ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.2, February 2008, pp.323−337
DOI: 10.3724/SP.J.1001.2008.00323

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

S-CBR:基于数据库模式展现数据库关键词检索结果*

彭朝晖[1,2,3+], 张 俊[1,2,4], 王 珊[1,2]

[1](中国人民大学 信息学院,北京 100872)
[2](教育部数据工程与知识工程重点实验室,北京 100872)
[3](山东大学 计算机科学与技术学院,山东 济南 250014)
[4](大连海事大学 计算机科学与技术学院,辽宁 大连 116026)

S-CBR: Presenting Results of Keyword Search over Databases Based on Database Schema

PENG Zhao-Hui[1,2,3+], ZHANG Jun[1,2,4], WANG Shan[1,2]

[1](School of Information, Renmin University of China, Beijing 100872, China)
[2](Key Laboratory of Data Engineering and Knowledge Engineering, Ministry of Education of China, Beijing 100872, China)
[3](School of Computer Science and Technology, Shandong University, Ji'nan 250014, China)
[4](School of Computer Science and Technology, Dalian Maritime University, Dalian 116026, China)
+ Corresponding author: E-mail: pengch@ruc.edu.cn

Peng ZH, Zhang J, Wang S. S-CBR: Presenting results of keyword search over databases based on database schema. *Journal of Software*, 2008,19(2):323−337. http://www.jos.org.cn/1000-9825/19/323.htm

Abstract: In this paper, a result presentation method of keyword search over databases named S-CBR (schema-based classification, browsing and retrieving) which includes three phases of result classification, user browsing, and retrieving is proposed. Firstly, S-CBR uses database schema and query keywords to produce the first-level classes automatically and classify results into them. For large classes, it further partitions them based on the content of keyword nodes. Furthermore, S-CBR gives each class a readable description, and presents the description and each result graphically to help users understand the results easily. Users also can retrieve the class which they are interested in based on the information of the first-level classes to find the results they need or acquire more relevant results. Experimental results verify the method's effectiveness and efficiency.
Key words: classification; result presentation; keyword search over database; database schema; heuristic search

摘 要: 提出一种基于数据库模式的数据库关键词检索结果展现方法 S-CBR(schema-based classification, browsing and retrieving),包括结果分类、用户浏览和再次检索 3 个过程.S-CBR 首先利用数据库模式和查询关键词自动产生第一级类别,将检索结果分配到各个类中;对于比较大的类,按关键词节点内容进行第二级分类;另外赋给每个类别一个类别描述,并将类别描述和每个结果图形化地展现出来,使用户容易阅读和理解检索结果.用户还可以根据 S-CBR 提供的结果类别模式信息对感兴趣的类别作进一步检索,以尽快找到所需结果或获取更多的相关结果.实验证明了 S-CBR 方法的有效性.
关键词: 分类;结果展现;数据库关键词检索;数据库模式;启发式检索

* Supported by the National Natural Science Foundation of China under Grant Nos.60473069, 60496325 (国家自然科学基金)
Received 2007-04-02; Accepted 2007-07-17

中图法分类号: TP311 文献标识码: A

关系数据库关键词检索(keyword search over relational databases,简称KSORD)使得普通用户能够使用关键词来检索关系数据库,而不需要了解数据库模式和SQL语言[1,2].目前,关于KSORD已有许多研究工作,从实现机制上可分为基于数据图的和基于模式图的两种.基于数据图的系统包括BANKS[3,4],ObjectRank[5],DETECTOR[6,7]等;基于模式图的系统包括DBXplore[8],DISCOVER[9],IR-Style[10],SEEKER[11],Hunter[12]等.各个KSORD系统对检索结果的定义有所差别:在ObjectRank系统中,检索结果被定义为单个元组,这个元组与查询相关但是不必包含查询关键词;而在DBXplorer,BANKS,DETECTOR,IR-Style等大多数系统中,每个检索结果必须包含查询关键词,通常,结果是来自多个关系的元组的连接(元组连接树).

检索结果面临的一个重要问题是,如何改进系统的结果展现,使用户能够迅速地理解结果,进而尽快找到所需的结果.KSORD结果展现并不是简单的问题[1]:一方面,系统检索结果是单个元组或多个元组构成的元组连接树,这种形式很不直观;另一方面,系统往往产生大量的结果,其中很多结果是相似的,尽管系统提供排序(rank)机制,但是排序的规则并不一定符合当前查询的需要,用户往往不容易从结果列表中发现自己所需的结果.进一步来说,好的结果展现应当不但能让用户读懂结果,还能主动提供一定的启发信息,帮助用户作进一步的查询.

本文针对返回结果为元组连接树的KSORD系统,提出一种基于数据库模式的结果展现方法S-CBR (schema-based classification, browsing and retrieving).S-CBR采用按结构分类和按内容聚类相结合的方法,将检索结果组织成两级类别,有助于用户快速浏览结果.在此基础上,用户可以对某些感兴趣的类别做进一步的检索.与其他的KSORD结果组织方法相比,S-CBR是一种新的KSORD结果展现架构,其优点在于,使用户能够观察到可能的检索结果的全貌,给用户提供了启发信息,能够支持用户作进一步更深入的检索.

本文第1节介绍本文所要用到的基本概念.第2节详细介绍S-CBR方法的体系结构和算法.第3节介绍实验评估结果.第4节介绍与本文相关的工作.第5节总结全文并提出未来工作.

1 基本概念

为便于描述问题,我们首先定义要用到的基本概念,其中的定义1~定义4基于文献[3,12],并根据本文的需要作了少许修改.

定义1(模式图). 一个关系数据库的模式可以表示为一个无向的模式图$G(V,E)$,V是节点集,E是边集.其中,对于数据库中的每个关系R,G中有一个相对应的节点$u_R\in V$;对于数据库中每一对关系R和S,如果R和S之间存在主外码联系,G中就包含一条无向边$\langle u_R,u_S\rangle\in E$.

图1所示为DBLP[13]数据库的模式图.DBLP数据库由Author,Paper,Write,Cites这4个关系组成,这几个关系之间存在着主外码联系,在模式图中,用阿拉伯数字表示边的序号.

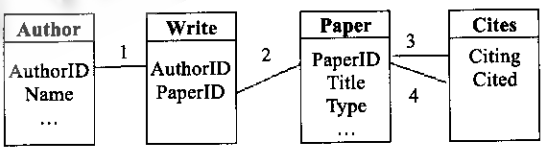

Fig.1 DBLP database schema graph

图1 DBLP数据库模式图

定义2(数据图). 一个关系数据库的数据可以表示为一个无向带权的数据图$G_D(V,E)$,V是节点集,E是边集.其中,对于数据库中的每个元组t,G_D有一个相对应的节点$u_t\in V$,本文中,我们对数据库中的元组和数据图中的节点不加区别;对于数据库中的每一对元组s和t,如果s和t之间存在主外码联系,G_D中就包含一条无向边$\langle u_s,u_t\rangle\in E$;$G_D$中的每个节点和边被预先按照一定规则赋予一个权重[3].

定义3(关键词查询). 用户的一次输入被定义为一个关键词查询Q,由一组关键词$k_1,\ldots,k_n$组成($n\geq 1$),表示为

$Q(k_1,\ldots,k_n)$.

若节点包含关键词作为其属性值的一部分,则称该节点和该关键词是相关的,并称该节点为关键词节点.通常,KSORD 的第 1 步是利用 RDBMS 提供的全文索引,对每一个关键词 k_i 确定与其相关的关键词节点集合 S_i.

定义 4(结果树). 关键词查询的一个结果是一棵有根无序的带权树(元组连接树),它是数据图的一个子图,包含每个 S_i 中的至少 1 个节点,而且每个叶子节点必然是关键词节点.

结果树的根称为信息节点,因为它将所有的关键词节点连接起来,并尽可能多地反映了关键词节点之间的关系.

每棵结果树有一个相关性分数,它是根据各节点和各边的权重来计算的,结果树应该按照相关性分数降序排列,系统返回用户所要求的 Top-k 结果树.

例如,在 DBLP 数据库上,给定关键词查询(Hristidis,Papakonstantinou),KSORD 系统将从 DBLP 的数据图中检索出 Top-k 结果树,每棵结果树其实是 DBLP 数据图的一个子图.图 2 给出了其中一棵结果树的示例,其含义是 Hristidis 和 Papakonstantinou 合作了一篇论文,论文题目是"Efficient IR-Style Keyword Search...".

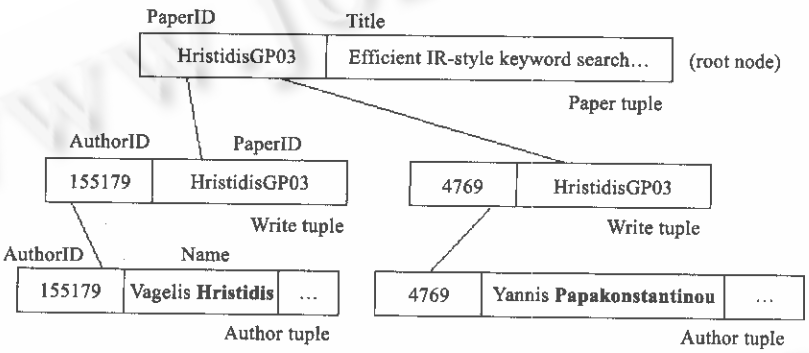

Fig.2 A result tree of DBLP database (data subgraph)

图 2 DBLP 数据库的结果树(数据子图)

定义 5(结果树模式). 利用数据库模式信息对结果树的节点和边作标号,即可得到结果树模式,它是一种有根无序的标号树.其标号采用如下两个规则:

规则 1(树节点标号). 假设节点 $t\in$关系 R,则 t 的标号为$[R]$.

规则 2(树边标号). 假设边$\langle t_1,t_2\rangle$,其中,$t_1\in$关系 R_1,$t_2\in$关系 R_2,假定边所对应的外码和主码属性分别为 R_1 的属性$(A_1,\ldots,A_i,\ldots,A_r)(1\le i\le r)$和 R_2 的属性$(B_1,\ldots,B_i,\ldots,B_r)(1\le i\le r)$,则$\langle t_1,t_2\rangle$的标号为$\{(A_1,\ldots,A_i,\ldots,A_r),(B_1,\ldots,B_i,\ldots,B_r)\}$.

例如,图 3 即为图 2 结果树的结果树模式,为简化起见,图 3 中省略了边的标号,该结果树模式的含义是:两个作者合作写了一篇论文.

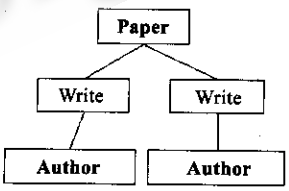

Fig.3 An example of result-tree-schema

图 3 结果树模式示例

2 S-CBR 方法

我们首先要解决的问题是给出一种良好的组织结果树的方法,从而提高用户浏览结果的效率.大量的观察和实验证实[14],许多结果树具有同样的结构,表达了相似的语义.例如在,DBLP 数据库中,关键词查询(Jim Gray, Transaction)会产生很多结果,其中一些结果的含义是 Jim Gray 写了关于 Transaction 的论文,另一些结果的含义是 Jim Gray 的论文被关于 Transaction 的论文引用,还有一些结果是 Jim Gray 的论文引用了关于 Transaction 的论文.这样,将这些结果树按照结构的异同组织成不同的类别,对每个类别给出可读性好的类别描述,就有助于用户快速浏览结果.

对于每次查询,S-CBR 方法首先从模式图出发,利用查询关键词信息,生成结果树模式,每个结果树模式代表一种结构(语义),作为一个类别,这样就形成了检索结果的第一级类别;然后,S-CBR 将系统返回的 Top-k 结果按照结构的异同分配到第一级类别中,对于包含结果数量较大的类别,再按照各个结果树的关键词节点的内容异同作聚类,形成第二级类别;将这两级类别按照一定规则排序后,S-CBR 将类别描述和每个结果用图形化的方式展现出来.在此基础上,用户可以对感兴趣的类别作进一步的检索.S-CBR 方法的框架如图 4 所示.

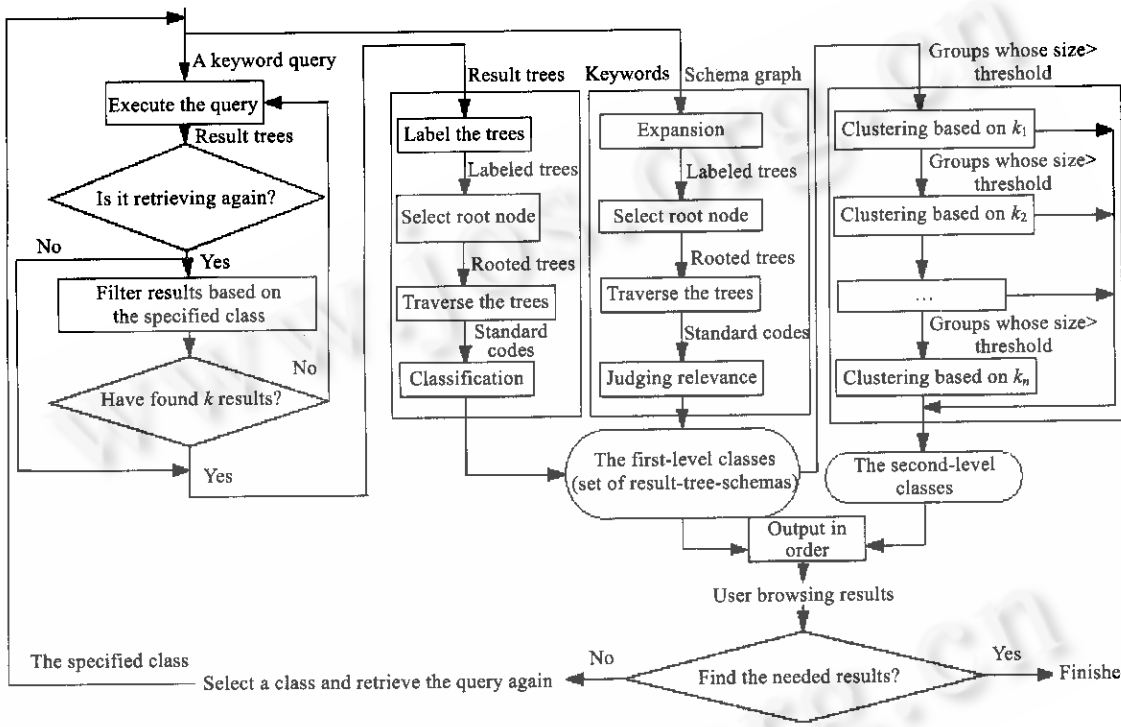

Fig.4 Framework of S-CBR

图 4 S-CBR 框架

2.1 结果树和类别的匹配

我们先介绍如何将一棵结果树分类到某一个类别中,然后在下一节介绍如何产生这些类别.我们知道,每次查询产生的结果树是一棵有根无序树,对它加标号之后得到的结果树模式是一种有根无序的标号树,我们采用判断树同构的方法实现结果树的分类.也就是说,若一棵树的结果树模式和一个类别(本身也是结果树模式)同构,则将这棵树分到这一类中,否则不属于这一类.这样,我们就把结果分类问题转化为有根无序标号树的同构判断问题.下面介绍关于树同构的一些结论,包括定理 1~定理 3 的内容,其详细说明可参见文献[12].

定理 1. 两棵有根有序标号树是同构的,当且仅当它们的先序遍历编码相等.

定义 6(标准编码). 设 T 是一棵有根无序标号树,从 T 派生的所有有根有序树为 $T_1,T_2,\ldots,T_n$,它们的先序遍历编码分别为 $S_1,S_2,\ldots,S_n$.我们称其中的最小编码 $S_{\min}$ 为 T 的标准编码.

算法 1. 生成标准编码(GetStandardCode 算法).

输入:有根无序标号树 T 的根节点 *root*,特殊符号"#"和"$"("#">"$">所有的标号).

输出:T 的标准编码 *St*.

Begin

01 *St*←*label*(*root*)+"$"; //"+"表示字符串连接,函数 *label* 获取节点或边的标号

```
for 从 root 到其儿子的每条边 e do
    St2←label(e);
    获取 e 的另一个端点 n;
    将 St2+GetStandardCode(n)+"$"插入到集合 S 中;
将 S 中的字符串按升序排列;
for S 中的每个串 s do
    St←St+s;
return St+"#";
End
```

定理 2. 算法 1 正确地计算了有根无序标号树 T 的标准编码.

定理 3. 两棵有根无序标号树是同构的,当且仅当它们的标准编码相等.

由定理 3 可知,只要对结果树模式计算它们的标准编码,然后判断标准编码是否相同,就可以完成结果的分类.但是,由于 KSORD 系统检索机制的原因,同一结构的不同结果树,它们的根可能是不对应的,这样会影响分类的正确性.例如,图 5 和图 2 的树不相同(两个作者合作的论文题目不同),但是具有相同的结构,都表示两个作者合作写一篇论文.然而,它们的根节点并不对应,这样,两棵树对应的结果树模式作为有根树就不是同构的.因此,我们需要重新确定每棵结果树的根,确保同一结构的树的根节点是对应的;同时,根节点还应该包含尽可能多的信息.显然,在图 5 中,结果树的根节点应为 Paper Tuple.

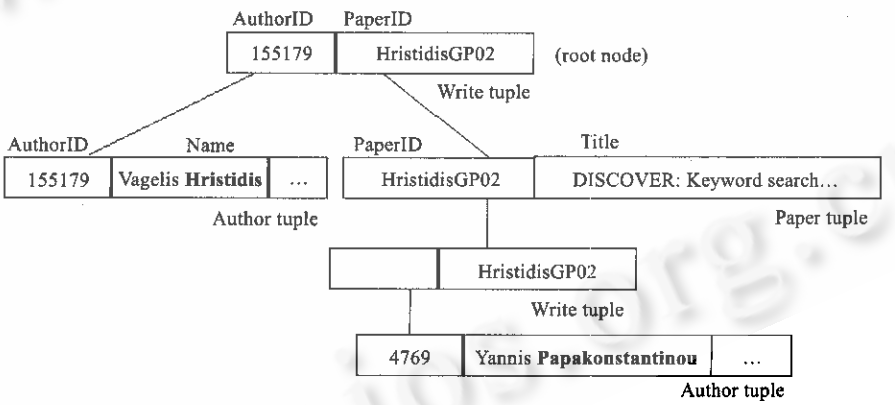

Fig.5 A result tree in the same pattern with the tree in Fig.2

图 5 与图 2 的结果树同结构的一棵结果树

我们首先考虑选择度最大的节点,若满足条件的节点有多个,则从中选择最靠近树的中心的节点.通常,满足上述两个条件的候选节点只有 1 个.如果这样的节点仍有多个,则分别以每个候选节点为根,采用算法 1 分别计算树的标准编码,选择产生最小标准编码的候选节点为根.如果有多个候选节点产生同样的最小标准编码,我们就可以选择其中任意一个作为根节点,因为它们产生同样的标准编码,对树同构的判断没有影响.

2.2 类别生成

S-CBR 产生的第一级类别其实就是结果树模式.我们首先介绍结果树模式的生成,然后介绍它的排序.

2.2.1 结果树模式生成

S-CBR 采用扩展的方法从模式图出发产生结果树模式.

定义 7(扩展). 在模式图 G 中,e 是节点 u 和 v 之间的边,则称 v 是扩展 u 得到的,并称这个扩展使用了产生式 $u \rightarrow uev$.

将图 1 中 DBLP 模式图的 4 个关系 Author,Paper,Write,Cites 简称为 A,P,W,C,则可以生成的产生式集合是

$F=\{A\rightarrow A1W,W\rightarrow W1A,W\rightarrow W2P,P\rightarrow P2W,P\rightarrow P3C,C\rightarrow C3P,P\rightarrow P4C,C\rightarrow C4P\}$.分别从模式图的每个节点出发,使用这些产生式可以扩展得到不同的结果树模式.图 6 给出了 DBLP 模式图上的 6 个结果树模式的示例以及从 A 节点开始生成它们的扩展步骤.

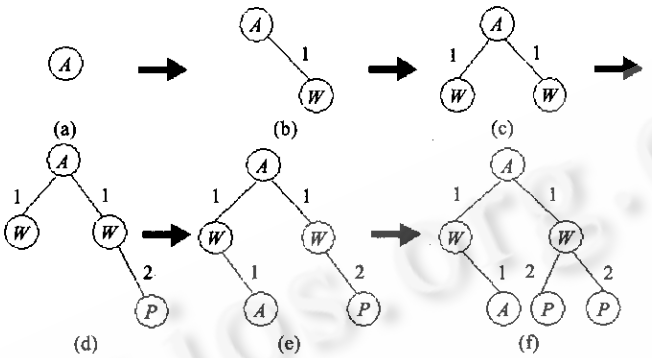

Fig.6 The result-tree-schemas expanded from the schema graph

图 6 从模式图扩展产生的结果树模式

算法 2 给出了生成结果树模式的算法,其中,需要我们指定结果树模式的最大允许大小(节点个数)MaxSize 作为程序结束的条件.

算法 2. 生成结果树模式.

输入:模式图 G,结果树模式最大允许大小 MaxSize,关键词查询 $Q(k_1,\ldots,k_n)$.

输出:结果树模式集合 *PTSet*.

V:队列,存储等待扩展的无根树

Z:队列,存储和 V 中元素相对应的有根树

Begin

01 **for** 模式图 G 中的每一个关系 $R_i,i=1,\ldots,m$ **do**

02 根据 G 生成产生式集合 F;

03 以 R_i 为根,生成单节点的无根树 T,将 T 插入到 V 的队尾;

04 由 T 生成有根树 T',将 T' 插入到 Z 的队尾;

05 调用扩展算法 Expand,获得部分结果树模式的集合 *PTSet*1;

06 将 *PTSet*1 的元素加入到 *PTSet*;

07 从 G 中删除关系 R_i;

End

算法 2 中调用了算法 3,算法 3 使用无根树作为扩展的基础,每次扩展一条边,产生一棵新的无根树,然后使用第 2.1 节中的方法选定根节点,把它转化为有根树(结果树模式).对有根树计算其标准编码,然后判断它和队列中已经产生的树是否同构,若不同构,则把新产生的树加入到队列中.

算法 3. 树扩展(Expand 算法).

输入:队列 V,Z,产生式集合 F,结果树模式最大允许大小 MaxSize,关键词查询 $Q(k_1,\ldots,k_n)$.

输出:部分结果树模式的集合 *PTSet*1.

Begin

01 **while** V 不为空 **do**

02 取出 V 的队首元素 U,取出 Z 的队首元素 U';

03 **if** *IsRelevant*(U') **then** 将 U' 插入到 *PTSet*1 的队尾;

04 **if** U 的大小<MaxSize **then**

```
05        for U 的每一个节点 n do
06            从 F 中获取 n 可以使用(符合规则 3 至规则 5)的产生式集合 F1;
07            for F1 的每一个产生式 f do
08                从 n 出发利用 f 扩展一条边,形成一棵新树 T;
09                由 T 生成有根树 T',调用算法 1 生成 T'的标准编码 S_T';
10                if S_T' 和 Z 中每棵树的标准编码都不相同 then
11                    将 T'插入到 Z 的队尾,将 T 插入到 V 的队尾;
12    从 V 中删除队首元素 U,从 Z 中删除队首元素 U';
End
```

需要注意的是,算法 3 使用产生式对每个节点作扩展时,选择的产生式必须满足以下 3 个规则,否则可能扩展出一些无效的结果树模式.这些无效结果树模式不对应任何结果树.

规则 3. 在扩展节点 R 时,只能选用形如 $R\rightarrow ReS$ 的产生式.

规则 4. 若关系 R 和 S 之间存在"外码→主码"联系,R 是外码所在关系,S 是主码所在关系,节点 R 是使用产生式 $S\rightarrow SeR$ 扩展节点 S 得到的,则在扩展 R 时,不能选用产生式 $R\rightarrow ReS$.

例如,关系 W 和 A 之间存在"外码→主码"联系,W 是外码所在关系,A 是主码所在关系,在图 6(e)中,节点 A 使用 $A\rightarrow A1W$ 扩展出了节点 W,节点 W 又使用 $W\rightarrow W1A$ 扩展出一个新的节点 A,这样得到的图 6(e)是不能与任何结果树相对应的.假设存在一棵结果树与图 6(e)相对应,这棵树对应节点 W 的元组为 $w\in$关系 W,对应两个节点 A 的元组为 $a_1\in$关系 A,$a_2\in$关系 A,那么,w 的外码既是 a_1 的主码又是 a_2 的主码,因此,元组 a_1,a_2 的主码相同,它们是同一个元组,这在实际的结果树(数据子图)中是不可能出现的情况.所以,图 6(e)是无效的结果树模式.

规则 5. 若关系 R 和 S 之间存在"外码→主码"联系,R 是外码所在关系,S 是主码所在关系,则在扩展节点 R 时,产生式 $R\rightarrow ReS$ 最多只能选用一次.

算法 3 中调用了算法 4,算法 4 判断有根树队列 Z 中的结果树模式与当前查询的相关性,按照结果树的定义,叶子节点一定是关键词节点,如果结果树模式的叶子节点所对应的关系不包含当前查询的任何关键词,则该结果树模式与当前查询不相关.例如,假设我们在关系 Write 上不建立全文索引,则关系 Write 的所有元组都不是关键词节点,图 6(d)的叶子节点 W 对应的关系就不包含任何关键词,那么,图 6(d)就与当前查询不相关.

算法 4. 结果树模式与关键词查询的相关性判断(IsRelevant 算法).

输入:结果树模式 T',关键词查询 $Q(k_1,\ldots,k_n)$.

输出:T'是否与当前查询 Q 相关.

```
Begin
01    for T'的每一个节点 n do
02      if n 是叶子节点 then
03          if n 所对应的关系的元组不包含任何关键词 then
04              return false;
05    return true;
End
```

定理 4. 算法 2 能够从模式图 G 产生所有不大于 MaxSize 的与当前查询相关的不同构的有效结果树模式.

S-CBR 规定,只有与当前查询相关的有效的结果树模式才可以作为第一级类别.定理 4 说明,算法 2 产生的结果树模式都可以作为第一级类别.结合第 2.1 节和文献[12]中的定理 4.1,可以很容易地证明定理 4.

2.2.2 结果树模式排序

作为第一级类别而产生的结果树模式,应当按照合理的顺序展现给用户.需要注意的是,一些结果树模式类别中可能不包含任何结果,其可能的原因有二:一种是数据库数据本身产生的结果树全都不具备这些模式;另一种是因为系统只返回 Top-k 结果,而这 Top-k 结果恰好都不具备这些模式.我们对结果树模式采用如下规则排序:

(1) 将不包含任何结果的结果树模式都排在包含结果的结果树模式后面;

(2) 对包含结果的模式,以其包含结果的最大相关性分数作为该模式的分数,按此分数降序排列各模式;

(3) 对不含结果的模式,按照它们的大小由小到大排列,结果树模式越小越排在前面(符合人们的阅读习惯).对于相同大小的结果树模式,我们提出如下打分机制对它们打分,按分数将其降序排列.

对结果树模式的打分,需考虑其节点和边的权重.我们根据应用特征,预先给模式图的每个节点和边赋予一个权重,并将这个权重作为结果树模式对应节点和边的权重.节点的权重应反映该节点关系的重要程度,边的权重应反映关系间主外码联系的重要程度,节点和边的权重越大,它们的分数越高,结果树模式的分数也越高.

假定节点 u 的权重表示为 $N(u)$,最大的节点权重为 $N_{\max}$,则节点 u 的分数为

$$NScore(u)=\log(1+N(u)/N_{\max}).$$

假定边 e 的权重表示为 $W(e)$,最大的边权重为 $W_{\max}$,则边 e 的分数为

$$EScore(e)=\log(1+W(e)/W_{\max}).$$

结果树模式的分数是

$$Score=\sum_e EScore(e)\times\lambda+\sum_u NScore(u)\times(1-\lambda),$$

其中,λ是控制因子,它决定边和节点的分数在计算结果树模式分数时分别所起的作用大小.

2.3 内容聚类

在第一级分类之后,发现某些类别所包含的结果树仍然很多,考虑根据内容实现第二级分类.因为用户最敏感的是他们输入的查询关键词,所以,考虑根据关键词节点的内容来分类.各个关键词不能被同等看待,实际上,不同的关键词在数据库中有不同的出现频率.例如查询(Gray,Database),Gray 只出现在少数元组上,而 Database 出现在大量元组中.可以根据与低频率关键词相关的节点的内容异同来对结果数量超过阈值的第一级类别作进一步划分.在本例中,首先根据 Gray 分类,这样,与不同的 Gray(如 Jim Gray 和 W.A. Gray)相关的结果树就分离开了.用户只需检查每一个子类别的标签(Jim Gray)或(W.A. Gray),就能决定哪个子类是他们所感兴趣的.

内容聚类的过程如图 4 所示,即首先将关键词根据其在数据库中的出现频率排序,出现频率等于与该关键词相关的关键词节点的个数,假定排序后的关键词顺序为 $k_1,k_2,\ldots,k_n$.然后,我们检查各类别的结果数目,对数目超过阈值的类别,首先按照关键词 k_1 分组,即如果两棵结果树中与 k_1 相关的节点的内容相同,则将这两棵树分为一组;否则,它们属于不同的组.若新产生的组的元素个数仍然大于阈值,我们将继续基于关键词 k_2 分组,直到各组的元素个数都小于阈值或者所有的关键词使用完毕.需要说明的是,内容聚类产生的组都作为第二级类别.

2.4 结果展现

S-CBR 采用图形化的用户接口将检索结果图示出来,使用 Windows 资源管理器的风格展示两级类别,如图 7 所示.第 2.2.2 节已介绍了第一级类别的排序规则,第二级类别的排序是以其包含的结果的最大相关性分数作为本类别的分数,按分数降序排列.对于每一类中的结果树,都按照它们的相关性分数降序排列.

S-CBR 对每个类别给出一个类别描述,第一级类别描述以图 3 所示的结果树模式为基础,二级类别描述即是产生该类时所依据的关键词的值.S-CBR 将类别描述和每个结果都以直观的“树”的形式图示出来,当鼠标点击界面左侧的两级类别时,S-CBR 在界面画板中图示该类的类别描述;当点击类中的树时,S-CBR 在画板中图示所选择的结果树,例如,图 7 画板中显示的就是第 7 类结果的第 3 子类的第 1 棵结果树,表示 Gray 的合作者 Raymond A. Lorie 写了关于 Database 的论文.

为了提高结果和类别描述的可读性,S-CBR 在图示时做了一些工作:一是类描述和结果树都可以由系统管理员为关系和属性名称配置别名,为边配置方向.这样,对用户隐藏了数据库的模式信息,展示的内容更接近自然语言;二是可以预先配置每个关系只选择部分属性在结果树中显示,以突出主要信息;三是每棵结果树按照其结果树模式的标准编码图示,这样,同类的结果树的图示结构相似,便于用户浏览.

对于不包含结果的第一级类别,我们使用特殊符号(*)将其标出,例如图 7 中第 10 类之后的类都是空类别,用户可以对空类别再次进行检索.

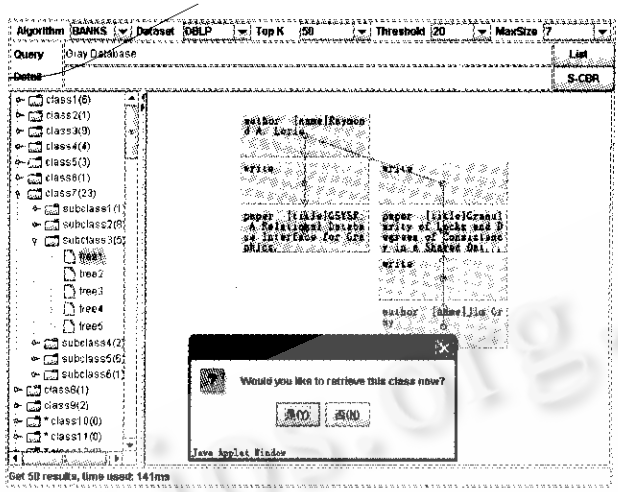

Fig.7 The result tree in GUI and the heuristic search interface

图 7 GUI 中的结果树及启发式检索接口

2.5 启发式检索

系统返回的是 Top-k 结果,然而,一个潜在的问题是,用户并不知道如何选择这个 k 值:如果 k 太小,则可能找不到需要的结果;如果 k 太大,则会检索出很多无用的结果,使用户感到疲倦.

S-CBR 方法的一个优点是根据模式图预先产生类别模式,将不包含结果的类别模式也显示出来,给用户提供了全局信息,使用户能够了解到可能的所有结果的结构(语义).这些类别模式(如图 3 所示)就是我们所说的启发信息,用户能够在此基础上作启发式检索.具体来说,对于空类别,用户可以双击该类别,系统对这一类再次检索,检索出来的 Top-k 结果全都属于这一类;对于非空的类别,也可用同样的方式再次检索,获取该类的更多结果.这样,用户开始检索时可以先用较小的 k 值,然后根据反馈的启发信息再次查找,最终找到所需要的结果.这种启发式检索对于用户提交查询时不知道如何确定 Top-k 的 k 值这种情况非常有效.

启发式检索的接口如图 7 所示.对于启发式检索的实现,我们采用了简单的结果过滤的方法,如图 4 所示,即只有与指定的类别相匹配的结果才允许返回给用户.这种实现方法的优点是在 KSORD 检索引擎的外部实现结果匹配,不需要修改 KSORD 内核,能够适用于多种 KSORD 系统.

3 实验评估

我们使用 Java 实现了结果分类系统,系统接收用户输入,传给 KSORD 系统;对于 KSORD 返回的结果,用户可以选择两种结果展现方式:List 和 S-CBR.List 是传统的结果展现方法,只是简单地将结果按相关性分数降序排列出来.我们的实验使用 AMD844*4 CPU 和 4G 内存的曙光服务器,操作系统是 Windows 2000 Advanced Server,数据库使用 Oracle9i,KSORD 系统选用我们基于文献[3]实现的 BANKS 系统,通过 JDBC 连接 Oracle9i.数据集采用 DBLP 真实数据的一个子集,所产生的数据图包括 497 000 个节点和 567 000 条边.我们在 Author 的 Name 属性,Paper 的 Title 和 Type 属性上建立了全文索引.

3.1 对参数MaxSize的评估

MaxSize 是 S-CBR 方法的一个重要参数,选择 MaxSize 的值需要综合考虑各种因素.在本节的每个实验中,我们都是随机产生 100 个查询,计算平均值.查询所用的关键词随机抽取自数据集,去掉了生僻和没有实际含义的词.

3.1.1 MaxSize 和第一级类别个数的关系

一般来说,MaxSize 越大,产生的结果树模式和第一级类别就越多.但是,因为只有与当前查询相关的结果树

模式才能作为第一级类别,所以对于给定的 MaxSize,第一级类别的个数还受到查询关键词的影响.图 8 给出了当 MaxSize 从 5 增长到 10、关键词个数分别为 2~5 的关键词查询所产生的第一级类别的个数情况.可以看出,随着 MaxSize 的增多,第一级类别个数增长得很快;而对同一个 MaxSize,查询关键词的个数越多,第一级类别个数也越多.这是因为关键词个数越多,从模式图扩展产生的结果树模式的叶子节点所对应的关系就越可能包含关键词(参见算法 4),这个结果树模式就越可能作为第一级类别.另外,在 DBLP 数据库的模式图中,从一个可能包含关键词的关系(Author 或 Paper)只能扩展到一个不包含关键词的关系(Write 或 Cites,因为没有建立全文索引),而以不包含关键词的关系为叶子节点的结果树模式不能作为第一级类别,所以,图 8 曲线中存在水平线段.

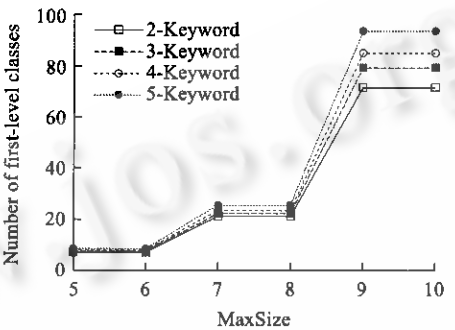

Fig.8 The number of produced first-level classes

图 8 产生的第一级类别的个数

3.1.2 系统效率分析

使用两种结果展现方式分别运行系统,然后比较它们的运行效率.List 所用的时间即是系统检索产生 Top-*k* 结果的时间,S-CBR 的时间由两部分构成:一部分是 List 时间;另一部分是结果分类时间 Classification. Classification 也由两部分组成:一部分是产生结果树模式的时间 Result-Tree-Schema;另一部分是对结果作结构分类和内容聚类的时间.在下面的效率分析中,将分别比较 List,S-CBR,Classification 和 Result-Tree-Schema 这 4 个值.

我们将查询关键词个数固定为 3、返回结果数固定为 100,观察随着 MaxSize 的变化,系统各部分所需要的运行时间.从图 9 可以看到,List 时间显然不受 MaxSize 影响;Result-Tree-Schema 时间随着 MaxSize 的增加呈指数增长;Classification 时间基本上与 Result-Tree-Schema 相同(分别代表 Classification 和 Result-Tree-Schema 的两条曲线基本重合).说明分类的主要时间消耗在结果树模式的产生上,而对结果的分类操作,包括树的遍历、内容聚类等的效率是很高的.总体上看,在 MaxSize 不大于 7 时,Classification 对系统的效率基本没有影响;到 MaxSize 为 9 时,Classification 对系统的效率产生了比较明显的影响.

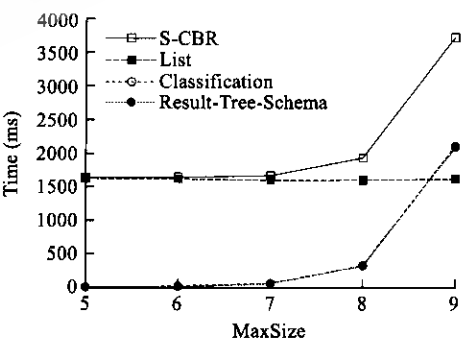

Fig.9 Efficiency (*KeywordNum*=3,*ResultNum*=100)

图 9 效率(*KeywordNum*=3,*ResultNum*=100)

固定 MaxSize 为 9、Top-k 的 k 值为 100,我们观察随着关键词的变化系统效率的变化情况.从图 10 可以发现,虽然树的查询相关性判断中涉及到了关键词,但是关键词个数对 Result-Tree-Schema 和 Classification 时间基本没有影响;随着关键词个数的增多,List 时间增加很快,当关键词个数超过 3 时,List 时间超过了 Classification 时间.总体来看,关键词个数越多,Classification 对系统效率的影响相对越小.

固定 MaxSize 为 9、关键词个数为 3,我们观察随着检索结果数 k 的变化系统效率的变化情况.从图 11 看到,随着 k 值增加,List 时间缓慢增加,Classification 时间则保持恒定,说明结果分类的效率不受检索结果数的影响.

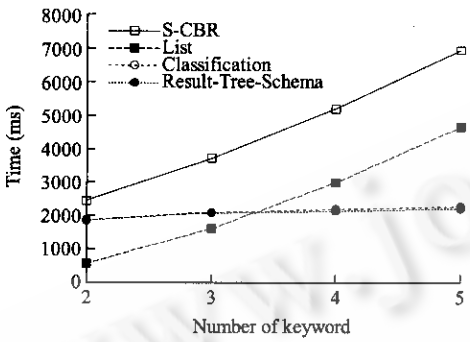

Fig.10 Efficiency (MaxSize=9,k=100)

图 10 效率(MaxSize=9,k=100)

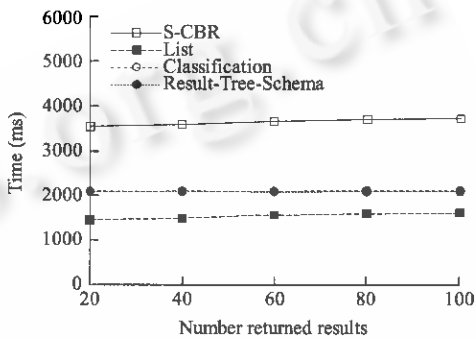

Fig.11 Efficiency (MaxSize=9,*KeywordNum*=3)

图 11 效率(MaxSize=9,*KeywordNum*=3)

3.1.3 MaxSize 的选择

MaxSize 并非越大越好,一方面会影响系统检索的效率;更重要的是,如果一棵结果树很大,这个结果就很难被理解,缺乏实际意义,通常的模式图系统(如 DISCOVER,SEEKER 等)产生的结果的节点数一般不超过 6 或 7.所以,我们的 S-CBR 方法通常可设置 MaxSize 为 7.由图 9 可知,当 MaxSize 为 7 时,S-CBR 和 List 的效率几乎相同.

MaxSize 的选择还要受关键词个数的影响,关键词数量大,结果本身所包含节点数就比较多.这时,需要适当加大 MaxSize 以提供给用户相关的结果.例如,关键词个数超过 3 时,可将 MaxSize 从 7 增加到 9.由图 10 可知,关键词个数超过 3 时,Classification 对系统造成的影响相对减少,MaxSize 为 9 时,S-CBR 从效率上可被用户接受.

3.2 S-CBR检索效果评估

3.2.1 效果评估用例

对于 KSORD 结果展现效果,至今尚没有权威的评价标准,我们使用文献[11]实验所用的 20 个例子来测试 S-CBR 的效果.文献[11]的 20 个例子是用来测试 SEEKER 系统的查准率和查全率的,并非专门为我们的 S-CBR 方法构建,拿它来测试 S-CBR 的效果,相当于是随机取得的测试用例,具有说服力.对于这 20 个例子,文献[11]给出的查询任务是查询某个作者发表的关于某个主题的论文,本实验中,我们再增加一项查询任务,即查询某个作者发表的被某个主题论文引用了的论文.显然,这两个任务的结果分别属于一种第一级类别,它们的类描述如图 12 所示.

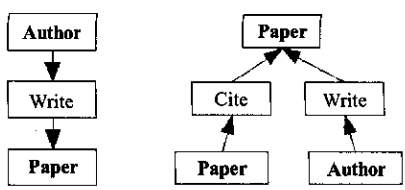

Fig.12 Class description: task 1 (left), task 2 (right)

图 12 类描述:任务 1(左)、任务 2(右)

设 MaxSize 为 7,首次查询的结果数为 100,观察在 List 和 S-CBR 两种方式下两个任务的检索结果,见表 1.对 List,给出了满足条件的第 1 个结果和最后一个结果在结果列表中分别所处的位置以及符合条件的结果数目.例如,查询 1 中的 10/13(4)表示满足任务 1 的第 1 个结果排在第 10 位,最后一个结果排在第 13 位,符合条件的结果共有 4 个.对 S-CBR,我们给出首次检索和再次检索(启发式检索)时,满足条件的一级类别所在的位置和该类所包含结果的个数.例如,查询 2 中的 1(5)表示再次检索时,第 1 个结果类别满足任务 1,该类包含 5 个结果.

Table 1 Search effectiveness evaluation of List and S-CBR

表 1 List 和 S-CBR 检索效果评估

No.	Keyword query	Task 1			Task 2		
		List	S-CBR		List	S-CBR	
			The first time search	The second time search		The first time search	The second time search
1	Knuth algorithm	10/13(4)	2(4)		15/15(1)	3(1)	
2	Turing article	No needed results	3(0)	1(5)	No needed results	6(0)	1(0)
3	Codd relational	3/66(24)	2(24)		67/81(10)	4(10)	
4	Fagin proceeding	No needed results	5(0)	3(0)	No needed results	9(0)	6(0)
5	Hartmanis NP	3/15(13)	2(13)		No needed results	8(0)	6(0)
6	Dijkstra programming	4/11(8)	2(8)		28/29(2)	3(2)	
7	"Jim Gray" article	1/23(23)	1(23)		24/100(59)	2(59)	
8	Salton "Information retrieval"	1/27(27)	1(27)		28/87(8)	2(8)	
9	"C.J. Date" database	3/18(16)	2(16)		19/96(31)	3(31)	
10	"Marvin Minsky" article	7/15(9)	2(9)		No needed results	9(0)	6(0)
11	Dewitt database	1/40(40)	1(40)		41/100(29)	2(29)	
12	Molina peer-to-peer	No results					
13	"Jennifer Widom" database	1/6(5)	1(5)		No needed results	11(0)	1(7)
14	"Jiawei Han" "data mining"	No results					
15	Bernstein concurrency	1/19(18)	1(18)		18/99(55)	2(55)	
16	Ullman database	1/27(19)	1(19)		14/99(60)	2(60)	
17	"Rakesh Agrawal" "data mining"	No results					
18	"Juris Hartmanis" complexity	1/29(29)	1(29)		30/30	2(1)	
19	Foster grid	No results					
20	Stonebraker object	1/3(3)	1(3)		4/92(46)	2(46)	

表 1 中有 4 个查询标注"检索不到结果"(no needed results),表示没有返回任何结果.这是因为文献[11]中的实验针对 DBLP 全集,而本实验使用的是 DBLP 的子集,在这个子集中没有符合任务的结果是正常的.有些查询标注"无"(no results),表示 List 下的查询有返回的结果,但是都不满足查询任务.原因既可能是满足任务的结果都在 Top-100 以外,也可能是本数据集中根本没有满足任务的结果.这时,使用 S-CBR 的启发式检索就能够判断是哪一种情况.

3.2.2 一级分类效果分析

由表 1 可知,KSORD 按照自身排序规则返回的结果顺序并不一定符合用户当前查询的需要,例如,在查询 3 中,满足任务 1 的 24 个结果分布在第 3 个~第 66 个结果中.这种情况在任务 2 中表现得更为明显,例如,在查询 7 中,满足任务 2 的 59 个结果分布在第 24 个~第 100 个结果中.这种情况下,用户若使用 List 方式浏览结果,效率会非常低.而 S-CBR 方式下,结果按照结构(语义)组织成不同的类,只需浏览类描述即可,大幅度提高了用户的浏览效率.例如,在查询 3 中,第二类结果就满足任务 1,同样,在查询 7 中,第 2 类结果即满足任务 2,很容易找到.

有一些查询任务是跨类别的,即满足条件的结果分布在不同的类中,这时需要把所有类描述浏览一遍,然后深入到若干相关的类中查找结果,这时的用户检索效率比 List 方式还要高.因为由图 8 可知,通常一级类别的个数并没有多少(其中还有空类别),可以快速浏览,还可以利用二级分类描述提高一级类别中的结果的浏览效率.

在获得表 1 的数据过程中我们发现,S-CBR 的一级类别排序规则是合理的,特别是将空类别从小到大排列,符合人们的阅读习惯,容易找到自己的所需.

3.2.3 二级分类效果分析

S-CBR 的二级分类即内容聚类,它将较大的一级类别作了进一步划分,进一步降低了信息超载.而且,基于内容的分类提供了较好的信息发现功能.例如查询 7(“Jim Gray”,Article),满足任务 2 的一级类别包括 59 个结果,即 Jim Gray 的论文被 59 篇论文引用.在内容聚类之后可以发现,这 59 篇论文每篇引用 Jim Gray 论文多少次,我们可以很容易地看到.再如查询 15(Bernstein,Concurrency),满足任务 1 的一级类别包括 18 个结果,内容聚类之后可以发现有两个 Bernstein,其中,Philip A. Bernstein 写了 16 篇论文,Bernstein 写了 2 篇论文.

3.2.4 启发式检索效果分析

在 S-CBR 方式下,若首次查询没有返回满足任务的结果,可以对某一种类别作启发式检索.在表 1 中,查询 2 的任务 1 和查询 13 的任务 2 给出了启发式检索的应用实例.在这两个任务中,首次检索所得的满足任务的类别都不包含任何结果,这时,仅对该类别再次检索即可得到所需的结果.这两个任务之所以首次检索没有得到结果,是因为满足任务的结果不在 Top-100 中,而在查询 2、查询 4、查询 5、查询 10 的任务 2 中,再次检索也没有得到结果,是因为本数据集中确实不包含满足任务的结果.

用户还可以使用启发式检索来获取更多满足条件的结果,首先指定 Top-k 的 k 值,然后对感兴趣的类别再次检索,即可获得 k 个属于该类别的结果,若返回的结果数小于 k,则说明数据库中该类别的结果都已经返回.采用这种浏览-再检索的模式,用户首次检索时就不需要在 k 值的选择上动脑筋了.

4 相关工作

在 KSORD 的研究中,有许多方法用于展现结果.BANKS[3]使用嵌套表格显示结果.文献[15]对结果格式作了改进,提高了结果可读性.DbSurfer[16]使用类似于 Windows 文件夹浏览树的形式显示找到的浏览路径(trail).DBXplorer[8]显示结果的生成过程,并以表格的形式显示结果元组.然而,这些工作都没有解决大量结果的组织问题.

TreeCluster 方法[14,17]是 KSORD 结果聚类方面的第一份工作,它是一种事后聚类方法.将系统产生的 Top-k 结果首先按照结构聚类,对于较大的类再按照内容聚类,把结果组织成两级类别,并图形化地把结果和类别描述并展示出来.S-CBR 是一种分类和聚类相结合的方法,它利用数据库模式图事先产生第一级类别,将结果分类,然后采用与 TreeCluster 方法类似的内容聚类把比较大的类做进一步划分.与 TreeCluster 相比,S-CBR 优点在于能够为用户提供所有可能的结果模式,用户能够根据 S-CBR 提供的类别信息(包括空类别),对感兴趣的类别做进一步的检索.从性能上比较,S-CBR 要预先产生第一级类别,在 MaxSize 较大时对系统的效率会有比较明显的影响,而 TreeCluster 仅仅对产生的结果作事后聚类,对系统效率的影响较小.

Hunter[12]是一种带有预处理的模式图系统,在预处理过程中,系统生成不同的候选元组集连接树模式,在查询过程中,用户要选择某一种模式,然后系统去检索与这一模式相匹配的结果.这实际上提供了一种对结果的分类方法.文献[15]也提到了类似的思想,但是没有从分类的角度去考虑,也没有公布方法的实现机制.Hunter 和文献[15]的方法都只能在 KSORD 系统的检索引擎内部实现,与系统紧密耦合在一起,其主要目的在于支持查询的执行,而不是结果的展现和理解.因此,所提供的分类方法还很初步,没有图形化的界面,也不支持用户的进一步检索. S-CBR 方法的侧重点在于结果的展现和进一步检索,提供了两级分类,并且在 KSORD 系统的外部实现,适用于返回结果为元组连接树的所有 KSORD 系统.Hunter 中的候选元组集连接树模式和 S-CBR 的结果树模式非常类似,因而,S-CBR 产生结果树模式的方法与 Hunter 中的类似,都采用从模式图扩展的方法,但具体的实现算法不同.

S-CBR 中首先要解决类别的产生问题.在 Web 上,Google[18]、新浪[19]等网站都提供了网页目录,这些目录主要是根据主题的不同来设置的,从内容上对网页作了分类.而对于单库上的 KSORD 来讲,检索结果源于一个专业数据库,直接按内容分类得不到很有意义的结果.但是,KSORD 系统能够获取 RDBMS 的模式信息,这是 Web 检索所不具备的条件.同时,KSORD 结果的结构特征反映了结果本身的语义,所以,S-CBR 利用数据库模式信息,按照检索结果的结构来产生分类目录,并根据与关键词查询的相关性对产生的类别作一定的裁剪.

传统的分类方法包括判定树归纳分类、贝叶斯分类、后向传播分类等[20].针对 KSORD 结果特征,S-CBR 没有使用这些方法,而是把结果树分类问题转化为标号树(结果树模式)的同构判断问题.这与 TreeCluster[14,17] 中把结果树聚类转化为标号树同构判断的方法相似,不同之处在于,TreeCluster 的类别中还包含关键词信息.

信息检索通常是一个反复尝试的过程,系统应该能够根据用户的反馈生成更符合用户要求的查询结果[21]. KSORD 系统也需要考虑如何展现查询结果,以便获得用户的反馈.S-CBR 在这方面做了开创性工作.它给用户提供所有可能的结果类别作为启发信息,根据用户的反馈作再次检索,给用户提供所需的结果.

5 结束语

本文介绍了 KSORD 系统的一种新的结果展现方法 S-CBR.其创新之处在于:

(1) 提出一种按结构分类和按内容聚类相结合的 KSORD 结果组织方法,将检索结果组织成两级类别,适用于返回结果为元组连接树的所有 KSORD 系统,提高用户浏览结果的效率.

(2) 给用户提供启发信息,使得用户能够观察所有可能结果的类别信息(包括空类别),支持用户对感兴趣的类别作进一步检索,以帮助用户尽快找到检索结果.这种启发式检索对于用户提交查询时不知道如何确定 Top-k 的 k 值这种情况非常有效.

(3) 给出了结果类别的排序规则和打分公式.

在今后的工作中,我们将根据用户使用 S-CBR 的情况,对各个数据库确定用户最常使用的结果树模式,在查询开始之前推荐给用户,以便用户可直接选择检索感兴趣的模式.

References:

[1] Wang S, Zhang KL. Searching databases with keywords. Journal of Computer Science and Technology, 2005,20(1):55–62.

[2] Hulgeri A, Bhalotia G, Nakhe C, Chakrabarti S, Sudarshan S. Keyword search in databases. IEEE Data Engineering Bulletin, 2001, 24(3):22–31.

[3] Bhalotia G, Hulgeri A, Nakhe C, Chakrabarti S, Sudarshan S. Keyword searching and browsing in databases using BANKS. In: Agrawal R, Dittrich K, Ngu AH, eds. Proc. of the 18th Int'l Conf. on Data Engineering. Dallas: IEEE Computer Society, 2002. 431–440.

[4] Kacholia V, Pandit S, Chakrabarti S, Sudarshan S, Desai R, Karambelkar H. Bidirectional expansion for keyword search on graph databases. In: Böhm K, Jensen CS, Haas LM, Kersten ML, Larson P, Ooi BC, eds. Proc. of the 31st Int'l Conf. on Very Large Data Bases. New York: ACM, 2005. 505–516.

[5] Balmin A, Hristidis V, Papakonstantinou Y. ObjectRank: Authority-Based keyword search in databases. In: Nascimento MA, Özsu MT, Kossmann D, Miller RJ, Blakeley JA, Schiefer KB eds. Proc. of the 30th Int'l Conf. on Very Large Data Bases. San Fransisco: Morgan Kaufmann Publishers, 2004. 564–575.

[6] Ding BL, Yu JX, Wang S, Qin L, Zhang X, Lin XM. Finding top-k min-cost connected trees in databases. In: Proc. of the 23rd Int'l Conf. on Data Engineering. Dallas: IEEE Computer Society, 2007. 836–845.

[7] Cai HY, Yao JL, Wang S. DETECTOR: A universal on-line keyword search system over relational database. Journal of Computer Research and Development, 2007,44(1):119–125 (in Chinese with English abstract).

[8] Agrawal S, Chaudhuri S, Das G. DBXplorer: A system for keyword-based search over relational databases. In: Agrawal R, Dittrich K, Ngu AH, eds. Proc. of the 18th Int'l Conf. on Data Engineering. Dallas: IEEE Computer Society, 2002. 5–16.

[9] Hristidis V, Papakonstantinou Y. DISCOVER: Keyword search in relational databases. In: Bernstein PA, Ioannidis Y, Ramakrishnan R, eds. Proc. of the 28th Int'l Conf. on Very Large Data Bases. San Fransisco: Morgan Kaufmann Publishers, 2002. 670–681.

[10] Hristidis V, Gravano L, Papakonstantinou Y. Efficient IR-style keyword search over relational databases. In: Freytag JC, Lockemann PC, Abiteboul S, Carey MJ, Selinger PG, Heuer A, eds. Proc. of the 29th Int'l Conf. on Very Large Data Bases. San Fransisco: Morgan Kaufmann Publishers, 2003. 850–861.

[11] Wen JJ, Wang S. SEEKER: Keyword-Based information retrieval over relational databases. Journal of Software, 2005,16(7): 1270−1281 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/1270.htm

[12] Zhang KL. Research on new preprocessing technology for keyword search in databases [Ph.D. Thesis]. Beijing: Renmin University of China, 2005 (in Chinese with English abstract).

[13] DBLP bibliography. 2004. http://www.informatik.uni-trier.de/~ley/db/index.html

[14] Peng ZH, Zhang J, Wang S, Qin L. TreeCluster: Clustering results of keyword search over databases. In: Yu JX, Kitsuregawa M, Leong HV, eds. Proc. of the 7th Int'l Conf. on Web-Age Information Management. LNCS 4016, Berlin: Springer-Verlag, 2006. 385−396.

[15] Aditya B, Chakrabarti S, Desai R, Hulgeri A, Karambelkar H, Nasre R, Parag, Sudarshan S. User interaction in the BANKS system: A demostration. In: Dayal U, Ramamritham K, Vijayaraman TM, eds. Proc. of the 19th Int'l Conf. on Data Engineering. Dallas: IEEE Computer Society, 2003. 786−788.

[16] Wheeldon R, Levene M, Keenoy K. DbSurfer: A search and navigation took for relational databases. In: Williams MH, MacKinnon LM, eds. Proc. of the 21st Annual British National Conf. on Databases. LNCS 3112, Berlin: Springer-Verlag, 2004. 144−149.

[17] Wang S, Peng ZH, Zhang J, Qin L, Wang S, Yu JX, Ding BL. NUITS: A novel user interface for efficient keyword search over databases. In: Dayal U, Whang KY, Lomet DB, Alonso G., Lohman GM, Kersten ML, Cha SK, Kim YK, eds. Proc. of the 32nd Int'l Conf. on Very Large Data Bases. New York: ACM, 2006. 1143−1146.

[18] Google Web directory. http://www.google.cn/dirhp?hl=zh-CN

[19] Sina Web directory. http://dir.iask.com/

[20] Han JW, Kamber M. Data Mining: Concepts and Techniques. New York: Academic Press, 2000.

[21] Baeza-Yates R, Ribeiro-Neto B. Modern Information Retrieval. New York: ACM Press, 1999.

附中文参考文献:

[7] 蔡宏艳,姚佳丽,王珊.DETECTOR:基于关系数据库通用的在线关键词查询系统.计算机研究与发展,2007,44(1):119−125.

[11] 文继军,王珊.SEEKER:基于关键词的关系数据库信息检索.软件学报,2005,16(7):1270−1281. http://www.jos.org.cn/1000-9825/16/1270.htm

[12] 张坤龙.数据库关键词搜索的预处理新技术研究[博士学位论文].北京:中国人民大学,2005.

彭朝晖(1978－),男,山东临清人,博士,讲师,主要研究领域为数据库信息检索.

王珊(1944－),女,教授,博士生导师,CCF高级会员,主要研究领域为高性能数据库新技术,数据库信息检索,数据仓库与 BI 技术.

张俊(1971－),男,博士,副教授,主要研究领域为数据库信息检索.