

基于路径权重的 XML 文档相似度仿真研究

赵艳妮^{1,2}, 郭华磊³, 马军生³

(1. 陕西职业技术学院 计算机科学系, 陕西 西安 710100;

2. 西安理工大学 自动化与信息工程学院, 陕西 西安 710048;

3. 西安通信学院 信息服务系, 陕西 西安 710106)

摘 要:针对 XML 文档查询效率低和准确度不理想的问题,提出一种基于路径权重的树相似度算法。该算法以树节点信息相似度和树结构相似度为出发点,依据信息组织主次分明的客观规律,信息按照重要程度依次排列在树的各个层次,树节点信息自上至下重要程度逐渐减弱。根据距离根节点越近的节点表示的信息越重要,最低层信息的重要性最小的特点,依照树节点在 XML 文档树中的层次自动计算该节点的路径权重,克服了传统 XML 文档树相似度计算中树节点信息权重平均分配或手工设置的缺点,解决了 XML 文档树的相似度自动计算问题,实现了 XML 查询树与文档树的快速匹配。仿真结果表明,该算法在大量 XML 文档检索方面查询效率、查准率和查全率都得到有效改进。

关键词:相似度;路径权重;查询树;文档树

中图分类号:TP391.9

文献标识码:A

文章编号:1673-629X(2016)09-0197-04

doi:10.3969/j.issn.1673-629X.2016.09.044

Simulation Research of XML Document Similarity Based on Path Weighting

ZHAO Yan-ni^{1,2}, GUO Hua-lei³, MA Jun-sheng³

(1. Department of Computer Science, Shaanxi Vocational & Technical College, Xi'an 710100, China;

2. School of Automation and Information Engineering, Xi'an University of Technology, Xi'an 710048, China;

3. Department of Information Service, Xi'an Communication College, Xi'an 710106, China)

Abstract: In order to realize the rapid and accurate retrieval of the XML document information, a tree similarity algorithm based on path weight is proposed. It considers the tree node information similarity and structural similarity, and the information is arranged in each level of the tree in accordance with the degree of importance by object rules of primary and secondary information organization, making the degree of importance for tree node information weakened from up to down. According to the characteristics that the node with closer distance from the root node represents the more important information, and the lowest level of the information has minimal importance, the path weight is calculated automatically in accordance with the tree node in XML document tree level, which overcomes the disadvantage of equally distribution or manual setting for tree node information weigh in the traditional XML document, and solves the similarity calculation of XML document tree, and realizes the fast matching of XML query tree and document. Simulation shows that the algorithm is improved in query efficiency, precision and recall.

Key words: similarity; path weight; query tree; document tree

0 引言

随着 XML 技术的广泛应用,如何在海量的 XML 文档中快速准确地检索到有效信息逐渐受到重视,成为近几年的热门研究领域之一。由于 XML 文档检索受到内容和结构的双重约束,因此 XML 文档的查询需要考虑关键字相似度和结构相似度两个因素^[1]。关

键词相似度的研究比较成熟,而结构相似度的研究是 XML 信息检索的难点。目前针对 XML 结构相似度的研究有两种:一是结构变换,穷举查询树的所有可能结构,然后与文档树匹配,该方法查准率高,但效率非常低;二是建立数学模型,建立查询树与文档树相似度的数学模型,该方法检索效率高,但查准率和查全率还有

收稿日期:2015-12-08

修回日期:2016-04-05

网络出版时间:2016-08-01

基金项目:国家自然科学基金资助项目(61272284);陕西省自然科学基金(2014JM8354);陕西省教育重点实验室科技项目(13JS083)

作者简介:赵艳妮(1982-),女,博士研究生,讲师,研究方向为模式识别。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20160801.0907.052.html>

待提高^[2-3]。

文中在文献[4]的基础上,提出一种自动计算查询树与文档树相似度的方法,克服了需要用户设置路径权重的缺点。该方法根据有效信息在查询树中的位置确定信息的重要程度,自动计算路径权重并赋予相应路径,综合考虑有效节点信息和有效节点间的结构关系来计算查询树与文档树之间的相似度,满足用户对 XML 文档的精确匹配和模糊匹配。

1 树相似度

文中将用户查询信息和 XML 文档都用树型结构表示,即将 XML 文档转换为一棵标记树的数学模型,树节点表示元素或元素属性,树节点的父子关系表示“元素-子元素”或“元素-属性”的关系,叶节点表示文本、关键字等信息^[5-6]。

设 QT 为用户查询树,DT 为文档树, SIMS (QT, DT) 表示查询树与文档树的相似度。查询树 QT 和文档树 DT 的相似度首先考虑 QT 和 DT 包含节点信息的相似度,QT 和 DT 包含节点信息相同的数量直接影响信息的相似度,相同的越多则相似度越高,相同的越少则相似度越低。但是树节点信息的相似度仅仅反映查询树与文档树节点信息的相似性,忽略了树节点之间的结构相似度^[7],因此在考虑树节点信息相似度的基础上,树节点的结构相似度也非常重要,事实上文档中信息都是有关联的^[8]。QT 和 DT 的相似度由树节点信息相似度和树结构相似度共同影响,只有树节点信息越相似且树节点结构越相似时 QT 与 DT 的相似度才大^[9]。

XML 树相似度如式(1)所示:

$$\text{SIMS}(\text{QT}, \text{DT}) = \text{SIMS}(\text{EQT}, \text{EDT}) \times \text{SIMS}(\text{LQT}, \text{LDT}) \quad (1)$$

其中,QT 为查询树;DT 为文档树; SIMS (QT, DT) 表示 QT 和 DT 的相似度; SIMS (EQT, EDT) 表示 QT 和 DT 的树节点信息相似度; SIMS (LQT, LDT) 表示 QT 与 DT 的树结构相似度。

1.1 树节点信息相似度

直观上 XML 树节点信息相同越多,则它们的相似度越高。文中改进了文献[4]提出的相似度方法:

$$\text{SIMS}(\text{EQT}, \text{EDT}) = \frac{|\text{EQT} \cap \text{EDT}|}{|\text{EQT}|} \quad (2)$$

其中, SIMS (EQT, EDT) 表示查询树 QT 和文档树 DT 的树节点信息相似度; EQT 表示 QT 的节点; EDT 表示 DT 的节点。

1.2 树结构相似度

树节点信息相似度仅仅反映了查询树的节点信息与文档树的节点信息的相似性,忽略了树的结构关系,

因此需考虑查询树与文档树的结构关系相似度^[1]。XML 树结构相似度如式(3)所示:

$$\text{SIMS}(\text{LQT}, \text{LDT}) = \sum_{i=1}^{\text{count}(\text{LQT})} \alpha_{uv} \frac{\min(\text{LQT}_{uv}, \text{LDT}_{uv})}{\max(\text{LQT}_{uv}, \text{LDT}_{uv})} \quad (3)$$

其中, SIMS (LQT, LDT) 表示查询树 QT 和文档树 DT 的树结构相似度; u, v 为 QT 中包含的节点; α_{uv} 表示在 QT 中节点 u, v 之间的路径权重; LQT_{uv} 表示在 QT 中 u, v 节点之间的路径所包含的边数量; LDT_{uv} 表示在 DT 中 u, v 节点之间的路径所包含的边数量。

路径权重定义如式(4)所示:

$$\alpha_{uv} = \frac{\text{Layer}_{uv}}{\sum_{i=1}^{\text{LC}(\text{QT})} \sum_{j=1}^{\text{LC}(i)} \text{Layer}_{uv}} \quad (4)$$

其中, Layer_{uv} 表示节点 u, v 路径所在的层数(路径所在层数从树最下层计算,最低层层数为 1,依次每增加一层加 1); $\text{LC}(\text{QT})$ 表示查询树 QT 路径层的数量; $\text{LC}(i)$ 表示第 i 层路径的数量。

这样既保证了所有路径的权重和为 1,又符合距离根节点越近的节点信息越重要的客观事实。客观上,查询树的根节点代表的信息是关键信息,所以被检索的 XML 文档必须保证根节点的匹配。检索时首先进行查询树根节点的匹配。将文档树根节点下所有节点进行解析,构建若干个文档子树,将这些文档子树以相似度的降序排列作为检索结果反馈给用户^[10]。

1.3 相似度计算举例说明

查询树 QT,文档树 DT_1 、 DT_2 和 DT_3 见图 1。

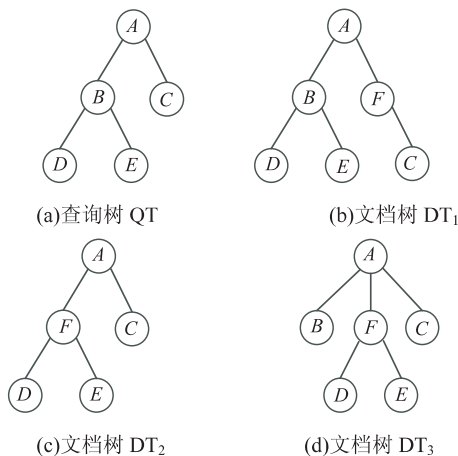


图 1 举例对象

查询树 QT 路径层数量 $\text{LC}(\text{QT})$ 为 2, $\text{Layer}_{AB} = \text{Layer}_{AC} = 2$, $\text{Layer}_{BD} = \text{Layer}_{BE} = 2$, $\text{LQT}_{AB} = \text{LQT}_{AC} = \text{LQT}_{BD} = \text{LQT}_{BE} = 1$ 。根据式(4)计算路径权重:

$$\alpha_{AB} = \frac{\text{Layer}_{AB}}{\text{Layer}_{AB} + \text{Layer}_{AC} + \text{Layer}_{BD} + \text{Layer}_{BE}} = \frac{1}{3} \quad (5)$$

同理, $\alpha_{AC} = 1/3$, $\alpha_{BD} = \alpha_{BE} = 1/6$ 。

(1) 文档树 DT_1 相似度。

其中, $LDT_{AB} = 1, LQT_{AB} = 1, LDT_{AC} = 2, LQT_{AC} = 1, DL_{BD} = QL_{BD} = 1, DL_{BE} = QL_{BE} = 1$ 。

文档树 DT_1 相似度如式(6)所示:

$$SIMS(QT, DT_1) = \frac{5}{6} \cdot (\frac{1}{3} \cdot \frac{1}{1} + \frac{1}{3} \cdot \frac{1}{2} + \frac{1}{6} \cdot \frac{1}{1} + \frac{1}{6} \cdot \frac{1}{1}) = 0.6944 \quad (6)$$

(2) 文档树 DT_2 相似度。

其中, $LDT_{AC} = LQT_{AC} = 1, LDT_{AD} = LQT_{AD} = 2, LDT_{AE} = LQT_{AE} = 2$ 。由于 DT_2 缺少 B 节点, 不存在路径 $A \rightarrow B, B \rightarrow D, B \rightarrow E, D$ 节点最近的祖先是 A 节点, 且路径 $A \rightarrow D$ 存在, E 节点类似。因此路径 $A \rightarrow D$ 和 $A \rightarrow E$ 的权重计算要对路径 $A \rightarrow C$ 进行权重分割, 最后求和。权重分割原理 $\alpha = \alpha_{AB}/n$ (n 为节点 B 的子节点数量)^[11]。在 DT_2 中, n 为 2, α_{AD} 定义如式(7)所示:

$$\alpha_{AD} = \alpha_{BD} + \frac{\alpha_{AB}}{2} = \frac{1}{3} \quad (7)$$

同理, $\alpha_{AE} = 1/3$ 。

文档树 DT_2 相似度如式(8)所示:

$$SIMS[QT, DT_2] = \frac{4}{5} \cdot (\frac{1}{3} \cdot \frac{1}{1} + \frac{1}{3} \cdot \frac{2}{2} + \frac{1}{3} \cdot \frac{2}{2}) = 0.8 \quad (8)$$

(3) 文档树 DT_3 相似度。

其中, $LDT_{AB} = LQT_{AB} = 1, LDT_{AC} = LQT_{AC} = 1, LDT_{BD} = 3, LQT_{BD} = 1, LDT_{BE} = 3, LQT_{CE} = 1$ 。

文档树 DT_3 相似度如式(9)所示:

$$SIMS[QT, DT_3] = \frac{5}{6} \cdot (\frac{1}{3} \cdot \frac{1}{1} + \frac{1}{3} \cdot \frac{1}{1} + \frac{1}{6} \cdot \frac{1}{3} + \frac{1}{6} \cdot \frac{1}{3}) = 0.6481 \quad (9)$$

2 算法比较分析

选取具有代表性的文献[6-8]中提出的算法, 以图2为研究对象进行比较研究, 验证文中相似度算法的有效性。其中, 文献[7]路径权重: $\alpha_{AB} = 0.25, \alpha_{AC} = 0.25, \alpha_{BD} = 0.25, \alpha_{BE} = 0.25$ 。结果如表1所示。

表1 匹配度对比

查询树	文档树	文献[6]	文献[7]	文献[8]	文中
QT	DT ₁	0.750 0	1.000 0	0.666 7	0.833 3
	DT ₂	0.687 5	0.875 0	0.666 7	0.694 4
	DT ₃	0.500 0	0.666 7	0.666 7	0.648 1
	DT ₄	0.300 0	0.800 0	0.640 0	0.800 0
	DT ₅	0.750 0	1.000 0	1.000 0	1.000 0

分析表1可知, 文献[6-7]的相似度算法都存在一定缺陷。QT和DT₅树节点信息和树结构都完全相

同, 匹配度应该等于1, 但文献[6]的相似度小于1, 与客观事实不符; QT和DT₁树结构不一致, 并且DT₁有F节点冗余信息, 文献[7]算法的相似度却等于1, 与客观事实不符。通过比较分析, 文中算法明显优于其他三种算法, 且符合客观事实。研究发现文档树的有效信息数量与相似度并不严格保持正比, 有效信息量多并且树结构相似度高的文档树相似度才高, 检索结果更接近查询信息。总之, 相似度越接近1效果越好。

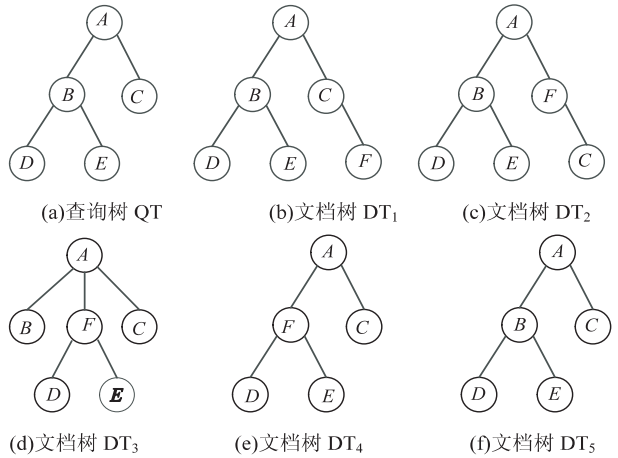


图2 研究对象

3 实验验证

将文献[6]、文献[7] (路径权重取平均值)、文献[8]中提出的算法与文中算法进行比较。实验数据: 500个XML格式的构件信息描述文档; 实验环境: i5-3210M处理器, 4G内存, Windows 7操作系统; 实验策略: Java语言在Eclipse3.7集成开发环境下以树节点按层优先搜索算法实现, 图表显示采用开源插件org.swtchart_0.9.0^[12-13]。具体结果如图3~5所示。

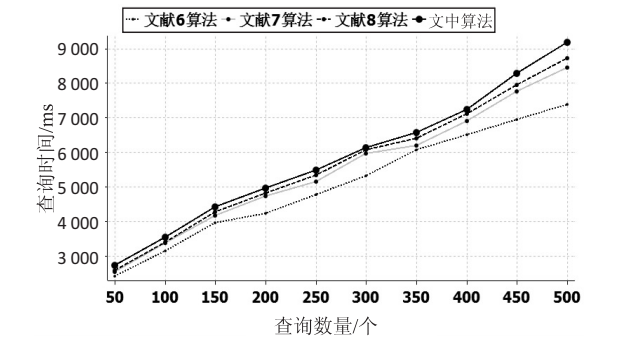


图3 查询效率

从查询效率、查准率和查全率三个性能指标对上述四种算法进行比较。为了避免算法的局限性, 给定了20棵完全不同的XML查询树, 性能曲线上的每个点表示20棵查询树的平均值^[14]。

图3说明在查询效率上文算法优于其他三种算法, 并且随着XML文档数量的增加, 查询速度优势逐渐变大; 图4说明在查准率上文算法高于其他三种

算法,主要原因是文中算法采用了距离根节点越近的路径代表信息越重要的策略,符合信息组织主次分明的客观事实;图5说明在查全率上文中算法比其他三种算法理想,文中算法从节点信息和结构信息两个方面考虑,增加了信息匹配率,提高了查全率。在查准率和查全率上,随着XML文档数量的增加,四种算法都逐渐下降,其他三种算法的下降趋势比文中算法较明显,符合客观事实。

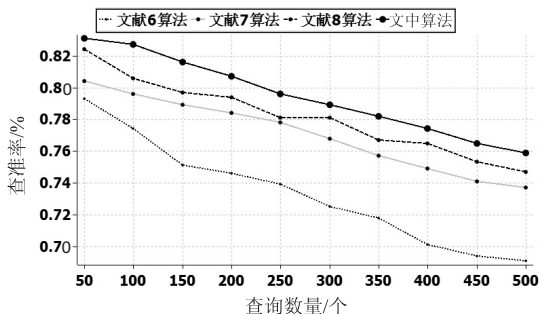


图4 查准率

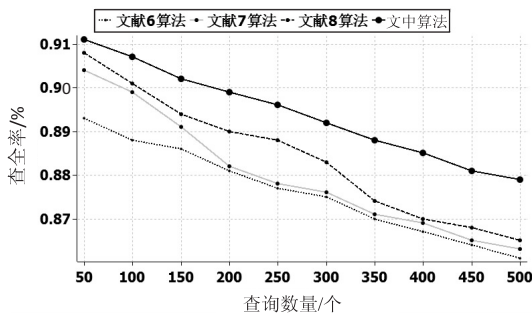


图5 查全率

4 结束语

针对XML文档信息的有效查询效率较低的问题,提出了基于路径权重的XML文档相似度匹配算法。依据日常工作生活中编辑信息的习惯,最重要的信息放在显著位置,例如XML的树根节点,根据信息的重要性依次排列在树的各个层次,距离根节点越近节点表示的信息越重要,依次排列,最低层信息的重要性最小。根据有效路径在查询树的层次自动计算该有效路径的权重,克服了人工指定路径权重的缺陷,计算简单有效。该算法有效信息量和有效节点间的结构关系共同影响查询树和文档树的相似度,克服了仅仅依靠信息匹配、忽略信息之间关联的局限性,使查准率和查全率得到显著提高。实验结果表明,该算法能够帮助用户方便快速地查询到有效信息,符合用户的需求。

参考文献:

- [1] Poonia A M, Jyothi V L. XML document retrieval by developing an effective indexing technique[C]//Proc of sixth international conference on advanced computing. [s. l.]: IEEE, 2014:120-123.
- [2] 魏珂,任建华,孟祥福. 基于查询片段松弛的XML小枝近似查询方法[J]. 小型微型计算机系统, 2013, 34(3): 508-514.
- [3] Rekha M, Rani N U. Efficient extraction of frequent elements from XML document using XML tree pattern matching[J]. International Journal of Advance Research in Computer Science and Management Studies, 2014, 2(3): 54-59.
- [4] 牛大伟, 苏龙超, 韩雨童, 等. 基于扩展倒排索引的不确定XML关键字查询算法[J]. 计算机应用与软件, 2015, 32(4): 247-251.
- [5] Liao H, Li X, Chen J. An accurate identification of extended XML tree pattern for XQuery language[J]. International Journal of Database Theory and Application, 2014, 7(5): 211-226.
- [6] 朱菁华, 王晓玲. 基于扩展查询表达式的XML关键字查询[J]. 计算机工程, 2014, 40(10): 25-31.
- [7] 张苗, 惠小强. 一种快速的XML文档验证算法[J]. 计算机技术与发展, 2015, 25(8): 123-127.
- [8] Piernik M, Brzezinski D, Morzy T. Clustering XML documents by patterns[J]. Knowledge and Information Systems, 2016, 46(1): 185-212.
- [9] 刘显敏, 李建中. 基于键规则的XML实体抽取方法[J]. 计算机研究与发展, 2014, 51(1): 64-75.
- [10] 于亚君, 姜瑛. 一种XML的树匹配改进方法[J]. 计算机工程与应用, 2012, 48(20): 177-181.
- [11] Piernik M, Brzezinski D, Morzy T, et al. XML clustering: a review of structural approaches[J]. The Knowledge Engineering Review, 2015, 30(3): 297-323.
- [12] 郭宪勇, 陈性元, 邓亚丹. 基于多核处理器的VTD-XML节点查询执行性能优化[J]. 计算机科学, 2014, 41(2): 179-181.
- [13] Mohan G B, Ravi T, Chandra J L E, et al. Duplicate detection in XML data using extended sub tree similarity function[J]. International Journal of Applied Engineering Research, 2015, 10(3): 7325-7334.
- [14] 覃章荣, 岑龙科, 任新文, 等. 基于中心实体逻辑分组的XML关键字查询算法[J]. 计算机工程与设计, 2014, 35(6): 2218-2223.