网络环境下海量信息的局部模式提取方法*

王腾蛟，唐世渭，杨冬青，刘云峰

(北京大学 计算机科学技术系，北京 100871)；

(北京大学 视觉与听觉信息处理国家重点实验室，北京 100871)

E-mail：tjwang@db.pku.edu.cn

http://www.pku.edu.cn

摘要：海量信息的模式提取是网络环境下海量信息集成研究的难点.给出了一种新的局部精确模式提取及其增量保持方法，通过探测目标集的路径距离，利用Hash类及其路径距离操作，将模式的生成规模控制在"模式直径"范围内，从而有效地抑制了模式膨胀.

关键词：海量信息；半结构化数据；模式提取；数据模型；信息集成

中图法分类号：TP393 **文献标识码**：A

随着Internet的发展和异构信息源集成技术的进步，对网络环境下的海量信息进行集成、分析处理并提供决策服务成为当前研究领域的新热点.针对处理技术的不同，海量信息大致可分为3类.第1类是传统的关系或面向对象数据库中的数据.这类数据源具有严格的、相对稳定的模式信息，因而容易操作.第2类是半结构化数据(semistructured data)[1].这类数据的特点是数据的结构不规则(irregular)或不完整(incomplete)，表现为数据不遵循固定的模式，结构隐含，模式信息量大，模式变化快，模式和数据统一存放等特点.第3类是近来发展很快的用扩展标识语言extensible markup language(XML)[2]表示的数据.

目前，对海量信息的研究主要集中在对半结构化数据和XML数据的数据模型、模式提取、查询表达及优化和DBMS集成服务上，其中模式提取是至关重要的一步，查询表达和系统的集成都强烈地依赖于提取的模式[3].人们对半结构化数据源的模式提取已做了很多工作，其中最具代表性的是文献[3]所提出的在OEM(object exchange model)上提取模式DataGuide的方法.DataGuide已被证实在许多方面发挥着重要作用，如浏览、查询表达、存储统计数据等.现在，XML已成为World Wide Web上数据表示与交换的新标准[2].虽然XML中的DTD(document type definition)可以提供模式信息，但在得不到DTD时，模式提取则是必不可少的.

以DataGuide为主流的模式提取方法都是提取出整个数据源的模式.这种建立全局模式的方法有一个重要的缺陷：文献[3]表明，在一个数据源上建立一个DataGuide等价于将非确定有限自动机(NFA)转变为确定有限自动机(DFA)，在最坏情况下，算法的复杂性为指数级，只能在数据源是树型结构或节点数量级较小的有向图中使用.而且，即使生成了全局模式，由于它过于庞大，系统要为它的维护付出很大的代价.在海量信息的集成环境下，直接应用DataGuide是不可行的，一般

* 收稿日期：2000-05-09；修改日期：2000-07-06

基金项目：国家重点基础研究发展规划973资助项目(G1999032705)；北京大学-IBM创新研究院资助项目

作者简介：王腾蛟(1973－)，男，山东济南人，博士生，主要研究领域为数据库，信息系统；唐世渭(1939－)，男，浙江宁波人，教授，博士生导师，主要研究领域为数据库与信息系统；杨冬青(1945－)，女，天津人，教授，博士生导师，主要研究领域为数据库与信息系统；刘云峰(1973－)，女，山东泰安人，助教，主要研究领域为管理信息系统.

采用牺牲精度的方法,即生成近似模式.这种方法虽然降低了时间复杂度,但同时也带来了误差,生成了实际并不存在的路径.

在一个实际的海量信息集成环境中,用户往往不需要得到全部数据源的模式信息,如在一个有关学校的半结构化数据库中,用户的一次访问可能仅需查询有关图书馆的信息,因而希望系统生成局部的、精确的模式信息.但是,困难在于半结构化的数据模型往往是一个连通的有向图(还可能有圈)结构,因为传统的提取全局模式的方法实质上是一种递归的穷尽扫描方法,因而无法将模式生成过程控制在局部.

目前对局部模式提取的研究还很少,但是在海量信息集成环境的应用中又迫切需要局部的模式信息.我们承担了国家重点基础研究发展规划973资助的"网络环境下海量信息组织与处理的理论与方法研究"中的"面向内容的海量信息集成、分析处理与服务"的研究课题.本文给出了一种新的海量信息局部精确模式生成方法,通过建立Hash类及其路径距离操作,将模式的生成规模控制在"模式直径"范围之内,从而有效地抑制了模式膨胀.用户可以依据自身需要"定制"所需模式规模的大小.解决了半结构化或XML数据源集成系统中因为对象数量级过高而不能生成精确模式的问题.同时,为了适应海量信息模式变化快的特点,我们还给出了相应的局部模式增量保持算法.

1 数据模型和全局模式提取

在这一节中,我们引用典型的半结构化数据的模型和模式提取方法来进行说明,并作为后面我们设计的海量信息局部模式提取的基础.

1.1 OEM数据模型

OEM数据模型是很具代表性的半结构化数据模型,最早由Stanford University的TSIMMIS项目组提出[4],并成功地应用于Lore[3]半结构化数据库管理系统.OEM是一个简单的、自描述的、嵌套的对象模型.从结构上看,OEM是一个带标记的有向图,节点表示对象,每个对象具有唯一的对象标识(oid),边表示了对象间的聚合关系.对象分为原子对象和复合对象两种,原子对象包含一个原子类型的值,如整型、实型、字符串、gif文件等等.复合对象的值是有序对〈标记,子对象〉的集合,每一个标记是对象与子对象之间关系的描述.每个OEM图都有一个根节点,从根节点至任意节点都应是可达的.

1.2 全局模式提取

定义1(标记路径). 一个OEM对象o的标记路径是一个或多个标记的序列$l_1,l_2,\ldots,l_n$,并且$\exists\ o_1,o_2,\ldots,o_n$,使得$\langle l_1,oid(o_1)\rangle\in value(o),\ldots,\langle l_i,oid(o_i)\rangle\in value(o_{i-1})$,其中$value(o)$是对象$o$的值,$i=1,2,\ldots,n$.

定义2(数据路径). 一个OEM对象o的数据路径是一个或多个标记和oid的交替序列$l_1,o_1,l_2,o_2,\ldots l_n,o_n$,使得形成一个从$o$开始的经过$n$个对象$(x_1,x_2,\ldots,x_n)$的具有$n$条边$(e_1,e_2,\ldots,e_n)$的路径,其中边$e_i$的标记为$l_i$,对象$x_i$的oid标识为$o_i$.

定义3(实例). 如果数据路径d中的标记序列等于标记路径l,则称数据路径d是标记路径l的实例.

定义4(目标集(target set)). 一个OEM对象s和它的标记路径l的目标集$t=\{o\mid l_1,o_1,l_2,o_2,\ldots,l_n,o\}$,其中$l_1,o_1,l_2,o_2,\ldots,l_n,o$是$l$的数据路径实例.该目标集记作$t=T_s(l)$.

定义5(DataGuide). 设某个OEM模型的根节点为s,该模型的DataGuide也是一个OEM模

型,设其根节点为 d,s 的每条标记路径在 d 中有且只有一条数据路径,并且 d 的每条标记路径都是 s 中的一条标记路径.

定义 6(Strong DataGuide). OEM 模型 s 和其 DataGuide d,设 $L_s(l)=\{m|T_s(m)=T_s(l)\}$,$L_d(l)=\{m|T_d(m)=T_d(l)\}$,如果对于 s 的任意标记路径均有 $L_s(l)=L_d(l)$,则称 d 是 s 的 Strong DataGuide.

例 1:图 1 是一个很简单的 OEM 模型和由全局模式提取算法得到的 Strong DataGuide 模式图.

Fig. 1 A simple OEM model and its Strong DataGuide schema graph
图1 一个简单的OEM模型和它的Strong DataGuide模式图

由于该算法在最坏情况下的复杂性为指数级,只有在数据模型是树型或节点数量级较小的有向图中使用(文献[3]中给出了实验数据,表明在无圈的情况下,节点数小于 10 000 时还是可行的).在海量信息的集成环境下,直接生成全局模式是不可行的,为了解决这一问题,我们给出下面的局部模式提取方法.

2 局部模式提取方法

针对海量信息集成系统环境中用户对模式信息的需求,我们设计了一种新的在海量信息环境下提取局部模式的算法.其方法是在生成目标集(target set)的过程中,通过探测目标集的路径距离,并建立操作"路径距离"的 Hash 类来监控模式的生成规模.

2.1 模式直径

模式直径是我们在生成局部模式之前建立的,并在生成的过程中遵循的搜索范围.依据模式直径,利用局部模式生成算法就可以实现"定制"模式大小,但是,我们通过在原型系统中的实验发现,如果模式直径选取不当,就可能会生成无意义的模式信息(如模式直径过小时,生成的模式图中没有包括源模型图中的任何叶节点,依据这样的模式,无法生成有用的查询).这里,我们给出计算模式直径的上界和下界算法.

算法 1.

输入:待查询的目标集 T.

输出:模式直径的下界 MinDistance 和上界 MaxDistance.

```
SearchDiameterBoundary(T,MinDistance,MaxDistance)
{
 for u∈T do
 {for all v≠u do L(v)  ω(u,v)
  L(u)=0; CurrentSet={u}
  while CurrentSet≠V do
  {v'=SearchNextNearNode(CurrentSet)
       //找出一点 v'∉CurrentSet,且∀ v∈CurrentSet,都有 L(v)≤L(v')
   CurrentSet=CurrentSet∪{u}
```

```
    for all v∉CurrentSet do
        if L(v)>L(v')+ω(v',v) then L(v)=L(v')+ω(v',v)
  }
  for all v≠u do
      if v.NextNode=NULL then LeafDistanceHash.Insert(v,L(v))
 }
 MinDistance=LeafDistanceHash.GetNearLeaf();
 MaxDistance=LeafDistanceHash.GetFarLeaf();
}
```

算法 1 的思想是以目标集(target objects)的顶点集合为中心,用 Hash 表记录搜索路径的距离.若存在顶点 u 至 v 的边,则 $\omega(u,v)=1$,否则 $\omega(u,v)=+\infty$. $SearchNextNearNode()$可以通过扫描图的邻接表实现,因而可以在 $O(n)$内完成.整个算法的复杂性为 $O(n^2\times|T'|)$.

2.2 局部模式提取

局部模式提取的难点在于控制模式生成过程,根据模式直径限制新目标集的发现.我们给出的算法 2 描述了局部模式生成的过程,其中动态建立的 MapHash 结构是一个三元组$\langle T,Sm,Depth\rangle$,记录了已发现的目标集和模式中对象的映射关系及目标集的路径深度.我们将该结构与相应的操作封装在一起构成 MapHash 类.

算法 2.

输入:源数据模型的根节点和某条标记路径.

输出:该标记路径对应的目标集的局部模式(顶为 Sm).

```
ExtractSchema(o,l) //o 是源数据模型的根节点,l 是 o 的一条标记路径
{
  T=MakeTarget(o,l)
  Sm=CreateObject();
  SearchDiameterBoundary(T,MinDistance,MaxDistance)
  Diameter=GetSchemaDiameter(MinDistance,MaxDistance)
  MapHash.Insert(T,Sm,0) //在 MapHash 建立初始信息,局部模式根节点的深度为 0
  RecursiveExtract(T,Sm,Diameter)
}
RecursiveExtract(T,Sm,Diameter)
{
  if MapHash.GetDepth(T)>Diameter then return;
        //搜索到深度大于模式直径的目标集时,到达递归出口
  p=GetAllOutEdge(T) //得到 T 中的每个顶点的全部出边
  for all distinct l in p do
  {
    T'=MakeTarget(T,l)
    Sm'=MapHash.Search(T')
    if Sm'≠NULL then
      { //Hash 表中已存在该目标集时,仅加入相应的边及其标注
        Sm.Value=Sm.Value∪{⟨l,Sm'⟩}
        return
```

```
        }
    else
        {
        Sm'=CreateObject()
        MapHash.Insert(T',Sm',MapHash.GetDepth(T)+1)
        Sm.Value=Sm.Value∪{(l,Sm')}
        RecursiveExtract(T',Sm',Diameter)
        }
  }
}
```

算法2通过广度优先扫描发现目标集,并追踪记录目标集的深度,把搜索深度大于模式直径作为递归出口,实现了抑制模式膨胀的目的.定理1给出了上述算法的正确性证明.

定理1. 由算法2对源数据模型(s)生成的局部模式(Sm)是局部精确的,即s的每条标记路径在d中最多只有一条数据路径,并且d的每条标记路径都是s中的一条标记路径.

证明:如果存在s的某条标记路径$l(l_1,l_2,\ldots,l_n)$,在d中有两条不同的数据路径$l_1,o_1,l_2,o_2,\ldots,l_i,o_i,\ldots,l_n,o_n$和$l_1,o'_1,l_2,o'_2,\ldots,l_i,o'_i,\ldots,l_n,o'_n$.设$o_i\neq o'_i$是两序列中第1个不等的对象标识,则有$o_{i-1}=o'_{i-1}$,$T=MakeTarget(o_{i-1},l_i)$,若$o_i=T$,则$o'_i\neq T$,这与目标集定义矛盾,故不存在不同的数据路径.同理可证,d的每条标记路径都是s中的一条标记路径. □

例2:图2给出的是一个简单的数据库研究中心的半结构化数据模型,我们通过上面给出的算法,提取有关Project部分的模式信息.

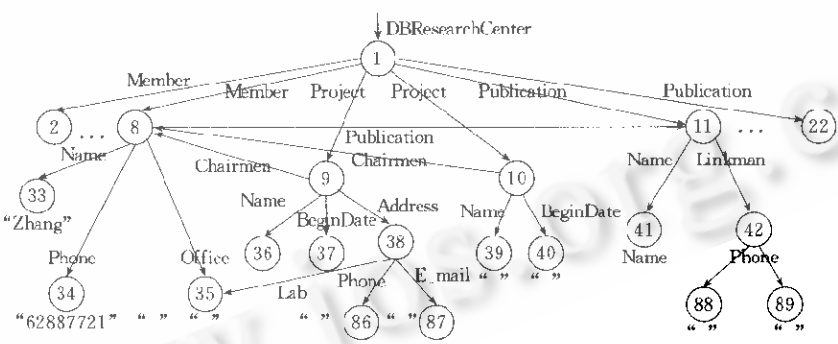

Fig. 2 A semistructured data model for a DBResearchCenter
图2 一个DBResearchCenter的半结构化数据模型

当执行算法2时,输入为根节点&1和其标注路径*DBResearchCenter.Project*.算法第1步MakeTarget得到第1个目标集为$T=\{\&9,\&10\}$,然后以该目标集为中心,生成如图3所示的局部模式图(模式半径为2).

在海量信息集成系统中,依据这个简单的局部模式图,就可以构造出如图4所示的局部模式框架提供给用户.

利用这个框架,用户可以很方便地构造查询.如查询李××在1998年主持的所有项目,可以写成:

SELECT *Project*

FROM *DBResearchCenter.Project d*

WHERE *GetYear*(*d.BeginDate*)="1998" and *d.Chairmen.Name*="李××";

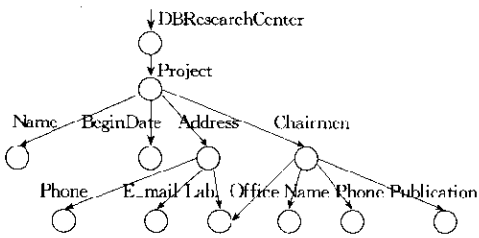

Fig. 3 Local schema of project part

图3 Project部分的局部模式

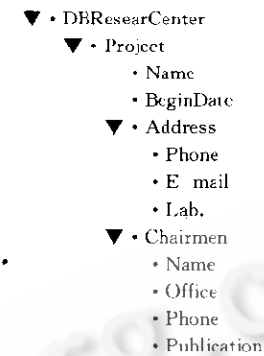

Fig. 4 User interface of local schema frame

图4 局部模式框架的用户界面

2.3 局部模式的增量保持

网络环境下海量信息的特点之一就是数据及其模式的易变性. 当数据源的信息发生变化后,我们以前的局部模式信息也就有可能"过时"了,因而需要保持局部模式与数据源信息的一致性. 算法3给出了增量保持的方法. 注意,因为模式是局部的,因而对于数据源的变化U,我们仅需要测试在模式直径范围内可能需要修改的对象或边.

算法3.

输入:目标集T,模式直径和对数据库的修改U.

输出:增量保持后的局部模式.

```
SchemaMaintance(T,Diameter,U)
{
  UpdatePointSet=GetAllUpdatePoint(U)
  for v∈UpdatePointSet do
    { if GetDistance(T,v)<Diameter then
        { MaintenancePointSet=GetMaintenancePoint(v)
          for Sm∈MaintenancePointSet do
            RecursiveMaintenance(Targetof(Sm),Sm,Diameter)
        }
    }
}
RecursiveMainTenance(T,Sm,Diameter)
{
  if MapHash.GetDepth(T)>Diameter then return;
  p=GetAllOutEdge(T)   //得到T中的每个顶点的全部出边
  for all distinct l in p do
  {
    T'=MakeTarget(T,l)
    Sm'=MapHash.Search(T')
    if Sm'≠NULL then
      { if (l,Sm')∉Sm.value then
        {
```

```
        if ⟨l,*⟩∈Sm.Value then
          Sm.Value=Sm.Value−⟨l,*⟩
        Sm.Value=Sm.Value∪{⟨l,Sm'⟩}
        if MapHash.GetDepth(Sm')>MapHash.GetDepth(Sm)+1
          MapHash.Update(T',Sm',MapHash.GetDepth(Sm)+1)
      }
      else
      {  Sm'=CreateObject()
        MapHash.Insert(T',Sm',MapHash.GetDepth(Sm)+1)
        if ⟨l,*⟩∈Sm.Value then
          Sm.Value=Sm.Value−⟨l,*⟩
        Sm.Value=Sm.Value∪{⟨l,Sm'⟩}
        RecursiveExtract(T',Sm',Diameter)
      }
  }
}
```

算法3首先排除了源数据模型中到目标集 T 的距离超过模式直径的修改点. 对于模式直径范围内的修改点 v,利用 MapHash 计算出局部模式图中所有包含 v 的顶点,记为 MaintenancePointSet,然后递归计算 U 对每个点及其子图的修改.

3 实验数据

局部模式生成算法的关键在于能否限制提取模式的过程,生成局部的精确模式. 定理1已经给出其正确性证明,另外,我们还在海量信息集成环境的原型系统中实验了局部模式提取算法的有效性. 表1是在 PIII 450上实验得到的数据,其中提取局部模式时的模式直径取值范围为5～8. 第3行数据中没有提取全局模式的时间,这是因为原算法复杂性为指数级,对于结点(对象)数目过高的图,无法得到运行结果.

Table 1 Experiment data for schema extracting

表1 模式提取实验数据

Data model①				Whole schema②			Local schema③		
Data resource④	Tree or not⑤	Objects⑥	Links⑦	Objects	Links	Time⑧(s)	Objects	Links	Time(s)
1	Y	300 150	301 200	4 409	4 678	231.6	572	1 006	42
2	N	3 227	5 714	7 114	9 225	34.2	792	815	0.9
3	N	24 500	60 431	5 607	7 446	—	1 402	3 044	15

①数据模型,②全局模式,③局部模式,④数据源,⑤是否树型,⑥对象,⑦链接,⑧时间.

4 总 结

本文针对网络环境下海量信息集成系统环境中用户对模式信息的需求,设计了一种新的提取局部模式的算法. 其方法是在生成目标集(target set)的过程中,通过探测目标集的路径距离并建立操作"路径距离"的 Hash 类来监控模式的生成规模,从而解决了因为对象数量级过高,而不能生成精确模式的问题.

利用数据源中的语义模型,针对对象间的语义相似度来控制模式膨胀,是我们今后继续研究的方向.

References:

[1] Abiteboul, S. Querying semi-structured data. In: Afrati, F., Kolaties, P., eds. Dabase Theory—— ICDT'97. Lecture Notes in Computer Science 1186, New York: Springer-Verlag, 1997. 1~18.

[2] Bray, T., Paoli, J., Sperberb-McQueen, C. Extensible markup language (XML) 1.0. W3C Recommendation. 1998. http://www.w3.org/TR/1998/REC-xml-19980210.

[3] Goldman, R., Widom, J. DataGuide: enabling query formulation and optimization in semistructured database. In: Bymatthias, J., ed. Proceedings of the 23rd International Conference on Very Large Data Base. Athens, Greece: Morgan Kaufmann Publishers, Inc., 1997. 436~445.

[4] Papakonstantinou, Y., Garcia-Molina, H., *et al*. Object exchange across heterogeneous information source. In: Yu, P.S., Chen, A.L.P., eds. Proceedings of the 11th International Conference on Data Engineering. Taipei: IEEE Computer Society, 1995. 251~260.

Extracting Local Schema from Massive Information in Network Environment*

WANG Teng-jiao, TANG Shi-wei, YANG Dong-qing, LIU Yun-feng

(*Department of Computer Science and Technology, Beijing University, Beijing* 100871, *China*);
(*National Laboratory on Machine Perception, Beijing University, Beijing* 100871, *China*)
E-mail: tjwang@db.pku.edu.cn
http://www.pku.edu.cn

Abstract: Extracting schema from massive information is very difficult for the research on massive information integration in network environment. A new method is presented in this paper, which is about extracting and incremental maintenance of local accurate schema. In this process, the algorithm control the scale of extracted schema within the 'schema diameter' by examining the path distance of the target set and using the Hash class and its path distance operation. This method is very efficient for restrain schema from expanding.

Key words: global information; semi-structured data; extracting schema; data model; information integration

* Received May 9, 2000; accepted July 6, 2000
Supported by the National Grand Fundamental Research 973 Program of China under Grant No. G1999032705; the Foundation of PKU-IBM Innovation Institute of China