第41卷 第8期 西 安 交 通 大 学 学 报 Vol.41 №8
2007年8月 JOURNAL OF XI′AN JIAOTONG UNIVERSITY Aug. 2007

八邻域网格聚类的多样性XML文档近似查询算法

衡星辰[1]，罗俊颉[2,3]，郭俊文[1]，覃征[1]，邵利平[1]
(1.西安交通大学电子与信息工程学院，710049，西安；2.陕西省人工影响天气办公室，710014，西安；
3.西安交通大学软件学院，710049，西安)

摘要：提出了一种基于八邻域网格聚类的多样性XML近似查询算法.首先给出了支持XML文档间语义距离计算的3种编辑操作代价模型，再利用XML文档间的语义距离建立XML文档的向量模型并设计基于八邻域网格的XML文档聚类算法，进而利用聚类过程中得到的物理和逻辑聚类中心对静态有序选择算法的查询评估策略进行优化，这样做只需定位聚类中心所在组群的局部范围，并在该范围内进行目标查询，而无需遍历整个XML数据库，从而快速返回满足用户需求的查询结果.经汽车外形智能化设计实验表明，所提算法的查询速度比静态有序选择算法平均提高了3～4倍.

关键词：多样性；近似查询；语义距离；八邻域；静态有序选择
中图分类号：TP391 **文献标识码**：A **文章编号**：0253-987X(2007)08-0907-05

Approximate Query Algorithm Based on Eight-Neighbor Grid Clustering for Heterogeneous XML Documents

Heng Xingchen[1], Luo Junjie[2,3], Guo Junwen[1], Qin Zheng[1], Shao Liping[1]
(1. School of Electronics and Information Engineering, Xi′an Jiaotong University, Xi′an 710049, China; 2. Weather Modification Office of Shaanxi Province, Xi′an 710049, China; 3. School of Software, Xi′an Jiaotong University, Xi′an 710049, China)

Abstract: An approximate query algorithm based on eight-neighbor grid clustering for heterogeneous XML documents is proposed. Firstly, the cost models of three editing operations used to compute semantic distance between XML documents are given. Secondly, by using the semantic distance, vector models of XML documents are built, and eight-neighbor grid based clustering algorithm for XML documents is designed. Thirdly, the query evaluation strategy for the static order-selective algorithm is optimized with the physical and logical clustering centers obtained from the process of clustering, thus the cluster center can only be located and queried within a local area of document groups without extending all over XML documents, and the query results satisfying the users' requirements can be returned rapidly. The experiments of intelligent design of automobile shape show that compared to static selectivity order algorithm, the efficiency of the algorithm is increased averagely by 3 - 4 times.

Keywords: heterogeneous; approximate query; semantic distance; eight-neighbor; static selectivity order

可扩展标记语言(XML)是基于Internet元数据的置标语言，它为Internet上的数据描述和交换提供了一种事实上的标准，可作为一种通用格式来存储和交换多种数据源的数据信息.目前，针对XML文档半结构化数据查询的研究主要包括查询重写策略[1-5]、计划松弛策略[6-8]和数据松弛策略[9]

收稿日期：2006-11-15. 作者简介：衡星辰(1980～)，男，博士生；覃征(联系人)，男，教授，博士生导师. 基金项目：国家重点基础研究发展规划资助项目(2004CB719401)；国家自然科学基金资助项目(60542004).

等，其中比较有代表性的算法是动态有序罚分算法[5]、静态有序选择算法[8]和动态加权剪枝算法[6]. 随着XML数据库规模的扩大，静态有序选择算法已不能及时地对查询结果按分值进行排序，从而使查询效率降低.

文档聚类是数据挖掘中的一项重要内容，它不但可以提高信息检索系统的查准率和查全率，还可以利用组织搜索引擎返回的结果，自动产生文档的层次簇或类，并利用簇或类对新文档进行归类. 因此，本文提出了一种基于文档聚类的多样性XML文档近似查询算法，试图解决上述问题.

1 基于语义的XML文档树间距

计算两棵树之间的距离等价于将一棵树转化为另一棵树时为计算所需要的一组编辑操作序列付出的最小代价[10]. 文本在文献[10]的基础上，在考虑了XML数据库中的根结点、叶子结点和中间结点的语义信息前提下，给出了基于语义的3种编辑操作代价的计算模型，而关于XML文档树间距的求解算法，限于篇幅，本文不再阐述.

1.1 重命名代价

重命名代价是指，用一个结点的标记去更新其他结点的标记所付出的代价. 例如，要将结点 n_1 被其祖先结点 n_2 所替换，计算步骤为：①根据XML模式先求出 n_1 和 n_2 间的最短路径 $p_{\min}(n_1, n_2)$；②统计 $p_{\min}(n_1, n_2)$ 经过的结点；③将这些结点标记的权重累加后作为重命名所付出的代价. 若路径上没有其他结点，则将结点 n_1、n_2 的权重之差作为代价，称之为“路过0问题”；若路径上出现同名结点，这说明结点 n_1、n_2 分别处于同名的2个结点，但所代表的语义上下文有所不同，此时的代价定义为无穷大，并称之为“再次路过问题”.

如何定义XML文档树中出现的每个结点的标记权重，是本文需要解决的问题. XML文档树中的一个结点标记的权重与3个因素有关：①前结点所处的位置；②结点出现的频率；③结点的逆文档频率. 在XML文档树中，结点所处的位置越高，出现的频率、逆文档频率越大，则该结点标记的权重就越大，计算式为

$$W(n_i)=\alpha\beta_{\text{naming}}\sum_{t=1}^{N}(G_t(n_i)/N)\,(N/N_{\text{normal}})\,F_{n_i}^{\text{ID}} \tag{1}$$

式中：N 表示结点 n_i 在XML数据库中出现的次数；N_{normal} 是结点出现的标准次数，即预先统计得出的所有结点出现的平均次数；N/N_{normal} 表示结点 n_i 出现的相对频倍，它的值越高，说明结点 n_i 出现的频率越大，结点标记的权重也越大；α 为平衡因子，用来调整结点与结点构成的结构谓词之间的频度差异；β_{naming} 为重命名因子，用来调整因为环境上下文引起的权重与重命名代价之间的转换等价度；$F_{n_i}^{\text{ID}}$ 表示结点 n_i 的逆文档频率，是衡量词语重要性的参数，即

$$F_{n_i}^{\text{ID}}=\lg(M/M_{n_i}) \tag{2}$$

其中，M 表示整个XML数据库中包含的XML文档数，M_{n_i} 表示包含结点 n_i 的XML文档数；$G_t(n_i)$ 是度量结点 n_i 在第 t 次出现时所在位置的重要性函数，即

$$G_t(n_i)=L_{n_i}/L_{\text{tree}} \tag{3}$$

其中，L_{n_i} 表示结点 n_i 在当前XML文档树中的层数，L_{tree} 表示当前文档树的总层数.

1.2 删除代价

由于一个结点的权重与该结点在XML数据库中出现的频率成正比，那么结点的权重越大，删除它所付出的代价就越小. 设结点权重的上限值为 $W_{\max}$，那么删除一个结点 n_i 的代价为

$$C_{\text{del}}(n_i)=\alpha\beta_{\text{del}}(W_{\max}-w_i) \tag{4}$$

式中：w_i 为结点 n_i 的权重；β_{del} 为删除因子，用来调整因环境上下文引起的权重与删除代价之间的转换等价度. 在特殊情况下，结点权重取上限值，则删除代价为0.

1.3 插入代价

插入结点就相当于在查询树中增加新的结点，插入结点的代价与新增结点的权重成正比，也就是说结点在XML数据库中出现的频率越大，则插入它的代价就越大. 插入结点 n_i 的代价为

$$C_{\text{inse}}(n_i)=\alpha\beta_{\text{inse}}w_i \tag{5}$$

式中：β_{inse} 为插入因子，用来调整因环境上下文引起的权重与插入代价之间的转换等价度.

对同样一个结点，本文计算编辑操作代价的结果是不同的，所以应用编辑操作来求解XML文档的树间距与2个XML文档的顺序相关，即 $d(s_i, s_j)\neq d(s_j, s_i)$. 为了便于计算，可按

$$d^{\text{Sema}}(s_i, s_j)=d^{\text{Sema}}(s_j, s_i)=(d(s_i, s_j)+d(s_j, s_i))/2 \tag{6}$$

来求解XML文档间的语义距离，使其与文档的顺序无关. 在式(6)中，s_i 和 s_j 分别代表2棵XML文档树，d 表示应用上述3种编辑操作求解 s_i 相对于 s_j 或 s_j 相对于 s_i 的树间距离，d^{Sema} 代表最终得到的

s_i 与 s_j 之间与求解顺序无关的语义距离.

2　多样性 XML 文档聚类算法

2.1　XML 文档向量化

从 XML 数据库中抽取 n 组且每组包含 k 个 XML 文档构成一个参照集 $R_S=\{(r_1,\cdots,r_k)^1,(r_{k+1},\cdots,r_{2k})^2,\cdots,(r_{(n-1)k+1},\cdots,r_{nk})^n\}$，其中$(r_i,\cdots,r_t,\cdots,r_{i+k-1})$的选取尽可能按照差异最大化的原则，这个差异由 XML 文档间距来评定. 通过$(r_i,\cdots,r_t,\cdots,r_{i+k-1})$，将 XML 数据库中的每一个 XML 文档 D 映射为一个 k 维的距离向量 $\boldsymbol{V}_D$，$\boldsymbol{V}_D$ 中的第 t 维坐标可以通过计算 D 与 r_t 之间的语义距离获得，即 $V_D[t]=d^{\mathrm{Sema}}(D,r_t)$. 这样，通过参照集 R_S，D 可被映射为 n 个 k 维的 $\boldsymbol{V}_D$，再将这 n 个 $\boldsymbol{V}_D$ 的平均值作为每个 D 在 k 维坐标平面上的投影 $\boldsymbol{V}_D[x_1,\cdots,x_k]$.

2.2　八邻域网格的 XML 文档聚类算法

若给定多样性 XML 数据库的规模 m，则聚类算法的主要步骤如下.

(1)利用 2.1 节的 XML 向量化方法，将 XML 数据库中的 m 个 XML 文档映射为二维坐标平面上的样本点，同时将二维平面进行等间距网格划分.

(2)将所有的样本点缓冲到数据缓冲列表中，聚类类别为 0.

(3)若数据缓冲列表为空，转步骤(10)，否则转步骤(4).

(4)取数据缓冲列表的首部元素可得到所在网格，若元素点落入网格垂直分界线，则属于下一个网格；若元素点落入网格水平分界线，则属于右面的网格.

(5)若网格内样本点数大于最少样本点数 $P_{\min}$，则将所在网格的坐标加入到八邻域连接缓冲列表中，否则将该网格内的所有样本点标为孤立点.

(6)若邻域连接缓冲列表为空，表明在当前类中已经找到了所有样本点，聚类类别增加 1，上转步骤(3).

(7)取邻域连接缓冲列表中的首部网格坐标，用当前聚类类别的标识颜色勾画结果，并显示区域在该网格内的样本点，且从数据缓冲列表中删除该网格内的所有样本点坐标.

(8)依次检测该网格周围的 8 个邻域网格，即$[i-1,j-1]$、$[i-1,j+1]$、$[i+1,j+1]$、$[i+1,j-1]$、$[i-1,j]$、$[i+1,j]$、$[i,j-1]$、$[i,j+1]$，若网格内样本点数大于 $P_{\min}$，将网格的坐标加入到邻域连接缓冲列表的尾部，否则将网格内的所有样本点丢弃.

(9) 上转步骤(6).

(10)依次对形成的每一个文档类的聚类中心进行计算.

2.3　选择聚类中心

聚类中心是指，对多样性 XML 数据库进行聚类后形成的各个组群中处于中心位置的 XML 文档. 中心文档的合理选择直接关系到本文算法的执行效果，在组群中，理想的聚类中心应选择查询答案最集中的位置，下面给出 2 种确定聚类中心的方法.

方法 1　将每个组群中处于物理位置最中心的一个 XML 文档确定为聚类中心，该方法可以通过计算 XML 文档间的欧氏距离来估算，简单、易于实现，但灵活性欠缺，称之为物理中心法.

方法 2　以物理中心作为基准点，在查询过程中根据组群中的查询结果分布动态地选取新的聚类中心，该方法可使查询结果的准确度更高，更贴近用户的查询需求，灵活性比较好，但计算代价较高，称之为逻辑中心法.

3　八邻域网格聚类的多样性 XML 近似查询算法

3.1　静态有序选择算法

静态有序选择(SSO)算法[8]主要由查询树松弛策略、罚分策略、近似匹配评估策略和最优选择估计策略 4 部分构成. 本文研究的重点是基于近似匹配评估策略的优化方法，其主导思想是：利用由原始查询模式树变换得到的查询谓词闭包对 XML 文档树进行精确匹配，如果匹配的结果数大于等于 K，则预示着有 K 个最优解已形成，算法终止；若匹配的结果数小于 K，利用最优选择估计策略，从查询谓词闭包中去除若干个谓词(相当于对查询树进行松弛)，然后继续进行精确匹配，直到精确匹配的结果数大于等于 K，或者查询谓词闭包为空为止.

3.2　XML 文档聚类的 SSO 优化算法

本文在 SSO 算法的基础上进行了改进，即在离线状态下，通过引入 XML 文档聚类算法对遵循多种 DTD 的多样性 XML 数据库进行聚类划分，形成以多个独立的 XML 文档为聚类中心的类群，而在 SSO 算法的近似匹配估算阶段，通过聚类中心来指导查询谓词闭包对这些组群进行深度和广度的有限搜索，算法流程如图 1 所示.

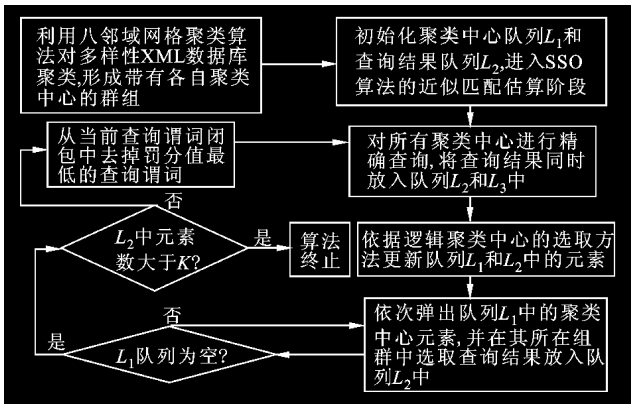


图1 八邻域网格聚类的多样性 XML 近似查询算法

4 实 验

4.1 多样性 XML 文档的聚类算法实验

在本实验中,首先利用自行设计的 XML 数据发生器生成 3 个汽车外形 XML DTD 的文档多样性 XML 数据库,其中包含 420 个 XML 文档,总计 200 000 个文档元素、20 000 个关键词,而 200 000 个文档元素共享了 150 个不同的元素名(结点标签),20 000 个关键词共出现了 100 000 次,并呈现均匀分布.接着,利用本文基于语义距离的 XML 文档树间距的计算方法,将所有文档间距分布在[0,1]区间,进而将所有文档间距均乘以 300 并取整,从中抽取 10 组文档间距最大的 XML 文档组成参照集,每组由 2 个 XML 文档构成.利用参照集,将 XML 数据库中的 420 个 XML 文档投影到 300×300 的二维坐标系上,再利用八邻域网格的 XML 文档聚类算法对 420 个 XML 文档的 XML 数据库进行聚类,聚类结果图 2 所示.

实验结果表明,本文的聚类算法基本上成功地将 420 个 XML 文档聚为源于 3 种 XML DTD 的 3 个文档类,其中包含 5 个孤立点、13 个噪声点,同时确定了每个类别的聚类中心.

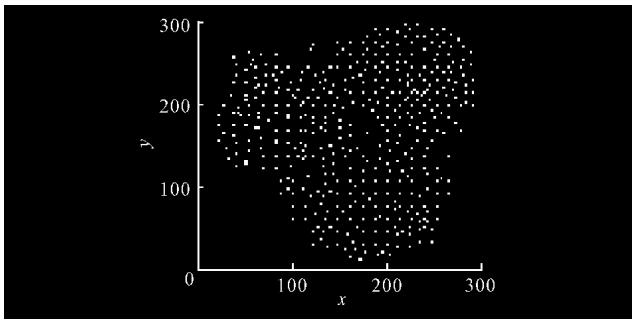


图2 多样性 XML 数据库聚类结果

4.2 多样性 XML 文档近似查询算法实验

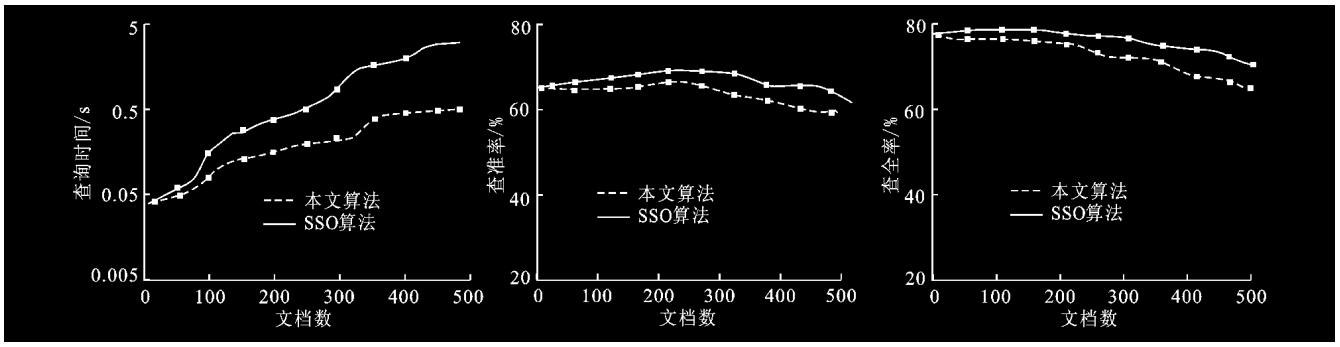
针对不同数据集和查询测试了本文算法,同时根据汽车外型 XML DTD 设计了 10 条具有代表性的类 XPath 查询路径,如表 1 所示.基于表 1 的 10 条类 XPath 查询路径,利用本文算法和 SSO 算法分别对包含 500 个 XML 文档的多样性 XML 数据库进行查询速度、查准率和查全率比较,结果如图 3 所示.

图 3 表明,本文算法在查询速度上优于 SSO 算法,随着 XML 数据库规模的增大,这种优势更加明显.这是因为,本文算法通过对 XML 数据库预先进行聚类、选取聚类中心,所以在 SSO 算法的近似估算阶段不需要遍历整个数据库,可直接定位在聚类中心所在的局部范围内进行目标结果的快速查询,从而比 SSO 算法的查询速度平均提高了 3～4 倍.另外,从图 3 还可以看出,SSO 算法的查准率、查全率性能略优于本文算法,这是因为 SSO 算法是随机地从 L_1 队列中输出元素,不同的聚类中心的输出顺序决定了预先是在哪些聚类中选取查询结果,并且直接利用预先计算好的查询结果和聚类中心的距离 d 间接地作为影响近似程度的标准,所以有可能在查询结果数大于 K 时,仍有一些与查询条件匹配的

表1 类 XPath 查询路径

路径标识	类 XPath 查询路径
Q1	Car/door/leftfront
Q2	Car/frontglass/FGKeypoint
Q3	Car/door/@coverbord/EHHlength
Q4	Car/carID/@frontview/wholewidth("1666")
Q5	Car_body[brand("sampas"),frontform("round"),rear("round")]
Q6	Car[@brand("golf"),overwidth("1825"),bootlength]
Q7	Car(door,coverbord[@EHHlength("1044"),@EHHwide("1200"),canopy])
Q8	Car[sideglass,frontglass(FGKeypoint("0-626-1245"))]
Q9	Car/door[@leftfront("750"),@rightfront("850")]//glass[FGKeypoint("0-626-1245")]
Q10	Car//carID("passat")/frontview[wholewidth("1666"),backview[wheelbase("2731")]]

注:Q1～Q4 属于类 XPath 绝对路径("/"代表父子关系标识符);Q5～Q8 属于包含拥有关系的类 XPath 绝对路径("[]"代表拥有关系标识符);Q9、Q10 属于包含拥有关系的类 XPath 相对路径("//"代表子孙后代关系标识符).



(a)查询速度　　(b)查准率　　(c)查全率

图 3　算法性能比较

聚类中心所在的聚类群没有被查到，但本文算法循环更新了 L_1 队列中保存的聚类中心，使得查准率和查全率的降低度控制在可以接受的范围内. 对于大规模的多样性 XML 数据库，本文算法的查询效率更具优势，它更适用于对查询的及时性要求较高，而对查询精度要求不高的用户. 实验结果同时表明，α、β_{inse}、β_{naming} 和 β_{del} 的取值会直接影响查询结果的准确率，这些值与当前采用的 XML 文档样本集有一定的关系，均取 0.5 是一种折衷的方法.

5　结　论

本文提出了基于文档聚类的多样性 XML 近似查询算法，该算法通过基于语义的 XML 文档树间距离的计算方法对 XML 数据库进行聚类划分，建立了带有多个聚类中心的聚类集，再利用这些聚类中心来指导 SSO 算法对 XML 数据库的遍历模式，使得 SSO 算法不需要查询整个 XML 数据库就能够返回用户需要的答案. 通过“九七三”智能化汽车外形设计实验表明，利用本文算法可提高查询效率，尤其在处理大规模 XML 数据库时，效果更加明显. 下一步的工作是，结合用户的查询需求和查询谓词罚分，来研究聚类中心的选择策略以及代价模型的参数学习算法，利用 XML 的索引技术提高 XML 数据的查询效率.

参考文献：

[1] Schlieder T. Similarity search in XML data using cost-based query transformations[C]// Proceedings of the 4th International Workshop on the Web and Databases. New York: ACM Press, 2001: 19-24.

[2] Chinenyanga T T, Kushmerick N. Expressive and efficient ranked querying of XML data [C]// Proceedings of the 4th International Workshop on the Web and Databases. New York: ACM Press, 2001: 1-6.

[3] Delobel C, Rousset M C. A uniform approach for querying large tree-structured data through a mediated schema [C]// International Workshop on Foundations of Models for Information Integration. Berlin: Springer-Verlag, 2001: 267-298.

[4] Fuhr N, Grossjohann K. XIRQL: an extension of XQL for information retrieval [C]// Proceedings of the SIGIR 2000 Workshop on XML and Information Retrieval. New York: ACM Press, 2000: 1-5.

[5] Schlieder T. Schema-driven evaluation of approximate tree-pattern queries [C]// 8th International Conference on Extending Database Technology. Berlin: Springer-Verlag, 2002: 514-532.

[6] Amer-Yahia S, Cho S, Srivastava D. Tree pattern relaxation [C]// Proceedings of 8th International Conference on Extending Database Technology. Berlin: Springer-Verlag, 2002: 496-513.

[7] Kilpelainen P. Tree matching problems with applications to structured text databases [D]. Helsinki, Finland: University of Helsinki, 1992.

[8] Amer-Yahia S, Lakshmanan L V, Pandit S. FleXPath: flexible structure and full-text querying for XML [C]// Proceedings of International Conference on Management of Data. New York: ACM Press, 2004: 83-94.

[9] Damiani E. The APPROXML tool demonstration [C]// Proceedings of 8th International Conference on Extending Database Technology. Berlin: Springer-Verlag, 2002: 753-755.

[10] Tai K C. The tree-to-tree correction problem [J]. Journal of the ACM, 1979, 26 (3): 422-433.

（编辑　苗凌）