/|U| \tag{2.12}$$

2.6 多条件支持度

令 $W \subseteq U$ 是 U 的子集，现在考虑 W 关于条件属性集的支持度。对于 $(U, C \cup D)$ 中的条件属性集 $X \subseteq C$，W 关于 X 的支持子集是

$$S_X(W) = W^{(U/X)^-} = \bigcup_{V \in U/X, V \subseteq W} V \tag{2.13}$$

相应的，W 关于 X 的支持度定义为

$$spt_X(W) = |W^{(U/X)^-}|/|U| \tag{2.14}$$

简单地，$S_X(Y)$ 意味着规则“X 蕴涵 Y”有强度 $spt_X(Y)$。

2.7 重要性和相对核

令 $\emptyset \subset X \subseteq C$，$\emptyset \subset Y \subseteq D$，$U/Y \neq U/\delta = U$。给定 $x \in X$，若 $S_{X-\{x\}}(Y) \subset S_X(Y)$，则称 x 在 X 中相对于 Y 是重要的，如果 $S_{X-\{x\}}(Y) = S_X(Y)$，则称 x 在 X 中相对于 Y 是不重要的。对于给定的 $x \in X$，定义 x 在 X 中的重要性（相对于 Y）为

$$sig^Y_{X-\{x\}}(x) = (|S_X(Y)| - |S_{X-\{x\}}(Y)|)/|U| \tag{2.15}$$

特别的，当 $X = \{x\}$ 时，用 $sig^Y(x)$ 来表示 $sig^Y_\emptyset(x)$，此时有

$$sig^Y(x) = sig^Y_\emptyset(x) = |S_x(Y)|/|U| \tag{2.16}$$

所有在 X 中相对于 Y 是重要的属性 $x \in X$ 组成的集合称为 X 相对于 Y 的核。

2.8 相对约简

在决策表中，我们关心的是决策属性的预测和决策规则的生成，所以，相对于决策属性的知识约简更为合理和有意义[9, 52]。

在决策表中，相对于决策属性，不同的条件属性可能具有不同的重要性，我们用“重要性”来区分这一点：

定义 2.9. *令 C 和 D 分别为条件属性集和决策属性集，属性子集 $C' \subseteq C$*

关于 D 的重要性定义为

$$\sigma_{CD}(C') = \gamma_C(D) - \gamma_{C-C'}(D) \tag{2.17}$$

其中，

$$\gamma_C(D) = |pos_C(D)|/|U| \tag{2.18}$$

已经知道，找所有约简的时间复杂性是指数的。而找出最小约简是 $NP-hard$ 的。于是，对于这一操作，有了很多启发式的算法，例如 $Quick-Reduct$ 的方法[37]

QuickReduct(C, D, R)

输入：所有的条件属性之集 C

所有的决策属性之集 D。

输出：C 的约简 $R\,(R \subseteq C)$

1. $R \leftarrow \emptyset$
2. do
3. $\quad T \leftarrow R$
4. $\quad \forall x \in (C-R)$
5. $\quad\quad$ if $\gamma_{R\cup x}(D) > \gamma_T(D)$
6. $\quad\quad\quad T \leftarrow R \cup x$
7. $\quad R \leftarrow T$
8. until $\gamma_R(D) = \gamma_C(D)$
9. return R

2.9 规则生成

在决策表中，最重要的是决策规则的产生。

决策规则定义如下：

$$r_{ij} : dex(X_i) \rightarrow des(Y_j), Y_j \cap X_i \neq \emptyset \tag{2.19}$$

规则的确定性因子：

$$\mu(X_i, Y_j) = |Y_j \cap X_i|/|X_i|, 0 < \mu(X_i, Y_j) \leq 1 \tag{2.20}$$

当 $\mu(X_i, Y_j) = 1$ 时， r_ij 是确定的；否则是不确定的。

当然，一般在产生决策规则之前，先对决策表进行属性约简，能达到更简洁的结果。而且，对于多个规则，可能可以合并。[29, 59]

2.10 粗糙集应用中存在的辩证关系

在粗糙集作为数据挖掘算法的应用中，存在着很多辩证的关系，需要在两个矛盾之间取一个折衷点，这样的情况有很多种，下面分别讨论之。

2.10.1 键属性

原始数据一般存放在数据库的表中，而数据库的每个表结构中，都有一个称为“键”的属性，这个属性的定义一般是这样的：它的唯一的，它是不可空的。这样的定义是为了“区分”各个不同的记录（或者等价于数据挖掘中的*instance*），使得可以用“键”来唯一的标识一个记录。

然而，在数据挖掘工作中，无论采用什么技术，都是不考虑这一属性的。正是由于它的“唯一标识”的性质，使用其它的属性都变得“无关紧要”，因为关于某一记录的所有其它属性都能通过它得到。在粗糙集方法中，可以证明，虽然它可以任何精确的刻画讨论的集合（使任何集合都变成“精确集”），但是如果有“键属性”的存在，在做属性约简时，将把所有的其它属性都约简掉，这不是我们所希望的。理由如下：

1. 数据挖掘的目的是找出数据中存在的“模式”，只对单体有意义的东西，不能称之为“模式”。

2. 对已有数据作这样的区分，得到的结论对新的数据没有任何指导意义，因为新的数据和已有的数据在“键属性”上必然是不同的。

不仅是“键属性“，任何与“键属性”有等价关系的属性（即可以“唯一标识”事例的属性）都不在考虑之列，比如身份证号、学号、毕业证号等等。作为数据挖掘的原始数据，不应该包含这样的属性信息。

可以看到，虽然有些属性可以任何精确的刻画讨论的集合，却不能引入讨论的范围，可以说是一种“精确”和“粗糙”之间的折衷。

2.10.2 规则的确定性和条数

和关于“键属性”的讨论类似，在生成决策规则时，也有类似的折衷。

对于一个没有重复事例的决策表，如果把每一个事例都作为一条规则的话，显然，得到的所有的规则都是确定的。但是，这样的规则也将是无意义的。理由如下：

1. 由于事例没有重复，意味着规则的条数和事例一样多，数据量一般是庞大的，规则数也将是庞大的，要找到匹配的规则将耗费很长时间。

2. 由于事例没有重复，意味着每一条规则都不能应用于新的数据。

相对的，如果我们只考虑事例的某几个属性，生成少量的不是很确定的规则，匹配起来将会容易的多，而且，少量的属性描述意味着规则简单和易于匹配，新的事例也更容易和已有的规则相匹配，从而实现预测的目的。

3 对于经典粗糙集理论的几点思考

3.1 划分不是必须的

从第2章中可以看出，在对决策表模型应用粗糙集相关的算法时，划分是整个过程的第一步操作，也是所有后继操作的基础。然而，对于任给的一组属性集A，对于划分U/A的计算，其时间复杂度都至少是$O(n)$的，在数据量巨大时（这个假设在数据挖掘时几乎总是成立的），时间的花费将是巨大的。虽然我们可以用$cache$的方法来适当减少计算，但是空间的浪费又将是巨大的。下面我们来论证，“划分”这个操作，在进行规则生成过程中，不是必须的。

3.1.1 划分的实质是什么

划分的实质是什么呢？

从第2章中可以看出，划分的最直接目的是为了计算“下近似”，而“下近似”正是用来描述粗糙集的核心概念之一。从这一方面分析，划分的实质作用在于：把决策表中的数据按照不同的属性集，分成不同的类，然后可以通过这些类的隶属关系用下近似的概念来刻画我们关心的决策属性和条件属性集之间的关系。

从数学角度来分析，它的作用在于，对于任给的属性集$C' \subseteq C$,把决策表变成相应的分类集合U/C'。而对于决策属性集D，相应的分类集合U/D看作是集合的精确描述，而用“下近似”来表U/C与U/D之间的关系。这样，就把一个应用问题转化为一个纯数学的问题。

对于$\forall P \subseteq A$而言，设$U/P = \{P_1, P_2 \ldots P_n\}$，从划分的定义可以知道，$\forall x, y \in U/P, x \neq y$，我们有$x \cap y = \emptyset$。？？？？

3.1.2 必要性分析

下面我们来分析一下“划分”这一操作的必要性。

粗糙集理论和方法的基础概念是“上下近似”，其它概念都是围绕它们展开定义的。从上一节的分析可以知道，“划分”只是为了建立从应用问题到数学问题这样一个桥梁，那么，我们显然可以采用一种更直接的映射，从应用问题直接映射到上下近似，通过省去“划

分”这一步骤，简化算法的时间复杂度在规则生成算法中，实际关心的是“下近似”，本文提出的算法也是基于“下近似”概念的。

这一思想的可行性分析如下：

对于决策表模型$K=(U,A)$，其中，$A=C\cup D, C\cap D=\emptyset$代表决策表中的属性集，而$C$叫做决策表$K$的条件属性，$D$叫做$K$的决策属性，$U\neq\emptyset$是决策表中所有对象构成的论域。对于$\forall x\in U, \forall P\in A$, 用$^{x}P$记作元素$x$在属性集$P$下对应的值集。

对于两个属性集$P_1, P_2\subseteq A$，设$U/P_1=\{P^1{}_1, P^1{}_2\ldots P^1{}_r\}, U/P_2=\{P^2{}_1, P^2{}_2\ldots P^2{}_s\}$，$P_1$对于$P_2$的下近似为

$$hello$$

3.2 计算全部下近似不是必须的

3.3 精确下近似（或者说完全下近似）是不是必须的

3.4 重要性，支持度等等概念是不是必须的

3.5 约简是不是必须的

3.6 算法的复杂性分析和证明

3.7 其他算法的复杂性分析及证明或引用证明

3.8 关于覆盖率的证明

3.9 关于平均规则长度的对比及证明

3.10 关于属性排序的问题

可以考虑把分类少的属性放在前面，因为少的类等价于大的覆盖率。

3.11 要解释的概念

1. 结构主义和结构化
2. 公理化
3. 自恰

要举的例子

1. 几个公理体系（六种粗糙集）
2. 几种生成规则的算法

3.12 对规则生成方法的再认识

现在应用粗糙集理论生成规则的算法有。。。。。，它们都是先对数据进行约简，然后应用约简后的属性集生成规则。不同的算法在于约简算法的不同。然而，$Kohavi$ 和 $Frasca$ 用实验证明，数据库中最有用的子集并不一定是粗糙集中的相对核，甚至可能不包括完全的核属性集[11, 55]。在本论文研究的算法中，考虑的是不使用传统的属性约简算法，而是直接应用“下近似”的概念，从数据集来生成规则。这就需要对理论中的“下近似”的概念作进一步的认识。

3.13 对“下近似”概念的再认识

粗糙集理论，现在已经结构化和公理化，即它的理论核心可以归结为从几个不加证明的公理推导出来的一整套自恰的理论体系。例如：。。。。。可以看出，在粗糙集理论的公理化系统中，最核心的概念是“下近似”和“上近似”。从第 2 章中我们知道：在决策表模型 $(U, A), A = C \cup D, C \cap D = \emptyset$ 中，对于属性集 $AB \in CA \cap B = \emptyset$，可以把 A 看做在这个决策表中对于 B 的下近似，如果它们满足：

$$\forall a \in U/A, \exists b \in U/B(a \subseteq b)$$

由于这个运算是整个粗糙集理论中的基本运算，所以它的效率将直接影响到任何一个算法的效率。从定义可以看出，这个运算包含了如下几种操作：

1. 求划分（ $U/AU/B$ ）

2. 判断隶属关系（ $a \subseteq b$ ）

这个定义在应用时有以下几种用法：

1. 判断 A 是不是 B 的下近似，如果是，那么可以直接用 A 来描述 B ，也就是说 A 是 $A \cup B$ 的一个约简，如果 B 是决策属性，那么 A 就是 B 的相对约简。这也是采用“先约简，后生成规则”的算法的基本思想。

2.

   如果按这个定义来计算的话，结果会不会好一些呢？

3.14 关于新的想法

1. 上下近似，这两个概念是必须的吗？如果不完全用这两个概念呢？显然也不会影响规则的生成。如何证明呢？我们可以只用部分下近似就可以了，而且，这样的算法更加简单。

2. 比如我会用到这样的两种描述：

   a) $x = A$ 时，$\forall y = B$吗？

   b) $x =?$ 时，$\exists y = B|C$吗？

4 新算法和相关的理论依据

经典的粗糙集规则生成算法是基于“约简”的，很大一部分相关的优化算法也是针对“约简”操作所作的优化。具体如图 4.1 所示。 而

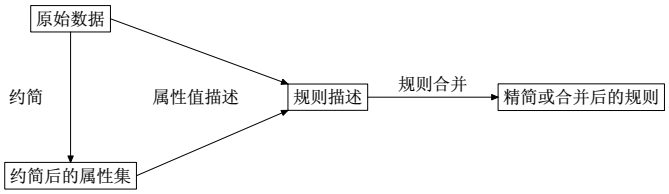

图 4.1 经典粗糙集规则生成算法

Fig. 4.1 Classic Rough Set Theory's Rules Generation Algorithm

我们所研究的这个新的算法是不依赖于约简的。形象地来说，如果用决策表系统来表示数据的话，关于经典算法和本文的算法的描述如图 4.2 所示。

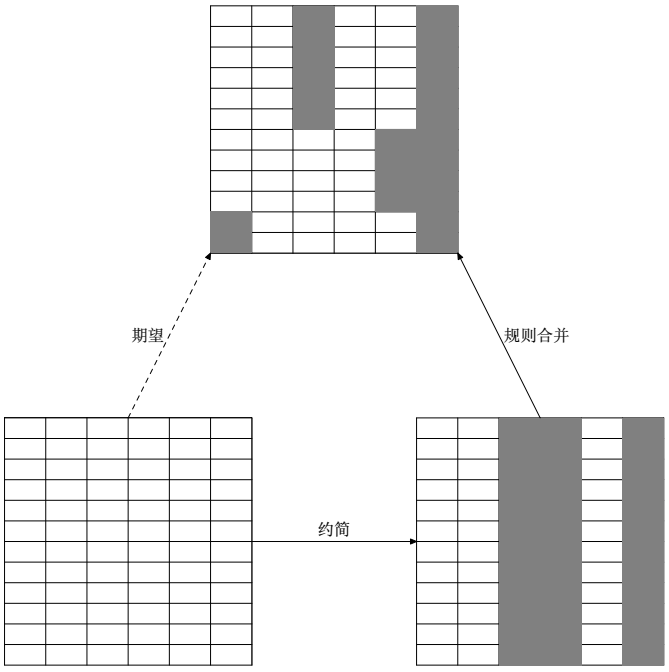

图 4.2 经典算法与本文算法的比较

Fig. 4.2 Comparison of Classic and Our Algorithms

算法如下：

Require: 对于决策表系统 $T=(U,C\cup D),\ C\cap D=\emptyset$
Ensure: 包含所有生成规则的集合 $Rules$
1: $Rules\leftarrow\emptyset$
2: 计算 C 的幂集并把其中的元素按集合的基从小到大排序，即 $Q=2^C=\{Q_1,Q_2\dots Q_r\}$，其中对于 $1\le x<y\le r$ 满足 $|Q_x|\le|Q_y|$
3: $S\leftarrow T$
4: $i\leftarrow 1$
5: 对于 Q_i 和 S 执行单属性规则生成算法 $SingleAttributeSetRulesGeneratingAlgorithm$ ($SASRGA$)(详见图4.4)，设生成的规则集为 R'
6: $Rules\leftarrow Rules\cup R'$
7: 计算 R' 在 S 中未覆盖的元素 S'
8: **if** $S'\neq\emptyset$ && $i<=r$ **then**
9: $S\leftarrow\{S',C\cup D\},i=i+1$
10: 跳到第 5 步执行
11: **end if**
12: 把 $Rules$ 作为结果返回

第 4.3 图 $ShortFirtsAlgorithm$ (SFA)

Require: 属性集 E 和决策表 $K=\{U,C\cup D\}$
Ensure: 规则集 R
1: $R\leftarrow\emptyset$
2: 对于 $\forall u\in U$，用二元组 $\{E_u,D_u\}$ 来表示，删除重复项，并设新的二元组集合为 U'
3: 对于 U' 中元素，按第一项分组，即把第一项相同的元素分到同一组
4: 删除组中元素个数大于 1 的组，只留下“单元素”组
5: 把这些组中的元素表示成规则的形式 $E_u\Rightarrow D_u$，并加入 R 中
6: 把 R 作为结果返回

第 4.4 图 $SingleAttributeSetRulesGeneratingAlgorithm$ ($SASRGA$)

可以看到，SFA中没有明确使用“划分”、“下近似”和“约简”这样的概念。但是它用到的其实是“下近似”概念在规则生成算法中的表现形式。下面的几节我们将要证明，我们的算法满足以下性质：

1. 对于一致决策表，SFA能达到100%的数据覆盖率
2. 对于所有的决策表，SFA与经典的粗糙集方法产生的规则具有相同的表现力和覆盖率
3. SFA的时间复杂度为$kkkk$，小于经典方法的ppp，与其它改进方法的时间复杂度相当
4. 与其它几种算法相比，SFA生成的规则具有更小的规则长度

如没有特别声明，以下几节采用如下约定:

对于决策表模型$K=(U,A)$，其中，$A=C\cup D, C\cap D=\emptyset$代表决策表中的属性集，而$C$叫做$K$的条件属性，$D$叫做$K$的决策属性。$U\neq\emptyset$是决策表中所有对象构成的论域。对于$\forall x\in U, \forall P\in A$, 用${}^{x}P$记作元素$x$在属性集$P$下对应的值集。并设$|U|=n, |C|=m, |D|=r$，$Q=2^{C}=\{Q_1,Q_2\ldots Q_t\}$，经典粗糙集算法得到的约简属性集为$R=Q_l=\{R_1,R_2\ldots R_s\}$。

4.1 覆盖率分析

定理 4.1. *对于一致决策表，SFA能达到100%的数据覆盖率。*

要证明这个定理，我们还需要用到下面的引理：

引理 4.2. $\forall A,B\in C, \forall x,y\in U, {}^{x}A\neq{}^{y}A\Rightarrow{}^{x}(A\cup B)\neq{}^{y}(A\cup B)$

证明如下：

证明. 用反证法：由已知，$\forall A,B\in C, \forall x,y\in U$，假设${}^{x}(A\cup B)={}^{y}(A\cup B)$

$\because A\subseteq(A\cup B), {}^{x}(A\cup B)={}^{y}(A\cup B)$

$\therefore {}^{x}A={}^{y}A$

显然这是与${}^{x}A\neq{}^{y}A$相矛盾的。

$\therefore$假设错误，命题得证。 □

下面我们用引理 4.2 来证明定理 4.1:

证明. 我们还用反证法：假设算法不能完全覆盖数据集，也就是说，在最后一次循环结束后，$S \neq \emptyset$，下面来证明，这是不可能的，即在这最后一步中，不可能存在这样的二元组——$({}^{x}C = {}^{y}C \&\& {}^{x}D \neq {}^{y}D)$。

在最后一次 $oneRule$ 过程中，设 $S = (U', C \cup D), U' \subseteq U$，此时，

$\because$ 由一致性的定义，对于 ${}^{x}D \neq {}^{y}D$ 的任意 $x, y \in U$，$\exists K \subseteq C$ 使得 ${}^{x}K \neq {}^{y}K$

$\therefore$ 对于 ${}^{x}D \neq {}^{y}D$ 的任意 $x, y \in U$，${}^{x}C \neq {}^{y}C$ (由上面证明的引理可知)

$\therefore$ 对于 ${}^{x}D \neq {}^{y}D$ 的任意 $x, y \in U'$，${}^{x}C \neq {}^{y}C$

亦即，${}^{x}C = {}^{y}C$ 与 ${}^{x}D \neq {}^{y}D$ 不能同时满足，假设错误，命题得证。

□

定理 4.3. *对于不一致决策表，SFA 能达到和经典算法一样的数据覆盖率。*

证明. 设不一致决策表为 $K = (U, A)$，在定理 4.1 中，我们已经证明，对于一致辞的决策表，SFA 可以有和经典算法一样的 100% 的数据覆盖率。对于不一致数据集中那些不一致的数据，由定义，它不可能出现在任一条规则中，也就是说，这些数据不能被经典算法和/或 SFA 中任何一条规则所描述，那么，如果从原数据集 U 中删掉全部不一致的数据，原数据集将变成一个新的一致的数据集 U'，原决策表变为 $K' = (U', A)$，这个我们在下面的引理 4.4 中给出证明。而对于一致决策表 K'，SFA 与经典算法都可以达到 100% 的数据覆盖率，也就是说，对于 U' 中的元素，都能被两种算法中的规则完全覆盖，而 $(U - U')$ 中的数据则都不能覆盖，所以，对于不一致决策表 K，经典算法和 SFA 形成的规则的数据覆盖率都为 $|U'|/|U|$，定理得证。 □

引理 4.4. *从不一致决策表中去掉全部不一致的数据项，决策表将变成一致的数据表。*

证明. 设 $K = (U, C \cup D)$ 为不一致决策表，由定义，其中存在这样的数据项 x, y 和属性集 M，使得 ${}^{x}M = {}^{y}M$ 但是 ${}^{x}D \neq {}^{y}D$。如果把所有这样的 x, y 都去掉，显然决策表已经没有了“不一致”的条件，成为一致的决策表，引理得证。 □

4.2 时间复杂度分析

定理 4.5. *在决策表模型中，设属性数共有m个，至少有一个属性不同的数据数为n个，决策属性只有r个，$r << m$，亦即，条件属性有$m-r$个，经典算法的最坏时间复杂度为$T_{classic}^{max} = O()$。*

证明如下：

证明. 这个呢？ □

对于一些改进的算法，如决策矩阵法，快速约简法。。。。等等。

定理 4.6. *在决策表模型中，设属性数共有m个，至少有一个属性不同的数据数为n个，决策属性只有r个，$r << m$，亦即，条件属性有$m-r$个，经典算法的平均时间复杂度为$T^{max} = O()$。*

证明如下：

证明. 这个呢？ □

定理 4.7. *在决策表模型中，设属性数共有m个，至少有一个属性不同的数据数为n个，决策属性只有r个，$r << m$，亦即，条件属性有$m-r$个。则SFA的最差时间复杂度是$O(2^m \times n^2)$*

下面我们来证明定理 4.7。

证明. 首先，各个条件属性分别为$\{A_1, A_2 \dots A_{m-1}\}$，决策属性为$A_m$，各个属性对应的划分数分别为$\{D_1, D_2 \dots D_m\}$，一般地，数据量是海量的，即有$m << n, D_i << n$。基于此，我们按以下步骤来分析：

对于计算好的属性集序列，每轮循环，设当前用到的属性集（含决策属性）为$B = \{B' \cup D\} = \{B_1, B_2 \dots B_k\}$，对应的划分数分别为$\{S_1, S_2 \dots S_k\}$，需要用的操作如下：

1. 对当前的数据集，得到每个数据对应的（条件属性值，决策属性值）二元组描述。这个操作只需要遍历数据集即可，复杂度为$t_1 = O(n)$

2. 把得到的二元组分组，并标记之，相当于先把二元组排序，再把排序后的二元组遍历，遍历时，如果访问的二元组与前一个相同，就归入上一组，不同，就新设一个组。由二元组的特点，排序可以用“快速排序”来做，而“快速排序”的复杂度，最坏为 $O(n^2)$，所以这一步的复杂度，最坏为 $t_2^{max} = O(n) + O(n^2) = O(n^2)$ 。

3. 设分组结束后，分得的组数为 z ,显然， z 的最大值为 $z^{max} = \prod_{i=1}^{k} S_i$ 。

4. 这次分组后，不能形成规则的数据集设为 Y，亦即对于 $\forall y \in Y$，均有 $\exists y' \in Y$, 使得 ${}^{y}B' = {}^{y'}B' \&\& {}^{y}D \neq {}^{y'}D$ 成立。这里，显然 $|Y|$ 最大值为当前分组数，即都不能形成规则，$|Y|^{max} = z^{max}$ 。

这样，算法的最差时间复杂度为：

当每一步都取最坏结果，且决策表不能约简时，每个数据都要用全部属性来描述才能区分，此时，共要循环 2^{m-1} 次，每次均不能分组，复杂度为 $t_2^{max} = O(n^2)$，这样，整个算法的最坏复杂度为 $T^{max} = 2^{m-1} \times t_2^{max} = 2^{m-1} \times O(n^2) = O(2^m \times n^2)$ □

这一结果说明，SFA 在最差时间复杂度上，与经典算法是相当的。

定理 4.8. *在决策表模型中，设属性数共有 m 个，至少有一个属性不同的数据数为 n 个，决策属性只有 r 个，$r << m$，亦即，条件属性有 $m - r$ 个。设 SFA 的平均时间复杂度是 $T^{mean} = O(??????)$*

要证这一定理，需要用到引理 4.9 和引理 4.10 。

引理 4.9. *设在算法中考虑的当前属性数（含决策属性）为 m，各属性对应的划分数分别为 $\{S_1, S_2 \ldots S_m\}$，数据总数为 n，则其形成的不同的（条件属性值，决策属性值）组合数的数学期望为 $E(n) = \Pi(1 - (\frac{\Pi-1}{\Pi})^n$ ，其中 $\Pi = \prod_{i=1}^{m} S_i$ 。*

证明如下：

证明. 显然，当 $n = 1$ 时

$$E(n) = E(1) = 1" \tag{4.1}$$

要求 $E(n)$，假设我们已经知道了 $E(n-1)$，那么，当增加第 n 个数据时，只有当它不同于以前出现的所有的数据，才会有数据项的增加。而当前共可能出现的组合数为 $\Pi = \prod_{i=1}^{m} S_i$，于是，由经典概率模型和数学期望的性质，我们有

$$E(n) = E(n-1) + \frac{\Pi - E(n-1)}{\Pi} \tag{4.2}$$

结合公式 4.2 和公式 4.1，令 $T = 1 - \frac{1}{\Pi}$，我们有：

$$\begin{aligned}
E(n) &= E(n-1) + \tfrac{\Pi - E(n-1)}{\Pi} \\
&= E(n-1) + 1 - \tfrac{1}{\Pi} \times E(n-1) \\
&= (1 - \tfrac{1}{\Pi}) \times E(n-1) + 1 \\
&= T \times E(n-1) + 1 \\
&= T \times (T \times E(n-2) + 1) + 1 \\
&= T^2 \times E(n-2) + T^1 + T^0 \qquad\qquad Kæy" \qquad (4.3) \\
&\cdots \\
&= T^{n-1} E(1) + T^{n-2} + \cdots + T^1 + T^0 \\
&= T^{n-1} + T^{n-2} + \cdots + T^1 + T^0 \\
&= \tfrac{T^n - 1}{T - 1} \\
&= \Pi(1 - (\tfrac{\Pi - 1}{\Pi})^n)
\end{aligned}$$

□

引理 4.10. *设在算法中考虑的当前属性数（含决策属性）为 m，即属性集为 $T_a = \{B_1, B_2 \ldots B_m\}$，其中 B_m 为决策属性。各属性对应的划分数分别为 $\{S_1, S_2 \ldots S_m\}$，当前要处理的数据总数为 n，我们把完成分组后，可以被规则覆盖的数据称为“可分辨数据”，不能被形成的规则覆盖的数据称为“不可分辨数据”，则不可分辨数据数据量的数学期望为 $F(n) = KKK$。*

证明如下：

证明. 设属性集 $T' = \{B_1, B_2 \ldots B_{m-1}\}$，令 Π^X 表示属性集 X 中全部可能的组合数，E_x^Y 表示当属性集为 Y 数据量为 x 时可能形成的组合数的数学期望，显然 E_x^Y 可由引理 4.9 来计算。

由定义，显然，当$n=1$时，没有不可分辨数据，即有

$$F(1)=0 \tag{4.4}$$

同样地，按经典概率论和数学期望的性质，我们有；当新增加一条数据，使数据量从$n-1$变为n时，有以下三种情况

1. 新数据与所有已知的数据属性组合都不相同，形成新的一种属性组合，此时$F(n)=F(n-1)$，这种情况出现的概率为$P_1=\frac{\Pi^{Ta}-E_{n-1}^{Ta}}{\Pi^{Ta}}$
2. 新数据与可分辨数据中的一个相同，此时，$F(n)=F(n-1)$，这种情况出现的概率为$P_2=\frac{E_{n-1-F(n-1)^{Ta}}}{\Pi^{Ta}}$
3. 新数据与不可分辨数据中的一个相同，此时，$F(n)=F(n-1)+1$，这种情况出现的概率为$P_3=\frac{E_{F(n-1)}^{Ta}}{\Pi^{Ta}}$
4. 新数据与原分辨数据中的几条数据形成新的不可分辨数据，此时，$F(n)=F(n-1)+M+1$，其中，M为原数据中新的不可分辨的数据量，这种情况出现的概率为P_4=????

由以上分析知：

$$F(n)=F(n-1)\times(P_1+P_2)+(F(n-1)+1)\times P_3+(F(n-1)+m+1)\times P_4 \tag{4.5}$$

□

下面我们来证明定理4.7。

证明. 首先，在决策表模型中，设属性数共有m个，至少有一个属性不同的数据数为n个，为方便讨论，决策属性只有1个，亦即，条件属性有$m-1$个。设各个条件属性分别为$\{A_1,A_2\ldots A_{m-1}\}$，决策属性为$A_m$，各个属性对应的划分数分别为$\{D_1,D_2\ldots D_m\}$，一般地，数据量是海量的，即有$m<<n, D_i<<n$。基于此，我们按以下步骤来分析：

对于计算好的属性集序列，每轮循环，设当前用到的属性集（含决策属性）为$B=\{B'\cup D\}=\{B_1,B_2\ldots B_k\}$，对应的划分数分别为$\{S_1,S_2\ldots S_k\}$，需要用的操作如下：

1. 对当前的数据集，得到每个数据对应的（条件属性值，决策属性值）二元组描述。这个操作只需要遍历数据集即可，复杂度为$t_1=O(n)$

2. 把得到的二元组分组，并标记之，相当于先把二元组排序，再把排序后的二元组遍历，遍历时，如果访问的二元组与前一个相同，就归入上一组，不同，就新设一个组。由二元组的特点，排序可以用“快速排序”来做，而“快速排序”的复杂度，平均为 $O(n\log n)$，最坏为 $O(n^2)$ 所以这一步的复杂度，平均为 $t_2^{mean} = O(n) + O(n\log n) = O(n\log n)$。

3. 设分组结束后，分得的组数为 z，其数学期望为 z^{mean} =??? (引理 4.9)

4. 这次分组后，不能形成规则的数据集设为 Y，亦即对于 $\forall y \in Y$，均有 $\exists y' \in Y$, 使得 ${}^{y}B' = {}^{y'}B' \&\& {}^{y}D \neq {}^{y'}D$ 成立。其平均值同样要用数学期望算得：$|Y|^{mean}$ =??? (引理 4.10)

这样，算法的平均时间复杂度为：$T^{mean} = O(????)$

□

4.3 规则长度分析

我们定义了一个新的评测标准“平均规则长度”$MeanRulesLength$ (MRL)来评测不同算法生成规则的长度，这个标准既考虑了规则的数目，又考虑了相等数目下规则的长度，可以作为一个更加全面的标准[16]。

定义 4.11. *设规则集为 $R = \{r_1, r_2 \ldots r_n\}$，设 S 表示 $MeanRulesLength$，则 S 定义为：*

$$S = \sum_{i=1}^{n} attrNumber(r_i) \tag{4.6}$$

其中，$attrNumber(r_i)$ 表示规则 r_i 中出现的属性数。

对于算法 SFA，我们有：

定理 4.12. *SFA 产生的规则数不会多于经典粗糙集算法。*

证明. 显然，在极端情况下，决策表不能得进行约简时，所有的数据本身就形成规则，而且规则不能合并，此时经典算法和 SFA 产生的规则数将是相同的，都是数据总数。

在一般情况下，决策表可以约简或者规则可以合并时，由于 SFA 算法在第 2 步是先计算属性集的幂集的，此把它们按基的大小进行排列，这样，如果在经典算法中存在规则 r_i, r_j 是可以合并的，设合并后的规则为 r，那么，必然有 $attrNumber(r) < attrNumber(r_i)\&\&attrNumber(r) < attrNumber(r_j)$，而在 SFA 算法中，显然 r 中的属性集应该出现在 r_i 和 r_j 之前，也就是说，这样的规则一定会先被发现，换言之，即使有规则可以合并，也会在 SFA 算法中被提前发现。 □

定理 4.13. *SFA 的 MRL 不会长于经典粗糙集算法。*

要证明定理 4.12，需要用到下面的引理：

引理 4.14. *在对 Q_l 进行完 $SASRGA$ 之后，产生的规则集就是算法 SFA 最终产生的结果规则集。*

证明如下：

证明. 小case □

下面我们用引理 4.14 来证明定理 4.12：

证明. difficult □

4.4 实验环境

本论文的实验验证部分在 $GNU/Linux$ 平台下用 $haskell$ 语言来完成。

选用这样的实现环境，是出于以下几点考虑。

1. $GNU/Linux$ 平台是我个人比较熟悉的平台
2. $GNU/Linux$ 平台相当稳定，适合做计算量比较大的工作
3. $haskell$ 语言具有跨平台的特性，可以不加修改在各种主流操作系统上运行
4. $haskell$ 语言是一种“Lazy”的语言，它会自动地只做能完成任务的最少的运算

5. *haskell* 语言是函数式语言的特性，使得它在这个多核和多处理器的时代，可以无需修改就自动并行计算

6. *haskell* 简练的语法，使得开发实现更加简洁，更不易出错

7. *haskell* 函数式语言的特性，使得“设计模式”“类型转换”这类在面向对象开发中很有用的东西变得极其自然，自然到根本不用去想，自己就能处理

8. *haskell* 语言是一种“形式化”的语言，它比 $C/C++, java, C\#, perl$ 等等语言更接近于数学描述，这使得它更适合于描述理论性较强的算法，比如本文中的算法，更直观也更不容易出错

另外要加以说明的是，作为一个算法验证和比较的程序，本文的实验系统没有图形用户界面（*GUI*），而是提供了很多精练和通用的接口，可以直接在其它程序中使用。从 *haskell* 得到的另一个好处是，系统可以在各个层面无修改地重用。

在实验数据方面，本论文采用的数据有：取自 *UCI* 的 *Iris* 数据集和 $New-thyroid$ 数据集，*ctr* 数据集，学长提供的学生贷款数据集 *Loan* 等。

由于经典粗糙集方法和一些改进的方法都是针对离散型数据的，所以对于这些数据集中的连续型数据，我们都首先进行了数据离散化处理，这样，就有了共同的比较基础。在数据离散化方面，用的是 *Rosetta* 软件提供的 *BROrthogonalScaler* 方法。当然，还有其它的数据离散化方法，但是通过第 4 章的分析可以知道，算法是与离散化方法无关的。

4.5 重要代码分析

自定义的主要数据结构如下：

```
1 type Attribute = String
2 type Value     = String
3 type Element   = M.Map Attribute Value
4 type MayRule = Rule
5
```

```
type Rule=(M.Map Attribute Value, M.Map Attribute Value)
type DecisionAttribute=(S.Set Attribute, S.Set Attribute)

data RoughSet = RoughSet{
        attributes :: DecisionAttribute,
        elements   :: S.Set Element
} deriving Show
```

主要函数原型如下：

```
combination :: [a] -> [[a]]
genAttributeArray :: RoughSet -> [DecisionAttribute]
covered :: Rule -> Element -> Bool
onceRule :: RoughSet -> DecisionAttribute -> [Rule]
rules' :: RoughSet -> [DecisionAttribute] -> [Rule]
sigNotCoverd :: RoughSet -> Rule -> RoughSet
notCovedRoughSet :: RoughSet -> [Rule] -> RoughSet
rules :: RoughSet -> [Rule]
elementShow :: Element -> String
descRule :: Rule -> String
coverRate :: RoughSet -> [Rule] -> Double
ruleLength :: Rule -> Int
meanLength :: RoughSet -> [Rule] -> Double
genAttributeCombine :: DecisionAttribute ->
        [DecisionAttribute]
subValue :: Element -> DecisionAttribute -> MayRule
generateMayRule :: RoughSet -> DecisionAttribute ->
        [MayRule]
filterMayRule :: [MayRule] -> [Rule]
allNeeded :: RoughSet -> (Int, [String], Double, Double)
writable :: (Int, [String], Double, Double) -> String
buildRouset :: String -> RoughSet
```

5 结论和展望

5.1 实验结果

数据集部分数据见表 5.1，表 5.2，表 5.3，表 5.4，表 5.5，表 5.6：

表 5.1 数据属性

Tab. 5.1 Attributes of the data set

数据集	属性个数	元素个数	数据类型
$Iris$	5	150	连续
$New-thyroid$	5	215	连续
$Yellow-small(balloons)$	4	20	连续
ctr	10	21	离散
$Loan$	8	2059	离散

表 5.2 *iris* 部分数据

Tab. 5.2 Part of data set iris

sepal length(cm)	sepal width(cm)	petal length(cm)	petal width(cm)	class
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3.0	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
5.0	3.6	1.4	0.2	Iris-setosa
5.4	3.9	1.7	0.4	Iris-setosa
4.6	3.4	1.4	0.3	Iris-setosa
5.0	3.4	1.5	0.2	Iris-setosa
4.4	2.9	1.4	0.2	Iris-setosa
4.9	3.1	1.5	0.1	Iris-setosa

对各数据做实验的整体结果数据表 5.7，表 5.8，表 5.9 所示：

5.2 结论

5.3 展望

表 5.3 *new − threshold* 部分数据

Tab. 5.3 Part of data set new-threshold

Class	T3-resin	Serum thyroxin	Serum triiodothyronine	TSH	Maximal TSH
1	107	10.1	2.2	0.9	2.7
1	113	9.9	3.1	2.0	5.9
1	127	12.9	2.4	1.4	0.6
1	109	5.3	1.6	1.4	1.5
1	105	7.3	1.5	1.5	-0.1
1	105	6.1	2.1	1.4	7.0
1	110	10.4	1.6	1.6	2.7
1	114	9.9	2.4	1.5	5.7
1	106	9.4	2.2	1.5	0.0
1	107	13.0	1.1	0.9	3.1

表 5.4 *small − yellow* 部分数据

Tab. 5.4 Part of data set small-yellow

Color	Size	Act	Age	Inflated
YELLOW	SMALL	STRETCH	ADULT	T
YELLOW	SMALL	STRETCH	CHILD	T
YELLOW	SMALL	DIP	ADULT	T
YELLOW	SMALL	DIP	CHILD	T
YELLOW	SMALL	STRETCH	ADULT	T
YELLOW	SMALL	STRETCH	CHILD	T
YELLOW	SMALL	DIP	ADULT	T
YELLOW	SMALL	DIP	CHILD	T
YELLOW	LARGE	STRETCH	ADULT	F
YELLOW	LARGE	STRETCH	CHILD	F

表 5.5 *ctr* 部分数据

Tab. 5.5 Part of data set ctr

x1	x2	x3	x4	x5	x6	x7	x8	x9	y
c	6	y	E	m	h	h	a	m	m
c	6	n	E	m	m	h	ma	m	m
c	6	n	E	m	h	h	ma	m	m
c	4	y	E	m	h	h	ma	l	h
c	6	n	E	m	m	m	ma	m	m
c	6	n	B	m	m	m	a	he	lo
c	6	n	E	m	m	h	ma	he	lo
s	4	n	B	sm	h	lo	ma	l	h
c	4	n	B	sm	h	lo	ma	m	m
c	4	n	B	sm	h	m	a	m	m

表 5.6 贷款部分数据

Tab. 5.6 Part of data set loan

省份	入学	生年	生月	性别	院系	金额(百元)	还款
四川	3	1982	12	男	力学与工程学院	48	到期未还
辽宁	4	1984	2	女	力学与工程学院	54	到期未还
辽宁	4	1985	2	男	力学与工程学院	54	到期未还
福建	2	1981	11	男	力学与工程学院	46	到期还
山西	2	1984	4	男	力学与工程学院	46	到期未还
内蒙古	4	1985	10	男	力学与工程学院	54	到期前还
辽宁	2	1983	5	男	力学与工程学院	46	到期前还
辽宁	4	1983	8	男	力学与工程学院	54	到期未还
辽宁	4	1985	11	女	力学与工程学院	54	到期未还
辽宁	4	1986	4	男	力学与工程学院	54	到期未还

表 5.7 规则覆盖率比较

Tab. 5.7 Comparison of Covering Rate

数据集	经典算法	JohnsonReducer	Genetic	Holte's 1R	SFA
iris					100%
new-threshold					100%
yellow-small					100%
ctr					100%
loan					100%

表 5.8 生成规则数比较

Tab. 5.8 Comparison of Generated Rules

数据集	经典算法	JohnsonReducer	Genetic	Holte's 1R	SFA
iris					13
new-threshold					
yellow-small					
ctr					9
loan					95

表 5.9 MRL 比较

Tab. 5.9 Comparison of $MeanRulesLength$

数据集	经典算法	JohnsonReducer	Genetic	Holte's 1R	SFA
iris					2.105
new-threshold					
yellow-small					
ctr					1.095
loan					0.145

参考文献

[1] Xiaoqing Chen, Taorong Qiu, Qing Liu, and Houkuan Huang. The framework of temporal granular logic based on information system. In *IEEE GrC 2006*, pages 604–606, 2006.

[2] Yaohua Chen and Yiyu Yao. Multiview intelligent data analysis based on granular computing. In *IEEE GrC 2006*, pages 281–286, 2006.

[3] Yong Chen and Yuehui Chen. Affective computing model based on rough fuzzy sets. In *Cognitive Informatics, 2006. ICCI 2006. 5th IEEE International Conference on*, volume 2, pages 835–838, Beijing, July 2006.

[4] Marek Chuchro. On rough sets in topological boolean algebras. In Wojciech Ziarko, editor, *Rough Sets, Fuzzy Sets and Knowledge Discovery*, pages 157–160. Springer, 1994.

[5] Tingquan Deng and Yanmei Chen. On reduction of morphological covering rough sets. In *FSKD 2006*, volume 4223 of *LNAI*, pages 266–275, 2006.

[6] D. Dubois and H. Prade. Rough fuzzy sets and fuzzy rough setsr. *International Journal of General Systems*, 17(2–3):191–209, 1990.

[7] Tao Feng, Jusheng Mi, and Weizhi Wu. Covering-based generalized rough fuzzy sets. In *RSKT 2006*, volume 4062 of *LNAI*, pages 208–215, 2006.

[8] G. Gates. The reduced nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory*, 18(3):431–433, May 1972.

[9] D. Bell H. Wang, I. Duntsch. Data reduction based on hyper relations. In P. Stolorz R. Agrawal and G. Piatetsky-Shapiro, editors, *Proceedings of KDD?8*, pages 349–353. Kluwer Academic Publishers, Boston, 1998.

[10] J. Hughes. Why Functional Programming Matters. *Computer Journal*, 32(2):98–107, 1989.

[11] Ron Kohavi and Brian Frasca. Useful feature subsets and rough set reducts, 1994.

[12] Yee Leung, Wei-Zhi Wu, and Wen-Xiu Zhang. Knowledge acquisition in incomplete information systems: A rough set approach. *European Journal of Operational Research*, 168:164–180, 2006.

[13] Dao-Guo Li, Duo-Qian Miao, and Yi-Qi Yin. Relation of relative reduct based on nested decision granularity. In *IEEE GrC 2006*, pages 397–400, 2006.

[14] Hong-Ru Li, Wen-Xiu Zhang, and Hong Wang. Classification and reduction of attributes in concept lattices. In *IEEE GrC 2006*, pages 142–147, 2006.

[15] Jinjin Li. Topological methods on the theory of covering generalized rough sets. *Pattern Recognition & Artificial Intelligence(in Chinese)*, 17(1):7–10, 2004.

[16] Zhe Liu and Yijie Li. A new heuristic algorithm of rules generation based on rough sets. In *Business and Information Management, 2008. ISBIM '08. International Seminar on*, volume 1, pages 291–294, Wuhan, Hubei, China,, December 2008.

[17] J. N. Mordeson. Algebraic properties of spaces in rough fuzzy set theory. In *Fuzzy Information Processing Society, 1999. NAFIPS. 18th International Conference of the North American*, pages 56–59, New York, NY, June 1999.

[18] S. Nanda. Fuzzy rough sets. *Fuzzy Sets and Systems*, 45:157–160, 1992.

[19] Aleksander Ohrn. *Discernibility and Rough Sets in Medicine: Tools and Applications*. PhD thesis, Norwegian University of Science and Technology, 1999.

[20] E. Orlowska. Incomplete information – rough set analysis. 1997.

[21] E. Orlowska. Introduction: What you always wanted to know about rough sets. In *Incomplete Information – Rough Set Analysis*, pages 1–20, 1997.

[22] Z. Pawlak. Rough sets. *Internat. J. Comput. Inform. Sci.*, 11:341–356, 1982.

[23] Z. Pawlak. *Rough sets: Theoretical aspects of reasoning about data*. Kluwer Academic Publishers, Boston, 1991.

[24] Zdzislaw Pawlak and Andrzej Skowron. Rough sets and boolean reasoning. *Information Sciences*, 177(1):41–73, 2007.

[25] Zdzislaw Pawlak and Andrzej Skowron. Rough sets: Some extensions. *Information Sciences*, 177(1):28–40, 2007.

[26] Zdzislaw Pawlak and Andrzej Skowron. Rudiments of rough sets. *Information Sciences*, 177(1):3–27, 2007.

[27] J. Pomykala. Approximation, similarity and rough constructions. In *ILLC Prepublication series, University of Amsterdam*, volume CT-93-07, 1993.

[28] P. Rojanavasu and O. Pinngern. Extended rough fuzzy sets for web search agent. In *Information Technology Interfaces, 2003. ITI 2003. Proceedings of the 25th International Conference on*, pages 403–407, June 2003.

[29] P. Mitra S. Pal, S. Mitra. Rough-fuzzy mlp: Modular evolution, rule generation, and evaluation. *IEEE Trans. On Knowledge and Data Engineering*, 15(1):15–24, 2003.

[30] Qiang Shen and Alexios Chouchoulas. Rough set-based dimensionality reduction for supervised and unsupervised learning. *International Journal of APPLIED MATHEMATICS AND COMPUTER SCIENCE*, 11(3):583–601, 2001.

[31] R. Slowinski and D. Vanderpooten. A generalized definition of rough approximations, 1996.

[32] Adam Szmigielski. Rough mereological localization and navigation. In *Rough Sets and Current Trends in Computing*, pages 629–638, 2002.

[33] E. C. C. Tsang, D. Chen, D. S. Yeung, X. Z. Wang, and J. W. T. Lee. Attributes reduction using fuzzy rough sets. *IEEE Transactions on Fuzzy Systems*, 16(5):1130–1141, October 2008.

[34] I. Burhan Türksen. Fuzzy sets, multi-valued mappings, and rough sets. In *Rough Sets and Current Trends in Computing*, volume 2475 of *LNCS*, page 60, 2002.

[35] Uci machine learning repository, 2005.

[36] Aida Vitória and Jan Maluszynski. A logic programming framework for rough sets. In *Rough Sets and Current Trends in Computing*, volume 2475 of *LNCS*, pages 205–212, 2002.

[37] T. Wakaki, H. Itakura, and M. Tamura. Rough set-aided feature selection for automatic web-page classification. In *Web Intelligence, 2004. WI 2004. Proceedings. IEEE/WIC/ACM International Conference on*, pages 70–76, September 2004.

[38] G. Y. Wang. Attribute core of decision table. In *Rough Sets and Current Trends in Computing*, volume 2475 of *LNCS*, pages 213–217, 2002.

[39] Qing-E Wu, Tuo Wang, Yong-Xuan Huang, and Ji-Sheng Li. New research on fuzzy rough sets. In *Machine Learning and Cybernetics, 2006 International Conference on*, pages 4178–4183, Dalian, August 2006.

[40] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning – I. *Information Sciences*, 8:199–249, 1975.

[41] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning – II. *Information Sciences*, 8:301–357, 1975.

[42] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning – III. *Information Sciences*, 9:43–80, 1975.

[43] Lotfi Zadeh. Toward a generalized theory of uncertainty (gtu): an outline. *Information Sciences*, 172(1–2):1–40, 2005.

[44] Jamie MacLennan ZhaoHui Tang. *数据挖掘原理与应用–– SQL Server 2005 数据库*. 清华大学出版社, 北京, 2007.

[45] Ning Zhong. Rough sets in knowledge discovery and data mining. *Journal of Japan Society for Fuzzy Theory and Systems*, 13(6):581–591, 2001.

[46] Feng Zhu and Hua-Can He. The axiomization of the rough set. *Chinese Journal of Computers*, 23(3):330–333, March 2000.

[47] Feng Zhu and Hua-Can He. Logical properties of rough sets. In *Proc. of The Fourth International Conference on High Performance Computing in the Asia-Pacific Region*, volume 2, pages 670–671. IEEE Press, 2000.

[48] William Zhu. Informed recognition in software watermarking. In *PAISI 2007*, volume 4430 of *LNCS*, pages 257C–261, 2006.

[49] William Zhu. Properties of the fourth type of covering-based rough sets. In *HIS'06, AUT Technology Park, Auckland, New Zealand, December 13-15, 2006*, pages 43–43, 2006.

[50] William Zhu. Properties of the second type of covering-based rough sets. In *Workshop Proceedings of GrC&BI 06, IEEE WI 06, Hong Kong, China, December 18, 2006*, pages 494–497, 2006.

[51] William Zhu. Topological approaches to covering rough sets. *Information Sciences*, 177(6):1499–1508, 2007.

[52] William Zhu and Fei-Yue Wang. Reduction and axiomization of covering generalized rough sets. *Information Sciences*, 152:217–230, 2003.

[53] William Zhu and Fei-Yue Wang. Properties of the first type of covering-based rough sets. In *Proceedings of DM Workshop 06, ICDM 06, Hong Kong, China, December 18, 2006*, pages 407–411, 2006.

[54] 史开泉，崔玉泉. *S-粗集与粗决策*. 科学出版社, 北京, 2006.

[55] 张文修，吴伟志，梁吉业，李德玉. *粗糙集理论与方法*. 科学出版社, 西安, 2001.

[56] 杨志民，刘广利. *不确定性支持向量机*. 科学出版社, 北京, 2007.

[57] 汪杭军，方陆明，张广群. 粗糙集在林业信息管理中的应用, 2006.

[58] 袁修久，张文修. 模糊粗糙集的包含度和相似度, 2005.

[59] 马廷淮，赵亚伟，曾振柄，张海盛. 基于粗糙集的决策规则约简, 2003.

致谢

本论文的完成，有太多的人需要感谢。

首先，感谢我的导师李义杰教授，感谢您在两年多的时间里对我的关心、支持和帮助。本文的成稿，与您的指导和培育和密不可分的。另外，在研究生学习阶段，您的博学和睿智都让我受益匪浅，还有您严谨的治学态度和对科学真理的追求，都让我铭记在心。这两年多来和您相处所得的一切，都将是我人生的一笔财富。

其次，感谢电子与信息工程学院的各位领导和老师，还有参加论文评审工作的各位老师。我的成长同样也离不开你们的辛勤培养。在学校的这段日子里，我学习和生活的方方面面无不包含在你们的工作和劳动中，谢谢你们。

再次，感谢我的师兄弟们、师姐妹和同学们。在两年多的时间里，遇到很多生活和学习上各种大小问题，在这个大家庭里，都很好地得到了解决，我们的友谊会天长地久。尤其感谢我师姐，谢谢你对我说过的每一句话和做过的每一件事，那些日子会是我一生中最珍贵的财富和最美好的回忆。

最后，感谢所有关心着我的人，在硕士期间，有太多的人需要感谢，有忘记提到的，在这里也一并表示我最衷心的感谢。

攻读学位期间发表的学术论文及科研成果

Zhe Liu and Yijie Li. A new heuristic algorithm of rules generation based on rough sets. In *Business and Information Management, 2008. ISBIM '08. International Seminar on*, volume 1, pages 291–294, Wuhan, Hubei, China,, December 2008