一种基于预先索引的关系数据库关键词搜索方法

葛唯益 宗士强 尹文科
(信息系统工程重点实验室 南京 210007)

摘 要 关系数据库的关键词搜索面临的最大挑战在于满足需求的答案可能来自多个关系的元组的组合。现有主流方法通过定位每个关键词对应的元组并动态发现元组之间的关联来得到搜索结果。然而当数据库规模较大或模式复杂时,这些方法存在搜索效率低的问题;此外,这些方法因只能支持简单的关键词查询而实用性受到限制。为此,提出对元组的组合进行预先索引从而加快搜索,此外还对其索引效率及查询能力进行改进以提高系统的可用性。首先,为了提高搜索和索引效率,提出基于模式图的元组连接枚举技术,该技术利用无环模式图枚举合适的关系连接,将其转换为SQL语句在数据库中执行以得到可能的元组连接;其次,为了保证结果的紧致性,提出了1到m元组连接的预先索引与顺序搜索机制,该机制对元组连接进行由小到大的搜索,并限制所有包含已有结果的元组连接都不再参与搜索;最后,为了支持复杂查询,提出基于域的索引结构,为每个元组连接建立面向不同查询类型的域,通过查找多个域并对结果进行逻辑组合得到最终结果。实验表明,相比于已有技术,本技术具有较快的索引速度与较高的查询效率,并能提供如布尔查询、属性查询等的复杂查询能力。

关键词 关系数据库,关键词搜索,预先索引,紧致性,复杂查询
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.4.037

Keyword Search for Relational Databases Based on Offline Index

GE Wei-yi ZONG Shi-qiang YIN Wen-ke
(Science and Technology on Information Systems Engineering Laboratory, Nanjing 210007, China)

Abstract The biggest challenge for keyword search over relational databases is that results are often assembled from tuples in several tables. Dominant approaches find tuples hit by keywords and identify their joins on the fly to form results, which are rather inefficient for databases with large scale or complex schema. Besides, these approaches only support simple keyword query, which limits their practical usage. Regarding this, we proposed an alternative way by indexing joinable tuples offline to speed online search, and strove to improve its index efficiency and query capability for practical usage. Firstly, to improve search and index efficiency, we proposed an approach to utilize schema graph information to enumerate joinable tuples. This approach discovers all suitable joinable tables and translates them to SQL queries, which are sent to database interfaces for getting possible joinable tuples. Secondly, to ensure the compactness of results, we proposed an approach to index joinable tuples and searched them by order of their size. The approach selects rele-vant joinable tuples with small scales in advance and excludes those joinable tuples containing them from the next round of selection. Finally, to support complex query, we proposed a field-based index structure, which uses different fields for different search types. At search time, these fields are queried and their results are sent to perform logical operations to get the final results. Experiments show that, compared to existing approaches, our approach outstands in index and search efficiency and provides query capability close to the SQL.

Keywords Relational database, Keywords search, Offline index, Compactness, Complex query

1 引言

关系数据库与信息检索的融合催生了关系数据库的关键词搜索技术[1]。该技术具有两大优势:首先,由于无需学习结构化查询语言(SQL)以及数据库模式的知识,降低了普通用户搜索数据库的技术门槛;其次,有别于现有应用开发需要预先编制查询模板,其查询方式更为灵活。因此它吸引了研究社区的广泛关注[2-13]。

尽管大部分关系数据库支持全文搜索,但它们只能搜索同一物理表中指定列的相关元组[14-16]。为了正规化的需要,数据库的逻辑信息单元常常分散在多个物理表中,因此关系数据库关键词搜索的最大挑战在于需要组合多个物理表中具有关联关系且与查询相关的元组构成最终的答案。

为了解决该问题,现有数据库关键词搜索产品(如

到稿日期:2015-03-17 返修日期:2015-07-30 本文受国家自然科学基金(61402426),软件新技术与产业化协同创新中心资助。
葛唯益(1985-),男,博士,工程师,主要研究方向为信息检索、数据库、语义Web,E-mail:geweiyi@gmail.com;宗士强(1978-),男,硕士,高级工程师,主要研究方向为数据库、数据挖掘;尹文科(1987-),男,博士,工程师,主要研究方向为信息检索。

Apache Solr[1])需要用户事先编写 SQL 语句来指定元组连接并进行索引，而系统只能对这些指定的内容提供关键词查询服务，因此这类系统需要人的参与，自动化程度低，无法满足大规模应用的需要。为此，该领域的学者关注于如何自动发现元组连接。目前主流的思想[2-10]是通过定位每个关键词对应的元组并动态发现元组之间的关联来得到搜索结果。但这类方法因为需要在线查找元组关联，存在搜索效率低的问题。于是基于预先索引的搜索方法应运而生[11-13]，它们通过事先发现所有可能的元组连接，并对其进行索引，以此来支持高效的检索。但现有方法由于索引与返回最长的元组连接，因此存在索引效率低以及结果不紧致的问题。此外，所有的这些方法都没有考虑如何支持复杂查询的问题。

首先，本文认为作为查询结果的元组连接需要满足两个性质：查询相关以及紧致性。查询相关是指元组连接要满足查询要求；紧致性是指无法删除元组连接中的任何元组，使得剩下的元组连接依然满足查询相关性。为此，本文利用数据库模式图枚举不同长度的关系连接并将其转化为 SQL 语句，从而查询得到不同长度的元组连接，通过对这些元组连接进行预先索引与顺序搜索使得产生的结果满足紧致性要求。因为在索引阶段使用模式图而非数据图，所以提高了元组枚举的效率。

其次，数据库的关键词搜索需要支持复杂查询，从而保证系统的可用性。针对支持复杂查询的需求，本文提出了基于域的索引结构，该索引除了支持简单的关键词查询之外，还能支持属性值查询以及支持关键词之间的与或逻辑组合。

2 相关工作

关系数据库的关键词搜索与传统的文档搜索的最大区别在于其结果可能是多个元组的连接，为此现有商用数据库搜索系统需要用户通过 SQL 语句手动指定要索引的元组连接，但该类系统存在适用性问题。目前研究工作关注于如何自动地搜索元组连接的结果，它们可以分为 3 类。

2.1 基于数据图(Data Graph)的搜索方法

这类方法将关系数据转换为图结构，图的每个节点对应于数据库中的一条元组，图上的边对应于元组之间的主外键关联。对于给定的关键词查询，利用从关键词到元组的索引，找到包含该关键词的所有元组，即图中的关键词点的集合。给定一个图以及多组关键词点的集合，需要寻找一个简化子树将每组中至少一个关键词点连接起来，并且该简化子树包含所有关键词且权值最小，该问题是分组 Steiner 问题。不同的研究文献对简化子树的定义也不相同，如简化子树不考虑方向，称为元组连接树；简化子树考虑方向，称为有根有向树。因为分组 Steiner 问题被证明是 NP-hard 问题[17]，目前只能使用一些近似算法进行求解，例如常见的方法包括：反向搜索(backward search)[2,7]以及动态规划[10]等。

基于数据图的检索方法尽管容易理解与实现，但也存在一些问题：1)数据图以元组为节点，因此具有相当的规模，为了找到包含关键词的元组之间的连接关系，算法要执行大量的遍历和查找工作，因此该类方法存在搜索时间较长的问题；2)大规模的数据图需要进行物化，从而增加了存储的开销以及更新的负担；3)实际应用往往要求查询能够支持复杂的逻辑组合，而该类方法往往只能处理关键词之间“与”的关系，因此存在实用性问题。

2.2 基于模式图(Schema Graph)的搜索方法

为了充分利用 RDBMS 自身的查询能力来加快检索速度，出现了基于模式图的检索方法[3-5,9]。该类方法利用数据库模式来枚举可能包含查询结果的所有连接表达式，然后在数据库上执行这些表达式得到查询结果。模式图中每个节点代表数据库中的一个关系，两个点之间存在一条边则代表它们之间存在主外键关联。对于给定的关键词查询，首先定位到包含每个关键词的关系，并在模式图上利用分组 Steiner 算法枚举所有可能的关系连接树；其次根据一系列规则将关系连接树转换成 SQL 语句，并在数据库上执行，得到所有可能的简化子树；最后对简化子树进行排序，并把相关内容返回给用户。实际中，如果两个元组之间通过太多的中间连接建立关系，那么，它们间距离太过遥远，对用户的应用价值就很小。因此，在第一步枚举时，会对表达式的连接数进行限制。

由于模式图的规模较小，且能充分利用 RDBMS 的查询能力，该类方法的搜索效率比基于数据图的方法更高，但是也存在不足。首先，由于模式图的抽象性，该类方法生成的连接表达式往往容易查询得到空结果，从而降低了整个搜索的效率。其次，该方法同样不支持复杂的逻辑查询，存在实用性问题。

2.3 基于预先索引的搜索方法

上述两类方法由于都要在运行时动态发现元组之间的关联，因此存在检索效率问题，为此一些研究工作提出对元组连接预先建立索引。文献[11]通过选择那些实体关系中[2])的元组作为根，从这些根出发利用数据图枚举所有最长元组连接树，然后提取它们的文本建立虚拟文档，并对其建立索引以加快检索过程。该方法因为使用预先索引机制，其检索时效性远好于不用预索引的方法。但是由于该方法基于数据图，其索引过程较慢。文献[12]对此进行了改进，根据模式图计算 full disjunction，以此将所有关系融合成一张表。通过对表中的元组进行索引来解决快速搜索问题。由于其可以利用 SQL 查询完成计算 full disjunstion，因此索引效率有所改进。文献[13]在文献[12]的基础上进一步研究了结果的排序问题。

但是，文献[11-13]返回的搜索结果都是元组通过主外键关系连接产生的最长结果，不是紧致的，因此每条结果都存在无关信息，从而增加了用户的浏览负担。其次，如果两个元组距离太远(比如通过很多中间元组连接)，其实际意义也不大，因此在做元组连接时需要考虑连接的距离。最后，除了基本查询之外，还有些用户希望支持更为复杂的查询，例如对属性值查询，目前的搜索方法都无法支持。

3 问题定义

3.1 关系数据库

关系数据库由一系列关系构成。关系 R 的头部 $H(R)=\{a_1,a_2,\cdots,a_n\}$，表示其具有 $a_1,a_2,\cdots,a_n$ 这些属性。关系 R 的主体 $B(R)=\{t_1,t_2,\cdots,t_m\}$ 对应于元组的集合，每个元组

1) http://lucene.apache.org/solr

2) 实体关系是指主键不完全由外键所构成的关系

$t=\{(a_1,v_1),\cdots,(a_k,v_k)\}$是一系列属性-值对的集合。

如果关系 R 中的某一属性或属性集 $A\subseteq H(R)$ 的值唯一标识一条元组，且 A 的任何真子集无此性质，则 A 称为 R 的候选键。每个关系可能有多个候选键，其中的一个被选择的候选键称为主键。如果关系 R_1 中的属性或者属性组 A_1 引用了关系 R_2 的主键 A_2，则称 $R_1.A_1$ 为外键，引用条件为 $R_1.A_1=R_2.A_2$。

定义 1(关系连接) 由两个存在引用的关系($R_1.A_1=R_2.A_2$)作内连接(简称连接)形成新的关系 R'。R' 的头部 $H(R')=H(R_1)\cup H(R_2)$，R' 的主体 $B(R')=\{t\in B(R_1)\times B(R_2)\mid t.A_1=t.A_2\}$，即两个关系的主体作笛卡尔积之后满足引用条件的元组。

为方便起见，将关系连接的主体中的元组称为元组连接。元组连接如果由 m 次关系连接构成，则称为 m 元组连接。单个元组也称为 1 元组连接。需要注意的是，元组连接中的构成元组可以属于同一个关系，即关系的自连接。

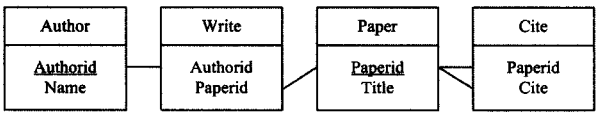
3.2 关系数据库的关键词搜索

定义 2(关键词查询) 关键词查询 Q 由一系列查询关键词 $kw_1,kw_2,\cdots,kw_q$ 以及它们之间的与或逻辑组合所构成。

因为关系之间存在引用，所以关系数据库查询的结果不仅仅局限于某一关系的元组，而可能是多个关系连接之后的元组连接。

定义 3(查询结果) 作为查询结果的元组连接需要满足两个性质：查询相关性以及紧致性。查询相关性是指元组连接需要满足查询要求。紧致性是指无法删除 m 元组连接中的任何元组，使得剩下的 $m-1$ 元组连接依然满足查询相关性。

其中，紧致性要求元组连接不应包含与用户理解无关的信息，从而减轻了用户浏览负担。例如，图 1 所示的是 DBLP 数据库的一个片段。



(a)

Author

rid	Authorid	Name
t1	a1	V. Hristidis
t2	a2	Y. Papakonstantinou
t3	a3	Y. Luo

Paper

rid	Paperid	Title
t4	p1	DISCOVER: Keyword Search in Relational Databases
t5	p2	Efficient IR-style Keyword Search over Relational Databases
t6	p3	Spark: Top-k Keyword Query in Relational Databases

Write

rid	Authorid	Paperid
t7	a1	p1
t8	a1	p2
t9	a2	p1
t10	a2	p2
t11	a3	p3

Cite

rid	Paperid	Cite
t12	p2	p1
t13	p3	p1
t14	p3	p2

(b)

图 1 示例数据库

图 1(a)是数据库的模式图，图中每块表示一个关系，有下划线的属性对应关系的主键，块之间的连线表示它们存在引用，连线两端的属性对应其引用条件。图 1(b)列出了每个关系包含的元组。假设用户输入的查询为“Hristidis AND Databases”，可能的答案包括：t1-t7-t4 以及 t1-t8-t5。虽然 t1-t7-t4-t13-t6 也满足查询条件，但该结果并不满足紧致性，因为可以删除 t13-t6 使得剩余的结果依然满足查询相关性。

4 基于预先索引的关键词搜索

相关工作已经指出，为了提高搜索效率，可以对元组连接进行预先索引，但是现有预先索引方法的结果不满足紧致性要求。为此，本文通过对 1 元组连接到 m 元组连接的预先索引与顺序搜索，使得产生的结果满足紧致性要求。此外，在索引阶段使用模式图而非数据图进行元组连接的枚举，提高了枚举的效率。具体来说，该机制包含如下 3 个阶段，如图 2 所示。

1)根据数据库中的外键引用生成数据库模式图，通过删边与复制操作去除模式图中的环路；

2)利用无环模式图枚举可能的 m 关系连接(m 从 1 到指定值)，将 m 关系连接转换为 SQL 语句在数据库中执行得到 m 元组连接；

3)结合紧致性要求对 m 元组连接进行预先索引与顺序搜索，得到最终的结果。

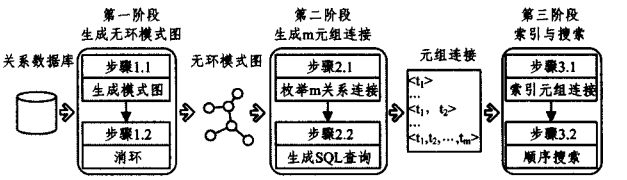


图 2 基于预先索引的关键词搜索的基本流程

4.1 生成无环模式图

数据库的关系通过主外键引用可以构成一个数据库模式图，该图的点对应于关系，边对应于关系间的引用。

定义 4(数据库模式图) 数据库模式图 $G_S=\langle V_S,E_S,L_E\rangle$ 是一个边带标签的无向多图(multigraph)。其中节点 $v_i\in V_S$ 表示一个关系；$(v_i,v_j)\in E_S$ 表示 v_i 与 v_j 间存在主外键引用；L_E 是一个标签函数，为每条边标记其对应的引用条件。

利用该定义，可以为任意给定的数据库生成其模式图。例如，图 1 中的数据库对应的模式图如图 3 所示。

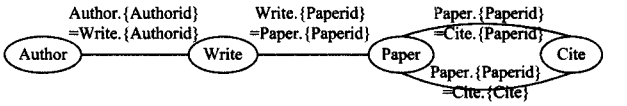


图 3 数据库模式图

在该例子中为了找到论文之间的引用关系，即 Paper-Cite-Paper 这样的结果，需要编写如下的 SQL 语句：SELECT p1.*,Cite.*,p_2.* FROM Paper p1,Cite,Paper p2 WHERE p1.Paperid=Cite.Paperid AND p2.Paperid=Cite.Cite。也就是说，需要通过给表 Paper 取两个不同的名字，才能进行连接操作，找出查询结果。归纳总结可以发现，对于存在环路的数据库模式图，需要消除其环路才能通过关系连接枚举元组连接。

为了消除环路，引入如下两个操作：删边以及复制。删边即删除环路中的边，使得有环的模式图变成无环。复制即对所删边中主键所在的关系进行复制和重命名，并将原先删除的边连接到新的关系，从而保留相应的引用。

例如，对于图 3 所示的数据库模式图，通过删除其中的一条边(如 Paper.{Paperid}=Cite.{Cite})，得到无环图，如图 4(a)所示。虽然消除了环路，但失去了 Paper 以及 Cite 之间的

一个引用，从而无法完成查找论文之间引用关系的任务。为此，需要进行复制操作。Paper 是 Paper.{Paperid}=Cite.{Cite}这条边的主键所在关系，因此对 Paper 进行复制和重命名得到 Paper2，并建立 Cite 与 Paper2 之间的关系。图 4(b)显示了复制之后的结果。

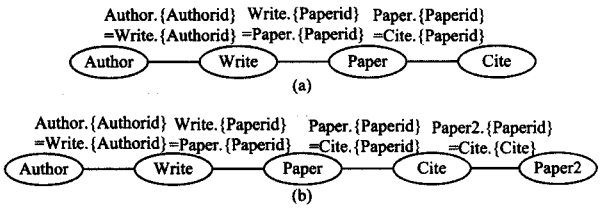


图 4 模式图的消环处理

需要指出的是，当模式图中存在多个环路时，只需将如上的步骤顺序执行多次，便能得到无环模式图。

4.2 生成元组连接

数据库的无环模式图存在各种可能的连通子图，每个连通子图都可以转换为 SQL 查询语句，从而查询得到元组连接。为了穷举可能的元组连接并对其进行索引从而提供快速查询，需要先枚举无环模式图的连通子图。

算法 1 枚举无环模式图的所有 $1,\cdots,m$ 关系连接

输入：无环模式图 G_S 以及整数 m
输出：所有 $1,\cdots,$m 关系连接
1. R 是关系连接的集合
2. Q 是关系集合的队列(每个元素是一个关系集合)
3. foreach node v_i in $V(G_S)$ do
4. if $B(v_i)\neq\emptyset$ and v_i 不是复制得到的关系 then
5. $\{v_i\}$加入 Q;
6. end
7. end
8. while $Q\neq\emptyset$ do
9. C=Q.pull();
10. if size(C)≤m then
11. if C 连通 then
12. 由 C 生成 G_S 的导出子图，将该子图加入 R;
13. end
14. if size(C)<m then/*为生成更长的关系连接做准备*/
15. idx 为 C 中关系在 G_S 中的最大下标;
16. T 为 G_S 中所有下标大于 idx 的关系构成的集合;
17. foreach t in T do
18. 将 C∪{t}加入 Q;
19. end
20. end
21. end
22. end
23. return R

定义 5(关系连接图) 关系连接图是无环模式图的连通子图，当关系连接图的节点数为 m 时，称其为 m 关系连接图(或 m 关系连接)。

算法 1 给出了 $1,\cdots,m$ 关系连接的生成算法。该算法的输入为模式图 G_S 以及参数 m，算法首先将包含元组的关系加入 Q 中(第 3－7 行)，接着枚举模式图节点集的所有大小不超过 m 的子集(第 14－20 行)，如果子集在 G_S 中是连通的，则利用其导出一个 G_S 的子图，并加入结果集中(第 11－13 行)。很容易看出该算法的最大时间消耗在于子集枚举，该步骤的时间复杂度为 $O(n(n-1)\cdots(n-m+1))$。另外，为了防止内容重复，作为种子的关系只能是原始关系，而所有通过复制重命名得到的镜像关系不在其中(第 4 行)。

为了从关系连接生成 SQL 查询语句，只需将引用条件作为 SQL 查询条件进行关系连接操作。利用数据库管理系统执行该查询语句，便能得到元组连接。

例如，对于 Paper-Cite-Paper2 的关系连接，可以转换得到如下的 SQL 语句：SELECT Paper.*, Cite.*, Paper2.* FROM Paper, Cite, Paper2 WHERE Paper.Paperid=Cite.Paperid AND Paper2.Paperid=Cite.Cite。查询关系数据库便得到诸如 t5-t12-t4 等元组连接。

4.3 预先索引与顺序搜索

为了提高搜索效率，对这些 $1,\cdots,m$ 元组连接进行预先索引。首先从每个元组连接中提取出各个属性值的文本信息，从而构成关于该元组连接的一个虚拟文档，然后利用文档搜索中成熟的倒排索引技术对虚拟文档建立索引，从而利用该索引支持关键词查询。

但是这样的搜索方式无法保证查询结果满足紧致性要求，因为 i 元组连接可能已经满足查询条件，但包含 i 元组连接的更长的结果也能搜索到。为此需要对如上的方法进行改进。具体来说，在索引时，除了虚拟文档，还需索引构成该元组连接的元组编号以及元组连接的长度。

例如，对于 t1-t8-t5 这样的元组连接，需要增加一个域 f_t，其值为{t1,t8,t5}，用来保存元组连接所来自的元组信息，此外增加一个域值为 3 的 f_l。

在搜索时，为了保证紧致性，即当 m 元组连接满足查询要求时，所有包含该元组连接的结果都不再考虑。因此，首先搜索长度为 1 的元组连接($f_l=1$)，并记录下结果来自的元组信息；然后搜索长度为 2 的元组连接，并删除包含 1 元组信息的结果。也即，如果该轮某结果对应的元组信息为 C，下一轮需要删除所有 f_t 的值包含 C 的结果。

5 复杂查询处理

为了降低问题的复杂度，现有的关键词查询系统都对查询的表达能力做了限制。例如，大部分系统都要求查询结果包含所有的关键词。但实际使用中，简单的关键词查询往往得不到准确结果，同时由于结果必须包含所有关键词，当关键词较多时用户又得不到任何查询结果。对于这种局限性，需要检索系统能够提供更加接近于 SQL 的查询表达能力。例如，除了简单关键词之外，还需要支持属性值查询以及关键词之间的与或逻辑组合。

属性值查询包含属性值约束、属性值范围约束以及属性值提问等，见表 1 的第二行。其中属性值(范围)约束是指查询结果必须包含给定的属性，且属性的取值需要包含术语(属性值约束)或者在某一区间内(属性值范围约束)。属性值提问是指查询特定属性的值，即查询结果能够直接给出指定属性的值。

为了支持这些查询，设计了基于域(field)的索引结构，并借助倒排索引工具(如 Lucene)对其建立索引。

首先，为了支持一般的关键词查询，即查找包含特定术语的元组连接，将元组连接作为一个虚拟文档，设置域为 f_{desc}，域的值为各属性值的文本信息。此外，为了保证紧致性，还增加了 f_t、f_l 两个域(见 4.3 节)。

其次，为了支持属性值(范围)约束，还需为元组连接的每

个属性设置一个域，并将属性值加入到域的值中。这样当进行属性值约束查询时（即查找属性是否包含某个术语），可以通过指定属性对应的域查找该属性包含某术语的元组连接。

再者，为了支持属性值提问查询，增加一个域 f_{attrs}，该域保存元组对应的所有属性名，通过该域可以快速定位到包含某属性名的所有元组。

最后，将这些不同的域进行逻辑组合，以支持逻辑查询。表 1 列出了系统支持的查询语法规则。可以看出，该查询语法具有较强的表达能力，因此系统具有较为宽泛的应用场景。

表 1　关键词查询支持的语法规则

〈术语〉::=〈单词〉|〈短语〉
〈属性值查询〉::=〈属性名〉":"〈术语〉|〈属性名〉":""["〈术语〉TO〈术语〉"]"|〈属性名〉":""{"〈术语〉TO〈术语〉"}"|〈属性名〉":?"
〈关键词〉::=〈术语〉|〈属性值查询〉
〈查询〉::=〈关键词〉|〈关键词〉[〈操作符〉]〈关键词〉|"("〈关键词〉[〈操作符〉]〈关键词〉")"
〈操作符〉::="AND"|"OR"

6　系统评估

本文使用两个公开数据集进行评价：DBLP 数据集（http://dblp.uni-trier.de/xml/）以及 IMDB 数据集（http://www.imdb.com/interfaces）。DBLP 数据集包含了 1.5GB 的 XML 数据，通过自行开发的转换工具将 XML 导入到 ORACLE 中，建立一个包含 4 个关系的 DBLP 关系数据库。与此类似，通过相应的工具对 IMDB 数据集 10GB 的文本数据解析后导入 ORACLE，建立 IMDB 关系数据库，其包含 21 个关系。

6.1　预处理

预处理包括了生成无环模式图、元组连接以及预先索引等操作。为了验证预处理时间、空间随关系连接的次数 m 变化的情况，分别进行了不超过 1、不超过 2、不超过 3 次关系连接，并索引相关的元组连接。图 5 与图 6 分别显示了预处理的时间和空间的消耗情况。可以看出，随着关系连接数的增加，时间和空间都有较大的增长，并且由于 IMDB 无论在数据量还是关系的数量上都远大于 DBLP，因此其索引所需的时间更多、空间更大。此外，对不超过 3 次的元组连接进行索引，DBLP 的时间为 4437s，空间为 7.72GB；IMDB 的时间和空间分别为 52339s 和 87.23GB。可以看出，面对大规模数据集时，处理的时间和空间消耗是可以接受的。

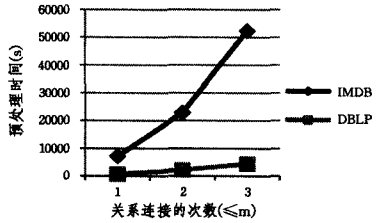


图 5　预处理时间随关系连接次数的变化

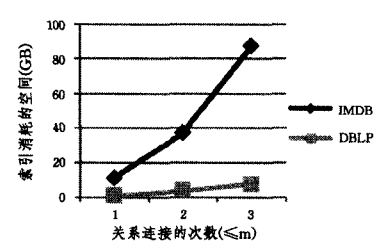


图 6　预处理空间（索引）随关系连接次数的变化

6.2　查询性能

从数据集中随机挑选一些单词作为查询关键词，用它们组合构成包含 2～6 个关键词的查询。图 7 示出了查询时间（使用不超过 3 的元组连接索引）随不同数量关键词查询的变化情况。实验表明，随着查询关键词的增加，查询时间总体呈上升趋势，但增长较慢，不会对响应时间造成太大的影响。

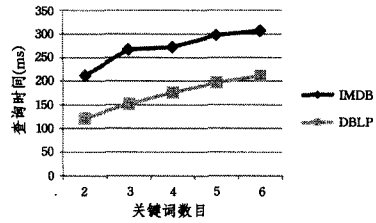


图 7　查询时间随关键词数量的变化

由于现有方法，尤其是基于预先索引的方法，都没有可供下载的算法，因此，通过查阅相关文献中的实验结果进行比较，如表 2 所列。比较发现：对于大小为 500MB 的数据集，EKSO[11] 查询返回完整结果的平均响应时间为 21～24s；ITREKS[12] 使用小规模的 DBLP 数据集（102MB）的响应时间达到 2～7s；Retune[13] 使用 420MB 的 DBLP 数据，其返回前 20 个结果的平均响应时间就已经达到 1s。比较这些公开的实验数据可以发现，相比于现有的基于预先索引的方法，本文所提方法具有更快的响应速度。

表 2　不同算法的查询响应时间

算法	数据集大小	查询时间
EKSO[11]	500MB	21～24s
ITREKS[12]	102MB	2～7s
Retune[13]	420MB	1s（前 20）
本方法	1.5GB	0.8s
本方法	10GB	0.3s

结束语　为了提高搜索效率，本文提出了一种基于预先索引的关系数据库关键词搜索方法。相比于已有基于预先索引的方法，本方法在索引的效率以及结果紧致性两个方面进行了改进。为了支持复杂查询，本文提出了基于域的索引结构。通过实现该系统，并进行相关评估，证明了方法的可行性。该方法的关键特征以及本文贡献总结如下：

1）针对现有预先索引方法在索引效率上存在的问题，提出了基于模式图的元组连接枚举技术。所提方法基于的模式图相比于数据图规模较小，且能充分利用 RDBMS 的查询能力，所以具有较高的索引效率。

2）针对现有预先索引方法在结果紧致性上存在的问题，提出了 1 到 m 元组连接的预先索引与顺序搜索机制。由于消除了结果中的无关信息，减轻了用户的浏览负担。

3）针对支持复杂查询的需求，提出了基于域的索引结构。该结构除了支持简单的关键词查询之外，还支持属性值查询以及支持关键词之间的与或逻辑组合。

4）在两个大规模的标准关系数据库上开展实验，验证了该系统的可行性，相比于现有基于预先索引的系统，本方法具有更高的搜索效率。

未来需要针对该领域出现的新的发展方向和形式（包括对象搜索[18,19]、自然语言问答[20,21]等）进行跟踪和研究。

参 考 文 献

[1] Agrawal R, Aliamaki A, Bernstein P A, et al. The Claremont Report on Database Research, 2008 [C/OL]. http://db. cs. berkeley. edu/claremont

[2] Bhalotia G, Hulgeri A, Nakhe C, et al. Keyword Searching and Browsing in Databases Using BANKS[C]// Proc. of the 18th Int'l Conf. on Data Engineering, 2002(ICDE 2002). San Jose: IEEE Computer Society Press, 2002: 431-440

[3] Agrawal S, Chaudhuri S, Das G. DBXplorer: A System for Keyword-based Search over Relational Databases[C]// Proc. of the 18th Int'l Conf. on Data Engineering, 2002(ICDE 2002). San Jose: IEEE Computer Society Press, 2002: 5-16

[4] Hristidis V, Papakonstantinou Y. DISCOVER: Keyword Search in Relational Databases[C]// Proc. of the 28th Int'l Conf. on Very Large Data Bases, 2002(VLDB 2002). Hong Kong: Morgan Kaufmann Publishers, 2002: 670-681

[5] Hristidis V, Papakonstantinou Y. Efficient IR-style Keyword Search over Relational Databases[C]// Proc. of the 29th Int'l Conf. on Very Large Data Bases, 2003(VLDB 2003). Berlin: Morgan Kaufmann Publishers, 2003: 850-861

[6] Kacholia V, Pandit S, Chakrabarti S, et al. Bidirectional Expansion for Keyword Search on Graph Databases[C]// Proc. of the 31st Int'l Conf. on Very Large Data Bases, 2005(VLDB 2005). Trondheim: ACM Press, 2005: 505-516

[7] He Hao, Wang Hai-xun, Yang Jun, et al. Binks: Ranked Keyword Searches on Graphs[C]// Proc. of the 2007 ACM SIGMOD Conf. on Management of Data, 2007(SIGMOD 2007). Beijing: ACM, 2007: 305-316

[8] Li Guo-liang, Ooi B C, Feng Jian-hua, et al. EASE: An Effective 3-in-1 Keyword Search Method for Unstructured, Semi-structured and Structured Data[C]// Proc. of the 2008 ACM SIGMOD Conf. on Management of Data, 2008(SIGMOD 2008). Vancouver: ACM, 2008: 903-914

[9] Luo Yi, Lin Xue-min, Wang Wei, et al. Spark: Top-k Keyword Query in Relational Databases[C]// Proc. of the 2007 ACM SIGMOD Conf. on Management of Data, 2007(SIGMOD 2007). Beijing: A. CM, 2007: 115-126

[10] Ding B, Yu J X, Wang S, et al. Finding Top-k Min-cost Connected Trees in Databases[C]// Proc. of the 23rd Int'l Conf. on Data Engineering, 2007(ICDE 2007). Istanbul: IEEE Computer Society Press, 2007: 836-845

[11] Su Q, Widom J. Indexing Relational Database Content Offline for Efficient Keyword-based Search[C]// Proc. of the 9th Int'l Data Engineering and Applications Symposium, 2005. Montreal: IEEE Computer Society Press, 2005: 297-306

[12] Zhan Jiang, Wang Shan. ITREKS: Keyword Search over Relational Database by Indexing Tuple Relationship[C]// Proc. of the 12th Int'l Conf. on Database Systems for Adavanced Applications, 2007. Bangkok: Springer-Verlag, 2007: 67-78

[13] Li Guo-liang, Feng Jian-hua, Zhou Li-zhu. Retune: Retrieving and Materializing Tuple Units for Effective Keyword Search over Relational Databases[C]// Proc. of the 27th Int'l Data on Conceptual Modeling, 2008. Barcelona: Springer-Verlag, 2008: 469-483

[14] Dixon P. Basics of Oracle Text Retrieval [J]. Bulletin of the Technical Committee on Data Engineering, 2001, 24(4): 11-14

[15] Maier A, Simmen D E. DB2 Optimization in Support of Full Text Search[J]. Bulletin of the Technical Committee on Data Engineering, 2001, 24(4): 3-6

[16] Hamilton JR, Nayak TK. Microsoft SQL Server Full-text Search [J]. Bulletin of the Technical Committee on Data Engineering, 2001, 24(4): 7-10

[17] Reich G, Widmayer P. Beyond Steiner's Problem: A VLSI Oriented Generalization[C]// Proc. of the 15th Int'l Workshop on Graph-Theoretic Concepts in Computer Science, 1989. Castle Rolduc: Springer-Verlag, 1989: 196-210

[18] Nie Z, Zhang Y, Wen J, et al. Object-Level Ranking: Bringing Order to Web Objects[C]// Proc. of the World Wide Web. 2005

[19] Zhang J, Shao R J, Zeng Y M. Research on Object-level Information Retrieval over Relational Databases [J]. Computer Science, 2012, 39(1): 142-147(in Chinese)
张俊, 邵仁俊, 曾一鸣. 对象级别的关系数据库信息检索技术研究[J]. 计算机科学, 2012, 39(1): 142-147

[20] Owda M, Bandar Z, Crockett K. Conversation-Based Natural Language Interface to Relational Databases[C]// 2007 IEEE/WIC/ACM/ International Conferences on Web Intelligence and Intelligent Agent Technology Workshops. 2007: 363-367

[21] Li H, Tian J W, Wang Y Y, et al. Ontology-based Natural Language Interface to Relational Databases [J]. Computer Science, 2010, 37(6): 200-205(in Chinese)
李虎, 田金文, 王缓缓, 等. 基于 Ontology 的数据库自然语言查询接口的研究[J]. 计算机科学, 2010, 37(6): 200-205

(上接第 166 页)

[8] Maric I, Yates R D. Bandwidth and Power Allocation for Cooperative Strategies in Gaussian Relay Networks[J]. IEEE Transactions on Information Theory, 2010, 56(4): 1880-1889

[9] Haohao Q, Xiang C, Yin S, et al. Optimal Power Allocation for Joint Beamforming and Artificial Noise Design in Secure Wireless Communications[C]// 2011 IEEE International Conference on Communications Workshops (ICC). 2011: 5-9

[10] Zou Y, Zhu J, Wang X, et al. Improving physical-layer security in wireless communications using diversity techniques [J]. Network, IEEE, 2015, 29(1): 42-48

[11] Ning Z, Ning L, Nan C, et al. Towards secure communications in cooperative cognitive radio networks[C]// 2013 IEEE/CIC International Conference on Communications in China (ICCC). 2013: 12-14

[12] Jain A, Kulkarni S R, Verdu S. Energy Efficiency of Decode-and-Forward for Wideband Wireless Multicasting[J]. IEEE Transactions on Information Theory, 2011, 57(12): 7695-713

[13] Ning K, et al. Performance Comparison Among Conventional Selection Combining[C]// 2009 IEEE International Conference on Optimum Selection Combining and Maximal Ratio Combining (ICC'09). 2009: 14-18