层次数据上关键字检索的结果聚合

胡 昊 何震瀛
(复旦大学计算机科学技术学院 上海 200433)

摘 要 由于使用方便等优点,数据库上的关键字检索技术使用户可以得到所需信息而不必书写复杂的 SQL 语句.但大部分现有的检索方法都关注通过连接操作得到包含所有关键字的元组连接树,忽略了对于检索结果的信息整合,这从某种程度上影响了用户对于检索结果的判断.文中提出并实现一种改进的关键字检索系统框架,在具有层次结构的属性指导下对得到的元组连接树结果做聚合操作,通过寻找最低层次最小覆盖聚合将关系更为紧密的元组作为更加相关的检索结果反馈给用户.文中还提出了基本的聚合算法并对其做改进从而减少了系统的响应时间.同时,为了改善用户体验,文中定义并给出了检索结果的摘要问题及其算法,使用户最大程度地了解检索结果.实验数据表明,文中的方法能够以较高的效率和较低的计算代价有效地完成检索结果的聚合和摘要.

关键词 关键字检索;聚合操作;层次结构;摘要算法
中图法分类号 TP311 **DOI 号**: 10.3724/SP.J.1016.2011.01986

Aggregate Keyword Queries on Hierarchy Relational Databases

HU Hao HE Zhen-Ying
(*School of Computer Science, Fudan University, Shanghai* 200433)

Abstract Keyword search (KWS) has been well accepted as a proven, user-friendly way to retrieve information, and recently applied successfully on relational databases. Today, this technique allows users to find pieces of information without having to compose complicated SQL queries. However, almost all the existing approaches focus on finding joined tuples matching a set of keywords and return the results as joining networks of tuples. In order to feed back the user more relative information, this paper formulates an expanding version of existing system to answer aggregate keyword queries over hierarchical relational databases in which the value of a specific attribute is organized in hierarchical structure. This version retrieves information in the form of MaxLMC(Max-Lowest hierarchy Minimum Coverage aggregate, which consists of tuples more similar and closer to each other) under the conduct of the above hierarchical structure. A Naïve algorithm is proposed to obtain MaxLMC and its enhancement is designed to reduce the system's responding time. Meanwhile, recognized that the number of returned answer might be extremely large in practical, we defined and studied the problem of effective exploration of large sets of aggregating tuples: summarization, a technique which has been applied to help the user find diverse aggregating tuples, thus can be used to improve the user experience. An extensive empirical evaluation using both real data sets and synthetic data sets is reported to verify the effectiveness and the efficiency of our methods.

Keywords keyword search; aggregate; hierarchy; summarization

收稿日期:2011-07-18;最终修改稿收到日期:2011-08-19.本课题得到国家科技重大专项基金(2010ZX01042-003-004)、国家自然科学基金(60703093,61033010,61073001)、国家"八六三"高技术研究发展计划项目基金(2009AA062803)和上海市科委现代服务业专项基金(10dz1511000)资助.**胡 昊**,男,1988 年生,硕士研究生,主要研究方向为 XML 数据管理、关键字检索.E-mail: huhao@fudan.edu.cn.
何震瀛,男,1977 年生,博士,讲师,主要研究方向为 XML 数据管理、图数据管理.

1 引　言

作为从结构化和半结构化数据中提取所需信息的一种方法，关键字检索获得广泛关注[1-6]. 近年来，数据库上的关键字检索方法大量涌现[1,4-5,7-8]. 为获得包含所有关键字的元组或元组连接树，现有方法往往导致冗余信息或无意义结果的产生. 另一方面，这些方法认为同一关系中的元组间相互独立，事实上，这些元组往往具有层次关系. 如何利用元组间的层次关系进行信息整合，为用户提供更好的体验，是研究者需解决的一个主要问题.

1.1 研究动机

例 1. 图 1 给出了某汽车销售的数据库，图 1(e)是该数据库的模式，图 1(a)～(d)是数据库中的实例片段. 图 1(e)中的有向箭头表示在模式之间存在有主键指向外键的关系. 当用户试图检索日系车中配备涡轮增压发动机的产品时，他可能提交的关键字组合为“涡轮增压”和“日本”. 表 1 展示了按现有技术的检索结果.

ClassKey	ClassName
…	…
02	零部件
0201	发动机附件系统
0201001	油泵
0201002	水泵
…	…
03	汽车
0301	乘用车
0302	商用车
…	…

(a) CLASS表

NationKey	NationName
01	中国
02	日本
03	美国
04	美国

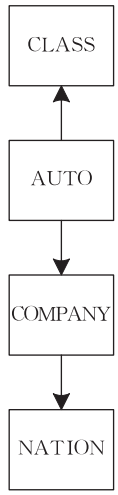
(d) NATION表

CompanyKey	NationKey	CompanyName	Partner	Establish	…
C001	01	一汽丰田	丰田汽车(日本)	2003	
C002	01	广汽丰田	丰田汽车(日本)	2004	
C003	01	一汽大众	大众汽车(德国)	1991	

(b) COMPANY表

AutoKey	ClassKey	CompanyKey	PriceInterval	Feature	…
A001	0301	C001	15～20	…,手自一体,涡轮增压…	
A002	0301	C002	10～15	…,手自一体,涡轮增压…	
A003	0301	C003	10～15	…,手自一体,涡轮增压…	
A004	0301	C001	20～25	…,手自一体,涡轮增压…	
A005	0301	C001	10～15	…,手自一体,涡轮增压…	
A006	0301	C002	10～15	…,手自一体,涡轮增压…	
A007	0302	C002	15～20	…,手自一体,涡轮增压…	
A008	0302	C002	25～30	…,手自一体,涡轮增压…	

(c) AUTO表



(e) 模式

图 1　一个层次关系数据库模式及其实例

表 1　一个可能的检索结果

	AutoKey	ClassKey	CompanyKey	PriceInterval	Feature	CompanyName	Partner	…
1	A001	0301	C001	15～20	…,手自一体,涡轮增压	一汽丰田	丰田汽车(日本)	
2	A002	0301	C002	10～15	…,手自一体,涡轮增压	广汽丰田	丰田汽车(日本)	
4	A004	0301	C001	10～15	…,手自一体,涡轮增压	一汽丰田	丰田汽车(日本)	
5	A005	0301	C001	10～15	手自一体,涡轮增压…	一汽丰田	丰田汽车(日本)	
6	A006	0301	C002	10～15	…,手自一体,涡轮增压	广汽丰田	丰田汽车(日本)	
7	A007	0302	C001	15～20	手自一体,涡轮增压…	一汽丰田	丰田汽车(日本)	
8	A008	0302	C002	25～30	…,手自一体,涡轮增压	广汽丰田	丰田汽车(日本)	

表 1 的结果只是简单的将符合条件的元组陈列给用户，这需要用户花费一定的时间来分析结果. 为使检索过程更加有效，将具有共同特征的元组聚合可帮助用户更加容易地熟悉返回结果. 图 2 是按照具体的每个公司聚合后的结果，圆圈中数字表示表 1 中元组的序号，每个圆圈代表一条元组. 同时，在每个公司内部，我们也按照层次关系将乘用车(0301)和商用车(0302)区分开来. 这样的结果由于特征鲜明更易于被用户接受.

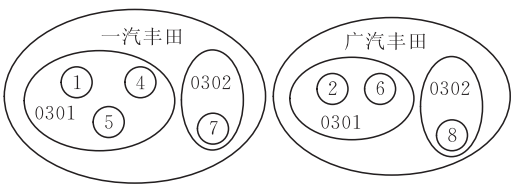


图 2　按照汽车公司分类聚合的结果

当然，用户也有可能比较关心汽车产品的价位，而对于汽车的品牌没有那么敏感. 因此，我们的聚合操作也应当能够在价格区间属性上展开.

本文中，为使用户获得更好的检索体验，我们利用层次关系数据对检索结果进行信息整合，即聚合

和摘要.在考虑这些问题时,我们将遵循以下原则:

(1)聚合粒度适当.检索结果集的聚合粒度的过小或者过大,都会影响用户分析结果.

(2)摘要集.为使用户能够在短时间内比较全面地了解聚合结果的特性,我们应有针对性地给出具有代表性的 k 个聚合结果即摘要集.

(3)良好的扩展特性.目前已有工作致力于关系数据库上关键字检索问题,为与这些工作结合,我们的改进应尽可能少地改变现有系统.

1.2 相关工作

近十年来,互联网搜索引擎的巨大成功使关系数据库上的关键字检索问题引起越来越多学者的关注,并已在业界许多成熟的商业数据库系统上有简单实现(如 Microsoft SQL Server、Oracle 和 IBM DB2 等).以往的研究更多地关注如何得到连接元组来获得同时包含所有关键字的元组连接树[1,4-5,7,9-16],并基于此提出了 DBXplorer[1]、DISCOVER[4]、BANKS[7]、SPARK[5]、SQAK[8]等具体的实现方法.

DISCOVER 系统是此领域实现较早也较为成功的检索系统,后续在此基础上的 SPARK、SQAK 等都是对 DISCOVER 的具有某方面特性的改进工作.为了使本文的方法能更好地与其它关键字检索的方法整合在一个系统中,我们选取 DISCOVER 作为扩展的基础.该系统利用数据库模式图,在模式图的基础上枚举可能包含查询结果的所有候选网络,然后依据查询表达式生成 SQL 语句投入数据库中进行处理并取回结果.DISCOVER 系统通过枚举所有连接表达式避免了在构造表达式时有可能出现的关键字与多个关系上多个属性值匹配的情况而导致的二义性.

对于检索结果的进一步处理等工作包括打分方法和结果聚合也都在文献[3,17-18]中有所研究.Zhou 等人在文献[17]中提出关系数据库检索结果的聚合问题,但该研究仅关注于一张表上的聚合操作.然而,实际应用中,信息会依据范式被分解成不同的部分存储于多张表中.若简单的将该方法应用于多张表的问题中会根据连接关系生成一张非常大的表,这是耗时且不实际的.另外,该问题中的属性上并无层次关系.SQAK 的工作也对类似的聚集操作予以支持[8],但需要由用户指定在哪些属性上进行聚集操作,这对于不熟悉数据库模式的用户来讲是十分困难的.而且,SQAK 中的聚集操作是对结果集进行聚集选择(即计算元组数目或求最大最小值等),并非在此基础上做信息整合.Buranasaksee 等人在文献[18]中也提出类似的问题,他们的解决方案是直接使用 DISCOVER[4] 系统得到结果以后再做聚合,并没有考虑依据关系上的其它特性(例如:层次特性)对结果进行更好的区分.因此,得到的结果有可能使关系不是很紧密的元组聚合在一起;此外,该方案也没有考虑到检索结果集的摘要问题.

Roy 等人在文献[19]一文中也提到了类似的 package 的摘要问题,并给出了基于贪心和随机算法的两种解决方案.这与我们的问题类似,但是由于我们的检索结果集合在共有属性上并无交集,因此我们设计了针对该问题的剪枝策略.

1.3 本文贡献和组织

本文的主要贡献如下:

(1)在具有层次关系数据的背景下提出检索结果的最小覆盖聚合和最低上层最小覆盖聚合概念,以此概念帮助用户更好地了解检索结果.我们还在此基础上提出了求解最低层次聚合的基本算法和增强算法.

(2)引入针对最低上层最小覆盖聚合族的摘要问题并提出了求解的贪心算法及具有剪枝的优化算法.

(3)基于 DISCOVER 提出具有反馈机制的检索系统结构.

本文第 2 节介绍系统概要、数据模型和问题定义;第 3 节给出求解 MaxLMC(S,Q)的 Group by 方法和按序连接方法;第 4 节介绍摘要问题并在 Baseline 方法上给出 3 个剪枝条件;第 5 节介绍实验;第 6 节总结展望本文工作.

2 概要与数据模型

2.1 系统概要

我们的系统在 DISCOVER 的基础上进行扩展,系统的结构示意如图 3 所示.

该系统由 6 部分组成.其设计遵循良好扩展特性的原则,最大限度地减少现有系统结构的更改.聚合模块(Aggregator)部分主要负责检索结果的聚合;摘要模块(Summarization)部分则对聚合结果做摘要.此处需注意的是返回结果时并非只返回摘要,而是在返回摘要的基础上先使得用户了解检索结果的特性,而后再返回其它所有结果.

检索时,首先由用户将用于检索的关键字集合 $\{k_1,k_2,\cdots,k_m\}$ 输入系统.依据数据库表项中的倒排索引,对于数据库中的每一张表,我们都可以得到包

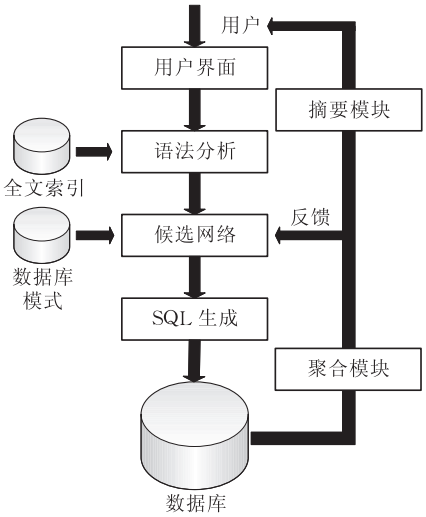


图 3　系统结构

含关键字 $k_1, k_2, \cdots, k_m$ 的元组集合 $R_i^{k_1}, R_i^{k_2}, \cdots, R_i^{k_m}$. 其中元组集合 $R_i^{k_j}$ 中表示关系 R_i 中属性上包含关键字 k_j 的所有元组. 然后，依据数据库模式图，在这些元组集合的基础上按照完整和无冗余的规则生成连接的候选网络(candidate networks).

候选网络经由 SQL Generator 生成对应的 SQL 语句并投入数据库管理系统进行检索.

我们的系统与 DISCOVER 不同的一点是我们不会一次将所有的候选网络都生成出来，而是按照候选网络的规模大小(例如，边的数目)分批生成. 这是由于先生成的结果可以对后续候选网络或 SQL 语句的生成是否执行起反馈和指导作用，过早的投入生成会造成计算资源的浪费.

对于生成的结果元组分别由 Aggregator 聚合和 Summarization 步骤摘要返还给用户.

本系统的提出是在 DISCOVER 的工作上进行的，因此与 DISCOVER 系统中相同的部分其模型和定义我们不再赘述，这些部分包括 Joining network of tuples、Keyword Query、MTJNT、Joining Network of Tuple Sets、Candidate Network 等定义，请参阅文献[4].

我们的工作主要在检索结果的聚合和摘要上，因此，下文分别给出两者定义.

2.2　聚合和覆盖

定义 1(划分).　设 S 是一个元组集合，$\pi \subseteq 2^S$. 如果下列条件成立则称 π 为 S 的一个划分：① $S = \bigcup_{P \in \pi} P$，②对于任意 $P_i, P_j \in \pi$，若 $i \neq j$，则 $P_i \cap P_j = \varnothing$.

定义 2(聚合关系).　设 $\alpha = \{S_1, S_2, \cdots, S_n\}$ 是一个有限元组集族，$S_i (1 \leq i \leq n)$ 是任意元组集合，$\leqq$ 是如下定义的二元关系：对于 α 中任意集合 S_i 和 S_j，如果存在一个函数 $F: \pi \rightarrow S_j$（π 是 S_i 的划分），则称 S_i 和 S_j 满足 $\leqq$，记作 $S_i \leqq S_j$，F 称为聚合函数. 如果满足下列条件则称为 α 上的聚合关系：对于 α 中任意集合 S_i 和 S_j，如果 $S_i \leqq S_j$，$S_j \leqq S_i$，则 $S_i = S_j$.

引理 1.　集族 α 上的聚合关系是一个偏序.

证明.　从略.

例如，$\alpha = \{S_1, S_2\}$，其中 S_1 为表 1 中 7 条元组的集合，S_2 为图 1 中 AUTO 表和 COMPANY 表做自然连接后的所有元组集合，我们知道，S_1 中的元组都能够在 S_2 中找到与之对应的(AutoKey 相同)但是属性比它本身多的元组. 现在将 S_1 中的元组按照公司(CompanyKey)划分，即所有相同的公司共同属于某一划分单元(注意：不同公司之间的元组也可能属于相同划分单元)，记此划分为 π. 定义其上的聚合函数 F，我们令 $F: \pi \rightarrow S_2$ 当且仅当该划分下所有元组同属于一个国家(即，在 S_2 中 NationKey 是一致的). 此时 F 是“公司”关于“国家”的聚合函数，我们称 S_1 的划分 π 满足国家上的聚合关系.

显然，满足这样聚合关系的集合很多. 例如，从 S_1 中去掉 $n (n < |S_1|)$ 条元组得到的新的集合都满足 π 上关于国家的聚合关系，因此，我们关心的是如何得到完备的这样的集合，后文中借助于 Group by 的方法实现这一点. 另一方面，由于可以在不同的属性上构造聚合关系，因此在构造过程中应当注意全面性. 一般地，我们将构造聚合关系的过程称为聚合操作.

定义 3(划分的覆盖).　设 S 是一个元组集合，$\pi = \{S_1, S_2, \cdots, S_m\}$，$\pi \subseteq 2^S$ 是 S 上的一个划分. D 是 S 上的字典，$Q \subseteq D$ 是 S 中出现的某些词，谓词 *contain* 表示集合中包含词项. 则 $Cov(\pi, Q) = \{S_i \mid S_i \ contain \ Q\}$ 是划分 π 关于 Q 的覆盖.

定义 4(最小覆盖聚合，MC).　设 $\alpha = \{S_1, S_2, \cdots, S_n\}$ 是一个有限元组集族. $Q \subseteq D$ 是 α 中出现的某些词，给定 S_i 和 π，$\pi \subseteq 2^{S_i}$ 是 S_i 上的一个划分，若 $\forall S_j \in \alpha, S_j \leqq S_i$，不存在划分 $\pi' \subseteq 2^{S_j}$，使得 S_j 上有划分的覆盖 $Cov(\pi', Q)$，则 S_i 是关于 Q 的最小覆盖聚合(MC).

定义 5(层次关系).　设 H 是一个有限集合，$<$ 是自反的、反对称的和传递的偏序关系，且：对于 H 中的任意元素 h_i，若 h_i 非极小元，如果存在一个函数 $F: h_i \rightarrow H', H' \subseteq H$，且 F 值域中的各个 $H' \subseteq H$ 互不相交，则称 F 为分层函数，称 $<$ 是 H 上的层次关系.

例如，CLASS 表中的 CLASSKEY 集合上的包含关系就是一个层次关系. 层次关系是针对集合中

的元素而定义的，而聚合关系是针对集族中的集合而定义的. 一般地，若 $h_i \prec h_j$，我们称 h_j 是 h_i 的上层. 为了说明最低上层关系，我们定义运算 MH，该运算返回层次集合上子集 $H' \subseteq H$ 中最小的上层，即 H' 的上确界. 形式化表述为：MH(H')=$\{h_i$|存在 $h_k \in H, \forall h_j \in H', j \neq k$，有 $h_j \prec h_k$，并且对于所有这样的 h_k, h_i 是其中的极小元或者 H' 中的最大元$\}$. 由于 H 中极大元不唯一，实际操作中，我们引入 τ，并令 $\forall h \in H$，h 是极大元，有 $h \prec \tau$.

引理 2. 若 MH(H')存在，则必唯一.

定义 6(最低上层最小覆盖聚合，LMC). 设 S 为关于 Q 的 MC，MH(S)表示 S 中层次关系属性上的最低上层，若对于任意的 $S_2 \subset S$，S_2 中包含所有 Q，不存在 $S_1 \subset S$，且 $\forall w_i \in Q, S_1$ *contain* Q，使得 MH(S_1)$\prec$MH(S_2)，则称 S_2 为关于 Q 的最低上层最小覆盖聚合(LMC). 若同时也不存在 $s \in S$，且 $S_2 \cup \{s\}$ 也是 LMC，则称 S_2 为 MaxLMC，此时 S_2 中所含的元组数目在满足最低上层最小覆盖聚合的条件下达到最多.

显然，关于 Q 的最大最低上层最小覆盖聚合(MaxLMC)是对最小覆盖聚合的一个细分.

例如，图 2 中$\{1,4,5,7\}$是在“一汽丰田”公司上的 MC，$\{1,4\}$和$\{7\}$是基于层次关系的 LMC，而$\{1,4,5\}$则是一个 MaxLMC，可以看出，所有 MaxLMC 是对 MC 的细分.

不同层次关系的最低上层最小覆盖聚合更多地反映了检索结果在认知上的粒度大小，而聚合则反映了检索结果在某一自身属性上的同质.

问题 1(MaxLMC(S,Q)). 给定关系数据库模式 S_c 及其实例 D_{S_c}，Q 是用户提交的检索，S 是对应于检索结果的元组集合. 求解：MaxLMC(S,Q)，S 关于 Q 的最大最低上层最小覆盖聚合族.

2.3 摘 要

尽管我们致力于寻找 MaxLMC，而这已经将可能的问题结果 LMC 大大减少，但是实际情况中 MaxLMC(S,Q)的数目仍然有很多. 可以想象，日系车中含有涡轮增压的产品数目本身就有许多，为使用户了解检索结果的特性，我们可以根据公司属性聚合结果，也可以根据价位属性聚合结果，这中间重要的一点是许多汽车会在不同的聚合集中有重叠. 因此，在计算 MaxLMC(S,Q)的基础上，我们应当进一步对结果做摘要，即从中选出 k 个具有代表性的最小覆盖聚合(一般为 5～10 个)，这使得用户在较短时间内能较全面地了解检索结果的特性.

问题 2(摘要). 给定 MaxLMC(S,Q)和 k，如何计算 MaxLMC(S,Q)的具有代表性的 k 个最低上层最小覆盖聚合 k-MaxLMC(S,Q). 换言之，这 k 个最低上层最小覆盖聚合可以最大限度地表示之前求出的 MaxLMC(S,Q).

文中，我们称 k-MaxLMC(S,Q)为摘要集. 摘要的目的是展示给用户一个简短的具有 k 个最低上层最小覆盖聚合的结果，同时，在这 k 个结果中可以最大限度地将检索结果元组的集合涵盖进来.

3 最低上层最小覆盖聚合算法

本节讨论如何在检索结果中得到最低层次的最小覆盖聚合. 为了将 SQL 语句的这一特性应用进来，我们可以在系统中 SQL Generator 生成语句时添加相应的 Group by 语句. 为此，我们先定义属性划分.

定义 7(属性划分). 设 A 是给定关系数据库模式 S_c 上的属性集合. $A=A^C \cup A^P \cup A^U$ 是关于 A 的一个划分，其中 A^C 表示分类属性，A^P 表示具有偏序关系的聚合属性，A^U 是其补集. A^C 和 A^P 统称为聚合属性.

例如，AUTO 表中 PriceInterval 是分类属性，CompanyKey 和 COMPANY 表中的 NationKey 等外键是具有偏序关系的聚合属性，这是因为若某汽车属于某公司，其必属于相应的国家，为此，在聚合时应先选择偏序次序较为靠前的属性. 我们可以在这两类属性上做聚合操作. 而 Feature 和 COMPANY 表中的 Establish 属性则不是分类属性.

在生成 SQL 语句时，我们应选择所有可能用于聚合操作的属性并在其上做 Group by 操作. 但是这样会增加系统生成的 SQL 语句数目从而增大开销. 因此，我们仅在偏序关系的聚合属性上做 Group by 操作，而分类属性上的聚合操作我们将在具体的 SQL 语句得到结果后进行，此操作只需要简单地遍历一遍结果元组，并将不同的类别元组置于不同的最小覆盖中即可.

若 SQL 语句中涉及到两个或以上的聚合属性 $A_1^P, A_2^P, \cdots, A_k^P$，则根据 A^P 的偏序关系选择划分更为细致的属性 A_i^P，使得 $\forall A_i^P, j \neq i$，有 $A_i^P \leqq A_j^P$(这里使用$\leqq$来表示 A_i^P 上聚合的结果之间聚合关系的比较，下同). 当然，有可能在 A_i^P 上并无聚合结果，此时根据 Feedback 机制使 SQL Generator 重新生成 SQL 语句，并选择仅在 A_i^P 之上(即 A_{i+1}^P)的属性

聚合.

3.1　基于 Group by 的方法

为求得 MaxLMC(S,Q)，根据定义，最低上层最小覆盖聚合是对最小覆盖聚合的一个细分，因此我们需要首先计算出最小覆盖聚合.

定理 1.　基于 SQL Generator 生成的带有 Group by 操作的 SQL 语句的执行结果为最小覆盖聚合 MC.

这与选择的 Feedback 机制有关，证明从略.

我们关心的是 LMC. 假设用户在检索时需要含有特性“a”和“b”的产品，根据候选网络生成 SQL 语句得到的可能连接结果如表 2，这些结果都是在公司 C 上 Group by 得到的.

表 2　一部分可能的连接结果(中间一列为公司)

AutoKey	层次	特征		AutoKey	层次	特征
A001	0301	a	C	A002	0301	b
A001	0301	a	C	A003	0302	b
A004	0302	a	C	A002	0301	b
A004	0302	a	C	A003	0302	b
A005	0301	a	C	A002	0301	b

易知，表中的 5 条结果已经是最小覆盖聚合，但是若考虑到层次关系，则只有第 1，4，5 条元组满足最低上层最小覆盖聚合. 这是因为第 1 条元组的最低上层为 0301，而第 2 条元组的最低上层为 03，因此我们保留第 1 条元组. 同理，第 4，5 条元组得到保留.

算法 1 给出求解最大最低上层最小覆盖聚合的 Naïve 方法，其中 MH(t_i)表示 t_i 元组上层次关系的最小上层.

算法 1.　Naïve MaxLMC(S,Q).

输入：最小覆盖聚合 S

输出：MaxLMC(S,Q)，结果用 R 表示，$R=\{R_{S_1}, R_{S_2},\cdots,R_{S_m}\}$，其中 R_{S_i} 表示层次 S_i 上的聚合结果

```
1. R=empty
2. for each tuple t_i in S
3. {  Lowest=true
4.    for each tuple t_j with the same value on joining
      attribute in S
5.    {  if(MH(t_j)≺MH(t_i))
6.          Lowest=false
7.    }
8.    if (Lowest=true)
9.       R_MH(t_i) = R_MH(t_i) ∪ {t_i}
10. }
11. return R
```

算法 1 求解最大最低上层最小覆盖聚合，它的输出为 MaxLMC(S,Q)，每一个 MaxLMC 中包含了所有相同最低上层的聚合结果. 算法的流程是：对于每一条元组都遍历与之连接属性上值相同的其它元组，若其它元组的最低上层层次都不低于该元组，则将该元组加入其自身的最低上层层次的 MaxLMC. 上例中的结果应为(以序号表示元组)：{1,5}，{4}.

假定在最小覆盖聚合 S 上共有 n 条元组，连接属性共有 d 种同的值，因此平均在每个连接属性的值上有$\left\lceil\frac{n}{d}\right\rceil$条元组，对于每一个值我们的算法复杂度为$\left(\frac{n}{d}\right)^2$，因此总的复杂度为 $O(d\times(n/d)^2)=O(n^2/d)$.

3.2　按序连接方法

注意到，上例中连接的结果 2、3 并未被选用. 因此若可以设计一种连接操作时避免生成 2、3 元组的方法，则将大大减少时间开销. 由于结果是由 SQL 语句生成的，因此有两种方案可供实现改进算法：(1)完全弃用数据库管理系统提供的 SQL 检索方式，根据层次关系实现系列操作；(2)在 SQL 语句在做最后一步连接操作前，根据层次上的最低上层选择在做连接时剔除掉部分结果.

本文中，我们选取后者. 在 SQL 语句最后做连接之前，有两个待连接的元组集，称之为 T_1 和 T_2，其中 T_1 和 T_2 中的每条元组分别包含 Q 的一个子集 Q'，并且在连接时对于 $t_i\in T_1$，$t_j\in T_2$，则 $t_i\bowtie t_j$ 的结果必须含有所有的 Q，记为 $Q_{t_i}\cup Q_{t_j}=Q$. 并且，T_1 和 T_2 中的元组都具有层次关系的属性，若某 T_i 中没有层次属性，则可认为 T_i 中的层次为 H 上的极小元，使得 MH(t_j)=MH($t_i\bowtie t_j$)，这样不影响后面的计算.

我们的改进算法是基于层次关系的传递性质，即若某 $t_i\in T_1$，它的层次为 MH(t_i)，则 t_i 的连接结果(用 $Join(t_i)$表示)的最低上层 MH($Join(t_i)$)满足：MH(t_i)≺MH($Join(t_i)$). 算法开始于两个先按照连接属性排序后按照层次偏序排序(同一层次的排序按字典序)的元组集 T_1，T_2(这可以在 SQL 中使用 Group by 和 asc 实现). 选择层次种类较多的元组集(假设为 T_1，没有层次属性的待连接元组集层次种类为 0)，在顺序遍历 T_1 的过程中，对于 T_1 中的每个元组 t_i，选择连接属性上值相同的 $t_j\in T_2$，设为$\{t_{j_0},t_{j_1},\cdots,t_{j_m}\}$，顺序遍历这样的 t_j 即可，且从 t_{j_0} 开始，MH($t_i\bowtie t_j$)层次较低的连接元组会按序聚合在

一起，根据最低上层条件，若对于t_{j_k}，$MH(t_i \bowtie t_{j_0}) \prec MH(t_i \bowtie t_{j_k})$，则 t_{j_k} 以后的元组都不必考虑，因而 $t_i \bowtie t_{j_k}$，$t_i \bowtie t_{j_{k+1}}$，…，$t_i \bowtie t_{j_m}$ 这些连接操作不必要执行.

算法 2. MaxLMC(S,Q).

输入：待连接元组集 T_1，T_2

输出：MaxLMC(S,Q)，结果用 R 表示，$R=\{R_{S_1}, R_{S_2}, \cdots, R_{S_m}\}$，其中 R_{S_i} 表示层次 S_i 上的聚合结果

```
1. R=empty
2. for each tuple t_i in T_1
3. {   for each tuple t_j in T_2 with the same value on
       joining attribute
4.     {   if (MH(t_i ⋈ t_j0) ≺ MH(t_i ⋈ t_jk))
5.             break
6.         else
7.             t=Jion(t_i, t_j)
8.             R_MH(t_i ⋈ t_j0) = R_MH(t_i ⋈ t_j0) ∪ {t}
9.     }
10. }
11. return R
```

定理 2. 算法 2 产生与算法 1 相同的结果.

需要说明的是：算法 2 当其中有一 T_i 上不具有层次属性时退化为算法 1. 此时对于 T_j 中的任一元组，都需要在 T_i 上连接，因此不节省连接操作. 此外，对于没有做连接的元组，在返回结果时我们将直接返回，这保证了返回所有结果

按算法 2 生成的对于某一连接属性的连接元组聚合得到的一系列 MaxLMC，我们按照聚合元组数目的降序在其上维护一个优先队列，使得 MaxLMC(S,Q) 在每一属性上都是有序的，这样的顺序便于用户查看，同时也为我们的摘要算法做好数据准备.

4 摘要算法

直接将所有的 MaxLMC(S,Q) 返还给用户（如 2.4 节所述）具有两个缺点. 一是 MaxLMC(S,Q) 的数目非常多；二是各 MaxLMC 之间有很大比例上的重叠. 我们做摘要的目的是找到 k 个具有代表性的 MaxLMC，给用户尽可能全面地提供检索结果并使其可以进一步掌握检索结果.

生成摘要的常用方法是聚类算法：我们可以在 MaxLMC 上定义两两之间的距离函数，然后应用各类聚类算法（例如，k-means）. 在将 MaxLMC(S,Q) 分成 k 类后，从每一类中随机选取一个聚合集即可. 但是，在当前的问题中定义距离函数是十分困难的.

本文中，我们根据最大覆盖原理设计了一种不同的方法用以求解该问题，目的是找到尽可能完整的覆盖检索结果集的 k 个代表 MaxLMC. 直观地看，这 k 个 MaxLMC 的并集中所含的元组数目最多，这能够更好地展示给用户检索结果的共有特性. 而且，从 LMC 的角度来看，用户可以从摘要集中构造出更多的 LMC.

例如，在图 4 中，实线和虚线框分别代表不同的分类属性上的聚合结果. 若 $k=2$，我们的摘要集应为 S_1，S_4，因为 $S_1 \cup S_4$ 中所涵盖的元素数目最多.

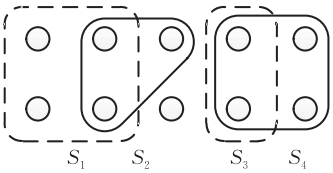


图 4 两类属性上的聚合结果

我们的问题是给定 MaxLMC(S,Q)，选择其中的 k 个集合使得这 k 个集合包含的元素最多. 最简单的解法是：对所有可能的 k 个集合的并集分别计算元素个数，并取使并集中元素个数最多的 k 个集合. 这具有指数的时间复杂度. 为解决这一问题，我们给出一个基本的贪心算法.

4.1 基本(Baseline)方法

算法 3 伪码如下，基本思想是：从最大的集合（元素数目最多）开始，记为 I，在每一轮迭代中都选择 S_i 使 $I \cup S_i$ 最大. 算法在 k 轮迭代后停止.

算法 3. Greedy Summary.

输入：MaxLMC(S,Q)$=\{S_1, S_2, \cdots, S_n\}$

输出：覆盖 I

```
1. I=empty
2. Let S_max be the largest set in MaxLMC(S,Q)
3. remove S_max from MaxLMC(S,Q)
4. I=I∪S_max
5. iteration=0
6. While iteration<k do
7.     S_max=Max_{S'∈MaxLMC(S,Q)}{I∪S'}
8.     remove S_max from MaxLMC(S,Q)
9.     I=I∪S_max
10.    iteration++
11. return I
```

算法 3 是 Maximum k-Set Cover 问题的经典贪心解法，Maximum k-Set Cover 问题是集合上最小覆盖问题[13]的最优化问题，是一个经典的 NP 完全问题. 上述贪心解法不能完全正确地求解 Maximum k-Set Cover 问题，但是能给出(1−1/e)的近似率[20].

引理 3.　给定 k-MaxLMC(S,Q)问题，令 I^{opt} 表示其最优解，I^{greedy} 表示 Baseline 贪心法的解，$C(I)$ 表示 I 中涵盖的元组数目. 则 $C(I^{greedy})/C(I^{opt})\geq(1-1/e)$. e 为自然对数的底.

4.2　改进(Improved)方法

观察到我们的问题中在某属性上的聚合结果是互不相交的，即，MaxLMC(S,Q)中的结果按照聚合属性的不同(假设其中涉及到 m 个聚合属性)可分为 m 类，其中每类上的最大最低上层最小覆盖聚合集 MaxLMC 之间互不相交. 例如，在图 4 中，若 S_1 和 S_3 是按照价格区间属性上求 MaxLMC 的结果，则它们是不相交的，这是因为对于某一汽车，其价格区间唯一，因此其属于且只属于一类中.

由于 m 类中每一类上的 MaxLMC 是对 MC 的细分，因此这 m 类中的集合之间互不相交. 形式化的表述为：MaxLMC(S,Q)按照聚合属性归类为 $\{M^1,M^2,\cdots,M^m\}$，其中每一类 $M^i=\{M_1^i,M_2^i,\cdots,M_{m_i}^i\}$ 中的聚合覆盖互不相交，且前文已述，M^i 中的各个 MaxLMC(以下简称聚合集)是按照规模降序排列.

例 2.　如图 5 所示，MaxLMC(S,Q)中有 $m=2$ 时两类聚合集族 a,b，数字表示 MaxLMC 中元组的数目，每一族上的各个聚合集互不相交. 我们先给出算法 3 的 3 个剪枝条件，然后分别对照此例阐述.

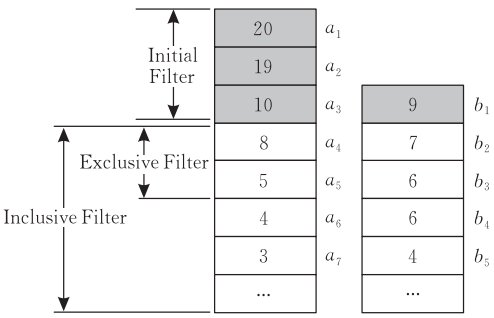


图 5　$m=2$ 时 MaxLMC(S,Q)可能的情况，数字代表每个 MaxLMC 中元组的数目

(1) Initial Filter. 若初始最大 MaxLMC 来自 M_1^i，则若 $|M_2^i|>|M_1^k|$ 对于 $k=1,2,\cdots,m,k\neq i$ 成立，M_2^i 应是下一个选择的 MaxLMC，而且可以继续选择直到出现 $k\neq i$ 使得 $|M_j^i|<|M_1^k|$ 为止.

例如，图 5 中，我们从元组最多的聚合集 a_1 开始选取，当选取第 2 个集合时，由于 a_2 中元组为 19 条，比 b_1 要多，因此不必考虑其它聚合集，将 a_2 作为结果输出即可. 同样的，a_3 也因此被输出.

(2) Inclusive Filter. 若当前摘要集为 I，并处于第 r 次迭代当中. 在考察 M^i 的时候，从 M_1^i 开始，若 $|M_1^i\cap I|=t$，$|I-\bigcup_{j=1}^{m_i}M_j^i|=x$，则在 M^i 中，我们只需要按序考察 $x-t+1$ 个 MaxLMC.

这是基于这样的原理：当按序考察集族 a 的聚合集时，若某 a_i 与 I 不相交，它对 $|I\cup a_i|$ 的贡献为 $|a_i|$，则 a_{i+1} 以后的集合均不必考虑，因为 $|a_{i+1}|\leq|a_i|$，其以后的对摘要集的贡献不会多于当前 a_i.

例如，图 5 中，假设灰色标记的 MaxLMC 的并集是已经选择的摘要集 I，当前为第 5 次迭代. 对于 a_4 来说，假设 8 条元组中有 3 条在 I 中出现(此例中这 3 条元组来自 b_1)，且 I 中共有 9 条元组来自集族 b(此例中均来自 b_1)，则在集族 a 中剩余的集合最多还可能与 I 相交的数目为 $9-3=6$，因此，在按序的 $6+1$ 个 MaxLMC 内必将出现一个不与 I 相交的集合(抽屉原理)，此后的聚合集均不必考虑，因为依据之前分析，一旦找到一条不与当前摘要集 I 相交的聚合集，以后集合均不必考虑.

(3) Exclusive Filter. 若当前摘要集为 I，并处于第 r 次迭代当中. 在考察 M^i 的时候，从 M_1^i 开始，若 $|M_1^i\cap I|=t$，则在 M^i 中，我们只需要考察到规模为 $|M_1^i|-t$ 的 MaxLMC.

例如，图 5 中，当前为第 5 次迭代，同上例，对于 a_4，有 3 条元组在 b_1 中，则 a_4 的贡献为 $8-3=5$，那么我们仅需考察到规模为 5 的 a_5 即可，因为 a_5 以后的 MaxLMC 规模都小于 5，贡献不可能比 a_4 大.

我们的改进的 Summary 方法即在基本的方法基础上应用上述 3 种剪枝策略得到. 具体伪码略.

5　实　验

为了验证文中所提出的关于聚合问题和摘要问题的改进方法，我们分别在真实数据和模拟数据上做实验. 真实数据来自电子工业部某研究所的保密数据. 为了验证本文方法，我们依照图 1(e)中给出的模式生成了模拟数据. 如图 1 所示，该数据库模式上共有 4 个分类属性和 3 个具有偏序关系的聚合属性. 生成数据时，我们指定了各分类属性和聚合属性上值相同的元组最多为 100 条，这也就是说，聚合得到的 MaxLMC 的最大规模不会超过 100. 在其它属性上我们均是从预先定义的字典中随机赋值.

另外，CLASS 表的层次最大深度为 5，平均深度为 3.12；其余三张表的元组数目如表 3.

表 3 元组数目表

AUTO	COMPANY	NATION
100000	1000	100

5.1 数据准备

对于真实数据和模拟数据，首先我们在其上做关键字的倒排索引，按照 DISCOVER 系统的方案生成在分类属性或聚合属性上带 Group by 的 SQL 语句. 然后将这些 SQL 语句投入数据库中.

本实验是在 Visual Studio 2010 平台上开发，用 C++ 编写，数据库使用 MySQL5.0 版本. 运行在 PC 上，Intel CPU，主频 2.8GHz，内存 8GB.

5.2 聚合结果

真实数据和模拟数据上我们都选取了 10 组用于检索的关键字. 由于聚合操作的两个算法之间涉及到不同的数据库操作，直接比较时间代价是不可行的. 因此我们在此比较按序连接方法的节省比，即按序连接方法的连接次数占基于 Group by 方法的连接次数的比例. 另外，由于有多个聚合属性，我们对每个检索都分别在每个聚合属性上做聚合，然后用总的节省连接次数除以总的连接次数做平均节省比.

由于在按序连接方法中我们不需要数据库管理系统来做连接，而是需要待连接元组集(见 3.2 小节)，因此在本实验中，我们将 DISCOVER 由候选网络翻译成 SQL 语句时从候选网络的连接操作处断开拆分成为个候选网络，然后分别生成 SQL 语句投入数据库中得到待连接元组集. 对于涉及到两次连接以上的候选网络，我们将指定最晚连接属性并从该连接属性处断开. 具体实现时，由于模式图已知，我们事先指定某些外键属性为最晚连接属性. 例如，对于本次实验的模拟数据，我们选择自然连接时的公共属性做为最晚连接属性(即例中的 ClassKey、CompanyKey 和 NationKey 属性). 需要注意的是，若某候选网络中同时具有两个以上的连接操作，则我们按序从首先出现最晚连接属性的那个连接操作处断开.

图 6 和图 7 分别是真实数据和模拟数据上的节省比率. 图中，横坐标是 10 次不同的查询，纵坐标则为节省比. 在图 6 中，Q_1 和 Q_6 的节省为 0，这是因为 Q_1 和 Q_6 的关键字分别来自两张不同的表，其中有一张表上没有层次关系，此时算法 2 退化为算法 1，并没有节省连接操作. 这在实际情况中时有发生，与具体的数据和检索都有关系. 而在模拟数据中，算法退化并不明显，这是由于我们随机赋值，因此包含不同的关键字的元组基本上都可以来自具有层次属性的关系，故而按序连接方法在此处没有完全退化.

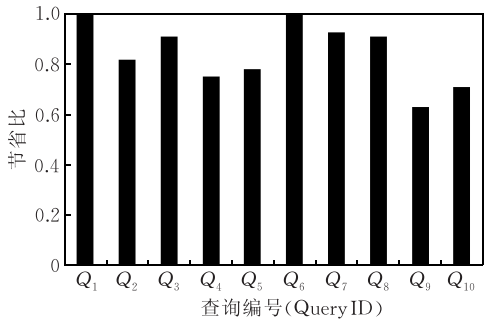


图 6 真实数据上的连接节省比

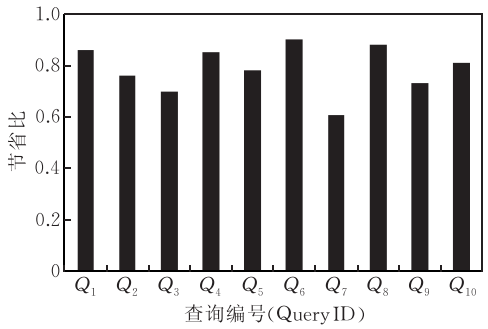


图 7 模拟数据上的连接节省比

5.3 摘 要

我们考察当 k 不同时及 MaxLMC 中分类 M 不同时 Improved 方法的效率差异.

真实数据上返回结果的平均元组数目为 1743 条，针对每个检索，我们都分别选定聚合属性为 2 个和 3 个时做摘要算法的比较，平均每次聚合生成 223 个($m=2$ 时)和 421 个($m=3$ 时)MaxLMC.

模拟数据上返回结果的平均元组数目为 5201 条，同样的，我们也在 2 个和 3 个聚合属性上做聚合，平均每次聚合生成 2210 个($m=2$ 时)和 3102 个($m=3$ 时)MaxLMC，这与真实数据差异较大，主要因为模拟数据的关键字分布比较随机，并非真实数据上呈明显的分布区分.

在做摘要时，贪心法随着 k 的增长时间代价增长很快，图 8、图 9 显示了这种情况，上方的曲线为基本的贪心方法随着 k 值从 5～25 变化所需的时间花费，其中纵坐标取时间的对数. 这是因为在 k 增长时，每一轮迭代都要遍历所有的元组集合用以判定最大覆盖.

图 8 考察了对 2 类属性上 MaxLMC 摘要的结果. 上方曲线为 Baseline 方法随着摘要集规模 k 的增长摘要时间的增长，下方曲线为具有剪枝的 Improved方法时间花费. 从数据上可以看出，剪枝以后的时间效率大幅提升. 同时，无论在真实数据和模拟数据上，我们的剪枝算法都能保证时间效率. 但是随着 k 的增长，剪枝算法时间代价仍近似指数增

长.考虑到实际应用中 k 值选取一般不会很大,因此我们的剪枝算法可以保证摘要的实时性.

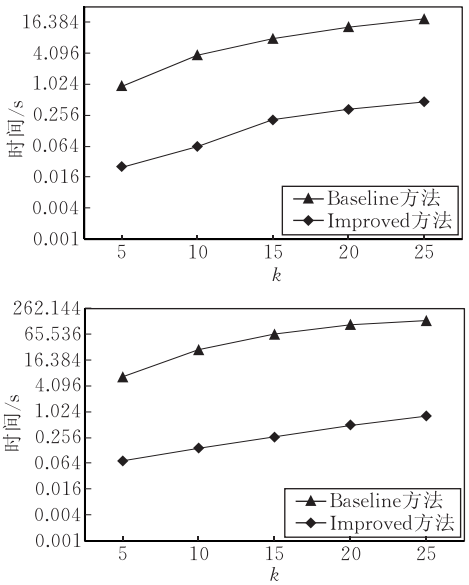


图 8　$m=2$ 时真实(上)和模拟(下)数据比较

图 9 考察了对 3 类属性上 MaxLMC 做摘要的结果,仍可从中看出剪枝后时间效率的大幅提升.当与图 8 中 $m=2$ 时情况对比,可以发现,我们的剪枝算法效率有所下降,但仍然可保证实时性.这是因为我们的剪枝是针对某一个属性上的聚合结果的剪枝,若 m 值增大,时间耗费也会随之逐渐增大.

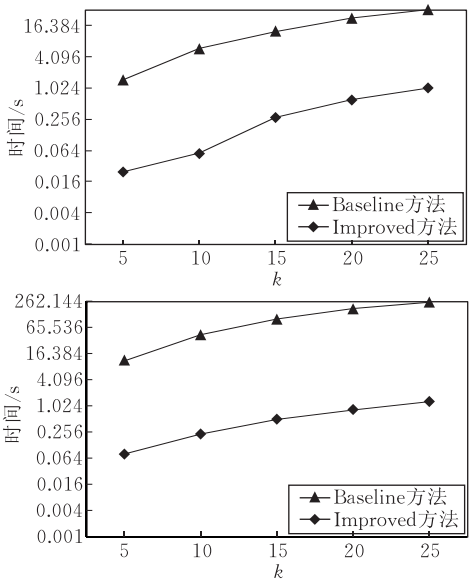


图 9　$m=3$ 时真实(上)和模拟(下)数据比较

6　总　结

本文在具有层次关系的关系数据库上提出了关键字检索结果的聚合问题和聚合结果的摘要问题,我们以 DISCOVER 作为基础,在其上进行扩展.此外,针对两个问题的基本算法我们又提出了具体的改进算法.分析和实验结果表明,改进算法的时间花费较小,具有明显优势.进一步的工作包括考虑更加有意义的反馈机制和对摘要结果进行打分,用以将聚合结果按照相关顺序返回.

参　考　文　献

[1] Agrawal S, Chaudhuri S, Das G. DBXplorer: A system for keyword-based search over relational databases//Proceedings of the 18th International Conference on Data Engineering. San Jose, California, USA, 2002: 5-16

[2] He H, Wang H, Yang J, Yu P S. BLINKS: Ranked keyword searches on graphs//Proceedings of the ACM SIGMOD International Conference on Management of Data. Beijing, China, 2007: 305-316

[3] Hristidis V, Gravano L, Papakonstantinou Y. Efficient IR-style keyword search over relational databases//Proceedings of the 29th International Conference on Very Large Data Bases. Berlin, Germany, 2003: 850-861

[4] Hristidis V, Papakonstantinou Y. DISCOVER: Keyword search in relational databases//Proceedings of the 28th International Conference on Very Large Data Bases. Hong Kong, China, 2002: 670-681

[5] Luo Y, Lin X, Wang W, Zhou X. SPARK: Top-k keyword query in relational databases//Proceedings of the ACM SIGMOD International Conference on Management of Data. Beijing, China, 2007: 115-126

[6] Liu F, Yu C, Meng W, Chowdhury A. Effective keyword search in relational databases//Proceedings of the ACM SIGMOD International Conference on Management of Data. Chicago, Illinois, USA, 2006: 563-574

[7] Bhalotia G, Hulgeri A, Nakhe C, Chakrabarti S, Sudarshan S. Keyword searching and browsing in databases using BANKS//Proceedings of the 18th International Conference on Data Engineering. San Jose, California, USA, 2002: 431-440

[8] Tata S, Lohman G M. SQAK: Doing more with keywords//Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD). Vancouver, BC, Canada, 2008: 889-902

[9] Qin L, Yu J X, Chang L. Keyword search in databases: The power of RDBMS//Proceedings of the ACM SIGMOD International Conference on Management of Data. Providence, Rhode Island, USA, 2009: 681-694

[10] Kacholia V, Pandit S, Chakrabarti S, Sudarshan S, Desai R, Karambelkar H. Bidirectional expansion for keyword search on graph databases//Proceedings of the 31st International Conference on Very Large Data Bases. Trondheim, Norway, 2005: 505-516

[11] Kimelfeld B, Sagiv Y. Efficient engines for keyword proximity search//Proceedings of the 8th International Workshop on the Web & Databases(WebDB 2005). Baltimore, Maryland, USA, 2005: 67-72

[12] Wen Ji-Jun, Wang Shan. SEEKER: Keyword-based information retrieval over relational databases. Journal of Software, 2005, 16(7): 1270-1281(in Chinese)
(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)

[13] Garey M R, Johnson D S. Computers and Intractability: A Guide to the Theory of NP-Completeness. New York: W. H. Freeman and Company, 1979

[14] Balmin A, Hristidis V, Papakonstantinou Y. ObjectRank: Authority-based keyword search in databases//Proceedings of the 30th International Conference on Very Large Data Bases. Toronto, Canada, 2004: 564-575

[15] Hristidis V, Hwang H, Papakonstantinou Y. Authority-based keyword search in databases. ACM Transactions on Databases Systems(TODS), 2008, 33: 1-40

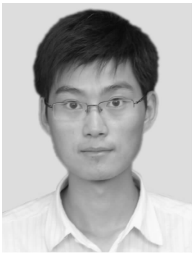
[16] Qin L, Yu J X, Chang L, Tao Y. Querying communities in relational databases//Proceedings of the 25th International Conference on Data Engineering(ICDE). Shanghai, China, 2009: 724-735

[17] Zhou B, Pei J. Answering aggregate keyword queries on relational databases using minimal group-bys//Proceedings of the 12th International Conference on Extending Database Technology. Saint Petersburg, Russia, 2009: 108-119

[18] Buranasaksee U, Porkaew K, Supasitthimethee U. Answer aggregation for keyword search over relational databases//Proceedings of the 2nd International Conference on Computer and Network Technology. Bangkok, Thailand, 2010: 477-482

[19] Roy S B, Amer-Yahia S, Chawia A, Das C, Yu C. Constructing and exploring composite items//Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD). Indianapolis, Indiana, USA, 2010: 843-854

[20] Hochbaum D S. Approximation Algorithms for NP-hard Problems: Approximating Covering and Packing Problems: Set Cover, Vertex Cover, Independent Set, and Related Problems. Boston: PWS Publishing Company, 1997: 94-143



HU Hao, born in 1988, M. S. candidate. His current research interests include XML data management and keyword search.

HE Zhen-Ying, born in 1977, Ph. D., lecturer. His research interests include XML data management and graph data management.

Background

Answering aggregate keyword queries on relational databases is a novel problem motivated these years, and has been well studied. This paper tackle aggregate keyword search problem under the consideration of there being relational databases in which the value of a specific attribute is organized in hierarchical structure. With the conduct of this hierarchical structure, we redefined the results for a given keyword query, called MaxLMC. MaxLMC takes both traditional keyword answers and new feature introduced by new scenario in this paper into account. Based on these definitions, we propose two algorithms to effectively compute answers called Group-by Based method and Sequentially Joining method. The latter is a enhanced version of the former method since it takes full advantage of hierarchical structure in processing. Recognized that the number of returned answer might be extremely large in practical, we defined and studied the problem of effective exploration of large sets of aggregating tuples, called summarization. It is a technique which has been applied to help the user find diverse aggregating tuples. This problem is a instance of a known maximum k-set cover problem which is NP complete. A classical greedy algorithm was introduced to tackle this problem, and with the feature of our aggregate keyword search, three pruning method was proposed to improve the efficiency of current greedy algorithm. Experiment using both real data sets and synthetic data sets is reported to verify the effectiveness and the efficiency of answering aggregate keyword search and our summarization method.

This research is supported by the National Science and Technology Major Project under grant No. 2010ZX01042-003-004, the National Natural Science Foundation of China under grant Nos. 60703093, 61033010, 61073001, the National High Technology Research and Development Program (863 Program) of China under grant No. 2009AA062803, the modern service industry of Science and Technology Commission of Shanghai Municipality under grant No. 10dz1511000.