DOI:10.3969/j.issn.1003-0972.2010.01.040

一种基于关系代数的 Apriori优化方法

尤 磊*,兰 洋,熊 炎

(信阳师范学院 计算机与信息技术学院,河南 信阳 464000)

摘 要:提出了一种采用关系数据库管理系统的数据处理能力实现关联规则算法的方法.结合 Apriori算法的思想与关系代数的理论,分析了采用 SQL语句实现 Apriori算法的理论可行性,并描述了算法的实现过程.在 Mushroom数据集上的实验验证了本文方法的简单高效性.

关键词:关联规则;关系代数;SQL语言;Apriori算法;频繁项集

中图分类号:TP311 **文献标志码**:A **文章编号**:1003-0972(2010)01-0156-05

An Optimized Aprior Implementation Based on Relational Algebra

YOU Lei*, LAN Yang, XIONG Yan

(College of Computer and Information Technology, Xinyang Normal University, Xinyang 464000, China)

Abstract: A method of algorithm implementation for association rules is proposed. It uses the power of data processing in relational database management system to mine data. By Combining Apriori algorithm with the theory of relational algebra, the feasibility of implementing Apriori algorithm by use SQL statement is analyzed, and then, the realization process of method is described. An experimental result on mushroom dataset shows that the method given in this paper is simple and efficient.

Key words: association rules; relational algebra; SQL; apriori algorithm; frequent itemset

0 引言

数据挖掘是从存放在数据库、数据仓库或其他信息库中的大量数据中挖掘有趣知识的过程[1].挖掘的对象是数据库或数据仓库中的大量数据.在挖掘过程中,挖掘程序从数据库或数据仓库中提取数据进行分析.

关系数据库是数据挖掘最流行的、最丰富的数据源,因此它是我们数据挖掘研究的主要数据形式[1].关系数据库系统的数据库管理系统,是为数据库的建立、使用和维护而配置的系统软件,具有强大的数据处理功能[2].

Apriori算法是第一个关联规则挖掘算法[3],目前对 Apriori算法的研究主要集中在对算法改进上[4].在实现过程中,Apriori及改进算法从数据库管理系统中提取数据而忽略了数据库管理系统的数据处理功能.对于存储在关系数据库或关系数据仓库的数据而言,可使用关系数据库管理系的通用的数据管理功能来实现挖掘算法.基于此,本文在分析关联规则的理论与关系代数理论的基础上,采用关系数据库的 SQL语言实现 Apriori算法.

1 相关理论

关联规则(Association Rules)挖掘是数据挖掘研究领域的一个重要研究方向,用于发现隐藏在大型数据集中的令人感兴趣的联系,发现的联系可以用关联规则或频繁项集的形式表示.

设 $I=\{I_1, I_2, \cdots, I_m\}$是项的集合,$D=\{T_1, T_2, \cdots, T_n\}$是一个事务数据库,其中每个事务 T是项的结合,使得 $T\subseteq I$ 每个事务有一个标识符,称为 TID.如果 I的一个子集 X满足 $X\subseteq T$,则称事务 T包含项目集 X.一个关联规则就是形如 $X\Rightarrow Y$的蕴涵式,$X\subseteq I$, $Y\subseteq I$, $X\cap Y=\varnothing$.

规则 $X\Rightarrow Y$在交易数据库中的支持度(Support)就是交易集中包含 X和 Y的交易数与所有交

收稿日期:2009-07-17;**修订日期**:2009-11-04; *.**通讯联系人**,E-mail: yleiou@163.com

基金项目:河南省教育厅自然科学基础研究计划项目(2010A520033);河南省教育厅青年基金项目(2009-QN-078)

作者简介:尤 磊(1981-),男,河南罗山人,硕士,助教,主要研究领域为数据挖掘、数据库技术.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

易数之比,记为 $s(X\Rightarrow Y)$,即

$$s(X\Rightarrow Y) = |\{T: X\cup Y\subseteq T, T\in D\}| / |D|.$$

规则 $X\Rightarrow Y$在交易数据库中的置信度(Confidence)是指包含 X和 Y的交易数与包含 X的交易数之比,记为 $c(X\Rightarrow Y)$,即 $c(X\Rightarrow Y) = |\{T: X\cup Y\subseteq T, T\in D\}| / |\{T: X\subseteq T, T\in D\}|$.

关系数据库是以关系模型为基础的数据库.关系数据库的标准语言 SQL语言是一种介于关系代数和关系演算的语言,其功能包括查询、操纵、定义和控制 4个方面,是一个通用的功能极强的关系数据库标准语言,已经被确定为关系数据库系统的国际标准,被绝大多商品化的关系数据库系统所采用[2].

关系模型中的关系操纵能力早期通常是用代数方法或逻辑方法来表示,分别称为关系代数和关系演算.关系代数是用对关系的运算来表达查询要求的方式,关系演算是用谓词来表达查询要求的方式.SQL语言是具有关系代数和关系演算双重特点的语言,是一种被关系数据库产品广泛使用的标准结构化查询语言[2].

在关系代数中选择运算[2]是按给定的选择条件选出符合条件的元组,可以表示为:

$$\sigma_{<选择条件>}(<关系名>),$$

投影操作选取关系的某些属性,可表示为:

$$\Pi_{<属性表>}(<关系名>).$$

根据支持度与置信度的定义以及关系代数中选择运算以及投影运算的定义,可采用关系代数的运算对关联规则的支持度与置信度的描述如下:

$$s(X\Rightarrow Y) = |\sigma_{<同时包含\ X和\ Y的交易数>}(<交易表>)| / |\sigma_{<交易数>}(<交易表>)|,$$

$$c(X\Rightarrow Y) = |\sigma_{<同时包含\ X和\ Y的交易数>}(<交易表>)| / |\sigma_{<包含\ X的交易数>}(<交易表>)|.$$

根据以上理论分析,结合关系数据库管理系统强大的数据操纵管理能力,在进行关联分析时,可采用关系数据库语言 SQL实现关联挖掘算法.

2　Apriori算法

Apriori算法开创性地使用基于支持度的剪枝技术,系统地控制候选项集指数增长,它使用逐层搜索的迭代方法,k项集用于探索 $(k+1)$项集,其主要步骤[1]有:

(1)连接步:通过 L_{k-1}与自己连接产生候选 k-项集的集合,记做 C_k.为方便计,假定事务或项集中的项按字典次序排序,设 l_1和 l_2是 L_{k-1}中的项集.$l_i[j]$表示 l_i的第 j项,L_{k-1}与自己连接的条件是它们的前 $k-2$个项相同,即

$$(l_1[1]=l_2[1])\wedge(l_1[2]=l_2[2])\wedge\cdots\wedge(l_1[k-2]=l_2[k-2])\wedge(l_1[k-1]<l_2[k-1]).$$

连接 l_1和 l_2的结果项集是 $l_1[1]l_1[2]\cdots l_1[k-1]l_2[k-1]$.

(2)剪枝步:C_k是 L_k的超集,即 C_k的成员也可以不是频繁的,但所有的频繁 k项集都包含在 C_k中.因一个项集是频繁的,则它的所有子集一定也是频繁的[3],如果 C_k的一个候选 k项集的 $(k-1)$子集不在 L_{k-1}中,则该候选不可能是频繁的,可以从 C_k中删除,压缩了 C_k.然后扫描数据库确定 C_k中每个候选项的支持度计数确定频繁项.

Apriori算法的步骤(2)是对步骤(1)产生的候选项集 C_k中的每一项进行支持度阈值的判断.对压缩后的 C_k,需要扫描数据库得到支持度计数确定频繁项.若采用常规方法,需要从数据库中读取数据集到内存中,再通过应用程序遍历的方式进行计数.在关系数据库中,可使用关系数据库对数据的操纵管理能力来实现对候选频繁项集的支持度计数.下面结合关系代数的理论以及 Apriori算法的思想进行描述.

设事务数据库的关系模型为 $R(TID, I_1, I_2, \cdots, I_m)$,每一个元组的数据采用二元变量[1]表示.如果项在事务中出现,则它的值为 1,否则为 0,其中 $I_1, I_2, \cdots, I_m$是项,$I=\{I_1, I_2, \cdots, I_m\}$是项的集合.根据 Apriori算法步骤(1)中,l_1和 l_2是 L_{k-1}中的项集.$l_1[j]$表示 l_1的第 j项,则有 $l_1[j]\in I$根据关系代数中的选择和投影运算,

$$l_1[1]l_1[2]\cdots l_1[k-1]l_2[k-1]$$

的支持度计数等价于关系模型 R中在数据列

$$l_1[1], l_1[2], \cdots, l_1[k-1], l_2[k-1]$$

上数据值同时为 1的元组的个数.可设视图 S,且满足

$$S=\sigma_{<l_1[1]=1\ AND\ l_2[2]=1\ AND\ \cdots\ AND\ l_1[k-1]=1\ AND\ l_2[k-1]=1>(<R>)},$$

则视图 S中元组的个数即是 $l_1[1]l_2[2]\cdots l_1[k-1]l_2[k-1]$的支持度计数.根据关系代数中函数运算[2]的 COUNT函数,可描述为

```
SELECT COUNT(*) FROM S,
```


© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

返回值是 S中元组的个数,即是

$$l_1[1]l_1[2]\cdots l_1[k-1]l_2[k-1]$$

的支持度计数.

根据以上描述,可综合运用关系代数中的选择运算、投影运算以及函数运算来实现对候选项集的计数.

3 Apriori算法的 SQL实现

一个关系模型的逻辑结构就是一张二维表,根据关系的定义,把表 1中的事务数据库可以看成一个关系,其关系模型可描述为:R(TID,面包,牛奶,尿布,啤酒,鸡蛋,可乐).

在此关系模式中,R是关系模式的名称,其属性有 TID、面包、牛奶、尿布、啤酒、鸡蛋、可乐.每一个元组的数据采用表 1中的二元变量[1]表示.

表 1 事务数据集的二元表示

Tab. 1 Binary transaction data set

TID	面包	牛奶	尿布	啤酒	鸡蛋	可乐
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1

首先从数据集中挖掘频繁 1项集,根据关系运算中的函数查询的描述,使用 SUM函数来统计每项出现的个数,对于{面包}项,采用一条 SQL语句:

SELECT SUM(面包) FROM R

来实现,返回的结果是关系 R中"面包"属性在元组值的和,即是项"面包"的计数.

同理,可得出其他几个属性项的计数.根据 SQL的语法结构,可实现如下:

SELECT SUM(面包), SUM(牛奶), SUM(尿布), SUM(啤酒), SUM(鸡蛋), SUM(可乐) FROM R.

对表 1中的事务数据集,可采用如上一条 SQL语句即可得到每个项目对应的计数.其返回结果的值为 4,4,4,3,1,2.根据支持度计数值 3,得出频繁 1项集:{面包,牛奶,尿布,啤酒}.

下一次迭代是根据频繁 1项集获取频繁 2项集,根据 Apriori算法思想,候选频繁 2项集的个数为 6个,分别是{面包,牛奶}、{面包,尿布}、{面包,啤酒}、{牛奶,尿布}、{牛奶,啤酒}、{尿布,啤酒}.根据关系运算的投影、选择以及函数查询,使用 COUNT函数来统计同时包含多个项出现的元组个数:对于{面包,牛奶}项集,实现如下:

SELECT COUNT(*) FROM R WHERE (面包 = 1) AND (牛奶 = 1),

返回结果为

3,

即{面包,牛奶}项集的计数.同理有如下 SQL语句

SELECT COUNT(*) FROM R WHERE (面包 = 1) AND (尿布 = 1),
SELECT COUNT(*) FROM R WHERE (面包 = 1) AND (啤酒 = 1),
SELECT COUNT(*) FROM R WHERE (牛奶 = 1) AND (尿布 = 1),
SELECT COUNT(*) FROM R WHERE (牛奶 = 1) AND (啤酒 = 1),
SELECT COUNT(*) FROM R WHERE (尿布 = 1) AND (啤酒 = 1).

由此得出频繁 2项集:{面包,啤酒},{牛奶,啤酒}.

根据以上讨论,可得出频繁 k项集发现频繁 $(k+1)$项集的过程.本文给出 Apriori算法采用 SQL实现的算法描述.设 $R(I_1, I_2, \cdots, I_n)$为事务数据集的关系模式,其中 I_n是事务数据集中对应的项,R的元组数据采用二元变量的形式描述,L_k为频繁 k项集,C_k为候选频繁 k项集,根据 Apriori算法的核心思想有 $C_k = L_{k-1} \times L_{k-1}$. $L_k[i][j]$表示 L_k中第 i个频繁 k项中的第 j个项,$len(L_k)$表示 L_k中频繁项的个数.对于上列中的频繁 2项集有:$L_2[1][1]$表示项面包,$L_2[1][2]$表示项啤酒.

算法:采用 SQL实现 Apriori算法,使用根据候选生成的逐层迭代得出频繁项集.

输入:事务数据集的关系模式 R;最小支持度阈值 min_sup.

输出:R中的频繁项集 L.

方法:

1) for (j = 1; j < = n; j + +) //循环对关系 R中的项计数获取频繁 1项集
2) {
3) if (SELECT SUM(I_j) FROM R) > = min_sup then
4) $L_1 = L_1 \cup \{I_j\}$; //如果项 I_j的计数不小于最小支持度则并入频繁 1项集
5) }
6) for(k = 2; $L_{k-1} \neq \varnothing$; k + +) //根据频繁

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

k-项集发现频繁 $(k+1)$-项集

7) {

8) $C_k = L_{k-1} \times L_{k-1}$; //连接步并压缩 C_k

9) for($i = 1$; $i <= len(C_k)$; i++) //循环对候选 k项集中的每个项集计数

10) {

11) if (SELECT COUNT(*) FROM R WHERE (C_k[i][1] = 1) AND (C_k[i][2] = 1) AND …AND (C_k[i][$len(C_k[i])$ - 1] = 1) AND (C_k[i][$len(C_k[i])$] = 1)) >= min_sup then

12) $L_k = L_k \cup \{C_k[i]\}$; //判断 $C_k[i]$中的项集是否满足支持度计数

13) }

14) }

15) return $L = \cup_k L$.

由上面实现过程可知,在采用 SQL 语句实现 Apriori算法时,可根据候选 k项集的项组合形成 SQL查询语句,提交至数据库管理系统执行后返回计数结果,再由此判断是否属于频繁项.即将确定候选项集计数的任务交给数据库管理系统处理.

4　实验

本文在 Intel酷睿 2双核 T8100处理器,1 024 MB RAM,160 GB硬盘的联想笔记本上,编程工具为 Delphi7.0、SQL Server2005数据库、ADO数据库引擎、采用 http://archive.ics.uci.edu/ml/datasets/Mushroom页面的蘑菇数据库 (Mushroom Data Set)为测试数据集,展开实验,该数据库有 8 124条记录,记录了蘑菇的帽子形状、帽子颜色、颈的形状、颈的颜色、气味、生存环境、是否有毒等 23种属性,每种属性有 2～12个枚举值.

实验前需将蘑菇数据库预处理为表 1所示的二元结构表示方式,即采用二进制数据表示记录中某个项出现与否,因蘑菇数据库每种属性有 2～12个枚举值,需要按照每一个属性 (第一个属性是期望输出类别 (P, E)除外)枚举值的个数 m扩展为 m个属性,属性的取值代表了项是否出现.预处理后的数据集共有 117个属性,8 124条记录,分别采用 SQL实现的 Apriori算法与循环数据集对频繁项计数的 Apriori算法,以随机提取数据集中的 2 000、2 500、3 000条数据为 3组测试数据,支持度阈值分别取 40%、50%、60%、70%、80%进行实验,运行时间 (ms)比较如图 1所示.

从图 1可以看出,在相同记录条数、相同支持度阈值的条件下,采用 SQL实现的 Apriori算法执行时间要小于循环数据集计数的方法实现.并且支持度阈值越小,时间相差越大,且从试验数据可以看出,在支持度阈值相同的情况下,记录条数的增加使支持度计数增加,算法执行时间也将有所减少.

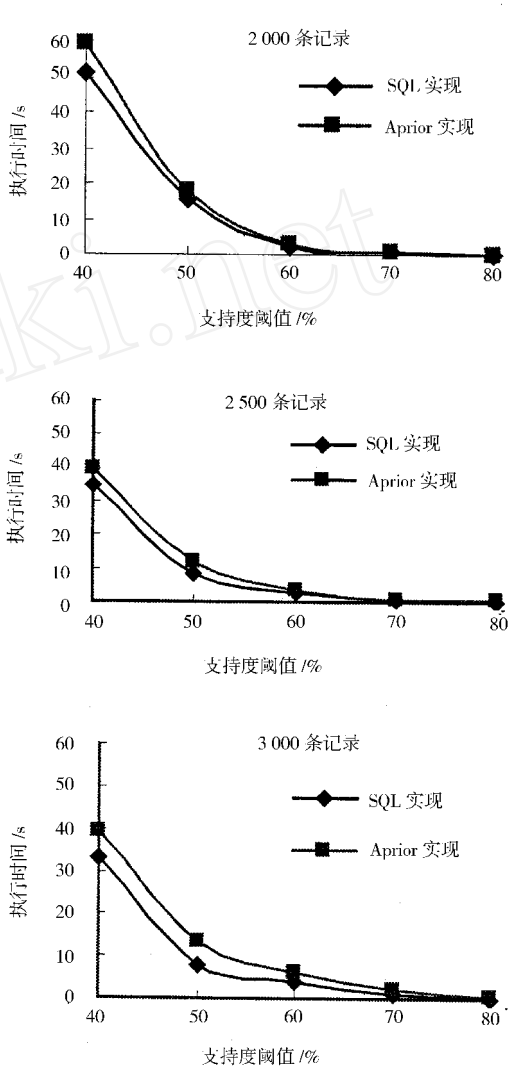

图 1　算法两种实现方法比较

Fig 1　Comparison of two implementation method of algorithm

5　结束语

数据挖掘过程需要处理大量的数据,而处理数据是数据库管理系统的优势所在.在挖掘存储在关系数据库中的数据时,可结合关系数据库系统的数据处理能力采用 SQL语句来实现挖掘算法.实验

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

表明,采用 SQL实现的 Apriori算法提高了挖掘效 率、简化了实现过程.

参考文献:

[1] Han J W, Kamber M. 数据挖掘概念与技术 [M]. 范 明,孟小峰,译. 北京:机械工业出版社, 2001.

[2] 萨师煊,王 珊. 数据库系统概论 [M]. 北京:高等教育出版社, 2000.

[3] Tan P N, Steinbach M, Kumar V. 数据挖掘导论 [M]. 范 明,范宏建,译. 北京:人民邮电出版社, 2006.

[4] Wu X D, Kumar V, Quinlan Ross J, et al. *Top 10 algorithms in data mining* [J]. Knowledge and Information Systems (S0219-3116), 2008, 14(1): 1-37.

[5] Hua K A, Jiang N, Villafane R, et al. *An algebraic approach to system performance analysis using data mining techniques* [C]//Proceedings of the 2003 ACM symposium on applied computing, March 09-12, Melbourne, Florida, 2003.

[6] 文继军,王 珊. SEEKER:基于关键词的关系数据库信息检索 [J]. 软件学报, 2005, 16(7): 1270-1281.

[7] 孟小峰,周龙骥,王 珊. 数据库技术发展趋势 [J]. 软件学报, 2004, 15(12): 1822-1836.

[8] Jeff S. *Mushroom dataset* [EB/OL]. (1987-04-27) [2009-04-18]. http://archive.ics.uci.edu/ml/datasets/Mushroom.

责任编辑:郭红建

(上接第 111页)

[5] 丁大均. 关于高层建筑中箱基和上部框架的共同作用 [J]. 建筑结构, 2002(8): 46-48.

[6] 罗立平,赵锡宏. 空间框架结构-厚筏-地基共同作用分析 [C]//上海高层建筑桩筏与桩箱基础设计理论. 上海:同济大学出版社, 1989.

[7] 李广信. 有关土的相互作用问题 [J]. 岩土工程学报, 1996, 18(6): 112-115.

[8] 谢晨智. 上部结构-筏板基础-地基土相互作用的有限元分析 [D]. 武汉:湖北工业大学, 2007.

责任编辑:郭红建

(上接第 123页)

[5] Atia A A, Donia A M, Yousif A M. *Synthesis of amine and thio chelating resins and study of their interaction with* Zinc(II), Cadmium(II) *and* Mercury(II) *ions in their aqueous solutions* [J]. Reactive & Functional Polymers (S1381-5148), 2003, 56: 75-82.

[6] 张超灿,庞金兴,李 曦,等. 核-壳型巯基胺螯合树脂的制备及其吸附性能 [J]. 武汉大学学报:理学版, 2001, 47(2): 189-191.

[7] 李 艺. PEI复合磁性分离材料的制备 [J]. 苏州大学学报:工科版, 2003, 23(4): 55-60.

[8] 忻新泉. 计算机在化学中的应用 [M]. 南京:南京大学出版社, 1986.

[9] 张延红,程国斌,马 伟. 利用 Oringin软件对吸附等温线拟合进行分析 [J]. 计算机与应用化学, 2005, 22(10): 899-902.

[10] 程国斌,张延红,王康平,等. 利用 Oringin软件实现化学实验数据的拟合分析 [J]. 化学教学, 2005(9): 46-48.

责任编辑:张建合

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net