# 摘 要

[聚类分析](http://baike.baidu.com/view/903740.htm)是根据指定的规则，将给定的数据集按照相似性进行分类的一种统计分析方法。聚类分析在许多领域得到广泛应用，例如计算视觉、生物医学、信息检索、数据挖掘和模式识别等。

[粗糙集理论](http://baike.baidu.com/view/223951.htm)是一种处理不完整性和不确定性问题的[数学工具](http://baike.baidu.com/view/750697.htm)，它能有效分析不精确、不一致、不完整的各类信息，能从数据本身提供的信息中发现有效的、潜在的知识。

Spark是由加州大学伯克利分校AMP实验室开发的内存计算分布式框架，主要针对海量数据处理和机器学习。Spark采用Scala语言实现，提供集成的语言编程接口，使用户可以非常容易地编写并行任务。Spark基于内存的计算比Hadoop的MapReduce快100倍，Spark基于硬盘的运算也比MapReduce要快10倍。

本文研究了基于粗糙集理论的MMR和MTMDP层次聚类算法，这两种算法都应用于分类数据。MMR算法应用了粗糙度的概念来选择分裂属性，选择对象数目多的叶子结点继续进行分裂。MTMDP算法应用了概率粗糙集模型中粗糙隶属度的概念来选择分裂属性，选择聚合度小的叶子结点继续进行聚类。这两种算法主要的优点如下：（1）能够处理聚类过程中的不确定性问题；（2）仅仅只需要用户输入希望生成的簇数目就能得到良好的聚类结果，具有良好的鲁棒性；（3）能够适用于大数据的处理。为了测试MMR和MTMDP算法的性能，我们在四个真实的数据集上进行了实验，并和一些不稳定的聚类算法进行了比较。传统单机的聚类算法无论是从效率上，还是从计算复杂度上都已无法满足海量信息的处理需要，云计算技术的发展为聚类分析提供了新的研究方向。为了突破面对海量数据时的计算瓶颈，本文在Spark平台上实现了MMR和MTMDP算法的并行化。实验结果表明MMR和MTMDP并行算法均有较好的性能，并且都适合海量数据的处理。

关键词：聚类分析； 粗糙集理论； 分类数据； 云计算

# Abstract

Clustering analysis refers to dividing a given dataset into similar groups according to given rules, and the technique is widely applied in many domains, such as computer vision, biology, medicine, information retrieval, data mining, and pattern recognition.

Rough Set Theory is a mathematical tool to deal with incomplete and uncertain problems. It can effectively analyze the inaccurate, inconsistent and incomplete information. It can find valid and potentially useful knowledge in large amount of data.

Spark is a memory computing distributed framework developed by California Berkeley AMP Lab, which mainly aims at the mass data processing and machine learning. Spark provides a language-integrated programming interface in the Scala programming language, making it easy for users to write parallel jobs. Spark runs programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.

This thesis studies two hierarchical clustering algorithms, MMR and MTMDP, for categorical data based on Rough Set Theory. The MMR algorithm employs the concept of roughness to search the partitioning attribute and choose the leaf node with more objects for further splitting. The MTMDP algorithm employs the concept of rough membership degree based on probabilistic rough set theory to search the partitioning attribute and determines the further clustering node with less cohesion degree. The main advantages of the two algorithms are as follows: (1) it is capable of handling the uncertainty in the clustering process; (2) it is a robust clustering algorithm because it enables the users to obtain stable results by only one input: the number of clusters; (3) it has the capability of handling large datasets. In order to evaluate performance of MMR and MTMDP, we carried out experiments on four real datasets. In addition, we compared the two algorithms with some unstable clustering algorithms. In terms of efficiency or computational complexity, the traditional clustering algorithms conducted on a single machine are unable to meet the needs of massive information processing. The development of cloud computing technology provides a new direction for clustering analysis. In order to break through the bottleneck of massive data computing, the parallelization of MMR and MTMDP is implemented on the Spark platform. Experimental results show that the MMR and MTMDP parallel algorithms achieve better performance, and they are suitable for handling large amounts of data.

**Keywords:** Cluster analysis; Rough Set Theory; Categorical data; Cloud Computing

目 录

[摘 要 III](#_Toc422787915)

[Abstract IV](#_Toc422787916)

[第1章 绪论 1](#_Toc422787917)

[1.1 本论文的背景和意义 1](#_Toc422787918)

[1.2 国内外研究现状 2](#_Toc422787919)

[1.2.1 聚类分析的研究现状 2](#_Toc422787920)

[1.2.2 粗糙集理论的研究现状 2](#_Toc422787921)

[1.2.3 云计算的研究现状 3](#_Toc422787922)

[1.3 本论文的主要内容 3](#_Toc422787923)

[1.4 本论文的结构安排 3](#_Toc422787924)

[1.5 本章小结 4](#_Toc422787925)

[第2章 相关知识概述 5](#_Toc422787926)

[2.1 数据挖掘 5](#_Toc422787927)

[2.1.1 数据挖掘的定义 5](#_Toc422787928)

[2.1.2 数据挖掘的基本过程 5](#_Toc422787929)

[2.2 聚类分析 5](#_Toc422787930)

[2.2.1 聚类分析的数据类型 5](#_Toc422787931)

[2.2.2 聚类分析中的相似度量方法 7](#_Toc422787932)

[2.2.3 聚类分析方法简介 8](#_Toc422787933)

[2.2.4 聚类分析的有效性 9](#_Toc422787934)

[2.3 粗糙集理论 10](#_Toc422787935)

[2.4 Spark平台介绍 11](#_Toc422787936)

[2.4.1 Spark的体系结构 12](#_Toc422787937)

[2.4.2 Spark的核心概念 12](#_Toc422787938)

[2.4.3 Spark编程接口 14](#_Toc422787939)

[2.5 本章小结 15](#_Toc422787940)

[第3章 算法设计与分析 16](#_Toc422787941)

[3.1 MMR算法介绍与分析 16](#_Toc422787942)

[3.1.1 MMR算法介绍 16](#_Toc422787943)

[3.1.2 MMR算例分析 18](#_Toc422787944)

[3.1.3 MMR算法分析 21](#_Toc422787945)

[3.2 MTMDP算法介绍与分析 21](#_Toc422787946)

[3.2.1 MTMDP算法介绍 21](#_Toc422787947)

[3.2.2 MTMDP算例分析 25](#_Toc422787948)

[3.2.3 MTMDP算法分析 28](#_Toc422787949)

[3.3 MMR和MTMDP算法的对比分析 29](#_Toc422787950)

[3.4 基于Spark的MMR和MTMDP并行算法设计 32](#_Toc422787951)

[3.4.1 Spark中的RDD操作 32](#_Toc422787952)

[3.4.2 MMR和MTMDP并行化的基本思想 32](#_Toc422787953)

[3.4.3 基于Spark的MMR和MTMDP并行算法的设计 34](#_Toc422787954)

[3.5 本章小结 39](#_Toc422787955)

[第4章 实验结果与分析 40](#_Toc422787956)

[4.1 实验平台 40](#_Toc422787957)

[4.1.1 硬件支持 40](#_Toc422787958)

[4.1.2 软件支持 40](#_Toc422787959)

[4.1.3 实验平台的部署 40](#_Toc422787960)

[4.2 实验过程与分析 43](#_Toc422787961)

[4.2.1 实验一：聚类结果分析 43](#_Toc422787962)

[4.2.2 实验二：串行和并行对比分析 47](#_Toc422787963)

[4.2.3 实验三：集群并行化加速比分析 49](#_Toc422787964)

[4.3 本章小结 51](#_Toc422787965)

[结 论 52](#_Toc422787966)

[致 谢 53](#_Toc422787967)

[参考文献 54](#_Toc422787968)

# 第1章 绪论

## 1.1 本论文的背景和意义

在日常生活、生产、科研工作中，经常要对被研究对象进行分类。研究和处理给定对象的分类常用的方法就是聚类分析。聚类分析是一种重要的人类行为，是根据指定规则，将给定的数据集按照相似性进行分类的一种统计分析方法。聚类[1]是使得同一个簇中的对象之间具有较高的相似性，而不同簇中的对象具有较大的相异性。聚类分析在许多领域得到广泛应用，例如计算视觉、生物医学、信息检索、数据挖掘和模式识别等[2]。

近年来聚类分析方面的研究发展迅速，诞生了许多优秀的聚类算法，但是许多已有的聚类算法存在一个共同的不足之处，它们在聚类过程中需要依赖以往的经验数据或者先验知识，否则无法达到好的聚类效果。[粗糙集理论](http://baike.baidu.com/view/223951.htm)是一种刻划不完整性和不确定性的[数学工具](http://baike.baidu.com/view/750697.htm)，它能有效分析不精确、不一致、不完整的各类信息，还可以对数据进行分析和推理，从中发现隐含的知识，揭示潜在的规律[3]。因此非常适合处理聚类分析这样的不确定性问题。粗糙集理论的最大优势在于，它在处理问题过程中不需要依赖任何以往经验数据或者先验知识。

随着互联网和信息行业的发展，数据正在迅速膨胀。比如一分钟内，微博和Twitter上新发的数据量就超过了10万，社交网络Facebook的浏览量超过600万，大数据时代对数据的处理又提出了新的需求。面对大规模数据集和高维数据类型时，传统的单机处理无论是从运算能力或时间效率都无已法胜任实际需求，随着云计算技术的发展为聚类分析提供了新的研究方向。

本文研究了基于粗糙集模型的两种分层聚类算法MMR（Min-Min-Roughness）和MTMDP（Maximum Total Mean Distribution Precision），这两种聚类算法应用于分类数据，克服了传统聚类算法在聚类过程中需要依赖以往的经验数据或者先验知识的缺点，仅仅根据输入的数据信息和希望的簇数目就能得到很好的聚类结果。本文实现了这两种聚类算法，并在四个真实的数据集上进行了测试，分析了两种算法的性能。本文还分析了在Spark平台上并行实现这两种算法的可行性，同时在云计算平台Spark[4]上并行实现了这两种聚类算法，并在Spark平台上测试了两种算法的加速比。

## 1.2 国内外研究现状

### 1.2.1 聚类分析的研究现状

在数据挖掘研究方向，聚类分类数据获得了人们大量的关注。Huang等人根据简单匹配度测量分类数据的相异性，扩展了标准的k-means算法，提出了k-modes聚类算法[5]。分类数据相异度测量是简单地根据不同属性值的数目，计算两个对象的相异性。k-modes算法改进了k-means，使用一种基于频率的方法更新中心，使聚类的代价最小。k-modes算法包含了所有k-means的优点，并且能够解释聚类的结果。但是k-modes算法依赖于初始模型和处理顺序，产生局部的最优结果。另外k-modes算法规定一个对象只能属于一个簇，并且属于每个簇的置信度都相同。在现实世界中，数据的边界是很难区分的，所以每个簇之间往往没有很明确的边界。所以Huang和Ng提出了基于k-modes算法基础的fuzzy-k-modes[6]算法，分配隶属度给不同的簇。但是该算法依然受到初始模型和处理顺序的影响，因此fuzzy-k-modes需要调整一个控制参数来获得更好的结果。在实际应用中我们不知道如何获得最优的参数，而最优参数的值往往依赖以前的经验和先验信息。

近年来粗糙集(Rough Set)理论在聚类分析方面引起了广泛的研究。比如Cheng和Wang基于粗糙集理论和香农理论改进了聚类算法，Lingras和West提出了粗糙的k-means算法，并且用该算法来分析学生Web的访问日志。Maji和pal提出了一种基于粗糙集理论的选择最佳生物属性的fuzzy c-medoids算法。以上提到的这些算法不是有收敛缺陷，就是有稳定性问题。Chen等人提出了一种适用于分类数据的基于粗糙集的层次聚类算法，但是该算法的时间复杂度达到了，其中*n*是指对象数目，*m*是指属性数目。

### 1.2.2 粗糙集理论的研究现状

粗糙集理论是Pawlak教授于1982年提出的一种能够定量分析处理不精确、不一致、不完整信息与知识的数学工具[7]。粗糙集理论最初的原型来源于比较简单的信息模型，它的基本思想是通过关系数据库分类归纳形成概念和规则，通过等价关系的分类以及分类对于目标的近似实现知识发现。由于粗糙集理论思想新颖、方法独特，已成为一种重要的智能信息处理技术，该理论已经在机器学习与知识发现、数据挖掘、决策支持与分析等方面得到广泛应用。

粗糙集理论发展三十多年来，无论在理论研究还是应用研究上均取得了良好的成果。从1992年至今,每年都召开以粗糙集为主题的国际会议，主要有RSCTC、RSFDGrC和RSKT。在国内从2001年开始每年举办粗糙集与软计算学术会议。此外还有很多基于粗糙集理论的KDD(Knowledge Discovery in Database)系统已经建立，其中最具有代表性的有KDD-R，RoughDAS&RoughClass[8]，LERS[9]，Rosetta和RIDAS[10]等。

### 1.2.3 云计算的研究现状

云计算[11]是一种新的服务模式和应用，它基于分布式的拓扑结构和通过网络连接大量的计算机节点，这样可以使用户得到超大规模的且具有伸缩性、高可靠性的计算机资源，从而能以低廉的价格来处理大规模数据或者得到充足的存储空间。面向企业级的云计算服务包括：基础设施服务（IaaS）、平台即服务（PaaS）和软件即服务（SaaS）三种，面向普通用户的云计算服务则更倾向于云存储和云安全。[云计算](http://baike.baidu.com/view/1316082.htm)是[分布式计算](http://baike.baidu.com/view/30655.htm)、[并行计算](http://baike.baidu.com/view/1666.htm)、[效用计算](http://baike.baidu.com/view/5246792.htm)、[网络存储](http://baike.baidu.com/view/600264.htm)、[虚拟化](http://baike.baidu.com/view/729629.htm)、[负载均衡](http://baike.baidu.com/view/51184.htm)、热备份[冗余](http://baike.baidu.com/subview/104445/10102121.htm)等传统[计算机](http://baike.baidu.com/view/3314.htm)和[网络技术](http://baike.baidu.com/view/25363.htm)发展融合的产物。

## 1.3 本论文的主要内容

本论文的研究内容如下：

（1）聚类分析的基本概念和主要方法，重点研究了层次聚类算法。

（2）粗糙集理论的基本概念：等价类、上近似、下近似、粗糙度、粗糙隶属度等。

（3）了解了Spark云计算平台的编程模型，重点研究了RDD的变换和动作。

（4）基于粗糙集模型的MMR层次聚类算法的基本概念和算法的实现，分析了MMR算法的优缺点。根据MMR算法的特点，结合Spark的编程模型实现了MMR算法的并行化。

（5）基于粗糙集模型的MTMDP层次聚类算法的基本概念和算法的实现，分析了MTMDP算法的优缺点，和MMR算法进行了对比分析。根据MTMDP算法的特点，结合Spark的编程模型实现了MTMDP算法的并行化。

（6）在Soybean、Zoo、Wisconsin Breast Cancer和Mushroom四个数据集上进行了实验，并且选择了三个常用的聚类结果评价指标Overall purity、ARI、NMI对MMR和MTMDP算法的聚类结果进行评估。

（7）在Spark云平台上对并行的MMR和MTMDP算法进行性能分析，完成了串行和并行对比分析和集群加速比分析两个实验。

## 1.4 本论文的结构安排

1. 简要地介绍了论文的研究背景、目的及意义。阐述了本文的主要方法和论文的主要内容与结构安排。
2. 概述了本文相关的理论知识。介绍了数据挖掘的定义和基本过程，详细介绍了聚类分析的数据类型、相似度量、主要方法和评价指标，还介绍了粗糙集理论的基本概念，最后介绍了实验平台Spark的体系结构和特点。
3. 详细介绍了MMR和MTMDP算法的设计与实现。首先介绍了MMR算法相关的定义，然后介绍了MMR算法的主要思想，然后给出了一个具体的实例分析MMR的具体过程。再详细介绍了MTMDP算法并与MMR算法进行了对比分析。最后介绍了基于Spark平台的并行算法设计与实现。
4. 首先介绍了Spark集群的搭建过程，实验的硬件环境和软件配置。然后介绍了基于Spark的MMR和MTMDP算法聚类结果分析实验、串行和并行对比分析实验、集群加速比分析实验三个实验的详细过程，并且分析了实验结果。

最后，对本文的研究工作进行总结，对未来的研究工作进行了展望

## 1.5 本章小结

本章首先介绍了本课题的背景和意义，说明了聚类分析在各领域的广泛应用，介绍了传统聚类分析方法的不足以及本论文拟研究的两种层次聚类算法MMR和MTMDP的特点。接着，介绍了聚类分析、粗糙集理论和云计算的国内外研究现状。最后概述了本论文的主要内容和结构安排。

# 第2章 相关知识概述

## 2.1 数据挖掘

数据挖掘(Data Mining)一般是指从大量的数据中通过算法搜索隐藏于其中信息的过程，数据挖掘又称数据库中的知识发现，是人工智能和数据库领域研究的热点问题。数据挖掘主要基于机器学习、模式识别、统计学、人工智能、可视化技术等，分析大量的数据，挖掘潜在的模式，获取的信息和知识可以广泛地应用于市场分析、欺诈检测、产品控制和科学探索等。

### 2.1.1 数据挖掘的定义

数据挖掘就是从大量的、不完整的、有噪声的、不一致的真是世界中的数据，提取出未知的、有用的信息和知识的过程[12]。其中数据源必须是真实的、大量的、含噪声的，挖掘出的知识是可理解、可运用的。从商业的角度上，数据挖掘可以描述为，按照企业既定业务目标，对大量的企业数据进行探索和分析，揭示隐藏的、未知的规律，并进一步将其模型化的先进有效方法。

### 2.1.2 数据挖掘的基本过程

数据挖掘包括三个阶段：数据准备、数据挖掘、结果的表达与解释。其中数据准备阶段又包括：信息收集、数据清理、数据集成、数据选择、数据变换、数据挖掘、模式评估和知识表示等步骤[13]。

## 2.2 聚类分析

将物理或抽象对象的集合分成相似的对象类的过程称为聚类，簇是数据对象的集合，这些对象与同一个簇中的对象彼此相似，而与其他簇中的对象相异[14]。尽管分类是一种有效的手段，但是它常常需要高昂的代价收集和标记大量训练元组集或模式，以便分类法使用它们对每个组建模。

在聚类分析中主要涉及三个要素，分别是相似性度量、聚类算法、聚类准则函数，一般的聚类分析方法是通过多次迭代执行找出使得聚类准则函数达到最优的聚类结果。一般聚类分析的流程图如图2-1所示。

### 2.2.1 聚类分析的数据类型

假设聚类的数据集合包含个数据对象，这些对象可能代表人、房子、文档、国家等，基于内存的聚类算法通常可以采用以下两种数据结构运行：

图2-1 聚类分析流程图

Y

数据预处理

定义距离函数

按给定规则进行聚类

评估和输出

满意

开始

结束

调整参数

N

（1）数据矩阵（data matrix，或称为对象与变量结构）：它用*p*个变量（也称为度量或属性）来表现*n*个对象，例如用年龄、身高、体重、性别、种族等属性来表现对象“人”。这种数据结构是关系表的形式，或者看成*n*×*p*（*n*个对象×*p*个变量）矩阵：

 (2-1)

（2）相异度矩阵（dissimilarity matrix，或称为对象－对象结构）：存储所有成对的*n*个对象两两之间的近似性，通常用一个*n*×*n*维的矩阵表示，其中是对象和对象之间的测量差或相异度，通常是一个非负的数值，当对象和越相似或“接近”时，其值越接近0；两个对象越不同，其值越大。其中，并且。

数据矩阵的行和列代表不同的实体，而相异度矩阵的行和列代表相同的实体。因而，数据矩阵经常称为二模(two-mode)矩阵，而相异度矩阵称为单模(one-mode)矩阵。很多的聚类算法都使用相异度矩阵进行聚类。

(2-2)

### 2.2.2 聚类分析中的相似度量方法

在数据挖掘中聚类分析中经常出现的数据变量类型有区间标度变量、二元变量、分类变量、序数型变量、比例标度变量、混合类型变量和向量变量等[15]。不同的数据类型之间有不同的相异度测量方法。

（1）区间标度变量：区间标度变量是一个粗略现行标度的连续度量。典型的例子包括重量和高度、经度和纬度坐标（如聚类房屋），以及大气温度。其距离度量一般包括欧几里得距离、曼哈顿距离以及闵可夫斯基距离。

欧几里得距离定义如下：

 (2-3)

其中和表示两个维的数据对象。

曼哈顿（或城市块）距离定义如下：

 (2-4)

闵可夫斯基距离定义如下：

 (2-5)

（2）二元变量：只有两个状态：0或1，0表示该变量不出现，1表示该变量出现。例如，给出一个描述病人的变量smoker，1表示病人抽烟，而0表示病人不抽烟。如果像对待区间标量那样处理二元变量会产生误导的聚类结果，所以二元变量要采用特定的方法来计算其相异度。如果一个二元变量的两个状态具有同等价值和相同的权重，则该二元变量是对称的。如果输出的状态不是同等重要的，比如疾病的判断，这样的二元变量是非对称的。

对称二元相异度定义如下：

 (2-6)

非对称相异度定义如下：

 (2-7)

其中表示两个对象，是对象和值都为1的变量数目，是对象值为1，但是对象值为0的变量数目，是对象值为0，但是对象值为1的变量数目，t是对象和值都为0的变量数目。

（3）分类变量：分类变量是二元变量的推广，它可以取于两个的状态值，例如颜色是一个分类变量，它可能有多种状态：红、绿、蓝等。

分类变量的相异度定义如下：

 (2-8)

其中，是匹配的数目（即对象和对象取值相同状态的变量的数目），表示总的变量数目。

（4）向量对象：在某些应用中，如在信息检索、文本文档聚类和生物学分类中，需要对包含了大量符号实体（如关键词和短语）的复杂对象（如文档）进行比较和聚类。为了测量复杂对象间的距离，通常期望放弃传统的度量距离计算，而引入非度量的相似度函数。

向量对象的余弦相似度定义如下：

 (2-9)

其中表示两个向量，是向量的转置，是向量的欧几里得范数，是向量的欧几里得范数，本质上是向量之间的夹角的余弦值。

### 2.2.3 聚类分析方法简介

没有任何一种聚类算法可以普遍适用于揭示各种多维数据集所呈现出来的多种多样的结构[16]。根据数据的积聚规则以及应用这些规则的方法，有多种聚类算法。主要的聚类算法包括以下几种。

（1）基于划分的方法：划分式聚类算法需要预先指定聚类数目或聚类中心，通过反复迭代运算，逐步降低目标函数的误差值，当目标函数值收敛时，得到最终聚类结果。典型的划分聚类算法有k-means和k-modes。

（2）基于层次的方法：它使用数据的联接规则，透过一种层次架构方式，反复将数据进行分裂或凝聚，将数据对象集的层次分解。典型的层次聚类算法有CURE和Chameleon。

分裂的层次聚类算法：使用自顶向下的策略。开始它把所有对象置于一个簇中，该簇是层次结构的根。然后它把根上的簇划分成许多个较小的子簇，并且递归地把这些簇划分成更小的簇，划分过程继续，直到每个对象自成一簇，或者达到了某个终止条件，例如达到了某个希望的簇数目，或者某个簇的直径都在某个阈值内。

凝聚的层次聚类：使用自底向上的策略，首先每个对象作为其簇，然后合并这些原子簇为越来越大的簇，直到所有的对象都在一个簇中，或者满足某个终止条件。

本文中采用分裂的层次聚类算法，并且一旦分裂执行就不能修正，所以如何选择分裂的属性和选择继续分裂的节点显得特别关键。

（3）基于密度的方法：只要“邻域”中的密度（对象或数据点的数目）超过某个阈值，就继续增长给定的簇。也就是说，对给定类中的每个数据点，在一个给定的区域内必须至少包含某个数目的点。这样的方法可以用来过滤“噪声”孤立点数据，可以发现任意形状的簇。典型的基于密度的聚类算法有DBSCAN、OPTICS和DENCLUE。

（4）基于网格的方法：该方法使用一种多分辨率的网格数据结构。它将对象空间量化为有限数目的单元，形成网格结构，所有的聚类操作都在网格上进行。基于网格的聚类算法常常与其它方法相结合，特别是与基于密度的聚类方法相结合。这种方法的主要优点是处理速度快，其处理时间独立于数据对象的数目，仅依赖于量化空间中每一维上的单元数目。典型的网格聚类算法有STING和CLIQUE。

（5）基于模型的方法：试图优化给定数据和数据模型之间的拟合。这类方法经常基于这样的假设：数据根据潜在的混合概率分布生成。典型的基于模型的聚类算法有EM（期望最大）。

目前针对聚类算法的研究主要着重于两个方面：第一，聚类对象的输入顺序以及初始聚类中心的选择方面。这是聚类研究中的一个重点，因为这两个因素会直接影响到聚类结果的好坏，如何能使最终的聚类结果达到全局最优是进一步研究的方向。第二，关于算法执行效率的方面。除了聚类的质量，效率也是算法优劣至关重要的考量因素。随着数据量的急剧增加，如何改进现有聚类算法，提高算法效率，使其能够快速有效处理大规模数据成为亟待解决的另一个研究课题。

### 2.2.4 聚类分析的有效性

聚类分析是一个富有挑战性的研究领域，它的潜在应用提出了各自特殊的要求，数据挖掘对聚类的要求很多，下面介绍一些常用的聚类分析评价指标。

（1）可伸缩性：能否无论是在小数据集还是包含百万个对象的大数据集上，都能够取得良好的聚类结果。

（2）处理不同数据类型属性的能力：能否处理区间变量、二元变量、分类变量、序数变量等混合的数据类型。

（3）发现任意形状的聚类：许多聚类算法基于欧几里得或曼哈顿距离度量来确定簇，基于这种距离度量的算法只能发现球状形的簇，但是簇的形状可能是任意的。所以能否发现不同形状的簇是很重要的。

（4）对于决定输入参数的领域知识需求最小：许多的聚类算法在聚类分析中要求用户输入一定的参数，比如期望的簇的数目，聚类结果又对于输入参数非常敏感，而面对实际的问题时恰当参数的选取很困难，对参数依赖过大会使聚类质量难以控制。所以能通过算法对相应数据判断自行聚类的算法更好。

（5）处理带噪声数据的能力：绝大多数真实世界中的数据库都包含了离群点、缺失、未知或者错误的数据。一些聚类算法对于这样的数据敏感，可能会导致低质量的聚类。

（6）增量聚类和对输入记录的次序不敏感：一些聚类算法不能讲新插入的数据合并到已经的聚类结构中去，只能重新聚类。一些算法以不同的输入顺序提供对象时可能会产生不同的聚类结果。开发增量聚类算法和对输入数据不敏感的算法具有重要的意义。

## 2.3 粗糙集理论

[粗糙集理论](http://baike.baidu.com/view/223951.htm)是一种刻划不完整性和不确定性的[数学工具](http://baike.baidu.com/view/750697.htm)，它能有效分析不精确、不一致、不完整的各类信息，还可以对数据进行分析和推理，从中发现隐含的知识，揭示潜在的规律[17]。粗糙集理论的最大优势在于，它在处理问题过程中不需要依赖任何以往经验数据或者先验知识。粗糙集近年来受到越来越多的关注，应用越来越广泛，已经成为人工智能理论及其应用领域中的研究热点之一。

（1）信息系统：一般情况下，结构化数据都可以用一张表来存储，每一行记录一个对象相关的事实。一张数据表又叫做一个信息系统。更准确的说，信息系统(IS)，通常可以按照下面的格式来描述：



是一个有限的非空的对象的集合，叫做全集；

是非空的有限的属性的集合；

，是指属性*a*的值域，是一个有限的无序的集合；

是一个映射，是对象关于属性的一个映射；

（2）不可分辨关系：对于任何的一个集合，都有一种不可分辨关系，不可分辨关系表示如下：

，和分别是对象*x*和对象*y*在条件属性*a*上的取值。在某属性集*B*上的不可分辨关系，就是在属性集*B*上取相同值的对象集合。

（3）等价类：不可分辨关系把全集划分成互斥的几个类，是根据划分的等价类，其中。明显地，如果两个对象属于一个相同的等价类，相对于属性集*B*这两个对象是不可分辨的。等价类也称为基础集。

（4）上近似、下近似：对任意的集合集合可以近似的用两个关于属性*B*的集合来描述，和，和分别叫做属性集*B*关于集合*X*的下近似和上近似。下近似表明了根据知识*B*判定肯定属于*X*的*U*中元素组成的集合，即如果一个对象，那么这个对象一定属于集合*X*。上近似表明了根据知识*B*判断可能属于集合*X*的*U*中的元素组成的集合，即如果一个对象，那么这个对象有可能属于集合*X*。

（5）粗糙度：集合的不精确性是由于边界区域的存在而引起的，集合的边界区域越大，其精确性就越低，为了更精确的表达这一点，引入了粗糙度这个概念，*X*是*U*的一个子集，集合*X*相对于属性集*B*的粗糙度定义如下：

 (2-10)

其中表示集合的对象数目，显然，如果，那么，集合*X*关于属性集*B*是没有边界的，即集合*X*关于属性集*B*是清晰的。如果，显然，那么集合*X*关于属性集*B*就是有边界的，*X*关于属性*B*是粗糙的。粗糙度反映了我们根据知识*B*对集合*X*的了解程度。

（6）粗糙隶属度：给定一个属性集*B*，给定一个对象，粗糙隶属度是指对象属于集合*X*的概率()，对象*x*属于集合*X*的粗糙隶属度用表示，计算公式如下：

 (2-11)

## 2.4 Spark平台介绍

Spark是近几年发展迅速的云平台，已经成为了Apache顶级的孵化项目，是由UC Berkeley AMP实验室开发的开源通用并行云计算平台，目的是加快数据分析的速度。Spark基于MapReduce[18]思想实现分布式计算，但是采用了内存分布式数据集，引进了内存集群计算的概念，可以将运算的中间结果存储在内存中，从而不再需要读写HDFS，因此Spark能更好地运行图应用、机器学习等迭代式的算法和交互式的数据挖掘算法。Spark内核是用Scala语言开发的，Scala是一门集成了面向对象编程和函数式编程的各种优点的语言，能将操作分布式数据集像操作本地集合对象一样容易，使用户可以非常便捷地编写并行任务。

### 2.4.1 Spark的体系结构

目前，Spark已经发展成为包含各式各样子项目的大数据计算平台，Spark的体系结构如图2-2所示。其核心框架是Spark，同时涵盖了支持结构化数据SQL查询与分析的查询引擎Spark SQL，提供了Hive命令接口的Shark方便了用户对数据的操作，提供了底层的分布式机器学习库Mlib、基于BSP模型的并行计算图框架GraphX、流计算框架Streaming，实现了Google的PageRank算法的Bagel等子项目。这些子项目在Spark上层提供了更高层、更丰富的计算框架。



图2-2 Spark的体系结构

### 2.4.2 Spark的核心概念

Spark是一个基于内存计算的大数据分布式编程框架，Spark的核心是RDD（弹性分布式数据集），RDD是Spark最基本的抽象，是对分布式内存的抽象使用，实现了以操作本地集合的方式来操作分布式数据集。RDD表示已经被分区，不可变的能够被并行操作的数据集合，RDD必须是可序列化的，可以缓存到内存中，省去了MapReduce大量的磁盘IO操作，提升了很大的效率。

RDD只能从持久存储或通过Transformations变换操作产生，相比于分布式共享内存（DSM）可以更高效实现容错，对于丢失部分数据分区只需根据创建该RDD的一系列转换记录即Lineage就可重新计算出来，而不需要做特定的Checkpoint。RDD和共享内存(DSM)的对比结果如表2-1所示。RDD的不变性，可以实现类似Hadoop MapReduce的推测式执行。RDD的数据分区特性，可以通过数据的本地性来提高性能，这与Hadoop MapReduce是一样的。RDD都是可序列化的，在内存不足时可自动降级为磁盘存储，把RDD存储于磁盘上，这时性能会有大的下降但不会差于现在的MapReduce。

总结RDD的特点如下：

（1）在集群结点上不可变的、已分区的集合对象；

（2）通过并行转换的方式创建如map、filter、join、groupBy等；

（3）失败时自动重建；

（4）可以控制存储级别，根据内存、磁盘等进行重用；

（5）必须是可序列化的；

（6）是静态类型的。

在RDD空间中，每个RDD都可以使用5个方面的特性来表示：

（1）分区列表（数据块列表）；

（2）计算每个分片的函数，根据父RDD计算出此RDD；

（3）对父RDD的依赖列表；

（4）对Key-Value RDD的Partitioner；

（5）每个数据分片的预定义地址列表（如HDFS上的数据块的地址）。

表2-1 RDD与DSM对比

|  |  |  |
| --- | --- | --- |
| 对比项目 | RDD | 分布式共享内存（DSM） |
| 读 | 批量或细粒度操作 | 细粒度操作 |
| 写 | 批量转换操作 | 细粒度操作 |
| 一致性 | 不重要（RDD是不可更改的） | 取决于应用程序或运行时 |
| 容错性 | 细粒度，低开销（使用Lineage） | 需要检查点操作和程序回滚 |
| 落后任务的处理 | 任务备份 | 很难处理 |
| 任务安排 | 基于数据存放的位置自动实现 | 取决于应用程序（通过运行时实现透明性） |
| 如果内存不够 | 与已有的数据流系统类似 | 性能较差 |

对于RDD可以有两种操作方式：变换（返回值仍然是一个RDD）与动作（返回值不是一个RDD）。变换如map、filter、groupBy和join等。变换操作是Lazy的，也就是说从一个RDD转换生成另一个RDD的操作不是马上执行，当Spark在遇到变换操作时只会记录需要这样的操作，并不会去执行，直到有动作操作的时候才会真正启动计算过程进行计算。动作是向应用程序返回值，或向存储系统导出数据的那些操作，例如，count（返回RDD中的元素个数），collect（返回元素本身），save（将RDD输出到存储系统）。在Spark中，只有在动作第一次使用RDD时，才会计算RDD（即延迟计算）。这样在构建RDD的时候，运行时通过管道的方式传输多个变换。

### 2.4.3 Spark编程接口

Spark用Scala语言实现了RDD的API。Scala是一种基于JVM的静态类型、函数式、面向对象的语言。Scala语言简洁（特别适合交互式使用）、有效（JVM上的强制静态类型），方便用户编写并行程序。

Spark由Master主节点和Workers从节点组成，用户编写的Spark程序被称为Driver程序，Dirver程序会连接到集群以运行Worker，Driver程序定义了一个或多个RDD，并调用RDD上的变换与动作。对RDD的变换与动作通过Scala闭包（字面量函数）来表示，Scala使用Java对象来表示闭包，并且都是可序列化的，以此把对RDD的闭包操作发送到各Workers节点。Workers存储着数据分块和享有集群内存，是运行在工作节点上的守护进程，当它收到对RDD的操作时，根据数据分片信息进行本地化数据操作，生成新的数据分片、返回结果或把RDD写入存储系统。如图2-3所示，用户的driver程序启动多个worker，worker从分布式文件系统中读取数据块，并将计算后的RDD分区缓存在内存中。

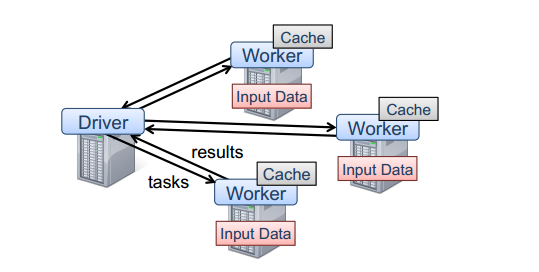


图2-3 Spark程序的运行图[19]

## 2.5 本章小结

本章首先介绍了数据挖掘的定义和基本过程。然后介绍了聚类分析的数据类型、不同数据类型的距离度量，再简要概述了主要的聚类分析方法，以及聚类分析常用的评价指标。本章还介绍了粗糙集的信息系统、等价类、上近似、下近似、粗糙度和粗糙隶属度等基本概念。最后介绍了本次实验平台Spark的体系结构、核心概念RDD和编程接口。