A Survey on Database Queries by using Dynamic Query Forms

Vinayak Jadhav[1], Amrit Priyadarshi[2]

[1, 2]Department of Information Technology, Dattakala Group of Institute of Technology, University of Pune, Pune, Maharashtra, India

Abstract: ***Large and Heterogeneous data is maintained by modern scientific databases and web databases. Over hundreds or sometimes thousands of relations and attributes are contained by these real world databases. Various ad-hoc queries are executed by the user on those databases. But the traditional predefined query forms are not sufficient to solve these ad-hoc query problems. Thus, this paper proposes Dynamic Query Forms (DQF), a new interface for database query forms. This is able to generate the query forms dynamically. The prime functions of DQF includes: 1) capture the preferences of users and rank those query form components. 2) Assist users to make their decisions. The invention of a query form is a user guided and an iterative process. At the end of iteration, the ranking list of form components is automatically generated by the system and the query form is added with the desired form components by the user. The rankings of form components are made using the captured user preferences. After the iteration, a user can fill the query form and view the query result after submitting queries. In such a way, a query form will be dynamically improved till the user satisfaction with the results of the queries. The expected F-measure for the query form goodness measuring is utilized in this approach. In DFQ, for estimating the goodness of a query form, a probabilistic model is developed. The effectiveness and the efficiency of DQF are proved after experimental evaluation and user study.***

Keywords: DQF, databases, query forms, F-measure.

1. Introduction

The most widely used user interface for querying database is 'Query form'. The developers and DBA's in various information management systems have designed and predefined the traditional query forms. The modern databases have become very large and composite with the rapid development of web informatics and scientific databases. In natural sciences, such as diseases and genomics, there are over hundreds of entities for biological and chemical data resources in the databases [1], [2]. Many web databases, like DBPedia and Freebase, usually have over thousands of structured web entities [3], [4]. Therefore, it is hard to structure a set of static query forms to satisfy different ad-hoc database queries on those composite databases. Many current database management and development tools, like SAP and MS Access, lets the user develop customized queries on databases, by providing several mechanisms. Conversely, the development of customized queries totally based upon manual editing's of user [5]. Those hundreds and thousands of data attribute will confuse the user if they are not familiar with database schema in advance. The non-technical users make usage of relational database a challenging task. Therefore, in recent years many researches were focused on database interfaces to assist users to query the relational databases without using SQL.

This paper proposes a Dynamic Query Form system (DQF), is a query interface capable of dynamically producing query forms for the users. Unlike traditional document retrieval, before identifying the final candidate, the users in database retrieval are mostly willing to execute several rounds of action [6]. The important feature of DQF is:

1) During the user interactions, capture the user interest.
2) Iteratively adapt the query forms.

Each of this iteration is made up of two types of user interactions. They are:

1) Query Execution, and
2) Query Form Enrichment.

The work flow of the DQF starts with a basic query form containing very few primary attributes of the database. It is then enriched iteratively by the means of interactions between the user and the system, unless the user is pleased with the query outcome. Mainly the study of query form components and the dynamic production of query forms are done in this paper. This paper proposes a dynamic query form system, generating the query forms in accordance with the run time desire of the user. The system presents an elucidation for the query interface in large and composite databases. The technique applies F-measure for approximation of the goodness of a query form [7]. Using this, we can rank and recommend the probable query form components, so that the users can filter the query form effortlessly. By using the proposed metric, to approximate the goodness of the protrusion and assortment of form components, have developed an efficient algorithms. As the DQF is an online tool and users usually expect quick response, Efficiency is very important.

Here, this paper proposes a dynamic query form generation technique which will assist users to dynamically generate query forms. The key idea is based on user preferences, to use probabilistic model to rank form components. By using both, runtime feedback and historical queries, system captures the user preferences. In the near future, the study should be concentrated upon the extensional use of this approach to non relational data. The remaining paper can be sorted out as: section 2 gives query forms and query results, together referred as Query form interface. Later in the section, we have reviewed the ranking metrics which is essential for the preferences to be given to the queries. And finally, the section ends with the estimation of ranking score.

Section 3 includes the briefly studied conclusion and the future work.

2. Literature Review

Literature on dynamic query forms can be classified into Query forms and query results, both are part of Query form interface, the ranking metrics for user preferences queries, and the estimation of ranking score.

2.1 Query form interface

This consists of two parts as explained in [8], namely Query forms and Query results.

2.1.1 Query forms

We have formally defined the query form in this section. Each of the queries does belong to a SQL query template. As ad-hoc join is not the part of the query form and is basically invisible to users, it is not handled in our approach by dynamic query forms. There are very limited number of options for users, if concerned to 'Aggregation' and 'Order By' in SQL. For instance, 'Order by' could only be 'decreasing order' and 'increasing order' and 'Aggregation' could only be AVG, MAX, and MIN and so on. To include these options, our approach can be easily extended by implementing them as dropdown boxes inside the user interface of the query forms [9].

2.1.2 Query Results

To choose whether a query form is required or not, there is not much time users have to waste on to go to every data instance in the query result. To add to this problem, a huge amount of data instances are generated as the output by many databases. To avoid this 'Many-Answer' problem [10], to show a high level observation of the query results earliest, we only yield a compressed result table. A cluster of actual instances are represented by each instance in the compressed table. Then, to view the comprehensive data instances, the user can briefly click through interested clusters. The compressed high level aspect of query results is proposed in [11]. Many one pass algorithms have been developed for generating the compressed view powerfully [12], [13]. Because of the efficiency issue, we have chosen the incremental data clustering framework [12], in this implementation. Different compressed views for user are proposed by different data clustering methods. Also, different clustering methods are proposed for different data types. We have implemented clustering just to offer a better aspect of the query results for the user. Any other clustering algorithm can be used if necessary.

2.2 Ranking Metrics

Query forms are designed such that it would return the user's desired result. To evaluate the quality of the query result, two traditional measures are available: Recall and Precision [7]. We can use the expected recall and expected precision to evaluate the query forms expected performance, because the query forms are capable of producing different queries for different inputs and different precisions and recalls can be achieved by different queries that outputs different query results. Naturally, current user's interest in query result is expected proportion of expected precision. The interest of user in approximated using the user's clickthrough on display of query results by the query form.

2.3 Estimation of Ranking Score

The ranking score estimation phase consists of only two phases. The two phases are: Ranking projection form components, and second is, ranking selection form components.

2.3.1 Ranking Projection Form Components:

The DQF has provided a two level ranked list for the purpose of projection of components. The first level is ranked entities. This level describes how to rank attributes of each entity and that too locally. The second level is the ranked list of attributes in the same entity. This level describes the ranked lists of attributes in the same entity. Instinctively, the entity should be ranked higher in list, if those entities have more number of high scoring attributes.

2.3.2 Ranking Selection Form Components:

The selection of attributes will be absolutely meaningless, if the selection attributes are not relevant to the currently projected entities. Therefore, for creating the selection components, first, the system should try and find out the relevant attributes. This section is further divided in three phases. The three phases are:

a) **Relevant attribute selection:**
b) In this, the related or similar attributes are selected. This attributes are then grouped.
c) **Ranking selection components:**
d) In this step, the components groups are acquired, taken from the first step. These components are then ranked according to their usage.
e) **Diversity of Selection components:**
f) Two selection components may have a number of redundancies or overlays. Therefore, a high diversity should be provided in order to select the recommended components. Diversity is the recent major topic in recommendation system and web search engines as proposed in [14] and [15]. Though, simultaneously maximizing the precision and the diversity is an NP-Hard problem [14]. In an interactive system, it cannot be implemented efficiently. In this DQF system, it is observed that the same attribute may construct the most redundant components. Hence, for each attribute, only the best selection components are recommended.

3. Conclusion

In this paper, we have proposed an approach for dynamic query form generation. This approach will help users to dynamically generate the query forms. The basic idea is based on a probabilistic model to order form components using the preferences of user. The user preferences are captured by the system using both, historical queries and runtime response. Experimental results have shown that the dynamic approach usually generates higher success rates and easier query forms compared with a static approach [8]. The form component rankings make it easier to customize the query forms for users. As development work, the study can be undertaken to extend our approach to non relational data.

We plan to develop number of methods to capture the user's preferences for the queries instead of click feedback. For example, a text box can be added to input some keyword queries and the query form for the user [9]. At each step, this can be incorporated in the form component rankings.

References

[1] J. C. Kissinger, B. P. Brunk, J. Crabtree, M. J. Fraunholz, B. Gajria, A. J. Milgram, D. S. Pearson, J. Schug, A. Bahl, S. J. Diskin, H. Ginsburg, G. R. Grant, D. Gupta, P. Labo, L. Li, M. D. Mailman, S. K. McWeeney, P. Whetzel, C. J. Stoeckert, J. . D. S. Roos, "The plasmodium genome database: Designing and mining a eukaryotic genomics resource", Nature, 2002.
[2] S. Cohen-Boulakia, O. Biton, S. Davidson, and C. Froidevaux, "Bioguidesrs: querying multiple sources with a user-centric perspective", Bioinformatics, 2007.
[3] DBPedia. http://DBPedia.org.
[4] Freebase. http://www.freebase.com.
[5] M. Jayapandian and H. V. Jagadish, "Automated creation of a forms-based database query interface", In Proceedings of the VLDB Endowment, 2008.
[6] S. Agrawal, S. Chaudhuri, G. Das, and A. Gionis, "Automated ranking of database query results", In CIDR, 2003.
[7] G. Salton and M. McGill, "Introduction to Modern Information Retrieval", McGraw-Hill, 1984.
[8] L. Tang, T. Li, Y. Jiang, Z. Chen, "Dynamic Query Forms for Database Queries", IEEE transactions on knowledge and data engineering, 2013.
[9] E. Chu, A. Baid, X. Chai, A. Doan, and J. F. Naughton, "Combining keyword search and forms for ad hoc querying of databases", In Proceedings of ACM SIGMOD Conference, 2009.
[10] S. Chaudhuri, G. Das, V. Hristidis, and G. Weikum, "Probabilistic information retrieval approach for ranking of database query results", ACM Trans. Database Syst. (TODS), 2006.
[11] B. Liu and H. V. Jagadish, "Using trees to depict a forest", PVLDB, 2009.
[12] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving data streams", In Proceedings of VLDB, 2003.
[13] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: An efficient data clustering method for very large databases", In Proceedings of SIGMOD, 1996.
[14] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong, "Diversifying search results", In Proceedings of WSDM, 2009.
[15] D. Rafiei, K. Bharat, and A. Shukla, "Diversifying web searchresults", In Proceedings of WWW, 2010.