

## MLCEA: 一种基于实体的XML关键字查询语义

黄静<sup>1</sup> 徐俊劲<sup>1</sup> 周军锋<sup>1,2</sup> 孟小峰<sup>1</sup>

<sup>1</sup>(中国人民大学信息学院 北京 100872)

<sup>2</sup>(燕山大学计算机科学与工程系 秦皇岛 066004)

(huangjingruc@ruc.edu.cn)

**摘要** 关键字查询方法为用户提供了友好便捷的查询方式,如何定义有效的查询语义是 XML 关键字查询要解决的基本问题.本文重点研究了 XML 关键字查询的语义,以实体作为基本语义单元,提出了最低公共实体祖先 LCEA 的概念,一个 LCEA 结点是描述现实世界完整信息单元的实体对象.在 LCEA 的基础上,提出了有意义的最低公共实体祖先 MLCEA,在为用户提供简单易用查询方式的同时,避免了漏解和返回无意义结果的现象.提出了计算 LCEA 和 MLCEA 的有效算法.最后根据不同的评价标准在 3 个数据集上通过丰富的实验数据对本文提出的方法进行了实验验证.

**关键词** XML; 关键字查询; 语义; LCEA; MLCEA

中图法分类号 TP391

## MLCEA: An Entity Based Semantics for XML Keyword Search

Huang Jing<sup>1</sup>, Xu Junjin<sup>1</sup>, Zhou Junfeng<sup>1,2</sup>, and Meng Xiaofeng<sup>1</sup>

<sup>1</sup>(School of Information, Renmin University of China, Beijing 100872)

<sup>2</sup>(Department of Computer Science and Engineering, Yanshan University, Qinhuangdao 066004)

**Abstract** Keyword search method provides users with a friendly way to query XML data, and the key problem in XML keyword search is defining effective semantics which is used to determine the data fragments returned to users. In this paper, we focus on providing users with an effective semantics, which is as important as efficiency. We propose a new semantics, LCEA (Lowest Common Entity Ancestor), based on the basic semantic unit, i.e. entity, to represent the result of XML keyword search. We then propose the notion of MLCEA (Meaningful Lowest Common Entity Ancestor) to accurately answer keyword queries over XML documents. Besides the two semantics, we also present two efficient algorithms to compute LCEA Set and MLCEA Set. The rich experimental results on various datasets according to different metrics verify the effectiveness of LCEA and MLCEA, and the high efficiency of our algorithms.

**Keywords** XML; keyword search; semantics; LCEA; MLCEA

### 1 引言

XML 已经成为 Internet 上数据发布和信息交换事实上的标准,大量的数据都以 XML 的形式来表示和传输,如何有效获取所需的信息已经成为学术界近期研究的一个热点问题. XQuery[1]可以精确表达用户的查询语义,但其正确使用的前提条件是用户必须了解

数据的模式信息并掌握复杂的查询语言语法.对于普通用户来说,很多情况下难以顺利得到所需的数据.关键字查询为用户提供了友好便捷的查询方式,用户不需要了模式信息和掌握复杂的查询语法,只需提交简单的关键字即可.

收稿日期:

基金项目: 国家自然科学基金项目(60573091);国家 863 计划(2007AA01Z155);教育部新世纪优秀人才支持计划基金项目

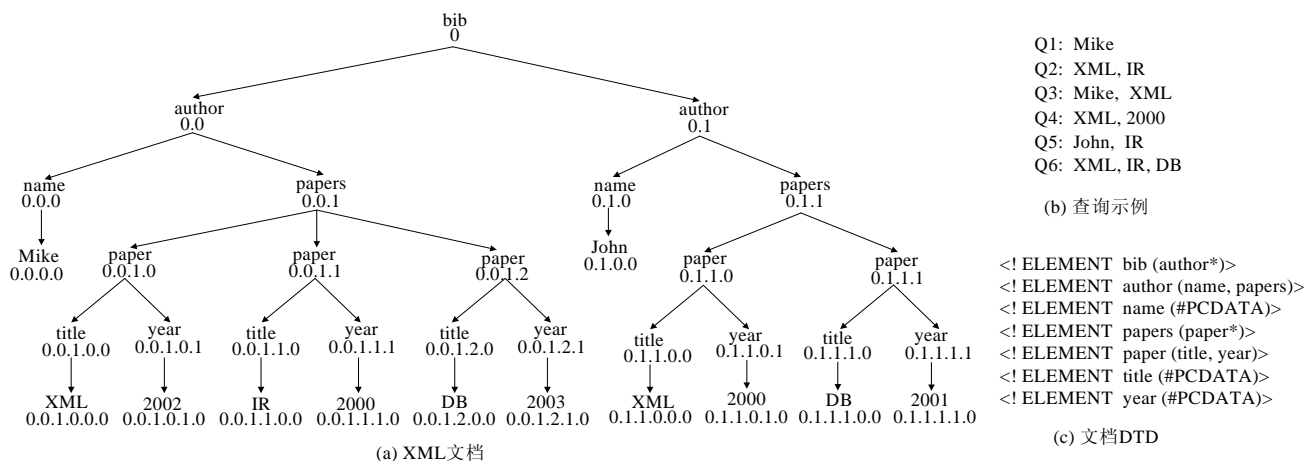


图1 带有 Dewey 编码的 XML 文档、查询示例和文档 DTD

对 XML 关键字查询来说,如何定义有效的查询语义是其要解决的基本问题.已有的基于树模型的关键字查询方法都以 LCA(Lowest Common Ancestor)为基础形成的.这种语义主要基于关键字接近的思想[2],认为包含所有关键字的最小数据片断就是一个好的结果. SLCA(Smallest LCA)[3]在 LCA 的基础上要求返回的 LCA 结点之间没有祖先后代关系,因此,只有部分 LCA 才是满足条件的 SLCA. XSearch[4]提出的结点相关联的概念和文献[5]提出的 VLCA(Valuable LCA)语义要求从匹配关键字的结点到它们的 LCA 的路径上不存在相同标签名的结点.由于这些语义都是仅考虑 XML 文档的结构特点,在实际使用中不可避免存在如下 3 个问题:(1)返回无意义的结果.(2)丢失有意义的结果.(3)无法处理单个关键字的查询.

本文考虑到 XML 文档中不同元素代表的信息类型的区别,将 XML 文档的元素进行分类,以实体作为基本语义单元来定义 XML 关键字的查询语义,提出了最低公共实体祖先(Lowest Common Entity Ancestor, LCEA)的语义.相比于 LCA, LCEA 代表的是具有实际意义的实体对象,能够自然有效地回答关键字查询,它不仅过滤了不相关的结果,而且不会丢失有意义的结果.在处理单个关键字查询的情况,LCEA 具有明显的优势.此外,通过分析关键字查询的特点,在 LCEA 的基础上我们提出了有意义的最低公共实体祖先(Meaningful Lowest Common Entity Ancestor, MLCEA)的语义,能够过滤一些特殊的无意义的 LCEA 结点.

本文的贡献可以总结为以下几点:

1. 提出了基于实体的查询语义 LCEA 和 MLCEA,有效的提高关键字查询的质量.
2. 提出了基于 LCEA 和 MLCEA 语义的有效查询算法.
3. 进行了全面的实验比较,实验结果表明本文的语义相比较已有的工作更加有效,所提出的算法同

样能高效的运行.

## 2 查询语义

本节将介绍 XML 关键字查询的一些基本概念以及文档结点分类的方法,给出 LCEA 以及 MLCEA 语义的定义,并以实例说明它们的有效性.

### 2.1 相关概念

**XML 数据模型:** 一个 XML 文档可以看成是带标签的有向树  $D(r, V, E)$ ,  $V$  表示结点集合,  $E$  表示边的集合,  $r$  是根结点.如图 1(a)所示,树中的结点表示文档中的元素,属性或者文本值,结点上的标签对应元素的标签,树中结点之间的祖先后代关系表示了元素之间的嵌套关系.我们使用 Dewey 编码[6]来唯一的标识每个结点,图 1(a)就是带有 Dewey 编码的 XML 文档.如果  $v$  是非叶结点,  $\text{tag}(v)$  表示它的标签,如果  $v$  是叶结点,  $\text{val}(v)$  表示它的值.

**定义 1:(关键字匹配集合).** 对给定的 XML 文档  $D(r, V, E)$  和关键字  $k$ ,用符号  $\text{KM}(k)$  表示  $D$  中所有匹配关键字  $k$  的结点集合,  $\text{KM}(k) = \{v \mid v \in V, k = \text{tag}(v) \text{ 或者 } k = \text{val}(v)\}$ .

### 2.2 结点分类

一个 XML 文档可以看成是由一些现实世界中的实体,属性,实体间关系组成,这与关系数据库中的 ER 模型相似.例如图 1(a)所示的 XML 文档中,有实体 **author** 和 **paper**,每个实体都有各自的属性, **author** 有属性 **name**.实体之间的关系由连接实体间的路径表示,例如一个 **author** 可以有一个或者多个 **paper** 后代结点.

根据 XML 文档中结点代表的语义,我们将结点分成实体结点、属性结点、值结点和连接结点四类:

1. 值结点是在文档 DTD 中不存在,但是在实际文档中存在于叶结点的那些结点.
2. 属性结点是只有一个值结点作为孩子结点的那

些结点.

3. 实体结点是对应到文档 DTD 中带\*号或者+号并且不是属性结点的那些结点.
4. 如果一个结点不是实体结点、属性结点也不是值结点,那么它是连接结点.一个连接结点的孩子结点可以是值结点外的其他结点类型.

考虑图 1(a)中的 XML 文档和图 1(c)中的 DTD, 结点 author 在 DTD 中是带\*号的结点并且不是属性结点,因此是实体结点. name 是 author 的属性结点.同样,paper 也是实体结点. bib 和 papers 是连接结点.图 1(a)中“Mike”结点是值结点.

### 2.3 LCEA 语义

属性结点和值结点都不能代表完整的语义信息, 实体结点才是基本的有意义语义单元,因为它代表了现实世界中的实体,是用户所需的信息.基于以上观察,我们提出最近公共实体祖先 LCEA 的概念.

#### 定义 2:(Lowest Common Entity Ancestor, LCEA)

给定  $m$  个结点  $n_1, n_2, \dots, n_m$ , 结点  $v$  是它们的最低公共实体祖先,当且仅当: (1)  $v$  是实体结点,且(2)  $v$  是  $n_i$  ( $1 \leq i \leq m$ ) 的祖先结点,且(3)不存在实体结点  $u$ ,  $u$  是  $v$  的后代,且  $u$  是  $n_i$  ( $1 \leq i \leq m$ ) 的祖先结点.则  $v = \text{LCEA}(n_1, n_2, \dots, n_m)$ .

LCEA 的基本思想是,给定关键字查询  $K = \{k_1, k_2, \dots, k_m\}$ , 对于结点  $n_i \in \text{KM}(k_i)$  ( $1 \leq i \leq m$ ), 如果结点  $v = \text{LCEA}(n_1, n_2, \dots, n_m)$ , 那么  $v$  是包含所有查询关键字的实体结点,并且是查询  $K$  的一个解.例如,对于查询 Q1, Mike(0.0.0.0) 是 LCA 结点,但不是 LCEA 结点,因为它不是实体结点,实体结点 author(0.0) 才是对应的 LCEA 结点,并且是这个查询的解.

**定义 3:(LCEA 集合)** 给定关键字查询  $K = \{k_1, k_2, \dots, k_m\}$  和 XML 文档  $D(r, V, E)$ , 查询  $K$  在  $D$  上的 LCEA 集合表示成  $\text{LCEASet}(K, D) = \{v \mid v = \text{LCEA}(n_1, n_2, \dots, n_m), n_i \in \text{KM}(k_i), 1 \leq i \leq m\}$ .

### 2.4 MLCEA 语义

**定义 4:(值结点属性).** 值结点  $u$  对应的属性用  $\text{Attr}(u)$  表示,若  $v$  是  $u$  对应的属性结点,那么  $\text{Attr}(u) = \text{Tag}(v)$ , 若  $u$  是属性结点,则  $\text{Attr}(u) = \text{Tag}(u)$ .

例如结点 XML(0.0.1.0.0.0) 的属性是 title.

**定义 5:(同名实体)** 对于实体结点  $u$  和  $v$ , 如果  $\text{Tag}(u) = \text{Tag}(v)$ , 那么  $u$  和  $v$  是同名实体,记作:  $u \sim v$ .

例如图 1(a) 中实体结点 paper(0.0.1.0) 和 paper(0.0.1.1) 就是同名实体.其实,考虑更一般的情况,这里同名实体可以理解成是逻辑等价的实体,标签名不一定相同,比如 article 和 paper, 就是逻辑等价的实体,这种逻辑等价性可以使用 ontology 中技术来判断,由于逻辑等价的实体常常表现为标签名相同,因此,本文

用同名实体来表示逻辑等价实体的概念.

一般来说,用户提交查询的时候,会使用两个同名实体的相同属性的关键字进行查询,如 paper 的 title 属性,而如果两个关键字对应相同实体的不同属性,则一般来说,用户需要的是同时满足这两个关键字的一个实体对象.对于图 1(a)中的 XML 文档,如果用户想找 2000 年发表的关于 XML 的文章,会输入如 Q4 的查询,可以求出两个 LCEA 结点:author(0.0) 和 paper(0.1.1.0). 第二个显然是符合用户需求的结果.第一个结果中, XML 和 2000 分别属于不同的文章,一篇文章的标题是 XML,另一篇文章发表于 2000 年,这个结果与用户的查询初衷是不符的.根据以上观察,我们提出了有意义的最低公共实体祖先 MLCEA 的概念.

#### 定义 6:(Meaningful Lowest Common Entity Ancestor, MLCEA)

给定  $m$  个结点  $n_1, n_2, \dots, n_m$ ,  $v = \text{LCEA}(n_1, n_2, \dots, n_m)$ .  $v$  是一个有意义的 LCEA, 即  $v = \text{MLCEA}(n_1, n_2, \dots, n_m)$ , 当且仅当不存在结点  $n_i, n_j$  ( $1 \leq i < j \leq m$ ) 使得:  $\text{Attr}(n_i) \neq \text{Attr}(n_j) \wedge \text{LCEA}(n_i) \sim \text{LCEA}(n_j) \wedge \text{LCEA}(n_i) \neq \text{LCEA}(n_j)$  成立.

**定义 7:(MLCEA 集合)** 给定关键字查询  $K = \{k_1, k_2, \dots, k_m\}$  和 XML 文档  $D(r, V, E)$ , 查询  $K$  在  $D$  上的 MLCEA 集合表示成  $\text{MLCEASet}(K, D) = \{v \mid v = \text{MLCEA}(n_1, n_2, \dots, n_m), n_i \in \text{KM}(k_i), 1 \leq i \leq m\}$ .

MLCEA 集合是 LCEA 集合的一个子集,排除了 LCEA 集合中关键字对应多个同名实体的不同属性的情况.例如对于查询 Q4, LCEA 结点 author(0.0) 不是 MLCEA 结点,因为关键字 XML 和 2000 分别属于两个同名实体(paper(0.0.1.0) 和 paper(0.0.1.1)) 的不同属性(title 和 year).可以看出,MLCEA 是比 LCEA 更加精确的语义.

## 3 查询算法

在一个 XML 文档上给定关键字查询  $K = \{k_1, k_2, \dots, k_m\}$ , 根据定义 3 和定义 7, 需要求出 LCEA 集合或 MLCEA 集合.本节将给出计算 LCEA 集合和 MLCEA 集合的有效算法

### 3.1 计算 LCEA 集合算法

查询  $K$  在文档  $D$  上的 LCEA 集合由包含所有查询关键字的 LCEA 组成.我们计算 LCEA 的算法基于以下性质:

**性质 1:**  $\text{LCEA}(n_1, n_2, \dots, n_m) = \text{LCEA}(\text{LCA}(n_1, n_2, \dots, n_m))$ .

**性质 2:**  $\text{LCEA}(n_1, n_2, \dots, n_m)$

$$= \text{LCEA}(\text{LCEA}(n_1, n_2, \dots, n_{m-1}), n_m).$$

根据性质 1 可以在计算 LCA 的基础上求 LCEA.

首先求得一个 LCA,如果是实体结点,那么它就是 LCEA;如果不是实体结点,那么从它出发向上直到根结点的路径上第一个遇到的实体结点就是 LCEA,如果不存在这样的祖先结点,则该 LCA 不是 LCEA.

性质 2 是查询求解思路.求解多个结点的 LCEA 时,可以先求部分结点的 LCEA,再用求得的 LCEA 与剩下的结点求 LCEA,从而加速求解.

我们采用倒排索引,将每个关键字匹配的结点索引在一起,并根据它们的 Dewey 编码从小到大排序.每个关键字按照匹配集合中结点的个数从小到大排序.另外,我们还有一个从结点到它最近实体祖先结点的索引.通过这个索引,可以找到任意结点的最近实体祖先.对于查询中每个关键字,根据倒排索引,就能找到每个关键字匹配集合,并且它们按照包含结点的个数从小到大有序.

算法1: 计算LCEA集合算法

```

KeywordSearch(K,D)
/* 输入:查询关键字K={k1,k2,...,km}, XML文档D */
/* 输出:LCEASet */
1 LCEASet=null;
2 getSortKMList();/* KM(k1),KM(k2),...,KM(km)有序 */
3 if (|K|=1) then
4   for v ∈ KM(k1)
5     EN=findEntity(v);
6     LCEASet=LCEASet ∪ EN;
7 else
8   LCEASet=getLCEAS(KM(k1),KM(k2));
9   for KM(k3) to KM(km)
10    if(LCEASet!=null) then
11      LCEASet=getLCEAS(LCEASet,KM(ki));
12 return LCEASet;
getLCEAS(S1,S2)
/* 输入:S1,S2为结点集合 */
/* 输出:LCEASet */
13 LCEASet=null;
14 for u ∈ S1
15   for v ∈ S2
16     L=getLCA(u, v);
17     LE=findEntity(L);
18     if(LE!=null) then LCEASet=LCEASet ∪ LE;
19 return LCEASet;

```

图 2 计算 LCEA 集合算法

图 2 中的算法 1 是计算 LCEA 集合的算法.函数 KeywordSearch(K, D)用来计算查询 K 在文档 D 中的 LCEA 集合.函数 getLCEAS(S<sub>1</sub>, S<sub>2</sub>)用来计算两个匹配集合 S<sub>1</sub>和 S<sub>2</sub>的 LCEA 集合.

首先设置初始 LCEA 集合为空,根据倒排索引,获得有序的所有关键字匹配集合(1~2 行).如果查询是单个关键字,则取匹配集合中的每个结点,调用 findEntity 函数找到它的最近实体祖先放入 LCEA 集合中(3~6 行).函数 findEntity(v)利用结点到最近实体祖先索引,返回结点 v 的最近实体祖先.

如果查询包含多个关键字,首先取匹配集合最小的两个关键字,调用 getLCEAS 函数计算包含这两个关键字的 LCEA 集合(行 8),再依次计算包含其它关键字的 LCEA 集合(9~11 行),最后返回结果(行 12).函数 getLCEAS(S<sub>1</sub>, S<sub>2</sub>)计算 LCEA 集合时,分别取 S<sub>1</sub>和 S<sub>2</sub>中的一个结点,求出所有结点组合的 LCEA 并返回(13~19 行).在计算两个结点的 LCEA 时,我们根据性质 1,先计算两个结点的 LCA,再调用 findEntity 函数找这个 LCA 的最近实体祖先,就是这两个结点的 LCEA(16~18 行).

### 3.2 计算 MLCEA 集合算法

MLCEA 集合是 LCEA 集合的一个子集,排除了 LCEA 集合中关键字对应多个同名实体的不同属性的情况.因此我们只要在算法 1 的 getLCEAS 函数中第 18 行求解两个结点 LCEA 的时候判断这个 LCEA 是否是 MLCEA 即可,只有 MLCEA 才放入 LCEA 集合中,这样得出的 LCEA 集合就是 MLCEA 集合,因为包含的每个 LCEA 都是 MLCEA.根据定义 6,如果两个结点对应两个不同的同名实体的不同属性,那么它们的 LCEA 就不是 MLCEA,其他情况则 LCEA 即是 MLCEA.

## 4 实验与分析

为了验证 LCEA 和 MLCEA 的有效性,我们将其与已有工作中提到的 LCA,SLCA 以及 XSearch 共 3 种语义进行了对比.

所有实验都是在处理器为 Intel 双核 2.0GHz,内存为 1GB,操作系统为 Windows XP Professional 机器上运行,实现语言为 Java.我们在三个数据集上进行了实验,分别是 SIGMOD Record[7], XMark[8]和 DBLP[9],大小分别为 500KB,115MB,130MB.

每个查询包含的关键字个数为 1 至 5 个.为了测试各种语义在关键字不同选择率的情况下的性能,我们将选择不同频率的关键字进行实验.根据关键字在文档中出现的频率不同,分为低频、中频、高频关键字,对应的频率分别为:1-100,101-1000,大于 1000.

我们将关键字查询转化成对应的 XQuery 查询,并在已有的 XQuery 查询引擎 X-Hiv/DB[10]上执行这些查询,将得到的结果作为标准结果,再与不同方法返回的结果进行比较,计算查准率和查全率.

### 4.1 查询质量

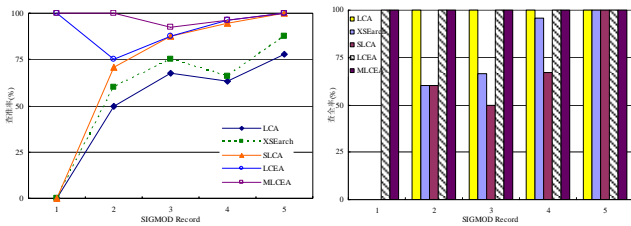


图3 SIGMOD Record 上查准率和查全率

我们通过比较 LCA, SLCA, XSearch, LCEA 和 MLCEA 的查准率和查全率来比较它们的查询质量. 各种语义执行 5 个查询, 分别包含 1 至 5 个关键字. 这里只给出在 SIGMOD Record 上结果, 如图 3 所示. 其它数据集上结论类似.

从图 3 可以看出, LCEA 和 MLCEA 在各种情况下查准率都比其它三种方法高, MLCEA 的查准率基本上可以达到 90% 以上. 因为它们返回的都是实体结点, 更符合用户使用 XQuery 进行查询求解的实际情况. 而 MLCEA 在 LCEA 的基础上排除了同名实体不同属性的情况, 因此比 LCEA 具有更高的查准率. SLCA,

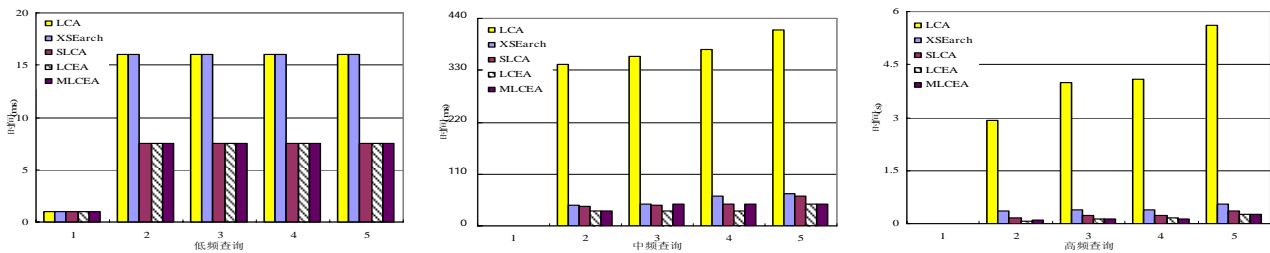


图4 DBLP上执行时间

XSearch 会丢失一些有意义的结果, 因此查全率比其他方法更低. LCA 的查全率能达到 100% 是以降低查准率为代价的. LCEA 和 MLCEA 在各种情况下都能将所有相关的结果返回, 因此查全率都为 100%.

最后, 从整体来看, LCEA 和 MLCEA 查准率和查全率比前三种方法都具有更好的效果, 因此查询质量比较高, 是有效的查询语义.

#### 4.2 执行时间

我们通过比较各种方法在不同数据集上执行时间来比较它们的查询效率. 每种方法在不同数据集上都执行 15 个查询, 这 15 个查询分为 3 组, 分别为(1)低频查询, 每个查询中, 至少包含一个低频关键字; (2)中频查询, 每个查询中, 至少包含一个中频关键字且不包含低频关键字; (3)高频查询, 每个查询中, 全是高频关键字. 由于篇幅关系, 这里只给出在 DBLP 上的执行结果, 如图 4 所示.

由于 LCA 方法需要计算所有的元素匹配情况, 所以耗费时间较长, 对于高频查询来说, 这点更明显. XSearch 由于要判断结点之间是否关联, 因此也要耗

费较多的时间. SLCA 的效率比 LCA 和 XSearch 高. LCEA 和 MLCEA 与 SLCA 效率接近. MLCEA 比 LCEA 效率差些, 因为 MLCEA 需要一些判断来排除一些没有意义的结果. 总体来说, LCEA 和 MLCEA 在各种频率查询上都具有较好的查询效率.

从以上两方面的实验结果来看, 本文提出的查询语义 LCEA 和 SLCEA, 相比于已有语义更有效.

## 5 结论

本文研究了 XML 关键字查询的有效性. 为了使得查询结果更加符合用户的需要, 我们分析了 XML 文档的特点, 对结点进行分类, 提出了以实体为基本语义单元的最低公共实体祖先 LCEA 和有意义的最低公共实体祖先 MLCEA 的概念, 并给出了基于这两种语义的有效查询算法. 实验结果验证了本文方法的有效性. 未来工作将考虑基于实体为基本单元对结果进行排序以及如何进行结果展示的问题.

## 参考文献

- [1] W3C. XQuery 1.0: An XML Query Language, <http://www.w3.org/TR/xquery..>
- [2] V. Hristidis, N. Koudas, Y. Papakonstantinou, and D. Srivastava. Keyword Proximity Search in XML Trees. In IEEE Trans. Knowl. Data Eng. 18(4), pages 525-539, 2006
- [3] Y. Xu and Y. Papakonstantinou. Efficient Keyword Search for Smallest LCAs in XML Database. In SIGMOD, 2005
- [4] S. Cohen, J. Mamou, Y. Kanza and Y. Sagiv. XSearch: A Semantic Search Engine for XML. In VLDB, 2003
- [5] G. Li, J. Feng, J. Wang, and L. Zhou. Effective Keyword Search for Valuable LCAs over XML Documents. In CIKM, 2007
- [6] I. Tatarinpv, S. D. Viglas, K. Beyer, J. Shanmugasundaram, E. Shekita and C. Zhang. Storing and Querying Ordered XML Using a Relational Database System. In SIGMOD, 2002.
- [7] <http://www.cs.washington.edu/research/xmldatasets/www/repository.html>
- [8] <http://www.xml-benchmark.org/>
- [9] <http://dblp.uni-trier.de/xml/>

---

[10] X-Hive/DB. <http://support.x-hive.com/xquery/index.html>.

**黄静**, 女, 1984 年生, 硕士研究生, 主要研究方向为 XML 数据库.

**徐俊劲**, 男, 1985 年生, 硕士研究生, 主要研究方向为 XML 数据库.

**周军锋**, 男, 1977 年生, 博士研究生, 主要研究方向为 XML 数据库.

**孟小峰**, 男, 1964 年生, 教授, 博士生导师, 主要研究方向为 Web 数据管理、XML 数据库、移动数据管理.