XML 文档的聚类算法及实现

学　　院　计算机科学与技术

专　　业　计算机科学与技术

年　　级　2006

姓　　名　李健

指导教师　张坤龙

2010 年 6 月 15 日

摘　　要

XML 文档聚类在很多应用方面起着重要的作用，比如查找相似文档，比较化学混合物，研究基因相似性等等。

为了更好地聚类 XML 文档集合，针对 XML 文档的结构特点，实现一种基于扩展邻接矩阵相似性计算的 k-means 算法。该算法通过计算模式文档和数据源文档之间的余弦相似度来确定两个 XML 文档之间的距离。在计算文档之间相似度方面，论文通过大量的数据和实验证明了它的算法效果较边集比较法理想。此外，传统的 k-means 算法对初始中心十分敏感，聚类结果随不同的输入而进行波动，容易陷入局部最优。为了消除这种敏感性，论文的算法还采用了快速查找初始中心点的思想，这样改进后的聚类算法不仅能够产生质量较高的聚类结果，而且还消除了对初始输入的敏感性。

关键字：聚类；k-means 算法；XML 文档结构；余弦相似度；边集比较法；初始中心；

ABSTRACT

The clustering of XML documents plays a crucial role in many applications such as in searching similar documents, in comparing chemical compounds, in finding genetic similarities,etc.

This paper implements a new k-means algorithm that bases on external adjacent matrix by computing similarity of XML documents for grouping XML documents ,according to XML documents' structure information. It determine the distance of each two XML documents by computing cosine similarity between model document and data source document, and it is compared with edge collection comparison in the calculation of similarity, which is proved that this new method is effective. What's more, the traditional k-means algorithm is very sensitive for selecting the initial center. Clustering solution fluctuates with different input, which is easy to reach local optimum. For eliminating this sensitivity, the algorithm uses the idea that can locate initial center quickly. Therefor, the modified clustering algorithm can not only generate better results,but also eliminate the sensitivity for input.

Key words: Cluster;K-means algorithm;XML document structure;Cosine similarity; Edge collection comparison;Initial center;

目　　录

外文资料

中文译文

致　　谢

第一章　　绪论

1.1　课题背景及来源

随着我国城市化和信息化进程的加快，互联网成为人们日常生活的必须品。每天都会有大量的人们浏览网页并搜索信息，它所蕴含的海量信息大大超过了用户可能的阅读量。这些信息很多是以XML作为数据表示和交换的标准。为了提高其效率，方便浏览更多的相关信息，就必须有一种管理网页的机制来把问题的相关内容整合起来，使得操作者能够尽快找到自己所需。对提高人们的生活效率和质量保证具有重要的意义。

XML的全称是可扩展标记语言，通过文档的元素节点之间的层次包含来表现数据之间的隶属关系。这种模式具有简单直观便于处理的优点，为了满足基于web的XML数据信息的挖掘的要求，提出一种基于结构相似性的聚类算法。聚类算法被广泛应用于机器学习、统计分析、模式识别以及数据挖掘与知识发现等不同领域，是分析数据并从中发现有用信息的一种有效手段。该方法是在根据XML文档的结构及其语义信息计算出相似度的基础之上对XML文档集合进行聚类，并进行了实验。

该课题以南开大学数据库与信息系统研究室为依托，属于863项目及天津市科技计划重点项目研究和开发的一部分。

1.2　国内外发展现状

XML（可扩展标记语言）正在成为Web数据交换的标准格式。随着XML格式的半结构数据的大量出现，如何处理和管理好大量的XML文档已经成为一个研究的热点。XML文档聚类作为XML数据处理的重要课题，是指将具有类似特征的XML文档聚集成簇。现今，国内这方面起步较晚，略有一些成果，相比之下，国外的发展速度显而易见，早已有优秀的科研人员投入到这方面的研究当中。

一个聚类算法的灵魂就是其相似度的计算，其思想主要是基于结构信息、语义信息及其二者的结合。

结构信息主要是XML文档的树形结构，现今比较成熟的方法有点集比较法、边集比较法和编辑距离法。语义信息就是利用XML文档每个节点的文本信息，这方面在国内外发展比较成熟，主要是基于路径的方法。本文涉及的算法就是二者的结合，既考虑到了结构信息，也带有语义信息。

1.3　数据挖掘技术概述

1.3.1　数据挖掘技术产生的背景

数据挖掘[1]是一个多学科交叉研究领域，它融合了数据库技术、人工智能、

机器学习、统计学、知识工程、面向对象方法、信息检索、高性能计算以及数据可视化等最新技术的研究成果。经过十几年的研究，产生了许多新概念和新方法。

数据挖掘之所以被成为未来信息处理的骨干技术之一，主要在于它以一种全新的概念改变着人类利用数据的方式。特别需要指出的是，数据挖掘技术从一开始就是面向应用的。它不仅是面向特定数据库的简单搜索查询调用，而且要对这些数据进行微观、中观乃至宏观的统计、分析、综合和推理。这里所说的知识发现，不是要求发现放之四海而皆准的真理，也不是要去发现崭新的自然科学定理和纯数学公式。所有发现的知识都是相对的，是面向特定领域的，同时还要能够易于被用户理解。

数据挖掘之所以吸引专家学者的研究兴趣和引起商家厂商的广泛关注，主要在于大型数据系统的广泛使用和把数据转换成有用知识的迫切需要。新的需求推动新的技术的诞生。随着信息技术的高速发展，数据库应用的规模、范围和深度不断扩大，已经从单台机器发展到网络环境。今年来由于数据采集技术的更新，如商业代码的推广、企业和政府利用计算机管理事务的能力增强，产生了大规模的数据。数以百计的数据库系统在运行，而且每天都在增加。决策所面临的数据量在不断增长，即使像使用IC卡和打电话这样简单的事务也能产生大量的数据。随着数据的急剧增长，现有信息管理系统中的数据分析工具已经无法适应新的需求。因为无论是查询、统计还是报表，其处理方式都是对指定的数据进行简单的数字处理，而不能对这些数据所包含的内在信息进行提取。人们希望能够提供更高层次的数据分析功能，自动和智能地将待处理的数据转化为有用的信息和知识。

数据挖掘的基础是数据分析算法[2]。数据分析是数据研究的基础，许多科学研究都是建立在数据收集和分析的基础上的。同时在目前的商业活动中，数据分析总是和一些特殊的人群的高智商行为联系在一起，因为并不是每一个人都能从过去的销售情况预测将来发展趋势或作出正确决策的。但是，随着一个企业或者行业业务数据的不断积累，特别是由于数据库的普及，人工去整理和理解如此大的数据源已经存在效率和准确性的问题。因此，探讨自动化的数据分析技术，为企业提供能带来商业利润的决策信息就成为了必然。

任何技术的产生总是有它的技术背景的。数据挖掘技术的提出和普遍接受是由于计算机及其相关技术的发展为其提供了研究和应用的技术基础。

归纳数据挖掘产生的技术背景，下面一下相关技术的发展起到了决定性的作用：

- 数据库、数据仓库和Internet等信息技术的发展；
- 计算机性能的提高和先进的体系结构的发展；
- 统计学和人工智能等方法在数据分析中的研究和应用。

数据库技术从 20 世纪 80 年代开始，已经得到广泛的普及和应用。在关系型数据库的研究和产品提升过程中，人们一直在探索组织大型数据和快速访问的相关技术。高性能关系数据库引擎以及相关的分布式查询、并发控制等技术的使用，已经提升了数据库的应用能力。在数据的快速访问、集成与抽取等问题的解决上积累了经验。现今，人们已经具备利用多种方式存储海量数据的能力。只有这样，数据挖掘技术才能有它的用武之地。这些丰富多彩的数据存储、管理以及访问技术的发展，为数据挖掘技术的研究和应用提供了丰富的土壤。

经历了几十年的发展，包括基于统计学、人工智能等在内的理论与技术性成果已经被成功地应用到商业处理和分析中。这些应用从某种程度上为数据挖掘技术的提出和反战起到了极大地推动作用。数据挖掘系统的核心模块技术和算法都离不开这些理论和技术的支持。因此，可以说，数据挖掘技术在继承已有的研究成果的基础上，摆脱了以前象牙塔式研究模式，真正开始客观地从数据集中发现蕴藏的知识。

1.3.2 数据挖掘技术的概念

数据挖掘的概念包含丰富的内涵，是一个多学科交叉研究领域。数据挖掘从本质上说是一种新的商业信息处理技术。数据挖掘技术把人们对数据的应用，从低层次的联机查询操作，提高到决策支持、分析预测等更高级应用上。它通过对这些数据进行微观、中观乃至宏观的统计、分析、综合和推理，发现数据间的关联性、未来趋势以及一般性的概括知识等，这些知识性的信息可以用来指导高级商务活动。

1.3.3 数据挖掘的应用

利用数据挖掘技术可以设计体育类软件，使得更好的分析每一位运动员及其对手的排兵布阵的相对优势，就像 IBM 公司开发的 Advanced Scout 一样。数据挖掘技术在美国银行和金融领域也是应用广泛。金融事务需要搜集和处理大量数据，对这些数据进行分析，可以发现潜在的客户群、评估客户的信用等。例如，美国银行大都使用数据挖掘工具，可以根据消费者的家庭贷款、赊账卡、储蓄、投资产品等，将客户分类，进而预测何时向哪类客户提供哪种服务。此外，数据挖掘技术还在电信、科学探索和信息安全中得到广泛应用。可见，数据挖掘正在深入我们的生活，充分说明了它的可利用性和高挑战性。

1.4 XML 技术概述

1.4.1 XML 技术的产生

XML[3]的全称是 Extensible Markup Language，意思是可扩展的标记语言，它是标准通用标记语言(Standard Generalized Markup Language,SGML)的一个子集。

在 80 年代早期，IBM 提出在各文档之间共享一些相似的属性，例如字体大小和版面。IBM 设计了一种文档系统，通过在文档中添加标记，来标识文档中的各种元素，IBM 把这种标识语言称作通用标记语言(Generalized Markup Language)，即 GML。经过若干年的发展，1984 年国际标准化组织(ISO)开始对此提案进行讨论，并于 1986 年正式发布了为生成标准化文档定义的标记语言标准(ISO8879)，称为新的语言 SGML，即标准通用标记语言。SGML 功能非常强大，是可以定义标记语言的元语言。

1998 年 2 月，W3C 发布了 XML1.0 标准，其目的是为了在 Web 上能以现有的超文本标记语言(HTML)的使用方式提供、接收和处理通用的 SGML。XML 是 SGML 的一个简化子集，它以一种开放的、自我描述的方式定义了数据结构。在描述数据内容的同时能突出对结构的描述，从而体现出数据与数据之间的关系。

W3C 组织于 2004 年 2 月 4 日，发布了 XML1.1 的推荐标准，这是最新的 XML 版本，不过目前大多数的应用还是基于 XML1.0 的推荐标准。

1.4.2 XML 技术的概念及用途

XML 技术[4]的概念包括很多，在这里简要介绍一下几个重要的定义。

XML 树结构，每一个 XML 文档都可以形成一种树结构，它从根部开始，然后扩展到最底端。本篇文章所利用的就是 XML 文档的树结构，根据其结构的相似性对 XML 文档进行聚类。

XML 元素，指的是从开始标签直到结束标签的部分。XML 元素是可扩展的，以携带更多的信息。

XML 应用于 Web 开发的很多方面，常用于简化数据的存储和共享。XML 可以将数据从 HTML 分离。通过 XML，数据能够存储在独立的 XML 文件中，这样就可以专注于使用 HTML 进行布局和显示，并确保修改底层数据不再需要对 HTML 惊醒任何改变。此外，XML 还可以简化数据共享和数据传输，从而使数据更加有用。

1.5 论文的研究内容和组织结构

本文的研究内容是设计并实现一个基于 XML 文档的聚类算法，该算法能将相似的 XML 文档聚集到一起，然后根据操作者的意图一起呈现出来，为在数据库系统上查找提供方便，自动显示查找的相关内容。

最终实现的算法应该有以下优点：

(1)实现了文档的聚类

(2)提高了搜索的准确率和查全率

本文先阐述当代现有的聚类算法，将各个算法一一详述，说明它们的特点和适用范围。分析本课题处理 XML 文档需要的条件，并且分析 XML 结构特点。

然后，在第三章会说明 k-means 算法的基本思想，根据 XML 文档结构的特点，

对原有的算法进行改善，使得可以直接对 XML 文档进行处理。由于 k-means 算法对初始中心点的选取具有很强的敏感性，对实验结果有直接并且重大的影响，所以本文采用快速选取初始中心点的思想，利用待处理的数据集的密度和位置分配，来确定初始中心点，这样大大避免了出现局部最优解的现象。

在第四章会看到本文新提出的一种计算 XML 文档相似度的算法，基于扩展邻接矩阵的 XML 文档结构及语义相似性的计算方法，并详细阐述其算法思想。并且在相似度计算方法方面，和边集比较法进行理论和实验上的对比，通过大量实验数据绘制出表格，得出结论。

最后提出结论和展望。

第二章　　聚类分析

2.1　聚类分析概述

2.1.1　聚类分析在数据挖掘中的应用

聚类是把一组个体按照相似性归成若干类别，它的目的是使得属于同一类别的个体之间的差别尽可能的小，而不同类别上的个体间的差别尽可能的大。数据挖掘的目标之一是进行聚类分析。通过聚类技术可以对源数据库中的记录划分为一系列有意义的子集，进而实现对数据的分析。

聚类分析可以作为其他算法的预处理步骤。利用聚类进行数据预处理，可以获得数据的基本概况，在此基础上进行特征抽取或分类就可以提高精确度和挖掘效率。也可将聚类结果用于进一步关联分析，以进一步获得有用的信息。

它还可以作为一个独立的工具来获得数据的分布情况。聚类分析是获得数据分布情况的有效方法。通过观察聚类得到的每个簇的特点，可以集中对特定的某些簇进一步分析，这在诸如市场细分、目标顾客定位、业绩评估和生物种群划分等方面具有广阔的应用前景。

此外，还可以完成孤立点挖掘。许多数据挖掘算法试图使孤立点影响最小化，或者排除它们。然而孤立点本身可能是非常有用的。如在欺诈探测中，孤立点可能预示着欺诈行为的存在。

2.1.2　聚类分析算法的概念与基本分类

定义 聚类分析的输入可以用一组有序对(X,s)或(X,d)表示，这里 X 表示一组样本，s 和 d 分别是度量样本间相似度或相异度(距离)的标准。聚类系统的输出是一个分区，若 $C=\{C_1, C_2, \ldots, C_k\}$，其中 $C_i(i=1,2,\ldots,k)$ 是 X 的子集，如下所示：

$C_1 \cup C_2 \cup \ldots \cup C_k = X$

$C_1 \cap C_2 = \Phi, i \neq j$

C 中的成员 C_1，C_2，...，C_k 叫做类，每一个类都是通过一些特征描述的，通常有如下集中表示方式：

- 通过类的中心或者类的中心点表示一个类
- 使用聚类树中的结点图形化地表示一个类
- 使用样本属性的逻辑表达式表示类

用中心表示一个类是最常见的方式，当类是紧密的或各向同性时用这种方法非常好，然而，当类是伸长的或各向分布异性时，这种方式就不能正确地表示它们了。

2.1.3　相似度计算

一个聚类分析过程的质量取决于对度量标准的选择，为了度量对象间的接近或相似程度，就需要一些相似性度量标准。用 s(x,y)表示样本 x 与样本 y 的相似度。当 x 和 y 相似时，s(x,y)的取值是很大的，当 x 和 y 不相似时，s(x,y)的取值是很小的。相似性的度量具有自反性 s(x,y)=s(y,x)。对于大多数聚类方法来说，相似性度量标准被标准化为 0<=s(x,y)<=1。

但是在通常情况下，聚类算法不是计算两个样本间的相似度，而是用特定空间中的距离作为度量标准来计算两个样本间的相异度。当度量两个样本间的距离时，可以用明科夫斯基距离函数、二次型距离函数、余弦距离函数和二元特征样本的距离度量，其中本文所利用的就是余弦距离函数。当计算类间的距离时，我们可以利用最短距离法、最长距离法、中心法、类平均法和离差平方和，基于 XML 文档的结构特性，本文采用类平均法[4]。

2.2 常用的聚类算法

2.2.1 基于划分的聚类

划分聚类方法[5]是最基本的聚类方法。像 k-平均、k-模、k-原型、k-中心点、PAM、CLARA 以及 CLARANS 等都属于划分聚类方法。

定义 给定一个有 n 个对象的数据集，划分聚类技术将构造数据 k 个划分，每一个划分就代表一个簇，k<=n。也就是说，它将数据划分为 k 个簇，而且这 k 个划分满足下列条件：每个簇至少包含一个对象，每一个对象属于且仅属于一个簇。

对于给定的 k，算法首先给出一个初始的划分方法，以后通过反复迭代的方法改变划分，使得每一次改进之后的划分方案都较前一次更好。所谓好的标准就是：同一簇中的对象越近越好，而不同簇中的对象越远越好。目标是最小化所有对象与其参照点之间的相异度之和。这里的远近或者相似度实际上是聚类的评价函数。

大多数聚类设计的评价函数多着重两个方面：每个簇应该是紧凑的，每个簇间的距离应该尽可能地远。

2.2.2 基于层次的聚类

层次聚类方法[6]对给定的数据集进行层次的分解，直到某种条件满足为止。具体又可分为凝聚的、分裂的两种方案。

凝聚的层次聚类是一种自底向上的策略，首先将每个对象作为一个簇，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，或者某个终结条件被满足，绝大多数层次聚类算法属于这一类。

分裂的层次聚类与凝聚的层次聚类相反，采用自顶向下的策略，它首先将所有对象置于一个簇中，然后逐渐细分为越来越小的簇，直到每个对象自成一簇。

层次凝聚的代表是 AGNES 算法。层次分裂的代表是 DIANA 算法。

2.2.3 基于密度的聚类

密度聚类方法[7]的指导思想是，只要一个区域中的点的密度大于某个域值，就把它加到与之相近的聚类中去。这类算法能克服基于距离的算法只能发现″类圆形″聚类的缺点，可发现任意形状的聚类，且对噪声数据部敏感。但计算密度单元的计算复杂度打，需要建立空间索引来降低计算量，且对数据维数的伸缩性较差。这类方法需要扫描整个数据库，每个数据对象都可能引起一次查询，因此当数据量大时会造成频繁的 I/O 操作。其代表算法有：DBSCAN、OPTICS、DENCLUE 算法等。

2.2.4 基于网格的聚类

STING 算法是一种基于网格的多分辨率聚类技术，它将空间区域划分为矩形单元。针对不同级别的分辨率，通常存在多个级别的巨型单元，这些单元形成了一个层次结构：高层的每一个单元被划分为多个第一层的单元。该聚类算法的质量取决于网格结构最底层的粒度。

STING 算法的主要优点是效率高，通过对数据集的一次扫描来计算单元的统计信息，因此产生聚类的时间复杂度为 O(n)。

第三章　　k-means 聚类算法分析

3.1　k-means 聚类算法概述

3.1.1　k-means 聚类算法定义

K-means 算法[8]以 k 为参数，把 n 个对象分成 k 个簇，以使簇内具有较高相似度，而簇间的相似度较低，相似度的计算根据一个簇中对象的平均值来进行。相似度的定义是划分的关键。

K-means 算法的基本思想是：随机的选取 k 个对象，每个对象初始地代表了一个簇的平均值或中心。对剩余的每个对象，根据其与各个簇中心的距离，将它赋给最近的簇。然后重新计算每个簇的平均值。这个过程不断重复，直到目标函数收敛。通常定义为公式(3-1)的目标函数，采用启发式方法使得目标函数最小。

$$E = \sum_{i=1}^{k} \sum_{p \in Ci} | p - m_i | \tag{3-1}$$

其中：E 为集合 U 中所有对象与相应聚类中心的均方差之和；p 为对象空间中的一个数据对象；m_i 为类 C_i 的均值。该函数旨在使生成的聚类结果簇尽可能的紧凑和独立。

对于已知数据集合分为几类的问题，k-means 划分方法是一个比较好的方法。K-means 算法属于划分方法，它需要有先验知识 k，即知道数据对象集合要聚为几类。在此，首先给出 k-means 的一般过程如图 3-1。

算法输入：聚类个数是k，以及包含n个数据对象的数据样本集U；

算法输出：满足方差最小标准的k个聚类；

算法步骤：

Step1 从 个数据对象中任意选择k个对象作为初始聚类中心；

Step2 根据每个聚类中所有对象的均值(中心对象)，计算样本集中每个对象与这些中心对象的距离，并根据最小距离重新对相应对象进行划分；

Step3 重新计算每个(有变化)聚类的均值(中心对象)；

Step4 循环执行Step 2到Step 3，直到每个聚类不再发生变化为止.

图 3-1　k-means 算法一般过程

3.1.2　k-means 聚类算法特点

k-means 算法是解决聚类问题的一种经典算法，这种算法简单、快速。对处理大数据集，该算法是相对可伸缩的和高效率的，因为它的复杂度是 O(n •k •t)，

其中，n 是所有对象的数目，k 是簇的数目，t 是迭代的次数。通常地，k<<n, 且 t<<n。这个算法经常以局部最优结束。

k-means 方法只有在簇的平均值呗定义的情况下才能使用。这可能不适用于某些应用，例如涉及有分类属性的数据。要求用户必须事先给出 k(要生成的簇的数目)可以算是该方法的一个缺点，而且对初始值敏感，遂于不同的初始值，可能会导致不同的聚类结果。K-平均方法不适合于发现非凸面形的簇，或者大小差别很大的簇。

3.2 快速选取初始点的 k-means 算法

3.2.1 快速选取初始点的提出

在k-means算法中，选择不同的初始聚类中心[9]会产生不同的聚类结果且有不同的准确率。因此，采用一开始就可以快速，尽可能准确地找到数据在空间分布上的初始聚类中心的方法，从而对数据进行划分。在用欧氏距离作为相似性度量的k-means算法中，相互距离最远的数据对象比随机取的数据对象更具有代表性，它们属于不同聚类的可能性比较大。如果能够寻找到是个相对距离最远的数据对象，它们分别代表了走个不同的数据集合，那么就可以找到与数据在空间分布上相一致的初始聚类中心．为了找到与数据在空间分布上相一致的、相似程度较大的数据集合，采取下列步骤：确定需要划分簇的个数是，计算数据对象两两之间的距离；找出距离最远的两个数据对象，形成两个独立的数据对象A_1和A_2，并将它们从总的数据集合U中删除；计算数据对象A_1，A_2与数据对象集合U中每一个样本的距离，找出在U中与A_1，A_2同时最远的数据对象，将它作为数据对象A_3，并从U中删除，重复上面的过程，直到确定了A_k个数据对象为止；再计算U中的各个样本数据到数据对象A_1，A_2，...，A_k的距离，将样本数据与距离最近的数据对象A_1，A_2，...，A_k进行合并，对应形成C_1，C_2，...，C_k对象集合；最后对C_1，C_2，...，C_k对象集合中所包含的数据对象分别进行算术平均，形成k个初始聚类中心。

假设在一个二维数据集U中，包含有10个数据样本，它们的分布如图3-2所示。假设要把它们划分为3类，按照上面的思想寻找初始聚类中心。首先计算数据集合U中数据样本两两之间的距离，可以得出A、H 之间的距离最远，那么将A、H作为数据对象A_1，A_2，并将它们从总的数据集合U中删除．再计算U中所有样本到A、H的相对距离，得出J与A、H的相对距离最远，将J作为数据对象A_3,并将它从U中删除。集合U中与A相邻近的对象是CD，这样便将CD与A_1合并形成集合C_1．集合U中与A_2相邻近的对象是EFIG，这样便将EFIG与A_2合并形成集合C_2。集合U中与A_3相邻近的对象是B，将B与A_3合并形成集合C_3。最后，分别对集合C_1、C_2、C_3中的数据样本进行算术平均形成3个初始聚类中心，应用k_means聚类

算法形成最终聚类. 这样得到的初始聚类中心与实际样本的分布更加相符，从而可以得到更好的聚类效果。

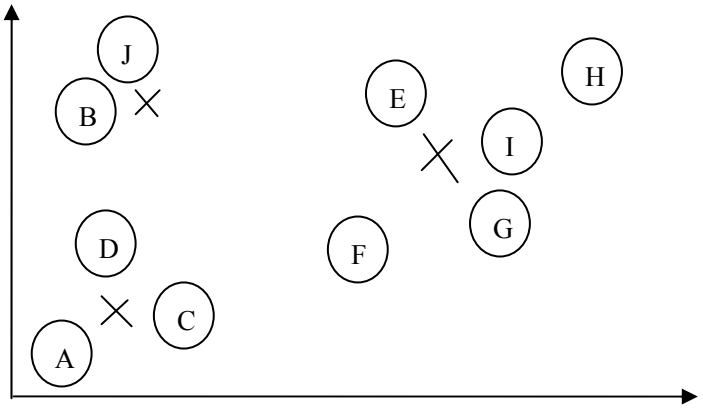

图3-2　二维数据样本分布

3.2.2　快速选取初始点的定义和相关过程

为了更好的说明改进的算法，给出以下几个定义：

定义1 数据对象i与集合U中样本点之间的距离为

$$d(i,U)=d(i,m),m\in U \tag{3-2}$$

定义2 数据对象i，j和集合U中样本点之间的距离同时最远定义为

$$d(U,i\mid j)=\max(\min(d(m,i),d(m,j)),m\in U) \tag{3-3}$$

因此，给出快速选取初始中心点的k-means算法过程如图3-3。

算法输入：聚类个数k，以及包含n个数据对象的数据样本集U；

算法输出：满足方差最小标准的k个聚类；

算法步骤：

Step1 计算数据样本集U中数据对象间的距离d(i，j)，找到集合U中距离最远的两个数据对象，将这两个数据对象作为A_1，A_2，并从集合U中删除这两个对象；

Step2 在U中找到与数据对象A_1，A_2的距离同时最远的数据样本，作为数据对象A_3，并从集合U 中删除该对象；

Step3 重复Step 2直到从U 中找到A_1，A_2，...，A_k，共k个数据对象为止；

Step4 从集合已，中找出与数据对象A_1，A_2，...，A_k距离最近的样本，合并生成所对应的集合C_1，C_2，...，C_K.

Step5 将最终形成的k个集合C_1，C_2，...，C_k中的数据对象分别进行算术平均，从而形成k个初始聚类中心；

图3-3　快速选取初始中心点的k-means算法过程

改进后的算法与传统的k-means算法相比，在计算初始聚类中心时会增加一些时间开销，但它的处理效率却大大提高了。该算法的复杂度和传统算法是一致的，是0(nkt)，其中，n是所有数据对象的数目；k是聚类的个数；t是迭代的次数．处理相同数据集时，改进后算法的迭代次数t与传统k_means算法迭代次数T的关系是t<=T。因为经过事先对数据对象的预处理，使得初始的聚类中心比盲目地指定初始聚类中心更接近最终的聚类结果，所以能够减少算法迭代的次数．当k<<n且t<<n时对处理大数据集是可伸缩的和高效率的．另外，该算法和传统算法一样都需要事先估计聚类的个数k，想得到最优解时必须试探不同的k值．所不同的是，传统算法得出的聚类结果会随不同的初始聚类中心产生波动，而改进后的算法会得到一个优化的、稳定的聚类结果。

3.3 计算 XML 文档结构相似性的常用方法

3.3.1 元素比较法

元素比较法[10]是指通过计算两个XML文档中共有元素的数量占这两个文档中所有元素数量的比例来反映文档之间的相似度。假设 $E(T)=\{e_1,...,e_i,...,e_n\}$表示文档树 T 中包含的所有元素，给定两颗树 T_1 和 T_2，$C(T_1,T_2)=\{c_1,...,c_i,...,c_m\}$表示 T_1 和 T_2 共有元素的集合，其中:

$c_i=e_{1j}=e_{2k}$,$LevelC(c_i)=Max(Level(e_{1j}),Level(e_{2k}))$,$C'(T)=\{c'_1,...,c'_i,...,c'_n\}$ 表示文档树 T_1,T_2 中非共有元素的集合。

则有计算文档树 T_1,T_2 之间相似度公式如公式(3-4)。

$$Sim(T_1,T_2)=\frac{\sum_{i=1}^{m}LevelC(c_i)}{\sum_{i=1}^{n}Level(c'_i)+\sum_{i=1}^{m}LevelC(c_i)} \tag{3-4}$$

3.3.2 边集比较法

这也是本文用作对比的方法。边集比较法是指通过计算两个 XML 文档树中相同的有向边的数量占这两个文档中所有有向边数量的比例来反映文档之间的相似度。

文档之间的相似度的计算公式如公式(3-5)。

$$Sim(T_1,T_2)=\frac{|R(T_1)\cap R(T_2)|}{|R(T_1)\cup R(T_2)|} \tag{3-5}$$

其中 T1 和 T2 表示两个文档树，R(T1)和 R(T2)是这两个文档树的边集，R(T1)∩R(T2)为两个边集中相同的有向边的集合，|R(T1)∩R(T2)|为两个边集中相同有向边的数目。R(T1)∪R(T2)为两个边集中所有的有向边的集合，|R(T1)∪

R(T2)|为两个边集中所有有向边的数目。

3.3.3 编辑距离法

编辑距离法是指利用编辑距离来衡量两棵树之间的相似度，其基本思想是将两棵树之间的距离定义为利用编辑操作实现一棵树到另一棵树转换所需的代价。树之间的编辑操作主要有三种：插入、删除和替代。

每一个编辑操作都有一个代价，令 λ 为一个代价函数，记为 λ(u)，其中 u 为一个编辑操作，T_1,T_2 为两颗 XML 文档树，S 为从 T_1 到 T_2 的编辑操作的序列，S={s_1,...,s_i,...,s_n},则 S 的代价是这些编辑操作的代价的总和，即：

$$\lambda(S)=\sum_{i=1}^{n}\lambda(s_i) \tag{3-6}$$

定义编辑距离为将 T1 转换到 T2 的最小的操作序列的代价，记为 ED(T_1,T_2)，即：

$$ED(T_1,T_2)=Min(\lambda(S))=MIN(\sum_{i=1}^{n}\lambda(s_i)) \tag{3-7}$$

为了对此类操作进行规范，插入和删除操作必须遵循两个原则：

(1)待插入的子树必须是源树的子树；待删除的子树必须是目标数的子树。

(2)待插入的子树及其后代树在此次插入操作前不能有其他编辑操作。

原则(1)保证了来源树倒目标数的正常转换；原则(2)防止了重复的编辑操作，保证了单个子树种编辑操作的代价最小。

图3-4中A,B,C,D,E,F分别代表XML文档的标签信息，以上三个文档结构，文档一到文档二需要将3、4节点对换，而文档三到文档二需要将1、3节点对换，但是如果用编辑距离算法来看，都是只通过两步的操作就可以完成。但是直观上看文档一和文档二的根节点相同，相似程度明显要高于文档二与文档三的相似程度，但是根据编辑距离算法两者的相似程度是一样的。所以编辑距离法对XML操作十分有效，但是在有的时候编辑距离法却不能很好的体现文档之间相似度的差异。

3.3.4 邻接矩阵

邻接矩阵用一个二维数组来表示图中顶点间的相邻关系的存储结构,无需列出顶点和弧,为图的描述提供了一种便利。

设G是一个图， V(G) 为 G的顶点集， E(G) 为 G的边集。 设G中有 n 个顶点 v_1，v_2，v_3…v_n；A=(a_{ij})n*n 为 G的邻接距阵，其中：

$$a_{ij}=\begin{cases}1 & v_iv_j\in E(G)\\ 0 & v_iv_j\notin E(G)\end{cases}\quad i,j=1,2,...,n \tag{3-8}$$

在这个式子中，只体现了连接与否的关系，比较简单，并没有很好的体现XML

文档的紧凑程度。也只能单单表示父亲与子女之间的关系，不能表示祖先与子孙之间的关系。

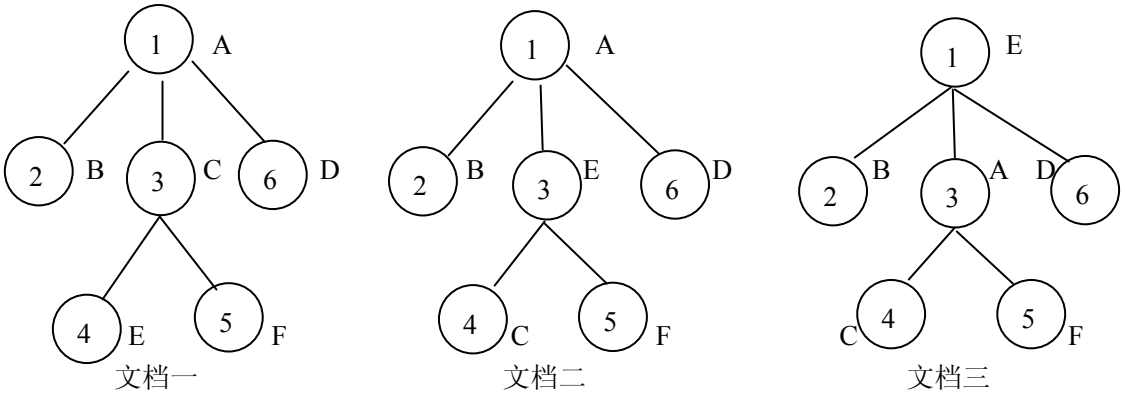

图 3-4　　三个 XML 文档

第四章　　XML文档结构及语义相似性

4.1　基于扩展邻接矩阵的相似性计算方法

4.1.1　XML 文档的预处理

XML 文档的 DOM 结构可以看作是该文档的树形结构[11]，其中节点属性看作是此节点的子节点，一个 XML 文档就可以看作是一个自上向下展开的树。如下图为一颗 XML 文档树，对此树进行编码，编码方式为深度搜索方式，即采用深度搜索方法遍历此树，然后为节点依次编码 1，2，3，4……，直到最后一个节点，记做节点编码。树中层的分配采用倒排方式，即树的叶节点所在层记做第一层，然后依次向上推第二层、第三层……，直到到达根节点，就像图 4-1 所示。

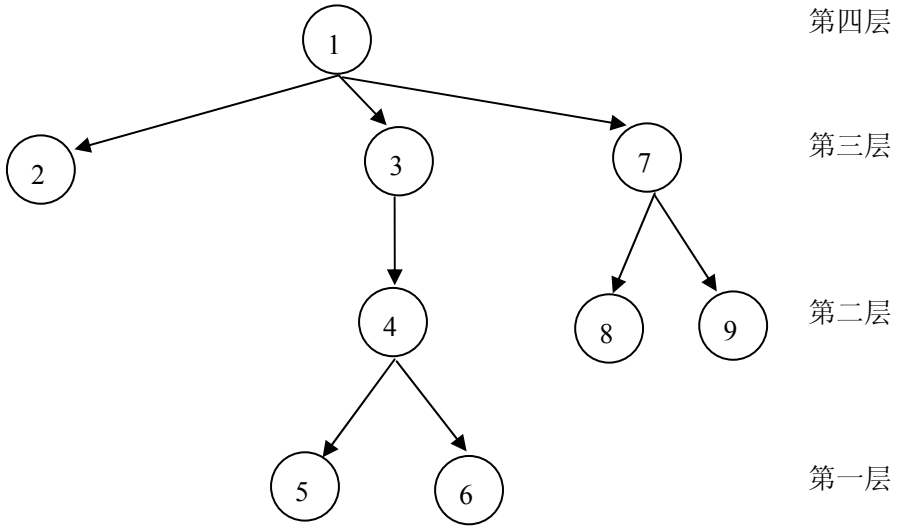

图 4-1　　XML 文档的树形结构

4.1.2　相关定义及概念

在介绍算法之前，我们先了解一下有关的定义。

模式文档：用户所提供的需求文档，在相似性比较中需要所有的其他文档与之相比较。

数据源文档：从数据源中提取的文档，在相似性比较中需要与模式文档相比较。

模式扩展邻接矩阵：用来表示模式文档结构及语义信息的邻接矩阵。

数据源扩展邻接矩阵：用来表示数据源文档结构及语义信息的邻接矩阵。

节点标签信息：即节点的语义，是节点的标志，亦是节点最重要的信息。

节点层信息：即节点在文档模型中处于哪一个层，是邻接矩阵中节点之间关系的度量标准。

节点编码信息：节点索引的唯一标识，在某个文档模型中不会有重复。

父节点信息：连接节点与节点间关系的信息，我们可以根据这个方便的找到每个节点的父节点及其祖先。

4.1.3 扩展邻接矩阵

首先介绍一下传统邻接矩阵的不足。传统意义上的邻接矩阵中只有 0 和 1 两个值，两个顶点之间有边相连则结构信息为 1，无边相连则结构信息为 0。这样的表示方法在描述树结构方面有以下几点不足：

(1)文档或图中的节点语义信息没有体现。

(2)对于树这种特殊的结构来说，只能体现树中父-子的关系，不能体现出祖先-子孙之间的关系。

(3)对于节点与节点之间连接的紧密程度的描述不够，仅仅是以 1 作为有连接的表示，而不能区别对待不同情况。

定义扩展邻接矩阵

G 是一个树， V (G) 为 G 的节点集合， E(G) 为 G 的祖先-后代关系。 设 G 中有 n 个节点 v_1，v_2，$v_3 \ldots v_n$； $A_{ij}=(a_{ij})n*n$ 为 G 扩展邻接距阵，其中

$$a_{ij}=\begin{cases} f_j \div f_i & v_iv_j \in E(G), i \neq j \\ 0 & v_iv_j \notin E(G), i \neq j \quad i,j=1,2,...,n \\ \phi & v_iv_j \notin E(G), i = j \end{cases} \tag{4-1}$$

f_i 代表 V_i 所在的层值，f_j 代表 V_j 所在的层值，ϕ 代表语义相似度。

此扩展的邻接矩阵有如下特点：

(1)模式文档矩阵大小为 n*n，其中 n 为模式文档中节点数量，数据源文档矩阵大小根据模式文档矩阵大小确定。

(2)用来表示结构信息有上（下）三角阵中剔除了左上右下对角线上的元素后剩余的元素，即 1+2+...+(n-1)=n(n-1)/2 个元素。

(3)用来表示语义信息的有上（下）三角阵中左上至右下对角线上的元素。

(4)用来表示结构信息的元素的取值遵循如下规则：

①如果两个节点具有父--子关系或祖先--子孙关系，则元素取值 Eij 为 Eij=子结点或子孙节点所在的层值÷父节点或祖先节点所在的层值。

②如果两个节点不具备以上的关系则元素值取为 0.

③如果两个节点中有任意一个节点为空节点，则元素取值为 0.

(5)模式邻接矩阵中的语义信息元素取值全为 1，数据源邻接矩阵中语义信息元素取值根据模式列表和比较列表中对应元素的相似程度取相应的值，取值范围在 0—1 之间。

假设有如图 4-2 两个 XML 文档。

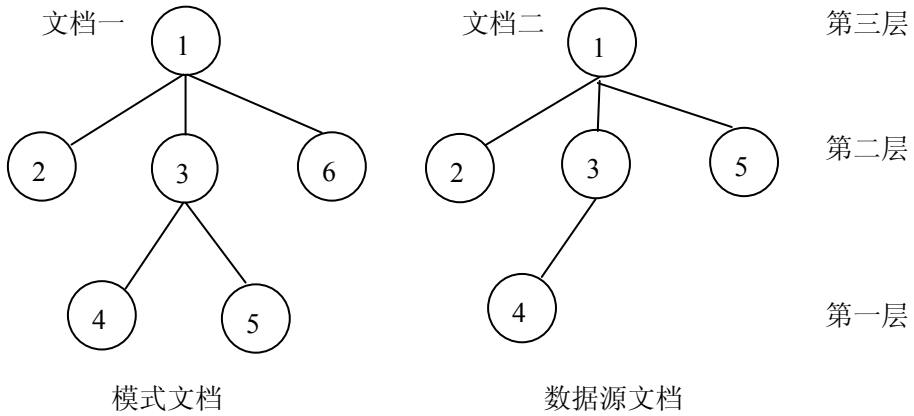

图 4-2　　模式文档与数据源文档

则模式文档扩展邻接矩阵：

$$\begin{pmatrix} 1 & 2/3 & 2/3 & 1/3 & 1/3 & 2/3 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/2 & 1/2 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

数据源文档扩展邻接矩阵：

$$\begin{pmatrix} 1 & 2/3 & 2/3 & 1/3 & 0 & 2/3 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

4.1.4　相似性计算

我们采用余弦距离公式来计算相似度[12]，下面给出两个扩展邻接矩阵 M1，M2，分别代表模式扩展邻接矩阵和数据源扩展邻接矩阵，然后我们讨论两个矩阵之间的相似性。

M1：$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix}$　　M2：$\begin{pmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & & \vdots \\ b_{n1} & \cdots & b_{nn} \end{pmatrix}$

在保证 M1，M2 中对应值位置不发生改变的前提下，我们可以把上面两个矩阵 M1，M2 看作是两个矩阵每行首尾相连的向量 d_1 和 d_2。

$$d_1 = (a_{11}, a_{12}, a_{13}, \ldots, a_{(n-1)n}, a_{nn})$$
$$d_2 = (b_{11}, b_{12}, b_{13}, \ldots, b_{(n-1)n}, b_{nn})$$

两个矩阵之间的相似性可以记做：

$$\cos(m1, m2) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}b_{ij}}{\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}^{2}}\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}^{2}}} = \frac{\sum_{k=1}^{n*n} m1_k m2_k}{\sqrt{\sum_{k=1}^{n*n} m1_k^{2}}\sqrt{\sum_{k=1}^{n*n} m2_k^{2}}} \quad (4\text{-}2)$$

4.2 实验结果

4.2.1 实验结果

基于 XML 文档结构的灵活性，为了更好地实现本课题的研究目的，本文做了三个实验加以说明。

实验一　两个 XML 文档之间的相似度

请看如图 4-3 所示的两个简单的 XML 文档。

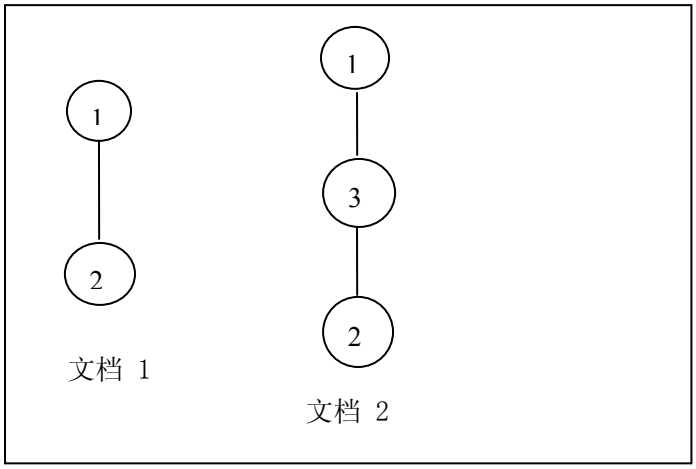

图 4-3　两个待比较的 XML 文档

可以看到文档 1 与文档 2 之间的差别就是文档 2 比文档 1 多了一个节点 3，并且在节点 1 与节点 2 之间。如果是利用边集比较法计算两个文档的相似度，那么得到的结果将是 0；然而，用本文提出的方法计算的相似度则不为 0。很显然，文档 1 与文档 2 之间并不是完全不相似。这两个文档是构造的文档，比较特殊。当一个 XML 文档的树形结构比较复杂的时候，向其中插入一个节点显然对其相似度的计算影响是不明显的。如果是单纯地考虑结构，此实验有一定的价值。实验的结果如图 4-4 和图 4-5。图 4-4 是用边集比较法进行的实验结果，图 4-5 是用基于扩展的邻接矩阵的计算方法得到的结果。

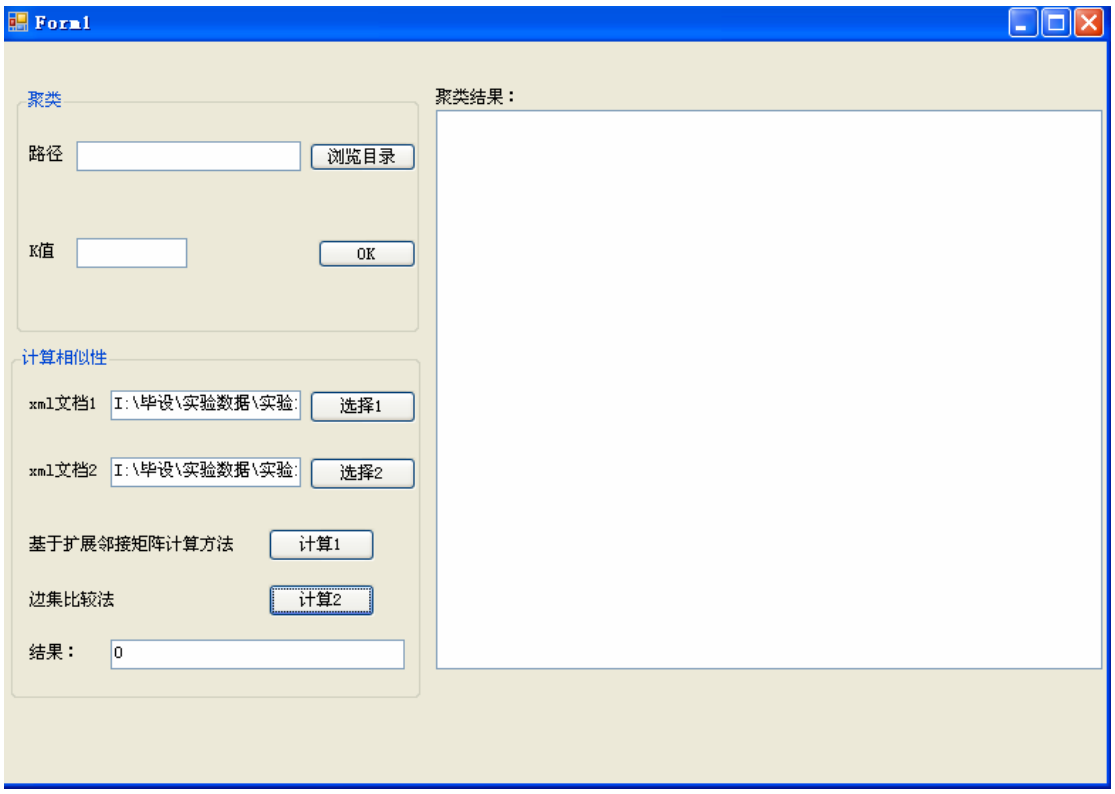

图 4-4　　边集比较法的实验结果

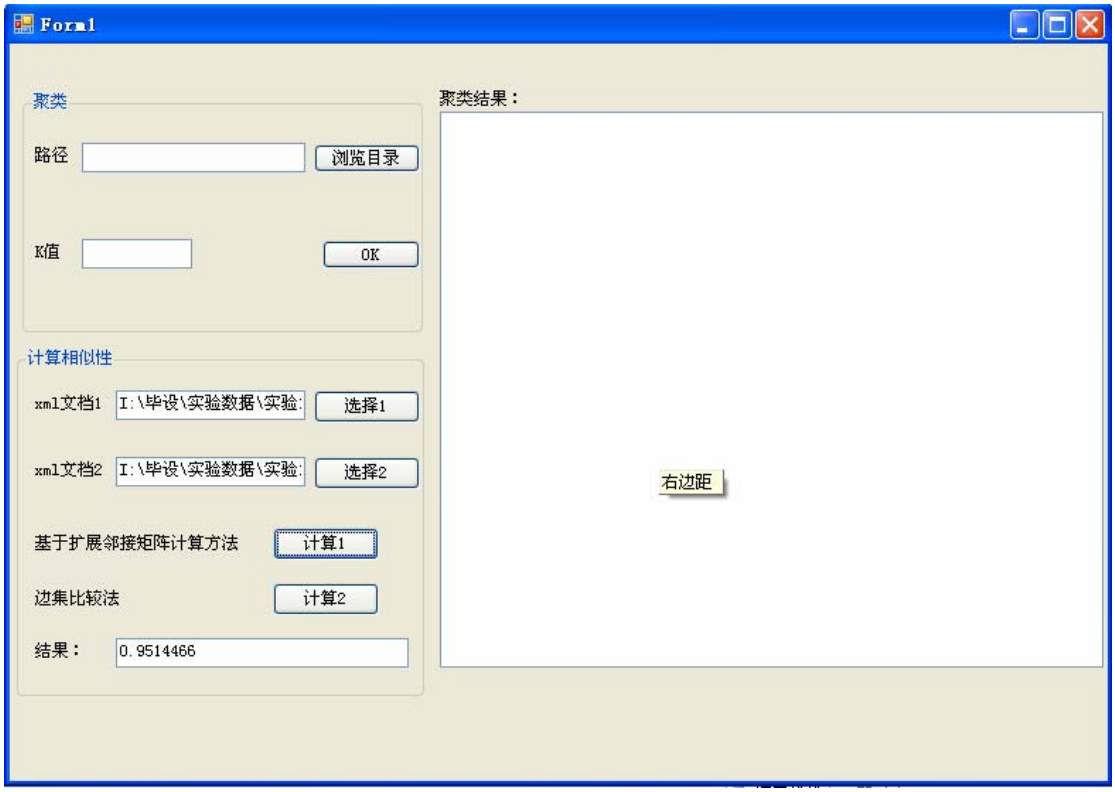

图 4-5　　基于扩展邻接矩阵的实验结果

由此实验，可以看出边集距离法只是针对 XML 的相连的结构提出来的，是基于相同的相连的边的比较，不考虑不相连的祖孙关系，只比较相连的父子关系是否相同；而基于扩展邻接矩阵的计算方法克服了边集比较法的局限性，不必局限在相连的父子关系上，是着眼于它们的子孙关系。而且它不仅考虑到 XML 结构的特点，而且还带有语义的考虑，有利于计算既含语义又考虑结构的分类。

实验二 相同 DTD 的 XML 文档聚类

DTD 是一套关于标记符的语法规则。它是 XML1.0 版规格的一部分,是 XML 文件的验证机制,属于 XML 文件组成的一部分。DTD 是一种保证 XML 文档格式正确的有效方法，可以通过比较 XML 文档和 DTD 文件来看文档是否符合规范，元素和标签使用是否正确。一个 DTD 文档包含：元素的定义规则，元素间关系的定义规则，元素可使用的属性，可使用的实体或符号规则。

XML 文件提供应用程序一个数据交换的格式,DTD 正是让 XML 文件能够成为数据交换的标准,因为不同的公司只需定义好标准的 DTD,各公司都能够依照 DTD 建立 XML 文件,并且进行验证,如此就可以轻易的建立标准和交换数据,这样满足了网络共享和数据交互。

为了进行上述的实验，本文构造了一个数据集，里面的 XML 文档都是基于一个 DTD。此数据集有 70 个 XML 文档，共分为 8 类，在每一个 XML 文档的名称上都有标记，其后缀就是它所属的类别。采用本文的方法，程序运行结果如图 4-6 所示，在这里我们可以看到每一个簇都分的比较正确。

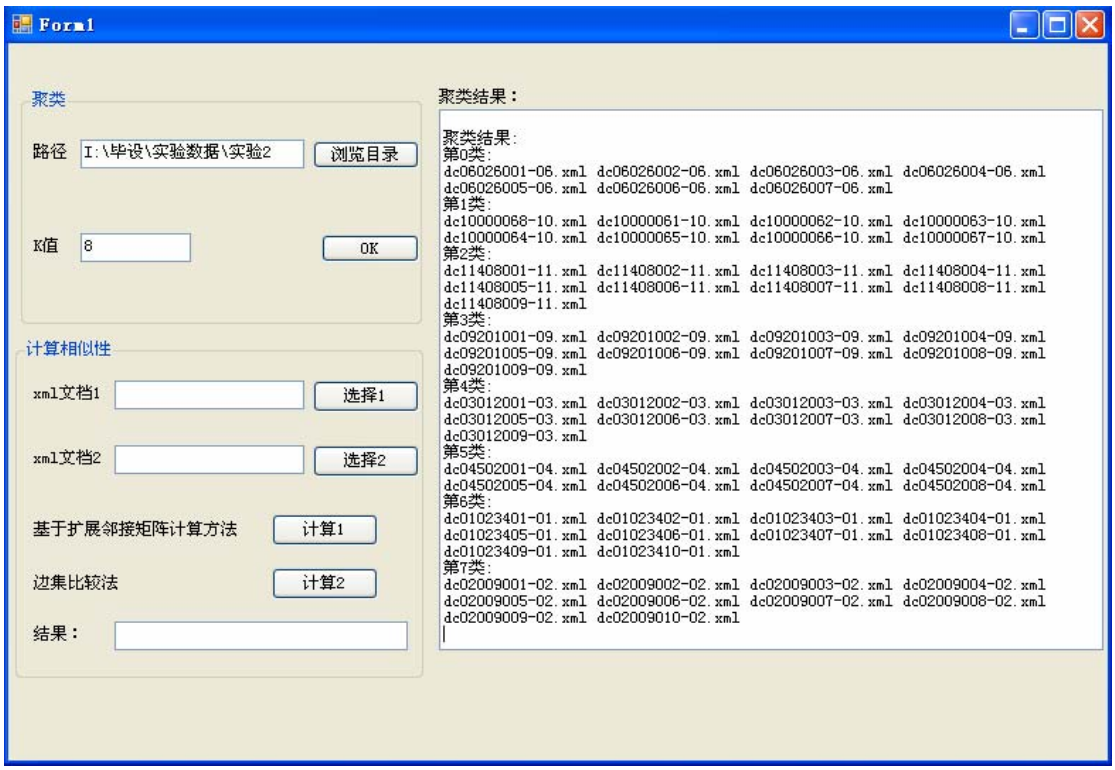

图 4-6　　聚类结果

采用边集比较法对数据集进行聚类，程序运行结果如图 4-7 所示。

图 4-7　边集比较法的聚类结果

上述 70 个文档，分为 8 个簇。簇之间的相似度相差并不大，说明聚类还是比较准确的。表 4-1 表明了文档集聚类之后的簇之间的均值。

表 4-1　各个簇之间均值比较结果

均值	01	02	03	04	06	09	10	11
01	1	0.8378	0.8167	0.8534	0.7252	0.7598	0.2257	0.8158
02		0.9998	0.9358	0.8893	0.7676	0.8306	0.2159	0.9349
03			0.9805	0.8727	0.7176	0.8228	0.2016	0.9569
04				0.9844	0.7254	0.8082	0.1964	0.8944
06					0.9896	0.7557	0.1844	0.7189
09						0.9599	0.1939	0.8034
10							0.9623	0.2014
11								0.9805

通过实验二，可以看到边集比较法和本文的方法在对相同 DTD 条件下的 XML 文档聚类结果相差不大，并且本文的方法和边集比较法的聚类效果都比较理想。

实验三 不同 DTD 的 XML 文档聚类

构造一个数据集，里面有 100 个 XML 文档，它们的类别在其名称中可以分辨出来，实验结果如图 4-8 所示。

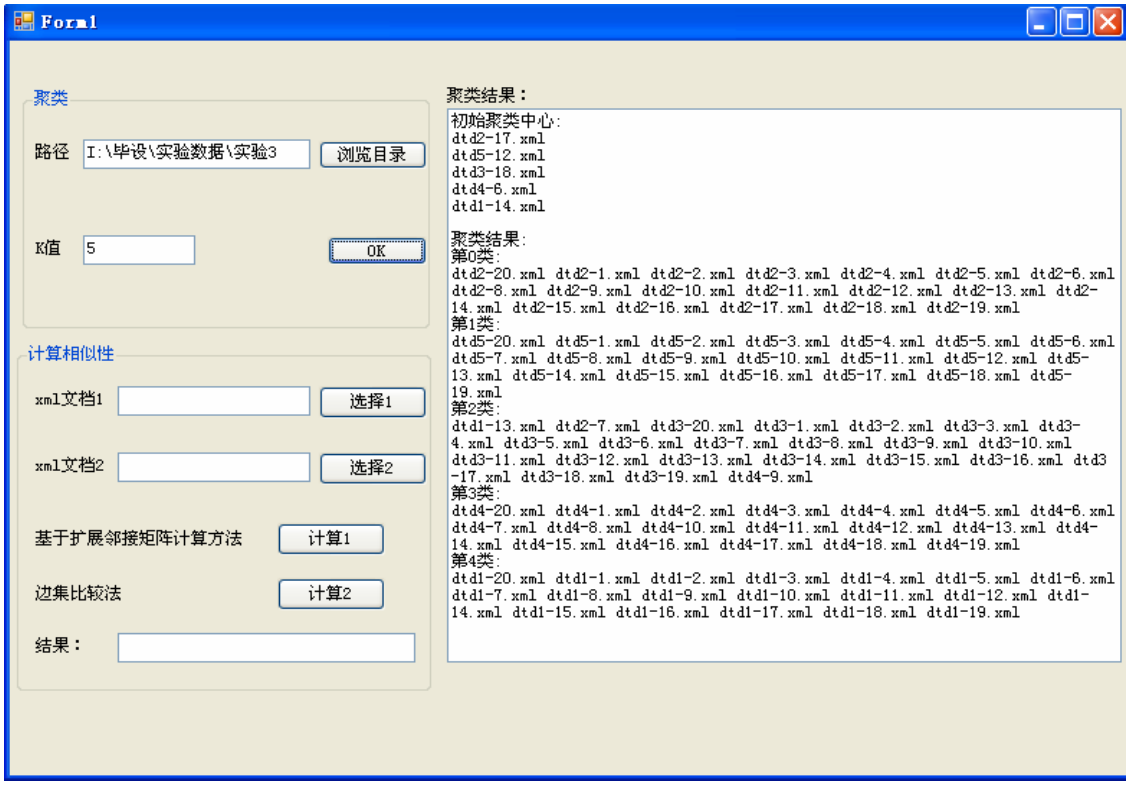

图 4-8　　聚类结果

我们可以看到，dtd4-9.xml dtd1-13.xml 和 dtd2-7.xml 文档分配错误，不过我们细看这几个文档可以看出它们都是一个空文档，所以它们可能被分配到所有簇当中，对结果的准确性其实不构成影响。

从实验二和实验三我们可以看出，本文提出的基于扩展邻接矩阵的计算相似度的聚类算法在准确性上，对于非大量数据是比较理想的。和现有的基于边集比较法的聚类方法相比，可以看出它还是有一定的优势的，除了不基于严格的结构信息以外，它还夹杂了一些语义信息的考虑，准确性也比较理想。

4.2.2　复杂度分析

空间复杂度　基于扩展的邻接矩阵相似性的聚类方法需要待处理文档的结构信息和语义信息，基于边集比较法的聚类方法仅仅需要文档的结构信息。文档的机构信息包括文档树的层数和每一层的元素的层数信息。因为一个文档的结构存储在RAM中，所以仅仅被消耗的记忆需要通过空间复杂度分析。

如果文档树的层数最大为N，每一层的平均元素数为M，在一个簇中需要的存储数量为M*N*8比特。所以，一个XML文档的最大层数为50，每一层中最多有元素50个，那么一个1k大小的簇需要消耗8M(50*20*8*1 *k*)的RAM。

时间复杂度　基于扩展的邻接矩阵相似性的聚类方法的时间复杂度至少为*O(m*m)*，*m*是文档元素的数目。对于大数量的数据这是不理想的。基于边集比较法的聚类算法避免了双方向比较的需要。它的时间复杂度是$O(m \times c \times p \times n)$：m

是文档的元素数目；c是簇的数目；p是迭代的数目；n是簇中不同元素的个数。

第五章　结论与展望

5.1　结论

本文总结并探讨了基于扩展邻接矩阵相似性计算的聚类算法所涉及到的一些关键技术和关键问题，并对国内外的研究现状也进行了详细介绍。针对现今聚类的诸多方法，在全面了解XML文档结构的特点和一些语义信息的基础上，本文提出了基于扩展邻接矩阵相似性计算的聚类算法，其目的是为了进一步提高XML文档聚类的准确度和效率。

本文开始重点介绍了k均值算法，并且在此基础上采用了快速选取初始中心点的方法，从而避免了出现局部最优解的情况。然后介绍了边集比较法，这是一种现今比较成熟的计算XML文档相似度的方法，在国内外使用比较广泛，主要是针对XML文档的结构信息和文档树的特点。最后引出本文提出的基于扩展邻接矩阵的相似性计算方法，虽然它是双向计算的方法，但是，并不像其他方法那样耗费更多的存储空间，并且这种方法克服了边集比较法单纯的依赖结构信息的特点，考虑到了少许的予以信息，这样就大大增加了使用的宽度，有利于更好地发展下去，充分利用有效的信息。

针对建立的XML文档数据集，利用本文的方法和边集比较法进行比较，实验证明本文的方法比边集比较法在某些方面更合理,比如实验一。它不仅仅依靠XML文档提供的结构信息，而且在树形结构方面区分不出来的情况下，也能利用语义信息来进行计算；此外，在准确度方面，本文的方法也不逊于边集比较法，证明了本文的方法的有效性和实用性。从而基本完成了本次的实验目的，达到了预期的效果。

同时也存在一些不足和问题。首先，本文提出的方法只是在少量数据集上进行了实验，并没有时间概念，不知道效率如何。还需要查找相当多的XML文档数据来进行实验，验证其效率。其次，本文提出的方法虽然与边集比较法进行了对比并实验，但结果显示对于边集比较法，本文的方法并没有什么特殊的优势，只是在某一些小的方面有所收获。还不能完全说明本文的方法比任何一种方法好，缺少长远的发展空间和突破点，这也是未来的研究方向之一。

5.2　展望

虽然本文的工作已经完成，但是还有很多问题需要我们去进一步探讨和研究。聚类方法现今已经被应用到很多领域，服务于各种行业，这方面的研究已经已经取得了很多成果，并且推出了一些原型系统。本文的方法是只针对XML文档来进行的聚类，考虑到了文档树的结构和语义两个方面，虽然该方法在本文中已经提出来，但是还有很长的路需要走。

XML文档主要是网络的组成部分，当今世界万维网正在以难以想象的速度发

展和扩张。每天都有数以万计的数据被增加到Web中去。其中很大一部分是网页信息。为了能够更有效地利用Web中近乎无限的各种信息，满足使用者的需要，提高网络的使用效率，基于XML文档的聚类是十分必要的。目前Web中已经有了许多技术成熟的聚类方法，而对于本文提出的方法，还没有被尝试，为此会需要大量的时间和实验。

参考文献

[1]Jiawei Han,Micheline Kamber. 数据挖掘概念与技术[M]. 北京：机械工业出版社. 2006:269-280.

[2]马敏辉. 基于粗糙集的多维数据分析算法研究[J],长春工程学院学报：自然科学版，2008，9(4)：77-79.

[3]潘海兰，吴翠红，葛晓敏. XML 及其在 MVC 模式中的应用[J],计算机技术与发展，2010，2：202-205.

[4]刘丹，宁云隆，于聪梅. XML 文档相似性的比较[J]，科学论坛，2009，2：64.

[5]杨厚群，何中市，雷景生. 基于划分的 XML 文档聚类研究[J]. 计算机科学，2008，35(3)：183-185.

[6]谢坤武. 文本挖掘中的层次聚类算法[J]. 湖北民族学院学报：自然科学版，2009，27(4)：415-419.

[7]胡学钢，王东波，吴共庆. 一种基于层次树的高效密度聚类算法[J]. 合肥工业大学学报：自然科学版，2008，31(2)：187-190，195.

[8]刘强，吴京慧. 优化初始中心点的 k-means 算法[J]. 信息技术，2009，2：71-74.

[9]曹志宇，张忠林，李元韬. 快速查找初始聚类中心的 k-means 算法[J]. 兰州交通大学学报，2009,28(6):15-18.

[10]杜新林，刘丹，董妍. XML 文档相似性的常用方法比较[J]. 长春大学学报，2009，19(6)：30-31.

[11]蒋悦，吴壮志，赵旭林等. 基于文档树的 XML 文件转换[J]. 计算机工程，2003，29(21)：97-99.

[12]张杰，卫金茂，刘丹. 基于 BFS 树的 XML 文档图结构相似性计算[J]. 计算机工程与设计，29(17):4603-4605.

[13]LlAN W，CHEUNG DW，MAMOULIS N，*et al*. An eficient and scalable algorithm for clustering xml documents by structure[J].IEEE Transactions on Knowled ge an d Data Engineering,2004，16(1)：82—96.

[14]ZHANG K，SHASHA D. Simple fast algorithms for the editing distance between trees and related problems[J]. SIAM J Comput，1989，18(6)：1245—1262.

[15]Ester M，Kriegel H P，Sander J. A density—based algorithm lot discoveri ng chlsters in large spatial databases with noise[C]//Proc 2nd lnt Couf on Kn1)wledge Diseovery and Data Mining Portland，1999，20：226—231.

[16]Pelleg D，Moore A. X—means：extending K—means with efficient estimation of the number of the clusters[C]//Proceedings of the 1 7th ICML，2000.

致　　谢

四年的大学生活即将结束，四年的时间想对于我的整个人生而言并不算很长，但是即将逝去的这个四年，对我的一生都将有着重要的影响。在这期间，我学到了很多，也交了很多朋友，得到了很多老师的帮助。

首先，我要感谢南开大学的卫金茂老师，在毕设的这段日子里，不断地给我帮助，将他的研究理念传授给我，使我受益匪浅。此外，卫老师他那严格谨慎、一丝不苟的治学作风和对科学事业的孜孜不倦、不懈追求的工作精神一直深深地感染着我，影响着我，更激励着我。他那种谦虚、勇于创新、严谨、踏实认真的作风，需要我在研究生阶段好好地学习，更值得我一生去学习。

我还要十分感谢张坤龙老师，是他不断督促我毕业论文的完成。他的幽默风趣和平易近人，让我在他面前更放松，无话不谈。他开阔的思路，使得我少走了很多弯路。除了学术外，张老师更注重对学生修养的培养，他看的书很多，不断向我们推荐好书，令我们受益匪浅。

感谢我的实验室成员张学良学长，他不仅在工作学习上给予了我很多启示和帮助，而且在生活上也很关心我。

感谢我的家人和众多关心我成长的亲友，他们在远方默默的支持着我，是他们无私的爱护、支持和鼓励，一直在伴随着我，使得我的学习与工作能够顺利进行。

在这里，我还要向所有帮助过我的人致以衷心的感谢!

李健

2010年6月