Top-k Ranking of Preference Query Results for XML Based on Similarity of Contextual States

Wei YAN*, Fei SUO

Computing Center, Liaoning University, Shenyang 110036, China

Abstract

The ranking method of XML preference query results is an important issue. For user’s preference query request, the query results are often user’s interested information. In order to satisfy the user’s personalized query requirements, this paper proposes the XML contextual preference model. When a user submits a query request, the user’s preference query results depend on the contextual conditions of the query. The context is modeled as a hierarchical structure. Based on similarity of contextual states, this paper clusters the preference queries with the similar contextual states. Furthermore, this paper proposes the top-k ranking method of preference query results. Finally, this paper presents the results of experimental evaluations which indicate the efficiency of the proposed approach.

Keywords: XML; Contextual Preference; Preference Query; Top-k Ranking

1 Introduction

At present, the XML data is widely applied on the Internet. The user often uses the XML document to store the information. The XML data becomes the de facto standard of information representation and information integration on the Web. With the wide application of XML data, the query processing of XML data becomes more and more important. The existing query language of XML uses path expression to query data. The XML query processing model does not consider the influence of user’s preferences on the query results. However, the personalized query request is proposed by the user, its query results are affected by the user’s preferences. Therefore, it is important to research the issue of preference query for XML data.

The technology of preference query gradually becomes a hot issue in the field of information retrieval. The research work about preference representation and query processing can be divided into two categories: one kind is qualitative approach, which expressed as a binary relation of partial ordering directly [1]; another kind is the quantitative approach, which used scoring function to represent the relation of preference indirectly [2]. In order to represent and handle user’s preference, Stefanidis et al. proposed the concept of context in the user’s preference query [3].

*Project supported by the National Nature Science Foundation of China (No. 61370075).
*Corresponding author.
Email address: yanweihome@126.com (Wei YAN).
The different users have different preferences depending on context. Context is modeled as a set of multidimensional attributes. In [4], Bunningen et al. proposed a knowledge-based context-aware preference model. The contextual preference expressed as the form of preference rules \( (C, P) \), in which the context \( C \) and preference \( P \) are logic expression. In [5], Amer-Yahia proposed a framework for defining user profiles and for enforcing them during personalizing XML search. In [6], the authors proposed an XML structural join-based twig caching scheme that allows preference queries to reuse the most beneficial structural join results.

Recently, the research work of preference query has a lot of research results. Arvanitis et al. proposed a preference-aware relational query answering system PrefDB, which used an extended relational data model and algebra to express different flavors of preferential queries [7]. In [8], Farnan et al. created a framework that empowers users with the ability to specify constraints on the kinds of plans that can be produced by the preference-aware optimizer PASQL to evaluate their queries. In [9], the authors adopted the relational database model, and proposed extensions that are specialized to handle preference data. The authors introduced a special type of a relation that is designed for preference data, and described operators on preference relations that can be embedded in SQL statements.

Based on the above research work, this paper proposes a ranking method of preference query for XML data. Firstly, this paper proposes the XML contextual preference model, which supports the user’s personalized query for XML data. Moreover, this paper presents the method of clustering contextual preference based on similarity between contextual states. Furthermore, this paper proposes the top-k ranking method of preference query results. The method of preference query for XML data can satisfy the user’s personalized query requirements and improve the interaction with the system.

2 Basic Knowledge

2.1 Modeling context

The definition of context [3] is as follows:

**Definition 1 (Domain)** A domain is an infinite countable set of values.

**Definition 2 (Contextual Attribute)** The contextual attribute is a countable set of the context names. Each contextual attribute \( C_i \) is represented by a context name and a domain \( \text{dom}(C_i) \).

**Definition 3 (Contextual Parameter)** The contextual parameter \( c_i \) is a finite set of contextual attribute \( C_i \). For a given application \( X \), the context environment \( C_X \) is a set of \( n \) contextual parameters \( \{c_1, c_2, \ldots, c_n\} \).

**Definition 4 (Contextual State)** The contextual state is an assignment of values to contextual parameters. The contextual state is a tuple with the values of the contextual parameters at time point \( t \), \( CS_X(t) = \{c_1(t), c_2(t), \ldots, c_n(t)\} \), where \( c_i(t) \) is the value of the contextual parameter \( c_i \) at time point \( t \).
2.2 Hierarchical structure of the context

This section establishes a context model, which is set of multidimensional attribute. The context is modeled as a hierarchical structure. The hierarchy of contextual attribute is the structure $(L, \prec)$: $L = (L_1, L_2, \ldots, L_{m-1}, ALL)$ of $m$ levels and $\prec$ is a partial order among the levels, such that $L_1 \prec L_2 \prec \ldots \prec L_{m-1} \prec ALL$. The upper bound of the hierarchical structure is the level $ALL$, and the lower bound of the hierarchical structure is the level $L_1$. The symbol $\text{dom}_{L_j}(C_i)$ represents the domain value of the contextual attribute $C_i$ of the level $L_j$. In this contextual hierarchy, the level $L_1$ represents the contextual parameter, and the level from $L_2$ to $ALL$ represents the contextual attribute.

![Hierarchical structure of the context](image)

In this paper, we consider the five contextual attributes: accompanying_people, location, time, age, and sex. Fig. 1 describes the hierarchical relationship of the context.

3 XML Contextual Preference Query Model

The XML document is semi-structured data format, which has hierarchical relationship and nested structure. The XML document can be represented as the data tree $T = (N, E, r)$ with a root node, where $N$ is set of data nodes, $E$ is set of edge between data nodes, and $r$ is the root node of data tree. The XML data node can be represented as a triple $<\text{id}, \text{label}, \text{text}>$, where $\text{id}$ uniquely identifies the data node, $\text{label}$ is the name of the corresponding element or attribute node, and $\text{text}$ is the corresponding element’s textual content or attribute’s value.

When the user queries XML document, the personalized query results are affected by the context environment of the user’s query. The same user’s query returns different query results under different context. In different context the user’s personalized query generates the different query results according to user’s interest. Thus, this section proposes the XML contextual preference query model.

Given an XML document $D$, the user’s contextual preference query for XML document can be expressed as the form $(cs, Path, score)$. Here, $cs$ represents the contextual state when the
user queries XML document. Path represents the path expression of the XPath query for XML document, and it uses the form of \( Path_1 \land Path_2 \land \ldots \land Path_n \). score represents the scores of user’s interesting degree for query path expression, and it is a value between 0 and 1.

4 Preference Clusters Based on Similarity of Contextual States

Firstly, this section calculates the distance between contextual parameters. In the hierarchical structure of the context, let \( lca(c_1, c_2) \) be the lowest common ancestor of contextual parameter \( c_1 \) and \( c_2 \). The path and depth distance between contextual parameters is defined as follows.

**Definition 5 (Path Distance)** The path distance \( \text{dist}_P(c_1, c_2) \) between contextual parameters \( c_1 \in \text{dom}_{L_j}(C_i) \) and \( c_2 \in \text{dom}_{L_k}(C_i) \): the path distance is equal to 0, if \( c_1 = c_2 \); the path distance is equal to 1, if \( c_1 \) and \( c_2 \) are contextual parameters of the lowest hierarchical level and \( lca(c_1, c_2) \) is the root node of their corresponding hierarchy; the path distance is computed through the \( f_P \) function \((1 - e^{-\alpha \times \rho})\), where \( \alpha > 0 \) is a constant and \( \rho \) is the minimum path length connecting them in the associated hierarchy.

**Definition 6 (Depth Distance)** The depth distance \( \text{dist}_D(c_1, c_2) \) between contextual parameters \( c_1 \in \text{dom}_{L_j}(C_i) \) and \( c_2 \in \text{dom}_{L_k}(C_i) \): the depth distance is equal to 0, if \( lca(c_1, c_2) \) is the root node of their corresponding hierarchy; the depth distance is computed through the \( f_D \) function \((1 - e^{-\beta / \gamma})\), where \( \beta > 0 \) is a constant and \( \gamma \) is the minimum path length between the \( lca(c_1, c_2) \) and the root node of the corresponding hierarchy.

**Definition 7 (Value Distance)** The value distance between two contextual parameters \( c_1 \) and \( c_2 \) is computed as:

\[
\text{dist}_V(c_1, c_2) = \text{dist}_P(c_1, c_2) \times \text{dist}_D(c_1, c_2).
\]

For example, the path distance between contextual parameter Christmas and weekend is \( 1 - e^{-3} \approx 0.95 \) in the Fig. 1, their depth distance is 1, their value distance is \( 1 \times 0.95 = 0.95 \). Whereas contextual parameter holidays and Christmas have value distance equal to \((1 - e^{-1 \times 1}) \times (1 - e^{-1/1})\). This means that the contextual parameter Christmas is more closely related to holidays than to weekend. In both examples, we assume that \( \alpha = \beta = 1 \).

**Definition 8 (State Distance)** Given two contextual states \( cs^1 = (c_1^1, c_2^1, \ldots, c_n^1) \) and \( cs^2 = (c_1^2, c_2^2, \ldots, c_n^2) \), the state distance is defined as:

\[
\text{dist}_S(cs^1, cs^2) = \sum_{i=1}^{n} w_i \times \text{dist}_V(c_i^1, c_i^2)
\]

where, each \( w_i \) is a specific weight of contextual parameter.

The weight of contextual parameter \( w_i \) is associated with the dimensions of the contextual parameter, according to the dimensions of the contextual parameter to assign the appropriate weights.
Based on the above distance between contextual states, the following section presents a method of clustering XML contextual preferences according to the similarity of contextual states. The similar contextual preferences have the same or similar score of interesting degree. This section uses the Euclidean distance formula to calculate the distance between the preference clusters. Assume that \( n \) is the number of preference clusters, \( \tau \) is an order \( \{ \tau_1, \tau_2, \ldots, \tau_n \} \) of preference clusters, \( \rho \) is another order \( \{ \rho_1, \rho_2, \ldots, \rho_n \} \) of preference clusters, \( \tau_i \) and \( \rho_i \) are \( i \)-th order of preference clusters respectively. The distance between order \( \tau \) and \( \rho \) of preference clusters uses the Formula (4) to calculate:

\[
dist_E(\tau, \rho) = \left( \sum_{i=1}^{n} (\tau_i - \rho_i)^2 \right)^{1/2}
\]  

(4)

The following Formula (5) calculates the minimum distance between orders of preference clusters:

\[
\text{mindist} = \min \ dist_E(\tau, \rho)
\]  

(5)

**Theorem 1** Given the minimum distance \( \text{mindist} \) of preference clusters, the contextual state \( cs_i \) of each preference cluster has the following property: Any similar contextual state \( cs_1 \) and \( cs_2 \) of preference clusters, their distance between contextual state of preference clusters \( \text{dist}_s(cs_1, cs_2) \leq \text{mindist} \).

**Proof** If the distance of two preference clusters less than or equal to the smallest distance threshold \( \text{mindist} \), the system merges two clusters. The distance threshold \( \text{mindist} \) of preference clusters represents the maximum distance between similar contextual state \( cs_1 \) and contextual state \( cs_2 \) of preference clusters. Therefore, any similar contextual state of preference clusters, their distance is satisfied \( \text{dist}_s(cs_1, cs_2) \leq \text{mindist} \).

In this way, the distance between the orders of preference clusters is not less than the minimum distance \( \text{mindist} \). The distance between two preference clusters less than or equal to the minimum distance, then the two preference clusters are similar. If the distance between each pair of preference clusters less than the minimum distance between the orders of preference clusters, the system will merge the two preference clusters.

## 5 Top-k Ranking of XML Preference Query Results

This section describes top-k ranking method of preference query results, which provides \( k \) ranking answers to satisfy the requirement of user’s preference query. According to the orders of preference clusters, the algorithm calculates the score of preference clusters in the contextual state, and then calculates the overall score of preference clusters. Based on Fagin’s threshold ranking algorithm [10], the algorithm returns \( k \) answers to the users.

**Algorithm 1** Top-k ranking algorithm.
Input The order of preference clusters \( \delta = \{\delta_1, \delta_2, \ldots, \delta_s\} \), the set of contextual state \( CS_i \subseteq \{CS_1, CS_2, \ldots, CS_s\} \), preference query \( Q \). Let \( B = (table_id, score) \) be a scoring table that can hold \( k \) scores. Let \( L \) be an array of size \( s \) storing the last scores.

Output Top-k answers scoring table.

Step 1 Repeat

Step 2 For \( i = 1 \) to \( s \) do

Step 3 Retrieve the next order of preference clusters from \( \delta \).

Step 4 Compute the initial score of the order of preference clusters: \( (\delta_i, score_0) \leftarrow \text{getNext}(Q_0) \)

Step 5 Update \( L[i] \) with the initial score of order of preference clusters.

Step 6 If \( \delta_i \in \delta \) then

Step 7 Get score of \( \delta_i \) in contextual state \( cs_i \) via a random access: \( score_i \leftarrow \text{getScore}_{cs_i}(\delta_i, Q_i) \)

Step 8 Calculate the overall score in contextual state set \( CS_i \): \( score(\delta_i|CS_i) = \text{Average} \sum_{cs_i \in CS_i} score(\delta_i|cs_i) \)

Step 9 Insert \( (\delta_i, score(\delta_i, CS_i)) \) into the correct position in \( B \).

Step 10 End If

Step 11 End For

Step 12 Until \( B[k].score \geq \prod_{i=1}^{s} L[i] \)

Step 13 Return \( B \).

6 Experiments Evaluation

The experiments use the XMark dataset to evaluate the effectiveness of the ranking method. The XMark dataset [11] stores the data in a large XML document, and simulates an auction site, used to evaluate the query processing capability of XML data in the real application. All the experiments were implemented in JDK 6.0, and performed on a system with 2.8GHz Pentium D processor with 2GB of RAM, and running on windows XP system.

The algorithm of comparison selects the Static Selectivity Order described in [12], henceforth referred to as SSO, which uses the method of selective evaluation to preference query processing. Our preference query and ranking method based on similarity of contextual states, henceforth referred to as PQRM.
6.1 Performance experiment of top-k query

This section evaluates query performance of top-k query. Fig. 2 shows the query execution time of PQRM and SSO method on XMark dataset. The $k$ value of top-k ranking algorithm varies from 50 to 700. It can be seen that the execution time of these ranking methods is increased with $k$ value. The performance of PQRM method is better than the performance of SSO method.

6.2 Top-k precision experiment

The top-k precision measures the ratio of the number of accurate answers among the first $k$ returned results with highest scores to $k$. Fig. 3 shows the comparison of the top-k precision of the PQRM and SSO method. The top-k precision of the PQRM method is always higher than the top-k precision of the SSO method. The average top-k precision of the PQRM method is 0.930, and the average top-k precision of the SSO method is 0.706. The reason is that the SSO method uses the strategy of selective evaluation, which evaluates the number of query results. If the number of query results is less than $k$, the SSO method removes penalty predicates, and then estimates again until the number of query results is $k$. In contrast, PQRM method considers the user’s preference, and clusters the preference queries with similar contextual states, and then quickly ranks the query results, so achieves the higher query accuracy.
7 Conclusions

In this paper, we propose top-k ranking method of XML preference query results based on similarity between contextual states. Firstly, the paper proposes XML contextual preference model to express user’s preference query intention. The user’s preference query results depend on the context environment of query. Moreover, the paper proposes the method of clustering preference query, which is based on similarity of contextual states. Furthermore, the top-k ranking method is proposed, which returned the query results quickly and satisfied the requirements of the user’s query as well. The efficiency of proposed approach is also demonstrated by experimental results. The future research direction is optimization of preference query further.

References