模糊XML关键字近似查询方法研究

李 婷 程海涛
(东北大学计算机科学与工程学院 沈阳 110819)

摘 要 在精确XML文档上的关键字查询方法的研究大多是基于LCA语义或者其变种语义(SLCA,ELCA等)开展的,将包含所有关键字的最紧致XML子树片段作为查询结果返回。但是这些基于LCA语义产生的查询结果中通常包含了大量的冗余信息,现实世界中存在着大量的不确定和模糊信息,因而如何从模糊XML文档中搜索到高质量的关键字查询结果是一个需要研究的问题。针对模糊XML文档上的关键字近似查询方法进行研究,通过引入最小连接树(MCT)的概念,提出在模糊XML文档上关键字查询的所有GDMCTs问题,并给出解决这一问题的基于栈的算法All fuzzy GDMCTs,该算法可以得到满足用户指定的子树大小阈值和可能性阈值条件的所有GDMCTs结果。实验表明,该算法在模糊XML文档上能够得到较高质量的关键字查询结果。

关键词 XML,关键字,近似查询,模糊,可能性
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.09.040

Research of Approximate Keyword Query on Fuzzy XML Documents

LI Ting CHENG Hai-tao
(School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China)

Abstract The study of keyword queries on crisp XML documents is carried out mainly based on the LCA semantics or its variant semantics (SLCA, ELCA), and the most compact XML subtrees containing all keywords are returned as the query results. However, the generated results based on the LCA semantics always contain a lot of redundant information, and there are a lot of uncertainty and fuzzy information exist in the real world. How to search the high quality results of keyword queries on fuzzy XML documents is an issue need to be studied. Aiming at investigating the method of approximate keyword query on fuzzy XML documents, firstly the concept of minimum connecting tree was introduced, All GDMCTs problem of keyword queries on fuzzy XML documents was proposed, and a stack based algorithm All fuzzy GDMCTs was given to solve the problem. The algorithm can get all the GDMCTs results satisfying the given subtree size threshold and possibility threshold. Experimental results show that the algorithm can get the high quality results of keyword queries on fuzzy XML documents.

Keywords XML, Keyword, Approximate query, Fuzzy, Possibility

1 引言

随着网络技术的快速发展,基于Web的信息管理变得越来越重要。XML已经成为Web上信息交换的标准,因而针对XML数据的数据管理成为一个重要的研究领域。现实世界中的数据往往存在着不确定性和不精确性,例如,若一个人的年龄大小是模糊的,只知道其年龄的可能取值及其可能性是{0.6/27,0.9/28,0.8/29,0.7/30},其中0.6/27表示年龄大小为27的可能性为0.6。XML的柔性特点可以很好地表示这些模糊数据,因此针对模糊XML数据的数据管理成为一个很重要的研究课题。

关键字查询是一种友好的查询方式。目前,对精确XML文档上关键字查询方法的研究主要是基于最低公共祖先(LCA)语义及其变种语义(SLCA[1],ELCA[2],VLCA[3])。研究者们提出了相关的算法寻找LCA(或者SLCA,ELCA等)节点,并将以LCA节点(或者SLCA,ELCA等)为根节点的子树作为结果返回。基于LCA语义及其变种语义的查询结果子树中虽然包含在标签中含有关键字的节点,但同时也包含了大量的冗余信息,使得查询结果的精度相对较低。针对该不足,一些研究者们对从LCA节点(或者SLCA节点)到含有关键字的节点的路径进行了研究,并将从LCA节点(或者SLCA节点)到包含关键字的节点的路径作为关键字近似查询结果返回[4-5]。最近,一些研究人员对概率XML数据上的关键字查询方法进行了研究[6-7]。其中文献[6]针对概率XML数据上的Top-k关键字查询问题进行研究,并提出PrStack和EagerTopK两种算法来求解具有最高概率值的前

到稿日期:2016-08-04 返修日期:2016-12-13

李 婷(1988-),女,博士生,主要研究方向为XML数据管理、XML关键字查询,E-mail:kitehyabc@163.com(通信作者);程海涛(1986-),男,博士生,主要研究方向为描述逻辑、本体。

K 个查询结果。而对于模糊 XML 数据上查询问题的研究成果主要集中在对其结构查询方法的研究这一方面[8-9]。

本文对模糊 XML 数据上的关键字近似查询方法进行研究,通过提出关键字查询,得出关键字查询的包含所有关键字的最小连接子树(minimum connecting tree)结果。这些最小连接子树满足其值不大于用户给定的子树大小阈值(subtree size threshold),并且其可能性值不小于给定的可能性阈值(possibility threshold)。

2　模糊 XML 数据模型

为了在 XML 中表示模糊数据,将模糊集和可能性分布理论引入 XML 数据模型中[10]。通过将元素节点与隶属度相结合来表示元素的可能度大小,将可能性分布与元素的属性值相结合来表示各个属性值的可能度及其它们之间的可能性分布。为此,在 XML 数据模型中引入一个模糊结构体"Val"和一个可能性属性 "poss"来表示在模糊 XML 文档中给定元素的可能性大小,其中,poss 在 (0,1] 之间取值。图 1 所示是一个模糊 XML 文档示例。

```
1. 〈Organization OName="University"〉
2.   〈Val Poss="0. 85"〉
3.     〈College CName="Software College"〉
4.       〈Employee EID="87001"〉
5.         〈Dist type="disjunctive"〉
6.         〈Val Poss="0. 8"〉
7.           〈ename〉Alisa York〈/ename〉
8.           〈position〉Lecturer〈/position〉
9.           〈telephone〉8687001〈/telephone〉
10.          〈office〉C1027〈/office〉
11.        〈/Val〉
12.        〈Val Poss="0. 9"〉
13.          〈ename〉Alisa York〈/ename〉
14.          〈position〉Professor〈/position〉
15.          〈telephone〉8687001〈/telephone〉
16.          〈office〉B1027〈/office〉
17.        〈/Val〉
18.      〈/Dist〉
19.    〈/Employee〉
20.      〈Student SID="201637001"〉
21.        〈sname〉John Smith〈/sname〉
22.        〈sex〉Male〈/sex〉
23.        〈email〉
24.          〈Dist type="conjunctive"〉
25.            〈Val Poss="0. 8"〉JSmith@gmail. com〈/Val〉
26.            〈Val Poss="0. 9"〉John_Smith@163. com〈/Val〉
27.            〈Val Poss="0. 7"〉JSmith@hotmail. com〈/Val〉
28.            〈Val Poss="0. 6"〉John_Smith@yahoo. com〈/Val〉
29.          〈/Dist〉
30.        〈/email〉
31.      〈/Student〉
32.    〈/College 〉
33. 〈/Val〉
34. 〈/Organization 〉
```

图 1　模糊 XML 文档示例

图 1 中,第 2 行〈Val Poss="0. 85"〉表示学院的名字是"软件学院"的可能性为 0. 85。为了表示元素属性值的可能性分布,在 XML 数据模型中引入模糊结构体"Dist"来表示元素属性值的可能性分布类型。对于模糊 XML 数据,对其采用两种可能性分布,即析取分布(disjunctive)或者合取分布(conjunctive)。例如图 1 中第 5－18 行是一个"Dist" 结构体,它表示雇员 Alisa York 的两种可能信息:1)她是讲师,可能性为 0. 8;2)她是教授,可能性为 0. 9。这两个可能信息之间满足的可能性分布类型是析取分布(disjunctive)。第 24－29 行同样是一个"Dist"结构体,它表示学生 John Smith 的邮箱地址的各种可能信息,这些不同的邮箱地址具有相应的可能性值,并且它们之间满足的可能性分布类型是合取分布(conjunctive)。

一个精确 XML 文档通常表示为一个有向树形结构,对于一个模糊 XML 文档 d,它同样可以表示为一个有向树形结构 $T=(V,E)$,其中,V 是树 T 中节点的集合,E 是树 T 中边的集合,可知 $E \in V \times V$。V 由两个节点集合组成,分别为一般节点集合 V_C 和模糊节点集合 V_F(Dist,Val)。树 T 的子树 T' 应满足条件:$V(T') \subseteq V(T)$,$E(T') \subseteq E(T)$。

3　最小连接树及其整体可能性值的计算

3.1　最小连接树

文献[4]提出一个最小连接树(minimum connecting tree)的概念,在一个 XML 树形结构 T 中,如果 $v_1, v_2, \cdots, v_m$ 是树 T 中的节点,那么节点 $v_1, v_2, \cdots, v_m$ 的最小连接树是树 T 中连接 $v_1, v_2, \cdots, v_m$ 的最小子树 T_M,并且最小子树 T_M 的根节点 $r(T_M)$ 是节点 $v_1, v_2, \cdots, v_m$ 的最低公共祖先(LCA)节点。对于在树 T 上的一个关键字查询 $\{k_1, k_2, \cdots, k_m\}$,关键字 k_1, $k_2, \cdots, k_m$ 的一个最小连接树(MCT)是包含这些关键字的节点 $g_1, g_2, \cdots, g_m$ 的最小连接树。为解决在查询结果中存在的数据冗余问题,文献[4]提出 Distance MCT(DMCT)和 Group DMCT(GDMCT)这两个概念,节点 $v_1, v_2, \cdots, v_m$ 的 MCT T_M 的 DMCT T_D 是满足以下 3 个条件的具有节点标签和边标签的最小子树:1)T_D 包含节点 $v_1, v_2, \cdots, v_m$;2)T_D 包含任意两个节点 (v_i, v_j) 的 LCA 节点 $\{u_1, u_2, \cdots, u_k\}$,其中 v_i, $v_j \in \{v_1, v_2, \cdots, v_m\}$,$i \neq j$;3)对于两个不同的节点 $n, n' \in \{v_1, \cdots, v_m, u_1, \cdots, u_k\}$,如果在 T_M 中从节点 n' 到 n 有一条长度为 L 的路径,那么在这条路径上除了节点 n 和 n' 不存在任何节点 n'' 满足 $n'' \in \{u_1, \cdots, u_k\}$。

对于一个 DMCT D 和一个 GDMCT G,如果 D 和 G 是同构的,假设 f 是 D 中节点到 G 中节点的映射,它同样产生一个从 D 的边到 G 的边的相应的映射,也称为 f,那么 D 和 G 满足以下两个条件:

(1)如果 n_D 是 D 中的一个节点,n_G 是 G 中的一个节点,并且 $f(n_D)=n_G$,那么 n_G 的标签中包含 n_D 的 id。

(2)如果 e_D 是 D 中的一个边,e_G 是 G 中的一个边,并且 $f(e_D)=e_G$,那么 e_D 的边上和 e_G 的边上具有相同的数字。例如下面是一个 GDMCT 结果:

$$u_1[a_1, a_2] \overset{2}{\leftarrow} u_0[c_1] \overset{2}{\rightarrow} u_2[a_5, a_7] \tag{1}$$

它可以由下面的 DMCTs 合并而成:$a_1 \overset{2}{\leftarrow} c_1 \overset{2}{\rightarrow} a_5$, $a_1 \overset{2}{\leftarrow} c_1 \overset{2}{\rightarrow}$

$a_7, a_2 \overset{2}{\leftarrow} c_1 \overset{2}{\rightarrow} a_5, a_2 \overset{2}{\leftarrow} c_1 \overset{2}{\rightarrow} a_7$。一个 GDMCT 的实例可以是一个 XML 数据树 T 的最小连接子树 MCT，也可以是一个 MCT 的 DMCT。

下面对模糊 XML 文档上的关键字近似查询方法进行研究，求解在给定可能性阈值 U 和子树大小阈值 K 的条件下关键字查询的所有 GDMCTs 结果。首先，对于关键字查询 $\{k_1, k_2, \cdots, k_m\}$，给出模糊 XML 文档上的所有 GDMCTs 问题的相关描述。

问题 1(所有 GDMCTs 问题) 给定一个 XML 树形结构 T，关键字 $k_1, k_2, \cdots, k_m$，一个子树大小阈值 K 和一个可能性阈值 U，求解关键字查询所有的 GDMCTs 就是寻找最小的元组(n,G)集合，其中 G 是一个根节点标签为$[n]$的 GDMCT，元组(n,G)满足以下条件：

(1)n 是关键字 $k_1, k_2, \cdots, k_m$ 的一个 LCA 节点，并且是一般节点。

(2)每个根节点为 n 并且大小上限为 K 的 DMCT 至少属于一个元组(n,G)中的一个 GDMCT，并且该元组的 GDMCT 所包含的每个 DMCT(或者 MCT 实例)的可能性值不小于给定的可能性阈值 U。

(3)如果任何 id 为 n_i 的节点从 $G(G \in (n,G))$中的节点 n'(标签$[n'] \in [n_1, \cdots, n_i, \cdots, n_m]$)中移除，那么至少有一个子树大小上限为 K 的 DMCT D'不属于任何元组，即使 D'的根节点为关键字 $k_1, k_2, \cdots, k_m$ 的一个 LCA 节点 n。

(4)节点 n'(标签$[n'] \in [n_1, \cdots, n_i, \cdots, n_m]$)中每个节点 n_i 包含来自关键字集合$\{k_1, k_2, \cdots, k_m\}$的相同关键字子集 S。

(5)G 不大于阈值 K。

对于一个 GDMCT(或者 DMCT)，它的值等于 GDMCT(或者 DMCT)边上的权值之和。例如上述 GDMCT 结果(1)中的 GDMCT 的大小为 4。

3.2 最小连接树的可能性值计算

为方便研究，用一个简化的树形结构来表示一棵模糊 XML 树。如图 2 所示，其中黑色实体圆圈节点表示一般的 XML 节点，长方形节点表示模糊节点(两个相连的模糊节点 Dist-Val 表示为一个模糊节点)。对树中的节点进行编号，并用节点号表示相应的节点，例如节点 9(n_9)表示标签里包含 A_3 的节点。可知，节点 1,2,3 等均为一般节点，节点 6,13,17 是模糊节点。元素和属性值的隶属度值被表示在连接该节点与其父节点的边上。未标记数值的边上的隶属度值默认为 1，表示其两端连接的节点中孩子节点相对父亲节点的隶属度值为 1。

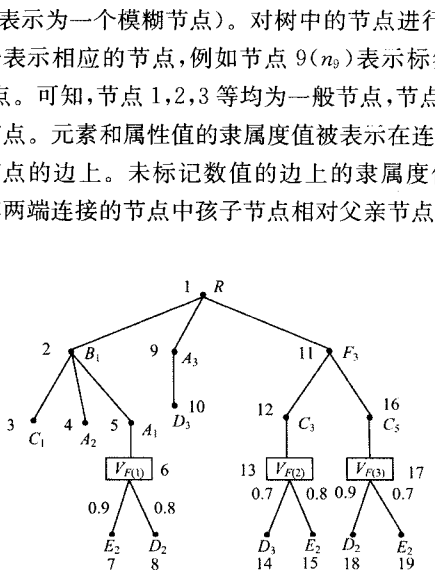

图 2 简化的 XML 树形结构

对图 2 中的 XML 树形结构提出关键字查询$\{E_2, D_2\}$，可知任何包含不同关键字的两个节点的最低公共祖先节点 $LCA(E_2, D_2)$是节点 1，节点 6，节点 11 和节点 17。在模糊 XML 文档中，模糊节点用来描述其孩子节点的可能性分布及其隶属度，结果子树中若包含模糊节点将使得查询结果含有冗余信息，因而模糊节点不能成为最小连接子树的组成节点。因此 $LCA(E_2, D_2)$应当返回节点 1，节点 5，节点 11 和节点 16，$MCT(E_2, D_2)$结果如图 3 所示。

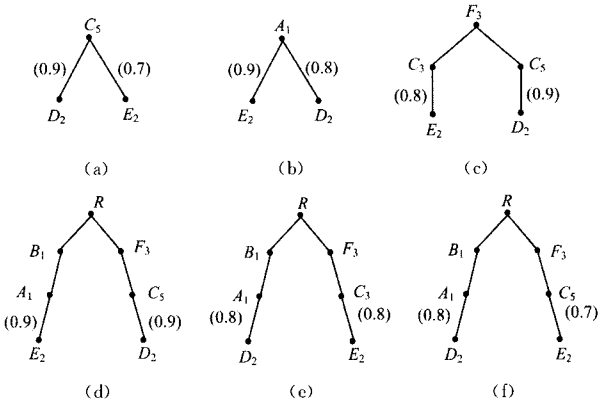

图 3 关键字查询的 MCT 结果

由于模糊节点不作为最小连接子树中的节点返回，因此模糊节点下的元素的隶属度值被标记在连接其父亲(或者祖先)节点和其孩子(或者后代)节点的边上(如图 3 所示)。一个 MCT 结果的整体可能性 P 包含两部分：1)MCT 的本地可能性，表示为 P_{local}。如果从 MCT 的根节点 n 到 MCT 中包含关键字的节点 v_i 路径上的隶属度值为 $\rho_1, \rho_2, \cdots, \rho_l$，那么 $P_{local} = \rho_1 \times \rho_2 \times \cdots \times \rho_l$。2)MCT 的存在可能性，表示为 P_{exist}。如果从模糊 XML 文档的根节点 r 到 MCT 的根节点 n 的路径上的隶属度值为 $\chi_1, \chi_2, \cdots, \chi_\kappa$，那么 $P_{exist} = \chi_1 \times \chi_2 \times \cdots \times \chi_\kappa$。一个 MCT 结果的整体可能性 P 通过式(2)计算：

$$P = P_{local} \times P_{exist} \tag{2}$$

例如图 3(a)的 MCT 结果中，其本地可能性值为 $P_{local} = 0.9 \times 0.7 = 0.63$，它的存在可能性值 $P_{exist} = 1$，因此它的可能性值为 0.63。DMCT 的整体可能性值同样可通过式(2)求解。对于关键字查询$\{E_2, D_2\}$，当 $K=5, U=0.7$ 时，查询的 DMCTs(GDMCTs)结果及其可能性值是：$\{n_7(0.9) \overset{1}{\leftarrow} n_5 \overset{1}{\rightarrow} n_8(0.8), 0.72\}$，$\{n_{15}(0.8) \overset{2}{\leftarrow} n_{11} \overset{2}{\rightarrow} n_{18}(0.9), 0.72\}$。对于第一个 DMCT 结果中的 $n_7(0.9)$，n_7 表示节点 7，其括号里的数字 0.9是 DMCT 结果的根节点 n_5 到 n_7 路径上的隶属度值，0.72是 DMCT 结果的整体可能性值。

4 模糊 XML 关键字近似查询

4.1 编码方式

区间编码是一个能够有效判断 XML 树形结构中两个节点祖先-后代关系的一种编码方式，每个节点被赋予一个区间编码$[start, end]$。给定一棵 XML 树，节点 u 和节点 v 是祖先-后代关系，当且仅当满足 $start(u) < start(v) \wedge end(v) < end(u)$，即祖先节点 u 的区间编码包含后代节点 v 的区间编

码。由于在模糊 XML 树形结构中存在模糊节点,因此普通的区间编码不能区分树中的模糊节点和一般节点。为了区分模糊节点和一般节点以及记录节点在文档相关路径上的模糊信息,对普通区间编码进行扩展,提出一种新的编码方式。对于一棵模糊 XML 树 T,对树中的每个节点采用一个六元组(*id*,*start*,*end*,*level*,*fuzzy identifier*,*membership degrees*)表示,其中 *id* 是树中节点的标识码,每个节点用唯一的标识码表示。对树 T 的所有节点进行先序遍历,每个节点在遍历时分别被访问两次并产生两个序号。一个是在遍历该节点的所有后代节点之前访问该节点产生的序号,记为 *start*;另一个是在遍历完该节点的所有后代节点后再一次访问该节点产生的序号,记为 *end*。*level* 表示节点在树形结构中所在的层次值,其中文档的根节点所在的层次值为 1。*fuzzy identifier* 用来表示一个节点是否是模糊节点,它的值是一个布尔值。当节点是模糊节点时,*fuzzy identifier* 的值为 1;当节点是一般节点时,*fuzzy identifier* 的值为 0。*membership degrees* 用来记录从模糊 XML 文档根节点到该节点路径上的模糊节点下的隶属度值,若从根节点到该节点的路径上没有模糊节点,则 *membership degrees* 的值为 1。例如(n_7,25,15,6,0,0.8)表示在树结构第 6 层上的一般节点 n_7,它的区间编码为(25,15),从根节点到该节点路径上的模糊节点下的隶属度为 0.8。

4.2　索引

针对下文将提出的模糊 XML 文档上的关键字近似查询算法,构建两个倒排索引列表 $I(k_i)$ 和 $I_A(k_i)$。当用户提出关键字查询$\{k_1,k_2,\cdots,k_m\}$时,对于每一个关键字 k_i,索引列表 $I(k_i)$中记录包含关键字 $k_i(1\leqslant i\leqslant m)$的节点,索引列表 $I_A(k_i)$ 记录列表 I 中包含关键字 k_i 的节点的祖先节点。图 4 是一棵采用六元组编码方式进行编码的模糊 XML 树形结构(为了清晰地显示,仅给出部分节点的编码),当用户在图 4 中的 XML 树上提出关键字查询$\{E_2,D_2\}$时,针对关键字 E_2 创建的索引如下:

$I(E_2)$=[(n_{10},14,15,6,0,0.9),(n_{27},48,49,6,0,0.8),(n_{37},68,69,6,0,0.7)]

$I_A(E_2)$=[(n_1,1,74,1,0,1),(n_2,2,27,2,0,1),(n_5,7,26,3,0,1),(n_6,8,25,4,1,1),(n_7,9,16,5,1,1),(n_{17},32,73,2,0,1),(n_{18},33,52,3,0,1),(n_{19},34,51,4,1,1),(n_{24},43,50,5,1,1),(n_{28},53,72,3,0,1),(n_{29},54,71,4,1,1),(n_{34},63,70,5,1,1)]

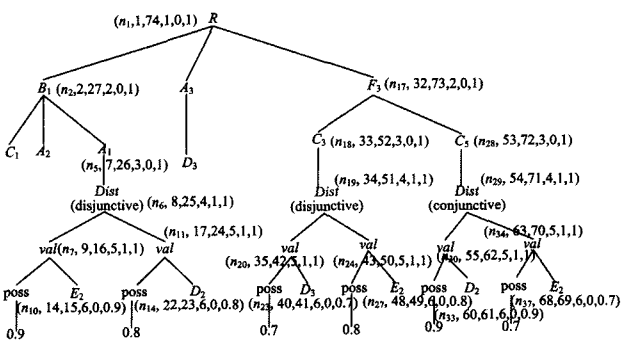

图 4　部分编码的 XML 树形结构

4.3　关键字近似查询算法

本节首先给出关键字近似查询算法(ALL fuzzy GDMCTs)的总体框架(见算法 1),该框架描述了被选择的节点列表如何以深度优先的方式进行遍历,节点如何被压入栈以及节点在栈中如何被弹出。首先从列表 I 和 I_A 中选择具有最小 *start* 值的节点压入栈 ST,当把节点压入栈 ST 时,在栈中相关节点的条目上记录栈中产生的 GDMCT(或者 DMCT)结果。当 top(ST). end$<n$. start(n 指将被压入栈 ST 的节点,该节点在列表中具有最小 *start* 值),在压入节点 n 之前,栈中不是 n 的祖先节点的节点 x_j 将被弹出,弹出节点 x_j 时它所记录的满足阈值条件的包含全部关键字的 GDMCTs 结果将被输出,而条目中节点 x_j 记录的包含部分关键字的 GDMCTs 结果将被移动到其父节点所在的条目上;然后将节点 n 压入栈 ST 中继续求解 GDMCTs 结果。All fuzzy GDMCTs 算法中栈 ST 的具体的操作过程如算法 2 所述,包括压入栈操作和弹出栈操作。给定一个关键字查询,All fuzzy GDMCTs 算法能够有效且及时地输出满足用户给定的子树大小阈值 K 和可能性阈值 U 条件下的 GDMCTs 结果。

算法 1　All fuzzy GDMCTs 算法的总体框架

All fuzzy GDMCTs (k_1,…,k_m,K,U) {

ST:栈,ST 中的条目 s 由(s. nodeID,s. GDMCTs)组成,其中 s. GDMCTs 是 GDMCTs 的列表;

1. 当输入关键字 k_1,…,k_m 时,得到并且访问列表 I 和 I_A;
2. 当 ST 非空栈 OR 在列表 I 和 I_A 中包含更多的节点时 do {
3. 从列表 I 和 I_A 中找到具有最小 start 值的节点 n;
4. 当 ST 非空栈 AND top(ST). end $<$ n. start do
5. POP(ST);//栈中不是节点 n 的祖先节点被 S 弹出
6. 当 top(ST). fuzzy identifier=1 时 do
7. POP(ST);//位于栈顶的模糊节点将被弹出
8. PUSH(n,ST);}}

算法 2　All fuzzy GDMCTs

输入:一个模糊 XML 文档 d,关键字 k_1,k_2,…,k_m

输出:在给定子树大小阈值 K 和可能性阈值 U 条件下的 GDMCTs

1. 得到索引列表 I(k_i)和 I_A(k_i) (1$\leqslant$ i $\leqslant$ m),{
2. For i=1,…,m do
3. 访问列表 I(k_i)和 I_A(k_i),
4. Return ($\bigcap_i^2 I_A(k_i)$)$\cup$($\bigcup_i I(k_i)$);// $\bigcap_i^2 I_A(k_i)$指计算在一次扫描中,至少在两个不同 I_A 列表中出现的节点。}

POP (ST) {

5. h$\leftarrow$pop(ST) and h. fuzzy identifier=0;
6. 从 h. GDMCTs 中输出和移除满足给定子树大小阈值 K 和可能性阈值 U 条件的包含所有关键字的 GDMCTs; // LCA 节点是 h 的 GDMCTs 结果
7. h′$\leftarrow$top(ST) and h′. fuzzy identifier=0;
8. For each GDMCT G in h. GDMCTs do{//将 h 剩余的 GDMCTs 移动到 h′
9. d=h. depth$-$h′. depth;
10. r$\leftarrow$root(G);// r. node ID=h. node ID
11. If degree(r)=1 AND e 是 G 中入射到 r 的边

then { // h 被弹出,h′代替 h 成为新的根节点

12. label(e) ← label(e)+d;// GDMCT 的边标签
13. label(r) ← h'. node ID;} // GDMCT 的节点标签
14. else {// h 不被弹出,仍然需要在 GDMCT 中,
15. 为 G 创建一个新的根节点 r',label(r') ←h'. node ID;
16. 从节点 r'到节点 r 增加边 e',label(e') ←d;}
17. If size (G)>K then drop G;// 修剪条件
18. If threshold (D) < U then drop D in G;//去除 G 中低于给定可能性阈值 U 的 DMCT 结果}
19. h'. GDMCTs ← h'. GDMCTs∪CreateNewGDMCTs(h. GDMCTs,h'. GDMCTs);
20. h'. GDMCTs ← Merge (h. GDMCTs,h'. GDMCTs);
21. When top(ST). fuzzy identifier=1,pop top(ST)}PUSH(n,ST){
22. Push s (n,∅) onto ST;//新的栈条目 s
23. For i =1,…,m do
24. If n contains k_i then s. GDMCTs ← s. GDMCTs ∪ {$n^i(\tau_w)$} // 上标 i 表示在部分 GDMCTs 中新的单个节点中包含的关键字,τ_w 表示从 G 的根节点到新节点路径上的各个隶属度值
CreateNewGDMCTs (h. GDMCTs,h'. GDMCT){ //创建包含多个关键字的新的 GDMCTs
25. NewGDMCTs← ∅;
26. For each G∈h. GDMCTs do
27. For each G'∈h'. GDMCTs do
28. If keywords(G) ∩ keywords(G')=∅,size(G)+size(G') ≤ K and the possibility of D (D')in G (G') is no less than U, then{ //把不相交的子树紧附于它们共同的根上
29. NewGDMCTs ← NewGDMCTs ∪ mergeGDMCTs (G,G');}
30. Return NewGDMCTs;}
31. Merge (h. GDMCTs,h'. GDMCTs){ //合并同构树
32. NewGDMCTs ← h. GDMCTs∪h'. GDMCTs;
33. For each G∈h. GDMCTs do
34. For each G'∈h'. GDMCTs do
35. If keywords(G)=keywords(G') and possibility of D (D') in G (G') is no less than U,通过从 G 到 G'的映射 μ,G,G'是同构的
36. And for every node $u^i(\tau_w)\in G, u'^i(\tau_w)\in G', \mu(u^i(\tau_w))=u'^i(\tau_w)$ then //合并具有相同关键字匹配的节点列表
37. Replace G,G' in NewGDMCTs by merge GDMCTs (G,G',μ);}
38. Return New GDMCTs;}

下面介绍算法 2 的具体执行过程。对于栈 ST,它存储的条目由(s. nodeID,s. GDMCTs)组成。s. GDMCTs 是 GDMCTs 的一个列表,以 s. nodeID 所标记的节点作为根节点。这些 GDMCTs 可能包含部分关键字,例如对图 4 中的模糊 XML 文档提出关键字查询$\{E_2,D_2\}$,栈处理节点过程中会产生一个部分 GDMCTs 结果 $n_{17}\xrightarrow{2}n_{27}{}^1(0.8)$,它只包含查询关键字的一个子集 E_2,并且用它里面的节点包含的关键字和从它的根节点到包含关键字的节点路径上的隶属度值注释。如 n_{17} 是关键字 E_2 和 D_2 的一个 LCA 节点(部分 GDMCTs 的根节点),$n_{27}{}^1(0.8)$中上标 1 表示节点 n_{27} 包含第 1 个关键字 E_2,数字 0.8 表示从节点 n_{17} 到节点 n_{27} 的路径上的隶属度是 0.8。由于模糊节点不作为 GDMCTs 中的组成节点,因此节点 n_{17} 到节点 n_{27} 的边上的路径长度记为 2。算法首先扫描列表 $I(k_i)$和 $I_A(k_i)$,列表 I 和 I_A 的节点被压入和弹出栈 ST,在每个主循环(算法 1 中第 2—8 行)迭代的最后,栈 ST 顶端的条目包含节点 n,它是到目前为止在 ST 中具有最大 start 值的节点。栈中的其他条目对应于节点 n 的祖先节点。在节点 n 被压入到栈 ST 之前,栈中的所有不是节点 n 的祖先节点的节点所对应的条目被弹出(通过算法 1 中第 4—5 行实现)。如果栈的顶端条目中的节点 h 的 *fuzzy identifier*=1,即顶端条目中包含的节点是模糊节点,那么节点 h 将被栈弹出,因为模糊节点不能成为 GDMCTs 中的组成节点。当一个条目 h 从栈 ST 中弹出,如果 h 是一般节点,那么任何来自 h. GDMCTs满足阈值条件的完整的 GDMCTs 被输出(算法 2 中第 6 行)。剩余的部分 GDMCTs 是包含部分关键字的,这里存在一种可能性,即 h 的父亲节点 h'的后代节点可能包含该部分 GDMCTs 错过的部分关键字。当 h'为一般节点时,h 的部分 GDMCTs 成为它的父亲节点 h'(一般节点)部分的(或者完整的)GDMCTs(需要注意的是,在 h 检查前,h'所在的条目可能已经有部分 GDMCTs 来反映在节点 h'的后代节点中找到的关键字)。将 h 的每个部分 GDMCT G 转移到 h'的 GDMCTs 的集合需遵循以下步骤:1)修改 G 来反映出新的根节点(算法 2 中的第 11—16 行);2)检查 G 是否满足修剪条件和阈值条件(算法 2 中的第 17—18 行)。

当对 h 的部分 GDMCTs 进行修改和修剪时,将其与它的父亲节点 h'的 GDMCTs 进行对比并且产生合适的 GDMCTs(算法 2 中第 19 行)。将 h 和 h'的 GDMCTs 进行合并,为从 h 到 h'的 GDMCTs 中能够被粘合在一起的 GDMCT 创建一个新的 GDMCT 来包含更多的关键字子集(算法 2 的 26—29 行)。最后,将从 h 到 h'的每一对同构的 GDMCTs 合并成一个 GDMCT(算法 2 第 20 行)以使产生的 GDMCTs 的数目最小。需要注意的是,当 h 被弹出栈时,以节点 h 为根节点的 GDMCTs 结果被输出(算法 2 第 6 行),而不是在它们最开始产生时(算法 2 第 19,25—30 行),其原因是将会有更多的 GDMCTs 与已经产生的 GDMCTs 进行合并(算法 2 第 20,32—37 行)。需要注意的是,若 h'为模糊节点,它将被 ST 弹出(算法 2 第 21 行),并且包含 h'的条目上产生的 GDMCTs 以及包含 h'的条目上产生的部分 GDMCTs 将一起被移到栈中 h'的父亲节点所在的条目中继续处理。算法 2 可以实现在模糊 XML 文档上的关键字近似查询的 GDMCTs 结果的求解,通过用户指定子树大小阈值 K 和可能性阈值 U,该算法可以求解子树大小阈值不大于 K 的 GDMCTs 结果,并且这些 GDMCTs 结果中的每个 DMCT (或者 MCT) 实例的可能性阈值均不小于 U。

5 实验测试

5.1 实验环境设置和实验数据集

所有实验均在 Intel(R) Core(TM) i3 CPU M 330 @ 2.13GHz处理器,3GB RAM 和内置 320G 硬盘的 Microsoft Windows 7 操作系统环境中进行,并采用 DBLP[11] 数据集进行实验测试。DBLP 是一个相对层次较浅的真实数据集,在

实验中采用一个 100MB 的 DBLP 数据集进行测试。首先采用文献[9]中使用的随机模糊信息产生方法将 DBLP 数据集转换为模糊 XML 数据集,将生成的模糊 DBLP 数据集记为 FDB,大小为 118MB。

5.2　精度和召回率测试

采用本文提出的 All fuzzy GDMCTs 算法在 FDB 数据集上进行关键字查询,对于每个关键字查询 Q_i,用户给定可能性阈值 U 和子树大小阈值 K,将通过关键字查询算法得出的不低于可能性阈值 U 和不大于阈值 K 的最小连接子树结果记为近似查询结果(Approximate Results),用 PR_i 表示。文献[9]提出的 LTwig 算法能够有效地在模糊 XML 文档上处理包含 AND,OR 和 NOT 谓词的小枝查询,并且得到不低于用户给定阈值的相应小枝查询的查询结果。对每个关键字查询 Q_i 构建相应的结构查询语句 SQ_i,采用 LTwig 算法得出相应构建的结构查询 SQ_i 在不低于可能性阈值 U 的条件下的查询结果,并将通过结构查询 SQ_i 得出的查询结果记为准确查询结果(Accurate Results),用 AR_i 表示。对于每一个关键字查询,测试关键字查询算法的精度和召回率,精度和召回率可以通过以下公式得出:

$$precision(\text{精度}) = |AR_i \cap PR_i| / |PR_i|$$

$$recall(\text{召回率}) = |AR_i \cap PR_i| / |AR_i|$$

在 FDB 数据集上提出 8 个不同的关键字查询,分别记为 $DQ_1, DQ_2, \cdots, DQ_8$,每个关键字查询由 1～4 个关键字组成。当给定可能性阈值 $U=0.6$ 时,对于不同的子树大小阈值 K ($K=5,7$),对比 All fuzzy GDMCTs 算法的精度和召回率,实验结果如图 5 所示。当 $K=5$ 时,对于不同的可能性阈值 U ($U=0.5,0.8$),同样对比 All fuzzy GDMCTs 算法的精度和召回率,实验结果如图 6 所示。实验结果表明,在用户给定可能性阈值 U 和子树大小阈值 K 的条件下,算法 All fuzzy GDMCTs 具有较高的查询精度和召回率,能够得到较为准确的查询结果。在同等的可能性阈值 $U(U=0.6)$ 的条件下,增大阈值 K 的值,All fuzzy GDMCTs 算法的平均查询精度和平均召回率会相应地提高;在同等的子树大小阈值 $K(K=5)$ 的条件下,降低可能性阈值 U 的值,All fuzzy GDMCTs 算法的平均查询精度和平均召回率也会相应地提高。这是因为在增大阈值 K 和降低可能性阈值 U 的条件下,All fuzzy GDMCTs 算法会得出更多相关的查询结果。因此,当用户对算法在给定的子树大小阈值 K 和可能性阈值 U 的条件下得出的查询结果不够满意时,可以通过适当增大子树大小阈值 K 或者降低可能性阈值 U 来得到更为精确和全面的关键字查询结果。

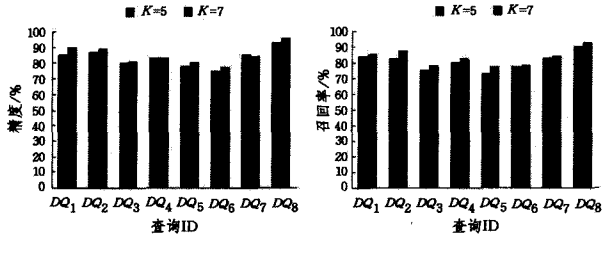

图 5　在不同子树大小阈值 $K(K=5,7)$ 的条件下的 All fuzzy GDMCTs 算法的查询精度和召回率($U=0.6$)

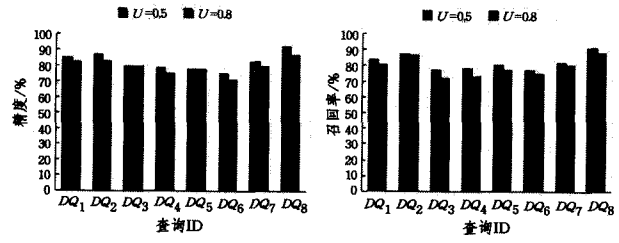

图 6　在不同可能性阈值 $U(U=0.5,0.8)$ 的条件下的 All fuzzy GDMCTs 算法的查询精度和召回率($K=5$)

下面将 All fuzzy GDMCTs 算法与文献[12]提出的 FIndex Loop 算法进行精度和召回率的测试。FIndex Loop 算法能够求解模糊 XML 文档上的关键字查询的 SLCA 结果及其可能性值,并且能够得到以 SLCA 节点为根节点的包含全部关键字的最小子树结果。在 All fuzzy GDMCTs 中,设定子树大小阈值 $K=15$,可能性阈值 $U=0.1$。在 FDB 数据集上分别提出关键字查询 $DQ_1, DQ_2, \cdots, DQ_8$,以针对每个关键字查询构建的相应的结构查询的查询结果为准确查询结果。对比 All fuzzy GDMCTs 算法和 FIndex Loop 算法的查询结果的精度和召回率,实验结果如图 7 所示。由测试结果可知,在一定的阈值条件下,与 FIndex Loop 算法相比,All fuzzy GDMCTs 算法具有更高的平均精度和平均召回率。

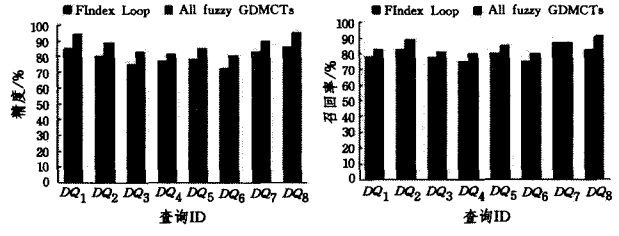

图 7　FIndex Loop 和 All fuzzy GDMCTs 算法的查询精度和召回率

5.3　时间花费和内存使用量测试

在未建立索引的情形下,All fuzzy GDMCTs 算法可以通过对文档中的节点进行先序遍历访问得到相应的关键字查询结果。在 FDB 数据集上分别进行关键字查询 DQ_1,DQ_3 和 DQ_5,在给定相同的子树大小阈值 K 和可能性阈值 U 的条件下($K=5,U=0.5$),分别测试 All fuzzy GDMCTs 算法在使用索引和未使用索引这两种情形下进行关键字查询的时间花费和内存使用量。在 All fuzzy GDMCTs(使用索引)中,索引通过 B+树进行存储,在进行关键字查询之前,首先对文档进行访问并根据所有实验测试所需的每个关键字分别创建节点列表 I 和 I_A。在算法的时间花费评估实验中,在使用索引的情形下算法的时间花费不考虑建立索引 I 和 I_A 所需要的时间(因为索引是在算法运行前对文档进行访问所创建的),仅考虑算法的运行时间。测试的实验结果如表 1 所列。其中 1985ms/89MB 表示在查询 DQ_1 中,All fuzzy GDMCTs 算法(使用索引)的时间花费为 1985ms,内存使用量是 89MB。结果表明,与算法不使用索引的情形相比,使用索引的情形下可

(下转第 226 页)

bute discretization for rough set theory based on information entropy [J]. Chinese Journal of Computers, 2005, 28(9): 1570-1574. (in Chinese)
谢宏,程浩忠,牛东晓. 基于信息熵的粗糙集连续属性离散化算法[J]. 计算机学报,2005,28(9):1570-1574.

[9] TAY E H, SHEN L. A modified chi2 algorithm for discretization [J]. IEEE Transactions on Knowledge and Data Engineering, 2002, 14(3): 666-670.

[10] SU C T, HSU J H. An extended Chi2 algorithm for discretization of real value attributes[J]. IEEE Transactions on Knowledge and Data Engineering, 2005, 17(3): 437-441.

[11] KURGAN L A, CIOS K J. CAIM discretization algorithm[J]. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(2): 145-153.

[12] JIANG F, SUI Y F. A novel approach for discretization of continuous attributes in rough set theory [J]. Knowledge-Based Systems, 2015, 73: 324-334.

[13] QIAN Q, CHENG M Y, XIONG W Q, et al. Reviews of binary ant colony optimization[J]. Application Research of Computers, 2012, 29(4): 1211-1215. (in Chinese)
钱乾,程美英,熊伟清,等. 二元蚁群算法研究综述[J]. 计算机应用研究,2012,29(4):1211-1215.

[14] PAWLAK Z. Rough sets[J]. Communications of the Acm, 1995, 38(11): 88-95.

[15] 王国胤. Rough 集理论与知识获取[M]. 西安交通大学出版社, 2001.

[16] XIONG W Q, WANG L Y, YAN C Y. Binary ant colony evolutionary algorithm[J]. International Journal of Information Technology, 2006, 12(3): 10-20.

[17] FAYYAD U M, IRANI K B. On the handling of continuous-valued attributes in decision tree generation[J]. Machine Learning, 1992, 8(1): 87-102.

[18] QUINLAN J R. C4. 5: programs for machine learning[M]. San Francisco: Morgan Kaufmann, 1993.

[19] KONONENKO I. Naive Bayesian classier and continuous attributes[J]. Informatica, 2010: 317-326.

[20] ZIARKO W. Variable precision rough set model[J]. Journal of Computer & System Sciences, 1993, 46(1): 39-59.

[21] XIONG W Q, WEI P. Binary ant colony evolutionary algorithm [J]. Acta Automatica Sinica, 2007, 33(3): 259-264. (in Chinese)
熊伟清,魏平. 二进制蚁群进化算法[J]. 自动化学报,2007,33(3):259-264.

(上接第 221 页)

以显著减少算法的时间花费,并且不会产生很大的内存使用量。

表 1 时间花费和内存使用量

查询 ID	All fuzzy GDMCTs /ms/MB	All fuzzy GDMCTs/ms/MB (未使用索引)
DQ_1	1985/89	16500/23.7
DQ_3	1190/73	14000/20
DQ_5	635/60	13200/18

结束语 本文针对模糊 XML 文档上的关键字近似查询方法进行了相关的研究。文中首先引入最小连接树(MCT)、距离最小连接树(DMCT)和成组距离最小连接树(GDMCT)的概念,给出最小连接树可能性值的计算方法。通过对传统的区间编码方式进行扩展以区分模糊 XML 文档中的模糊节点和一般节点,并记录节点在文档中所在路径上的模糊信息。提出关键字近似查询算法 All fuzzy GDMCTs 来求解在给定子树大小阈值 K 和可能性阈值 U 的条件下的 GDMCTs 结果。最后通过实验表明该算法能够获得较高质量的查询结果。

参 考 文 献

[1] XU Y, PAPAKONSTANTINOU Y. Efficient Keyword Search for Smallest LCAs in XML Databases[C]// Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 2005: 527-538.

[2] GUO L, SHAO F, BOTEV C, et al. XRANK: Ranked Keyword Search over XML Documents[C] // Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 2003: 16-27.

[3] LI G L, FENG J H, WANG J Y, et al. Effective Keyword Search for Valuable LCAs over XML Documents[C]// Proceedings of the sixteenth ACM Conference on Information and Knowledge Management. New York: ACM Press, 2007: 31-40.

[4] HRISTIDIS V, KOUDAS N, PAPAKONSTANTINOU Y, et al. Keyword Proximity Search in XML Trees[J]. IEEE Transactions on Knowledge and Data Engineering, 2006, 18(4): 525-539.

[5] BHALOTIA G, HULGERI A, NAKHE C, et al. Keyword Searching and Browsing in Databases using BANKS[C]// Proceedings of 18th International Conference on Data Engineering. New York: IEEE Press, 2002: 431-440.

[6] LI J, LIU C, ZHOU R, et al. Top-k Keyword Search over Probabilistic XML Data [C]// Proceedings of 2011 IEEE 27th International Conference on Data Engineering. New York: IEEE Press, 2011: 673-684.

[7] ZHOU R, LIU C, LI J, et al. ELCA Evaluation for Keyword Search on Probabilistic XML Data[J]. World Wide Web, 2013, 16(2): 171-193.

[8] MA Z M, LIU J, YAN L. Matching Twigs in Fuzzy XML.[J]. Information Sciences, 2011, 181(1): 184-200.

[9] LIU J, MA Z M, MA R Z. Efficient Processing of Twig Query with Compound Predicates in Fuzzy XML[J]. Fuzzy Sets and Systems, 2013, 229: 33-53.

[10] MA Z M, YAN L. Fuzzy XML Data Modeling with the UML and Relational Data Models[J]. Data & Knowledge Engineering, 2007, 63(3): 972-996.

[11] DBLP[EB/OL]. http://dblp.uni-trier.de/xml.

[12] LI T, MA Z M. Keyword Querying of Fuzzy XML[J]. Journal of Northeastern University: Natural Science, 2016, 37(7): 937-941. (in Chinese)
李婷,马宗民. 模糊 XML 关键字查询方法[J]. 东北大学学报:自然科学版,2016,37(7):937-941.