基于抽样的概念层次数据挖掘算法*)

An Algorithm for Concept Hierarchy Mining Based on Sampling

李 波

(重庆工学院计算机科学与工程系 建设工业集团博士后科研工作站 重庆400015)

Abstract This paper first presents some traditional Attribute-Oriented Induction(AOI)algorithm in data mining field and points out shortcomings of them as follows: (1)they couldn't deal with the unbalanced concept hierarchy; (2)the final generalized result doesn't refer to the distribution of real data set. Hence, we put forward an algorithm for concept hierarchy mining based on sampling, which samples the dataset first, and arranges the initial concept hierarchy, then scans the whole dataset, later organizes the concept hierarchy according to the statistics information, finally gets the generalized rule by calculating the information of leaves. It not only solves the above problems, but also has optimal time and space complexity.

Keywords Data mining, Attribute-Oriented induction, Concept hierarchy

一、引言

属性归纳算法的出现主要有以下几个原因。首先,虽然某些规律,如关联规则可以在基本概念层上发现[3,4],但是一些更让人感兴趣的规律一般只在更高的概念层上才能发现,并且表达得更简洁一些。因此有必要将数据库中的基本数据泛化到相对高的概念层上才能更有效地挖掘数据。其次,由于自发的挖掘会产生太多的规则而失去重点,因此一般推荐由用户来提出数据挖掘的要求,这样可以有限制地搜索相关的数据集来挖掘出相关的数据。最后,存在某些知识背景可利用,如概念层次。这不仅提高了挖掘的效率,而且可以反映出用户在挖掘过程中的一些控制,这样可以更有利于得到期望的泛化结果[2]。

传统的属性归纳算法主要有 LCHR 算法[5],AOI 算法[6],GDBR 算法[7]。LCHR 算法和 AOI 算法的时间复杂性和空间复杂性都很大,无法处理大规模的数据。GDBR 算法虽然有最优的时间复杂度和较好的空间复杂度,但是它没有考虑实际数据分布对概念层次和泛化规则的影响,因此最后的结果往往不是用户希望的。这三个算法都假设所有的叶子概念到顶层的概念的距离是一样的,而实际中存在大量不平衡的概念层次。考虑上面这些问题,我们提出了抽样的概念层次挖掘算法。通过抽样得到实际数据的近似分布,并以此对原概念层次进行调整。这样可以带来的两个好处是:(1)使概念层次可以反映数据的实际分布,也就是最后的泛化规则可以反映数据的分布;(2)可以把不平衡的概念层次转化为平衡的。因为我们的算法只对源数据文件扫描一次,所以可以达到最优的时间复杂度;再者,因为我们的算法只存储动态调整后概念的统计信息,所以可以取得常数的空间复杂度。

二、基于抽样的概念层次挖掘算法的描述

给定一个初始相关数据集和相应的概念层次,以及预先设定的泛化阈值来求出最后的泛化规则,这就是一个典型的属性归纳算法求解的问题。

必须指出的是:(1)由于初始相关数据集很大,因而考虑到运行效率,要尽量避免对其扫描多次;(2)正是由于初始相关数据集很大,一般不可能完全读入内存,这就要求数据能分配处理数据;(3)现实世界中存在大量不平衡的概念层次,所以要求算法能处理不平衡的概念层次;(4)用户关心的往往是主要数据的泛化规则,所以要求最后的泛化规则要有主要数据的详细信息,也就是说要考虑数据分布对泛化规则的影响。

基于上面的认识,我们提出了基于抽样的概念层次挖掘算法。它通过抽样(sample-data)取得初始相关数据集的近似分布,并以此对初始相关数据集进行扫描(scan-data)并转化为新概念层次的基本概念,并统计信息。等扫描完后,根据统计信息对概念层次进行的调整,收集(generalize—relation)调整后概念层次中叶子概念的统计信息,就可以得到最后的泛化规则。详细算法如图1所示。

```
Procedure ConceptHierarchyMiningBasedonSampling ( data_set_
    type,init_data_set,int thread,concept_hierarchy_tpye * old_ch)
Begin//基于抽样的概念层次挖掘算法
  For each A_i do
//抽样数据
Sample_data(inti_data_set,buf[i],sample_ratio);
//调整概念层次
new_CH [i] = arrange_concept_hierarchy (old_CH [i], buf [i],
  threshold);
end for
//扫描初始相关数据
scan_data(init_data_set,buf,prime_relation);
for each A_i do
final_ CH [i] = arrange_concept_hierarchy (new_CH [i], buf [i],
    threshold);
//根据概念层次调整泛化关系
generalize_realition(final_CH[i],prime_relation,threshold);
end for
output prime_relation;
end
```

图1 基于抽样的概念层次挖掘算法

其中,arrange-concept—hierarchy 的主要功能是根据各概念的统计信息,对原概念层次进行调整。其基本思路是:根据实行属性的泛化阈值,确定一个权值门限。然后计算各概念的出现频率数,如其权值大于权值门限,则保留;反之,对其进行泛化操作。当概念个数小于属性的泛化阈值时,基于这些概念建立新的概念层次。详细算法如图2所示。

```
Fifunction arrange_concept_hierarchy(concept_hierarchy_type CH,
```

*)本项目受到重庆市重点企业信息化项目(20007340)资助。李 波 博士后,研究方向,计算机网络。目前研究兴趣:数据挖掘算法及应用。

```
  buf_type buf_i) as concept_hierarchy_type
Begin
Buf_type prime_buf;
Integer total_occurrence;//buf-i 中所有叶子概念出现次数的总和
Concept_hierarchy_type new_CH;
Double thres_value,//权限门值
(1)Set_null(prime_buf);
do{
do{
(2)Total_occurrence=computer(buf_i);//计算 buf-i 中所有叶子概
   念出现次数
(3)Thres_value=1/T;//T 为属性泛化阈值
(4)For j=1 to count buf_i of do
 If(buf_i[j].Times/total_occurrence)>thres_value)then
  Move buf_i[j]into prime_buf;
   End if
  End for
(5)If(count of prime_buf+count of buf_i)<=T then
 Move all elements of buf_I into prime_buf;
 Flag_generization_ok=true;
 Break;
 Else if thre is no changes of prime_buf,buf_I then
 Break;
 End if
 End if
 T=T-count of prime_buf;
 }while(T>0);
 if count of buf_i=0 or count of buf is not changed or flag_general-
   ization_ok=true then break;
 end if
(6)get_generalized_concept(buf_i,1);//将 buf-i 中所有的概念向上
   泛化一层
 }while(count of buf>0);
(7)new_CH=get_concept_hierarchy(prime_buf,CH);//建立新的
   概念层次
 return new_CH;
 end
```

图2 arrange_concept_hierarchy 函数

三、概念层次调整举例

为了解释我们在上面提到的概念层次调整函数，我们通过一个属性的概念层次的调整实例来说明整个调整过程。

假设想要了解重庆各地区购买某种保险的人员的分布情况。又设有属性字段 Area 表示重庆的地理分布，其概念层次如图3所示。

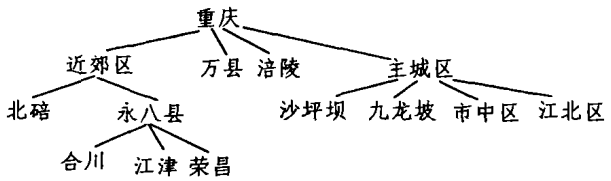

图3 Area 的概念层次

再设来自各地区的购买人数的数据如表1所示，其中概念名表示地区名，出现次数表示购买人数。设 sample－ratio＝10%，可以得到数据如表2所示，记录到缓冲区 buf 中。

首先，计算 total－occurrence＝100和 t＝1/T＝1/4＝0.25，然后对 buf 各结点进行扫描，计算 buf[i].Times/total-occurrence(1<=i<=10)，并与 t 比较。如 t 大于，则将该结点移入 prime 中，可得 prime＝{北碚}，重复(2)～(5)，直到 prime 没有变化，转(6)，buf 中各概念泛化一层，再转(2)重新计算，可得 prime＝{北碚、永八县、万县＋涪陵、主城区}，然后将原概念层次的中间结点入 prime，则 prime＝{北碚、永八县、万县＋涪陵、主城区、近郊区、重庆}，重建概念层次如图4所示。

表1 输入测试数据

概念名	出现次数
北碚	352
合川	103
江津	45
荣昌	100
万县	65
涪陵	74
沙坪坝	100
九龙坡	50
市中区	50
江北	61

表2 抽样数据

概念名	出现次数
北碚	36
合川	11
江津	3
荣昌	9
万县	7
涪陵	8
沙平坝	11
九龙坡	4
市中区	4
江北	8

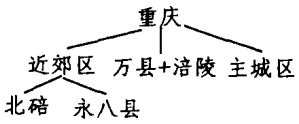

图4 调整后的概念层次

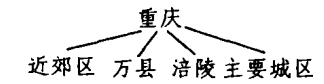

图5 未调整的泛概念层次

从图4中可以看出：概念层次低但占人数较多的北碚和永八县两个地区将出现在最后的泛化规则中，而概念层次高但占人数较少的万县和涪陵两个地区被合并了。这就使得最后的泛化规则反映了主要数据的详细泛化结果，而对次要数据只给出大致泛化结果。比较一下，我们可以得出，如果不调整原概念层次，则可以得到最后的泛化结果如图5所示。其最后的泛化结果概念层次没有反映出数据的分布。相比之下，可以看出调整后的概念层次更适合用户的需求。

四、基于抽样的概念层次挖掘算法分析

4.1 传统算法的性能分析

AOI 算法和 LCHR 算法有系统的输入与输出，只是在处理重复元组的方法上不同。AOI 算法是把重复元组插入到泛关系中。设 n 为相关数据集中元组的个数，p 为未排序的泛化关系表的大小，则每一个插入都至多要 p 个比较，因此 AOI 算法的时间复杂度为 O(n.p)。由于要保存输入的相关数据集，因此 AOI 算法的空间复杂度为 O(n)。LCH 算法是对泛化关系表进行排序，然后合并重复元组，设 n 为相关数据集中元组的个数，而最好的排序时间复杂度为 O(n.log(n))，即 LCHR 的时间复杂度为 O(n.log(n))。LCHR 算法的空间复杂度为 O(n)，用于保存输入的相关数据集。GDBR 算法的时间复杂度为 O(n)，因为它插入一个泛化元组只需 O(1)，其主要开销在读入相关数据集上，可以证明 O(n)时间复杂度是最优化的[3]。GDBR 算法的空间复杂度为 O(p)，$p=\prod_{i=1}^{m} t_i$，其中 m 为属性个数，t_i 为第 i 个属性最大不同属性个数。所以，当 n 很大时，由于 AOI 算法和 LCHR 算法要把整个初始相关数据

读入内存，导致大量使用虚拟内存，从而使性能迅速下降。而GDBR算法不会出现这样的情况。

4.2 基于抽样的概念层次挖掘算法分析

设η为抽样率，n为初始相关数据集中元组的个数，则提取抽样数据时间为$\eta \cdot n$。设对一概念进行概念层次调整时间为常数c_1，即第一次动态调整概念层次的时间为$c_1 \cdot \eta \cdot n$。设转化一个概念时间为一常数c_2，则对整个初始相关数据集的扫描时间为$c_2 \cdot n$。最后设第二次概念层次的调整和对prime-relation的调整时间为常数c_3，所以总的时间复杂度为$O(\eta \cdot n + c_1 \cdot \eta \cdot n + c_2 \cdot n + c_3) = O(n)$。

设元组的属性个数为m，各属性的不同概念数为n_i($1 <= I <= m$)，t_i为第i属性的泛化阈值，n(max)为属性的最大概念数，则Buf占空间为n(max)。初始概念层次$p = \Sigma c_1 \cdot n_i$，其中c_1为概念转化为概念层次的常数。最后的泛化规则prime-relation占空间为$s = \prod t_i$，所以总的空间复杂度为$O(nmax + p + s) = O(c)$，c为常数。

五、基于抽样的概念层次挖掘算法模拟测试

我们在以下环境中对算法进行了测试。开发环境：Microsoft Visual C++6.0 Windows NT 4.0 Server；运行环境：IBM兼容机，Intel Pentium II 233，128M RAM。对每一输入元组个数，模拟运算10次，取其平均值作为时间。具体数据如图6所示。

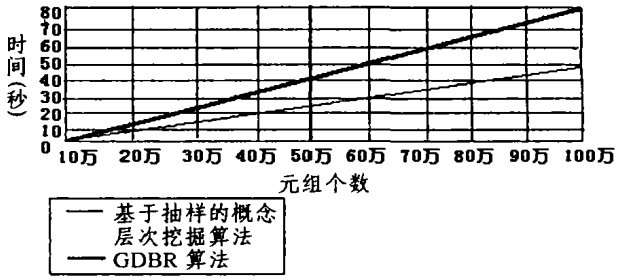

图6 测试结果

从图6我们可以看出基于抽样概念层次挖掘算法的执行时间与输入元组个数基本上是线性的，而且其处理大数据的能力是令人满意的，即具有较好的可扩展性。此外，我们还发现基于抽样的概念层次挖掘算法比GDBR算法执行时间短，这主要是因为基于抽样的概念层次挖掘算法在概念转化表(即原概念层次中的概念映射到调整后的概念层次中的概念的映射表)建立后，读入概念后，可直接得到其对应的泛化概念，而不用象GDBR算法那样读入一个概念后就要在全局概念表中查找，从而加快了扫描速度。

总结 本文分析了几种传统属性归纳算法，针对它们的不足，提出了基于抽样的概念层次挖掘算法，它不仅可以处理不平衡的概念层次，而且得到的泛化规则可以反映实际的数据分布。此外，这种算法具有最优的时间和空间复杂性。实验证明本文算法是有效的、可行的。

由于数据库的数据经常改变，并且重新计算一次泛化规则的代价非常昂贵，因此如何将基于抽样的概念层次挖掘算法与增量泛化和动态维护相结合，是值得我们进一步研究的内容。

参考文献

1 Han Jiawei, Fu Yongjian. Exploration of the Power of Attribute-Oriented Induction in Data Mining

2 Han Jiawei, Fu Yongjian. Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Database. http://www.cs.uregina.ca.

3 Agrawal R, Imielinski T, Swami A. Mining Association Rules Between Sets of Items in Large Databae. In Ptoc, 1993 ACM-SIGMOD Int. conf. Management of Data, Washington, D.C.: ACM Press, 1993. 207～216

4 Piatesky-Shapiro G, Frawley W J. Knowledge Discover in Database. AAAI/MIT Press, 1991

5 Han J, Cai Y, Cercone N. Data-Driven Discovery of Quantitave Rules in Relational Databases. IEEE Trans. Knowledge and Data Eng., Feb. 1993. 29～40

6 Cai Y N. Attribute-oriented Induction in Retaliation Databases. In: G. Piatesky-Spapiro, W. J. Frawley, eds. Knowledge Doscovery in Databeses, AAAI/MIT Press, 1991. 213～228

7 Carter C L. Efficient Attribute-Oriented Algorithm for Knowledge Discover from Large Databases. IEEE trans on Knowledge and data Engineering, 1998, 10(2): 193～208

(上接第74页)

结束语 本文在对文[2]提出的RRDM深入研究的基础上，对RRDM的基本定义进行了扩充，并提出了一种新的RRDM操作算子—分解算子，针对RRDM的基本概念和Rough关系操作，得出了一些相应的结论。

Rough关系数据库模型是对传统关系模型的扩充，它是为研究不确定性数据问题而发展起来的一个新学科，目前国内外对这个学科的研究还处于初级阶段，现在研究较多的是把Rough集理论和别的数学工具结合起来对信息系统处理，而真正把Rough集和关系数据库本身的理论结合起来进行研究的比较少。在Rough关系数据库理论研究方面还有很多不足，需要研究的问题还有很多，比如Rough关系演算、Rough域演算，基于Rough关系数据库的数据约束、知识抽取、函数依赖理论、不确定知识的度量等、Rough关系及其操作的性质等，从而建立它自己的完善的理论体系，使之更好地为实际应用服务。

参考文献

1 Dey D, Sarkar S. A probabilistic relational model and algebra. ACM Transactions on Database Systems, 1996, 21(3): 339～369

2 Theresa B, Petry F E, Buckles B P. Extension of the relational database and its algebra with rough set techniques. Computational Intelligence, 1995, 11: 233～245

3 Guan J W, Bell D A. Rough Computational methods for information systems. Artificial Intelligence, 1998, 105: 77～103

4 Lin T Y. An Overview of Rough Set Theory from the Point of View of Relational Databases. Bulletin of Internal Rough Set Society. Volume 1, Number 1

5 张文修，吴伟志. Rough集理论介绍和研究综述. 模糊系统与数学，2000, 14(4)

6 王国胤编著. Rough集理论与知识获取. 西安交通大学出版社，2001, 5

7 王能斌编著. 数据库系统原理. 电子工业出版社，2000, 1