基于关键字的数据库搜索研究综述

朱凡微, 吴明晖, 金苍宏, 吕 嘉, 应 晶
(浙江大学 计算机学院, 杭州 310027)

摘 要: 介绍了基于关键字的关系数据库搜索技术的研究成果, 从数据建模、体系结构、关键算法等几个方面详细分析和比较了各种技术的特点和优劣, 并指出了现有技术中存在的一些问题, 提出数据库关键字搜索技术未来的研究方向。

关键词: 基于关键字的搜索; 数据库模式; 相关性排序; top-*k* 查询
中图分类号: TP391 **文献标志码**: A **文章编号**: 1001-3695(2008)11-3238-05

Survey of keyword-based search over databases

ZHU Fan-wei, WU Ming-hui, JIN Cang-hong, LV Jia, YING Jing
(College of Computer Science & Technology, Zhejiang University, Hangzhou 310027, China)

Abstract: This paper introduced the related research of keyword-based search technologies over relational databases, analyzed and compared the models, architectures, and algorithms of the existing systems. Also, mentioned deficiencies and problems and prospected the future work.

Key words: keyword-based search; database schema; relevance ranking; top-*k* query

基于关键字的搜索引擎是人们在互联网上搜索信息的重要方式。然而, 现有的互联网搜索引擎主要针对静态网页进行索引。在网络应用中, 大量数据保存在后台数据库中, 以动态网页的形式提供给用户, 不能被现有的搜索引擎索引; 同时, 存储在数据库尤其是关系数据库中的数据, 由于需要特定的查询接口也不能被搜索引擎直接访问[1]。互联网上大量的关系数据库构成了 Deep Web 的主要组成部分, 也是现有互联网搜索技术所面临的一大难题。基于关键字的数据库搜索技术正是在这样的背景下成为数据库、信息检索、互联网等领域的学者共同关注的研究热点。

本文详细介绍了基于关键字的关系数据库搜索的研究现状, 比较数据库搜索中经典的数据模型和搜索模型, 概括出各种模型的优缺点, 并介绍了改进的模型。基于应用系统实例, 总结出数据库搜索系统的两种体系结构及各自特点, 着重对数据库搜索中的关键算法进行了分析评价, 并提出了改进意见。

1 研究现状

由于关系数据库是使用最为广泛的数据库, 也是 Deep Web 的主要构成部分, 基于关键字的关系数据库搜索主要也是针对关系数据库进行的。纵观近十年来该领域的研究成果, 可以分为三个阶段:

a) 第一阶段是对基础理论和算法的研究, 包括对关系数据库的建模、系统实现的体系结构和搜索排序算法的研究。

1998 年, R. Goldman 等人[2] 首先将信息检索中的邻域搜索方法引入数据库中, 在学术界掀起了数据库搜索引擎的研究热潮。2002 年, S. Agrawal 等人[3] 针对关系数据库实现了一个基于关键字的搜索引擎 DBXplorer。Aditya 等人提出将搜索结果进行排序的思想, 并实现了支持相关性排序和结果展现的 BANKS 系统[4]。V. Hristidis 等人[5] 在文献[3, 4] 的基础上对体系结构和算法优化作了进一步的研究, 提出了 DISCOVER 系统。2003 年 V. Hristidis 等人[6] 又将 IR 领域已经很成熟的排序策略运用于数据库搜索中, 提出 IR-Style 排序策略。2004 年 A. Balmin 等人[7] 在 Google 的 PageRank 算法启发下, 提出了一种基于权威(authority-based) 的相关性排序算法 ObjectRank。

b) 第二个阶段是对搜索准确性的探讨和提高, 研究主要涉及排序算法、搜索性能、搜索结果显示的优化等。

2005 年文继军等人[8] 对数据库搜索语言进行了扩展, 基于三类关键字, 即 Keyword、Keyword: Keyword、Keyword: 〈op〉Value 实现的 SEEKER 系统不仅可以支持属性值的搜索, 而且还可以对元数据进行搜索; 并对相关性评分机制作了改进, 将 top-*k* 结果返回给用户。2006 年 Liu Fang 等人[9] 通过对影响搜索有效性的关键因子的识别和规范化定义, 提出一种有效的新的排序策略, 并通过大量实验证明了搜索的有效性。Ding Bo-lin 等人[10] 针对寻找最小代价的 Steiner tree 问题提出了新的参数化解决方案, 将动态规划法与 Best-First 策略相结合, 有效地降低了原有搜索算法的时间/空间复杂度。Peng Zhao-hui 等人[11] 针对搜索结果的显示问题, 提出了一种新的聚类方法——TreeCluster。该方法包含模式聚类和关键字聚类两个步骤, 将同一主题的搜索结果归入同一个 cluster 中, 为用户提供了一种根据 cluster 的描述方便地选择感兴趣结果的方式。2007 年, He Hao 等人[12] 提出一种新的双层索引机制 Bi-level 索引, 很好地减小了逆向搜索的时间代价。Luo Yi 等人[13] 在 IR-Style 排序的基础上, 基于虚拟文档的模型, 提出了一种新的

收稿日期: 2008-01-30; 修回日期: 2008-04-18

作者简介: 朱凡微(1984-), 女, 福建人, 硕士, 主要研究方向为数据集成、数据库搜索(zhufw_1984@163.com); 吴明晖(1976-), 男, 江西人, 副教授, 博士研究生, 主要研究方向为软件工程; 金苍宏(1982-), 男, 浙江人, 硕士, 主要研究方向为 Web 服务组合; 吕嘉(1977-), 男, 江苏人, 博士研究生, 主要研究方向为面向方面的软件开发; 应晶(1971-), 男, 教授, 博士, 主要研究方向为软件工程、人工智能.

相关性排序方法,大大地简化了 IR-Style 排序中相关性评分的计算,同时提出了 Skyline Sweeping 算法和 Block Pipeline 算法,减少了查询执行期间对数据库的访问。

c) 第三阶段是对传统搜索系统的扩展,包括语义搜索的研究、分布式数据库搜索等。Zhang Jun 等人[14]提出了基于本体的语义搜索,利用领域本体的层次结构来实现关键字的概念比较。与之前系统采用的简单关键字匹配相比,这种方法很好地提高了查询的召回率(recall rate)和正确率(precision rate)。Yu Bei 等人[15]将面向单数据源的关键字搜索扩展为多数据源的搜索,解决了相关数据源的选择问题,将传统的基于关键字的数据库搜索扩展为支持分布式数据库的搜索。类似地,M. Sayyadian等人[16]也针对实际应用中存在的自治和异构的关系数据库,提出了将数据库模式匹配与结构发现技术结合的方法来发掘异构数据库中主外键关系,以实现对异构数据库高效快速的关键字查询。

2　关系数据库搜索模型

2.1　数据模型

对数据库建模是数据库搜索的第一步,数据模型的优劣不仅决定了搜索的准确性和完备性,同时也影响到搜索的效率。考虑到关系数据库中的关系通过主外键关系来连接这一特点,现有的搜索方法大都采用了有向图的方式来对数据库建模,主要可以分为模式图方法和数据图方法两类。

2.1.1　模式图方法

模式图(schema graph)方法将数据库中的每个关系表示为一个节点,关系之间的主外键联系表示为节点之间的有向边,由主键所在节点指向外键所在节点。图 1 是 DBLP[17]数据库的模式图。进行关键字搜索时,根据模式图和关键字集合生成候选网络(candidate network),通过执行候选网络对应的 SQL 语句来产生结果。采用这种方法的关键字搜索系统有 DBXplorer[3]、Discover[6]、IR-Style[6]等。

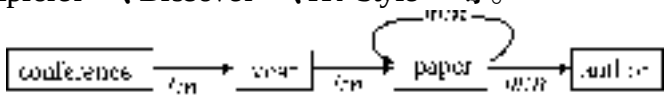

图 1　DBLP数据库的模式图

2.1.2　数据图方法

数据图(data graph)方法则是将数据库中的每个元组作为一个节点,元组直接的主外键关系表示为联系节点的有向边与模式图方法相同,也是由主键所在节点指向外键所在节点。DBLP 数据库的数据图[17]如图 2 所示。进行关键字搜索时,首先根据 DBMS 提供的索引机制选择出包含关键字的节点集,然后在数据库上搜索包含该集合的子图作为搜索结果。BANKS 系统[4]首先采用这种方法实现了数据库关键字搜索 。

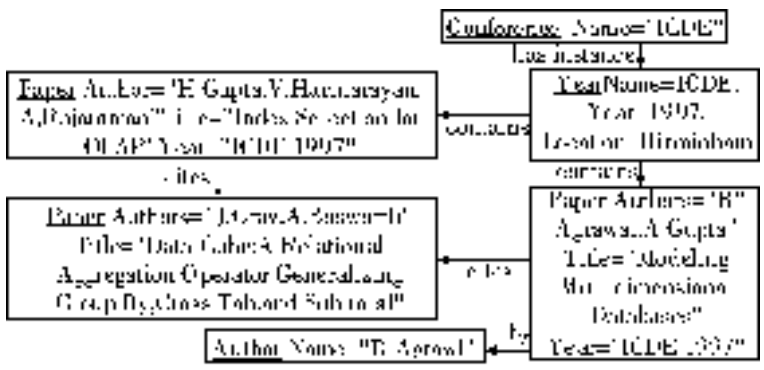

图 2　DBLP数据库的数据图

2.1.3　建模方法比较和改进

模式图方法只需在关系层面对数据库建模,存储的是数据库的结构信息,抽象程度高,在内存中存储的空间较小,但是在生成候选网络上耗费的代价较大;而数据图方法建模的粒度具体到每一个节点,存储了数据库中详细的数据信息,占用较大的内存空间,但是在搜索上具有较高的效率。这两种方法的有向图简单地通过有向边的指向体现了数据表或实体间的主外键联系,不能很好地支持相关性排序。因此对建模方法的改进主要有两方面[4]:a) 将模式图和数据图的方法相结合,在内存中保存简洁的模式图,而将详细的数据图存放于硬盘上;b) 为有向图的边赋予表示相连对象的相关性权重值,以支持相关性排序。

针对传统建模方式的不足,ObjectRank[7]系统采用权威传递图(authority transfer graph)对数据库建模,依据主外键联系在语义上紧密程度的不同,对每条边赋予一个相应的权威值,表示该联系的权威性,不仅很好地表达了数据库元组的语义信息,而且可以从语义相关的角度支持搜索结果的相关性排序,是数据模型研究上的一大突破。权威传递图包括权威传递模式图(authority transfer schema graph)和权威传递数据图(authority transfer data graph)。这两种有向图与原有的模式图和数据图的区别是,通过主外键相关的节点由正反向两条带权威值的边相连,权威传递图中的边不仅表示了对象之间的关联,而且反映了权威值在图中的流动状况。在 ObjectRank 系统中,对每一条边都创建了一条相应的反向边(backforward edge),并赋予权威值。这是因为在实际数据库中,权威值的传递不仅可以从主键节点传给外键节点,也可以从外键节点传给主键节点,而且正反方向传递的值可能不同。例如在 DBLP 数据库中,一篇文章的权重可以传递给他的作者;同样,一个作者的权重也可以传递给他的文章。图 3 是 ObjectRank 中对 DBLP 数据库建立的权威传递图。

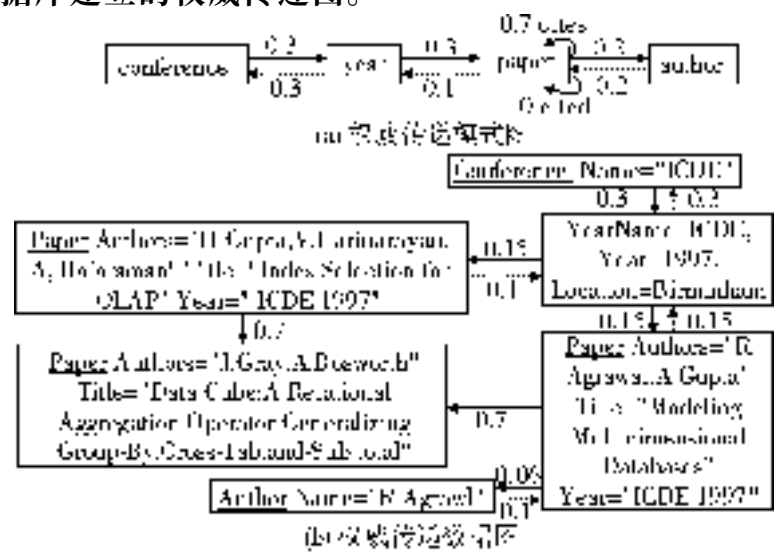

图 3　DBLP的权威传递图

2.2　搜索过程的形式化表示

为了描述数据库搜索的过程和算法,综合现有系统的建模方式[3~7,9,10,13~15,18],对数据库搜索过程中涉及到的关键元素定义如下:

定义 1　数据库模式。其形式化地表示为一个模式有向图 $SG(R_S, E_S)$。其中:$R_S = \{R_1, R_2, \cdots, R_n\}$,$R_i$ 是图中的一个节点,表示数据库中的一个关系;如果对于任意两个节点 R_i, R_j,存在 R_i 到 R_j 的主外键联系,则定义一条边 $\langle i, j\rangle \in E_S$。

定义 2　数据库数据。其形式化表示为一个数据有向图 $DG(N_D, E_D)$。其中:$N_D = \{N_1, N_2, \cdots, N_n\}$,$N_i$ 是图中的一个节点,表示数据库中的一个元组;如果对于任意两个元组 N_i, N_j,存在 N_i 到 N_j 的主外键联系,则定义一条边 $\langle i, j\rangle \in E_D$。

定义 3　搜索条件。其表示为一系列关键字的集合 $Q =$

$\{kw_1, kw_2, \cdots, kw_m\}$。

定义 4 中间结果。其表示为以包含关键字的节点以及一些辅助节点连接而成的元组树(tuple tree),一棵代表一个可能的搜索结果的元组树应满足三个条件:a) 每个叶子节点至少包含一个关键字;b) 每个元组在元组树中最多出现一次;c) 辅助节点的个数不能超过关键字的个数。其中,条件 a) 是完备性要求,保证该元组树满足搜索条件;条件 b) c) 保证不存在冗余,得到的元组树是最小的。

例如,图 4 表示的是一个数据库的模式图 $Q=\{k_1, k_2, k_3\}$,黑色的节点表示包含关键字的节点,则表示可能结果的元组树有四棵,如图 5 所示[3]。

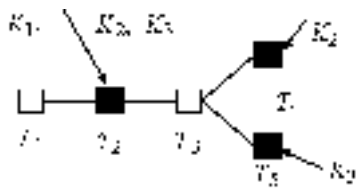

图 4 数据库模式的元组树

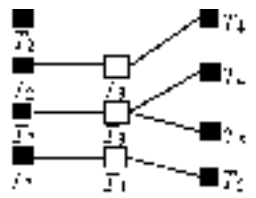

图 5 可能结果的元组树

定义 5 查询结果。其采用 SQL 语句查询中间结果得到的最后结果,表示为元组的自然连接。

2.3 数据库搜索模型的分类

根据搜索匹配的类型,可以分为关键字匹配和语义匹配两类。第一阶段(见第 1 章)的搜索技术多为精确的关键字匹配,即包含搜索关键字的元组满足搜索条件,而不包含关键字的元组则不满足搜索条件。支持语义匹配的搜索模型则是从语义理解的角度对关键字匹配的搜索模型作了改进,它不但考虑了搜索条件字面上的匹配,同时更注重搜索条件在概念上的比较。ObjectRank[7] 在排序算法中引入权威值,在一定程度上体现了语义匹配的思想,而在第二阶段出现的基于本体的语义搜索则是利用领域本体的层次结构关系,根据查询关键字的语义进行概念上的匹配,并采用向量空间模型来计算语义相似度。文献[14] 就是典型的语义匹配搜索系统,很好地提高了搜索的召回率。

根据搜索模型支持的结构,现有的数据库搜索技术可以分为两类:一类仅支持 AND 结构搜索,即返回的结果应当满足所有的查询关键字,早期的数据库搜索技术如文献[2 ~5] 都是仅支持 AND 结构的;另一类是既支持 AND 又支持 OR 结构的搜索,即搜索系统允许返回只满足部分关键字的结果,只是这些结果的相关性评分相对满足全部关键字的结果而言较低,2003 年之后大多数数据库搜索系统都是既支持 AND 结构又支持 OR 的,典型的如文献[6 ~10, 13 ~15, 19, 20] 等。

根据搜索的数据源不同,可以分为单数据源搜索和多数据源搜索。传统的数据库搜索都是针对单数据源,即在单个数据库中进行搜索。随着分布式数据库应用以及 Internet 中面向服务的体系结构的发展,支持多数据源搜索的技术也应运而生。文献[15,16] 详细介绍了在多数据源搜索中选择相关数据源以及对数据源排序的算法。

3 体系结构

数据库关键字搜索系统的体系结构可以根据是否直接查询实时数据库分为两大类,即在线搜索结构和离线搜索结构。在线搜索系统在用户的搜索请求到来时,直接查询实时数据库来生成查询结果,典型的系统有 DISCOVER、BANKS、IR-Style;离线搜索系统首先对数据库进行预处理,用户提交关键字时,是在预处理的结果中进行搜索,典型的系统有 DBXplorer、ObjectRank。在线搜索结构和离线搜索结构的对比如表 1 所示。

表 1 在线搜索和离线搜索对比

对比项目	在线搜索	离线搜索
是否需要预处理	否	是
查询目标	实时数据库	预处理后的数据库
响应时间	较长	较短
维护代价	低	高 数据库更新后需要重新进行预处理
适用情况	中小型数据库	更新较少的数据库
典型应用	DISCOVER、BANKS、IR-Style 等	DBXplore、ObjectRank 等

3.1 在线搜索体系结构

IR-Style[6] 是一个典型的在线搜索系统,它的体系结构是在 DISCOVER 的基础上完善发展而来的,主要分成以下三个模块:

a) IR Engine。当用户的搜索请求 Q 到来时,IR Engine 根据 DBMS 提供的全文索引,从每个关系 R 中选出满足搜索关键字的元组集 $R_Q=\{t\in R \mid score(t, Q)>0\}$。其中:$score(t, Q)$ 是元组 t 针对搜索请求 Q 的相关性排序函数。

b) CNG(candidate network generator)。它将 IR Engine 输出的相关元组集作为输入,并结合数据库模式来产生候选网络表示符合条件的元组的连接关系。

c) Execution Engine。这个模块将候选网络表示的结果转换为用 SQL 语言查询数据库得到的最终结果,并以 top-k 的形式反馈给用户。

3.2 离线搜索的体系结构

典型的离线搜索体系结构如 ObjectRank 系统。ObjectRank 分为预处理和查询两大模块。其中预处理模块在 DBXplorer 中也称为发布模块(publish),体现了离线搜索系统的特征。

a) 预处理模块。它通过 ObjectRank 执行模块来实现,将可供查询的数据库以及 ObjectRank 中需要的一些参数作为输入,输出所有关键字的 AuthoRank 索引。AuthoRank 索引与 RDMS 提供的全文索引不同,它是以 $\langle id(t), r^k(t)\rangle$ 形式来标志包含关键字 k 的元组 t。其中:$id(t)$ 表示元组 t 的编号;$r^k(t)$ 表示元组 t 关于关键字 k 的权威值。在 ObjectRank 中只对 $r^k(t)$ 大于某一阈值的元组进行索引。

b) 查询模块。该模块包括查询处理模块和数据库访问模块。其中,查询处理模块接收 AuthoRank 执行模块输出的 ObjectRank 索引和用户输入的搜索关键字,并结合搜索算法中所需的参数,在 AuthoRank 索引中进行搜索,输出满足条件的 top-k 的 id;数据库访问模块则根据这些 id 来查询数据库,将最终的结果返回给用户。

4 关键技术及算法

4.1 创建索引策略

对数据库中的关键字创建索引的优劣在很大程度上决定了搜索效率的高低和搜索结构的准确性。较为广泛应用的创建索引的策略主要有 IR 索引、符号表和 AuthoRank 索引。

IR 索引[4,6] 就是 RDBMS 支持的对文本属性的全文索引,它是一个逆排文件,以 $\langle k_i, occ(k_i)\rangle$ 的形式记录了关键字 k_i 在数据库中出现的位置。在线搜索系统采用的多是这种索引算

法,如 IR-Style、BANKS 等。

符号表[3]是 DBXplorer 系统中采用的索引方法,它继承 IR 索引的思想,但是利用专门的符号表来记录关键字和该关键字存在的元组编号。符号表的每一项形如 $\langle k_i, \mathrm{id}(k_i)\rangle$。

AuthoRank 索引[7]则是 ObjectRank 系统提出的。对于每个关键字,它记录的是它所在元组的编号和该元组针对该关键字的权威值 $\langle \mathrm{id}(t), r^k(t)\rangle$。该索引算法不仅记录了关键字的位置,同时也包含了关键字的权威信息,便于对结果进行排序。

以上三种索引算法的优缺点比较如表 2 所示。

表 2　索引算法比较

索引算法	优点	缺点
IR 索引	可以利用 RDMS 自带的全文索引	难以定位跨表连接得到的结果的位置
符号表	便于直接定位关键字所在元组,支持跨表连接	需要创建和存储符号表的额外代价
AuthoRank 索引	直接定位关键字所在元组,支持跨表连接,便于结果的相关性排序	维护符号表的代价;创建索引过程复杂;占用内存空间较大;维护代价较高

Bi-level 索引是在 BLINK 机制中提出的[12],它针对上述索引算法存在的内存需求大、最差情形下性能差的问题,解决了在搜索效率和空间需求上寻找一个平衡点的问题。Bi-level 索引包含顶层块索引(top-level block index)和块内索引(intra-block)两部分。首先将数据图划分为小块(block),由顶层块索引记录关键字到每一块的映射关系,以引导块间的搜索,而每一块的详细信息则存储在块内索引中,用来加速块内的查询。

4.2　相关性排序算法

对数据库关键字搜索的结果按照相关性进行排序是数据库关键字搜索中的一大突破。DISCOVER 系统中首次提出了排序的思想,之后很多研究都致力于排序算法的优化和改进,现有的算法主要有基于连接数排序、基于权重的排序、基于权威值的排序、基于 IR 索引的排序四种算法。

基于连接数的排序算法[3]是一种最简单的排序算法。根据关系数据库主外键连接的特点,可以很直观地认为,如果一个搜索答案是通过较多个元组的连接而得到的,那么它与搜索条件的相关性较低;反之,满足搜索条件的单个元组具有最高的相关性。因此基于连接数的排序算法就是按照结果所包含的连接数多少来进行排序,连接数越多,相关性越低。DISCOVER 系统采用的就是这种算法。

基于权重的算法[4]是继承了互联网中 PageRank 算法的思想,它考虑到数据库关系之间的主外键联系程度并非完全相同,而是依赖于数据库的语义,如在 DBLP 数据库中,paper 表和 writes 表之间的联系显然要比 paper 表和 cites 表之间的联系要强,因此,该算法对不同的主外键连接赋予不同的权重。同时,根据 PageRank 的思想,有很多边指向的节点,权重相对要高,因此算法根据指向节点的边的多少对节点赋予一个权重。算法主要包括以下步骤:

a) 根据主外键联系的强度给每条边赋予权重值 $\mathrm{Weight}_{edge}(u, v)$,权重值通常由领域专家给出。

b) 对于每条边 $\mathrm{edge}(u, v)$ 创建一条反向边 backward edge (v, u),反向边的权重值:

$$\mathrm{weight}_{edge}(v, u) = \mathrm{weight}_{edge}(u, v) \times x \times \log(1 + x)$$

其中:x 表示该边的内连(inlinks)数目。

c) 根据指向节点的边的数目给每个节点赋予权重值:

$$\mathrm{weight}_{node}(i) = \log(1 + x)$$

其中:x 表示该节点的入度数(in-degree)。

d) 对于搜索结果,将边权重和节点权重依据一定的函数进行组合得到最后的相关性得分,组合公式为

$$\mathrm{weight}_{total} = (1 - \lambda)\,\mathrm{weight}_{edge} + \lambda \mathrm{weight}_{node}$$

基于权威值的算法是基于权威传递数据图实现的[7],在权威传递数据图中每条边都带有一个权威值:

$$\alpha(e) = \begin{cases} 0, & \text{if } \mathrm{outDeg}(u, e) = 0 \\ \alpha(e)/\mathrm{outDeg}(u, e) & \text{if } \mathrm{outDeg}(u, e) > 0 \end{cases}$$

其中:$\mathrm{outDeg}(u, e)$ 表示节点 u 的出度。

基于权威的算法对每一个节点赋予一个关键字相关的权威值 r^w 和一个全局的权威值 r^G:

$$r^w = dAr^w + (1 - d)\, s / |S(w)|$$

其中:d 是一个调整因子;$A_{ij} = \alpha(e)$;$S(w)$ 是满足搜索条件 w 的元组集;s 是一个基本向量。1 表示该节点属于 $S(w)$;0 表示不属于 $S(w)$。

全局权威值是根据数据库的语义来赋予的,在不同的应用中具有不同的值,最后的结果为

$$r^{w,G}(v) = r^w(v) \times (r^G(v))^g$$

其中:g 是表示全局权威值在最终得分中所占的比重,可以由用户或管理员来指定。

基于 IR 索引的排序算法[6]主要利用了现有的一些 DBMS 在单个文本属性上对相关性排序的支持,将单个属性上的相关性得分组合成为最终的得分。对于查询 Q,单个文本属性的相关性得分是 DBMS 的 IR 引擎计算得出:

$$\mathrm{score}(a_i, Q) = \sum_{w \in Q \cap a_i} [1 + \ln(1 + \ln(tf))] / [(1 - s) + (\mathrm{sdl}/\mathrm{avdl})] \times \ln(N + 1)/(df)$$

其中:tf 表示关键字 w 在属性 a_i 中出现的频率;df 是 a_i 属性值包含 w 的元组数目。每个文本属性的值通过 combine 函数组合得到该元组数最终的分值:

$$\mathrm{combine}(\mathrm{score}(A, Q), \mathrm{size}(T)) = \left(\sum_{a_i \in A} \mathrm{score}(a_i, Q)\right) / \mathrm{size}(T)$$

其中:A 是元组树的所有文本属性的向量表示;$\mathrm{size}(T)$ 表示元组树的大小。上述四种排序算法的优缺点比较如表 3 所示。

表 3　排序算法比较

排序算法	优点	缺点
基于连接数排序	算法简单直观	无法对相同连接数的元组排序
基于权重排序	不同连接具有不同的权重 符合数据库的语义	权重值不易确定 需要领域知识
基于权威值排序	边和节点都有权威值 节点具有特定的和全局的权威值;可以调整影响因子控制排序结果	算法实现较复杂 需要权威传递图的支持
基于 IR 索引排序	可利用 DBMS 的 IR 引擎的支持;算法在 IR 领域上已成熟应用	不能突出 AND 语义的优先级;combine 函数过于强调不同元组中同一关键字的贡献

基于虚拟文档的排序算法[13]是对传统 IR 索引排序算法的一种改进,它将每个元组连接树(joined tuple tree)视为一个虚拟文档(virtual document)。因此,一个候选网络产生的所有结果将被建模为文档的集合,虚拟文档是与特定查询相关的,且在运行时动态创建。基于这种模型,可以直接计算每个元组连接树的相关性得分,而不必对每个文本属性进行评分再进行累加,每个元组连接树 T 的得分计算如下:

$$\mathrm{score}_a(T, Q) = \sum_{w \in Q \cap T_i} [1 + \ln(1 + \ln(tf_w(T)))] / [(1 - s) + sdl_T/(avdl_{CN \times (T)})] \times \ln(idf_w)$$

针对满足所有关键字的元组树具有比满足部分关键字的元组树更高的相关性得分这一要求,该算法引入完备性因子

(completeness factor):

$$score_b(T, Q) = 1 - [(\sum_{1 \le i \le m} (1 - T.i)^p)/m]^{1/p}$$

其中: $T.i$ 是在元组连接树中关键字 kw_i 经过规范后的频率表示。

该算法还通过定义元组树的大小因子(size normalization factor)来体现元组连接树的大小对关键字出现几率的影响,即元组连接树越大,关键字出现几率也越大,通过实验得到较有效的大小因子计算公式如下:

$$score_c = (1 + s_1 - s_1 \times size(CN)) \times (1 + s_2 - s_2 \times size(CN^{nf}))$$

基于虚拟文档的排序算法对每一个元组连接树的最终评分是上述三个评分函数的乘积:

$$score(T, Q) = score_a(T, Q) \times score_b(T, Q) \times score_c(T, Q)$$

5 结束语

数据库关键字搜索作为一个新兴的研究领域,在理论研究和应用系统开发上均取得了很大的进展,数据库关键字搜索在研究上不仅继承了互联网搜索和信息检索中的一些成熟技术,而且还根据关系数据库自身的结构化特点和语义信息,提出新的、更加适用于数据库搜索的算法和相关性排序策略,并在实际应用中取得了较好的效果。现有的数据库关键字搜索研究正在朝搜索性能优化以及与其他相关领域技术相结合的方向发展。然而,目前对数据库关键字搜索的研究也并存在一些问题: a) 现有研究仅仅是针对关系数据库,而在互联网上可能存在大量的非关系数据库,这些数据库也存储了大量的有用信息,如何实现对非关系数据库的关键字搜索将是未来的一个研究方向。b) 现有的数据库关键字搜索系统都是独立的一个系统,与现存互联网搜索系统没有接口,而对于用户的搜索需求,要求两者能够很好地结合起来。因此,实现非关系数据库关键字搜索,以及数据库关键字搜索和互联网搜索引擎的无缝连接还有待进一步的研究。同时,数据库的语义搜索和分布式数据库、异构数据库的搜索仍然是未来研究的热点。

参考文献:

[1] AVERLEE J, LIU L, ROCCO D. Discovering interesting relationships among deep webdatabases: a source-biased approach[J]. World Wide Web, 2006, 9(4): 585-622.

[2] GOLDMAN R, SHIVAKUMAR N, VENKATASUBRAMANIAN S, *et al*. Proximity search in databases[C]//Proc of VLDB '98. New York: [s. n.], 1998.

[3] AGRAWAL S, CHAUDHURI S, DAS G. DBXplorer: a system for keyword-based search over relational databases[C]//Proc of ICDE '02. California: [s. n.], 2002.

[4] BHALOTIA G, HULGERI A, NAKHEY C, *et al*. Keyword searching and browsing in databases using BANKS[C]//Proc of ICDE '02. California: [s. n.], 2002.

[5] HRISTIDIS V, PAPAKONSTANTINOU Y. DISCOVER: keyword search in relational databases[C]//Proc of VLDB '02. Hong Kong: [s. n.], 2002.

[6] HRISTIDIS V, GRAVANO L, PAPAKONSTANTINOU Y. Efficient IR-Style keyword search over relational databases[C]//Proc of VLDB '03. Berlin: [s. n.], 2003.

[7] BALMIN A, HRISTIDIS V, LPAPAKONSTANTINOU Y. Authority based keyword queries in databases using ObjectRank[C]//Proc of VLDB '04. Toronto: [s. n.], 2004.

[8] 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): 1270-1279.

[9] LIU Fang, YU C T, MENG Wei-yi. Effective keyword search in relational databases[C]//Proc of SIGMOD '06. Chicago: [s. n.], 2006.

[10] DING Bo-lin, YU J X, WANG Shan. Finding top-*k* min-cost connected trees in databases[C]//Proc of ICDE '07. Istanbul: [s. n.], 2007.

[11] PENG Zhao-hai, ZHANG Jun, WANG Shan. TreeCluster: clustering results of keyword search over databases[C]//Proc of WAIM '06. Hong Kong: [s. n.], 2006.

[12] HE Hao, WANG Hai-xun, YANG Jun, *et al*. BLINKS: ranked keyword searches on graphs[C]//Proc of SIGMOD '07. Beijing: [s. n.], 2007.

[13] LUO Yi, LIN Xue-min, WANG Wei, *et al*. SPARK: top-*k* keyword query in relational databases[C]//Proc of SIGMOD '07. Beijing: [s. n.], 2007.

[14] ZHANG Jun, PENG Zhao-hui, WANG Shan. Si-SEEKER: ontology-based semantic search over databases[C]//Proc of the 1st Conference on KSEM '06. 2006: 599-611.

[15] YU Bei, LI Guo-liang, SOLINS K. Effective keyword-based selection of relational databases[C]//Proc of SIGMOD '07. Beijing: [s. n.], 2007.

[16] SAYYADIAN M, LEKHAC H, DOAN A, *et al*. Efficient keyword search across heterogeneous relational databases[C]//Proc of ICDE '07. Istanbul: [s. n.], 2007.

[17] KAUSHIK R, KRISHNAMURTHY R, RAMAKRISHNAN R. On the integration of structure indexes and inverted lists[C]//Proc of ICDE '04. Boston: [s. n.], 2004.

[18] ZHANG Jun, PENG Zhao-hui, WANG Shan, *et al*. PreCN: preprocessing candidate networks for efficient keyword search over databases[C]//Proc of WISE '06. Wuhan: [s. n.], 2006.

[19] ZHAN Jiang, WANG Shan. ITREKS: keyword search over relational database by indexing tuple relationship[C]//Proc of DASFAA '07. Bangkok: [s. n.] 2007.

[20] 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[J]. 计算机研究与发展, 2007, 44(1): 119-125.

[21] WANG Shan, DU Xiao-yong, MENG Xiao-feng, *et al*. Database research: achievements and challenges[J]. Computer Science & Technology, 2006, 21(5): 823-837.

[22] SINGHAL A. Modern information retrieval: a brief overview[J]. IEEE Data Eng Bull, 2001, 24(4): 35-43.

[23] GROSSMAN D, FRIEDER O. Information retrieval: algorithms and heuristics[M]. 2nd ed. [S. l.]: Springer Publishers, 2004.

[24] MAMOULIS N, CHENG K H. Efficient aggregation of ranked inputs[C]//Proc of ICDE '06. Atlanta: [s. n.], 2006.

[25] KIMELFELD B, SAGIV Y. Finding and approximating top-*k* answers in keyword proximity search[C]//Proc of PODS '06. 2006.

[26] KACHOLIA V, PANDIT S, CHAKRABARTI S. Bidirectional expansion for keyword search on graph databases[C]//Proc of VLDB '05. Trondheim: [s. n.], 2005.

[27] LIU Zi-yang, CHEN Yi. Identifying meaningful return information for XML keyword search[C]//Proc of SIGMOD '07. Beijing: [s. n.], 2007.

[28] DONG X, HALEVY A. Indexing dataspaces[C]//Proc of SIGMOD '07. Beijing: [s. n.], 2007.

[29] WANG Shan, PENG Zhao-hui, ZHANG Jun. NUITS: a novel user interface for efficient keyword search over databases[C]//Proc of VLDB '06. Seoul: [s. n.], 2006.

[30] 王珊, 文继荣. 数据库和信息检索技术的融合[J]. 中国计算机学会通讯, 2006, 2(4): 18-24.

[31] 王珊, 张俊, 彭朝晖, 等. 基于本体的关系数据库语义检索[J]. 计算机科学与探索, 2007, 1(1): 59-78.