INCREMENTAL CONSTRUCTION OF QUERY FROM KEYWORD SEARCH

Indra. E[1], Bala Sendhil Kumar. G[1]

[2]*Assistant Professor - CSE, Christ College of Engineering and Technology, Puducherry - 605 010, India*
bsk.indrae@gmail.com

[2]*Assistant Professor (Senior Scale) - MBA, Christ College of Engineering and Technology, Puducherry - 605 010, India,*
guru.bsk@gmail.com

Abstract **- Nowadays Databases derives attention from users to precisely express their informational needs using structured queries. But database query construction cannot be performed well by most end users. Most end users uses keyword search which alleviates the usability problem at the price of query expressiveness. So there is an urge to move from keyword search to query construction in probabilistic manner by user interaction. This bridge the gap between usability of keyword search and expressiveness of database queries. IQ^P enables a user to start with an arbitrary keyword query and incrementally refine it into a structured query through an interactive interface.**

Keywords **- Query formulation, Incremental query construction.**

I. INTRODUCTION

Keyword search can be performed well by novice users, as it requires neither a-priori schema knowledge nor query construction skills. But it may return irrelevant or incomplete results. The Powerful tool of structured query retrieves the intended information from a database. But it requires typical expert users who must know the schema knowledge of database. In order to take advantages of both methods a new system of IQ^P is derived here. Using IQ^P , a user can benefit from both, a conventional ranking interface and a more controllable query construction interface. Conventional ranking interface allows the user to immediately identify the most common interpretation of her query. Controllable query construction interface enables the user to clarify her search intent step by step. IQ^P system consists of three components: 1) a framework that formally defines the process of incremental query construction; 2) a probabilistic model to estimate the probabilities of structural query interpretations; 3) an algorithm for generating the optimal query construction plan (QCP), which enables a user to obtain the intended structured query with a minimal number of interactions.

II. QUERY CONSTRUCTION INTERFACE

IQP query construction interface is designed to enhance the ranking centric approaches to database keyword search. Fig. 1 illustrates the user interface of IQ^P. The user interface of IQP consists of

1. A search field to input keyword queries,

2. A query construction window to present query construction options,

3. A query window listing structured queries, and

4. A result window for presenting search results.

When a user issues a keyword query, IQP provides the user with a ranked list of structured queries (as interpretations of the keyword query) and the corresponding results, which are presented in the query and result windows, respectively.

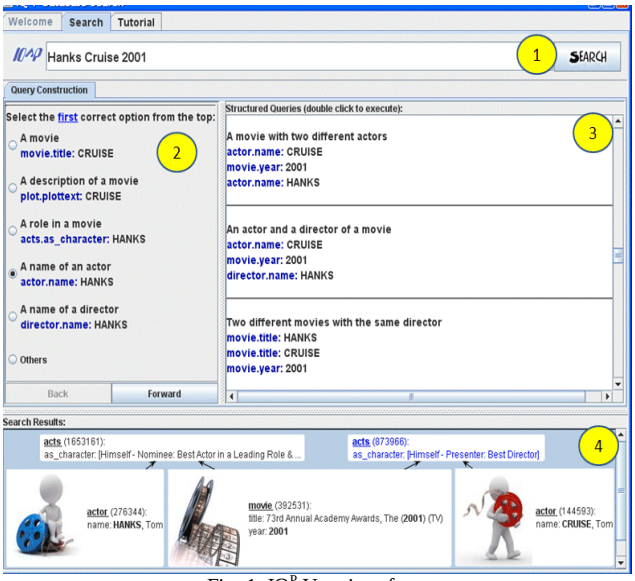

Fig. 1. IQ^P User interface.

III. QUERY CONSTRUCTION FRAMEWORK

First, we show how IQ^P translates a keyword query to a set of structured queries of a relational database. In IQ^P, a keyword is interpreted to an element of a structured query, if it matches the name or the value of that element. A set of keyword interpretations can be connected to form a structured query. The conditions of query interpretation guarantees that users typically assign one specific meaning to a keyword as well as it does not contain any redundant parts.

A. Query Construction Plan (QCP)

As shown in Fig. 1, in each query construction step of IQ^P, a user is presented with a list of query construction options, i.e., partial interpretations. She is supposed to select the

option that correctly interprets her keywords. The assumption here is a user decides on only one option at a time. If the option is a sub query of the intended query interpretation, the user accepts it. If the option is not a proper partial interpretation, the user rejects it. After the user accepts or rejects an option the interpretation space of the keyword query can be reduced accordingly. The user keeps evaluating the options one after another, until only one possible query interpretation is left. Such a query construction process can be modeled as a binary decision tree, which is called query construction plan.

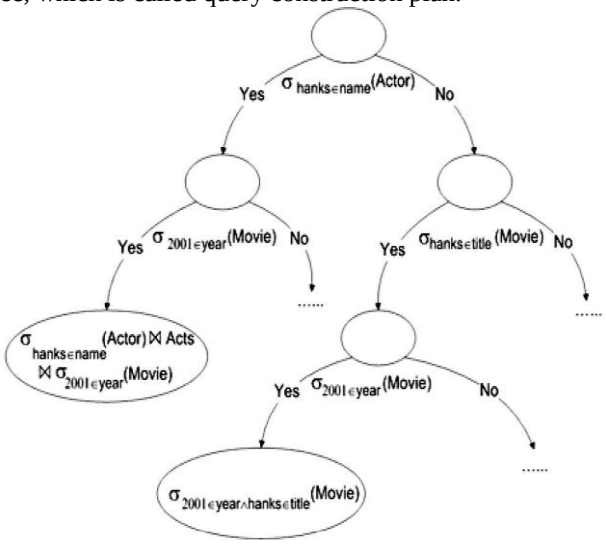

Fig .2.Query construction plan as a binary tree.

B. Query Construction versus Ranking

In the ranking centric approach the structural interpretations of a keyword query are ranked based on their probability of matching the user's intent. QCP for ranking is an unbalanced tree. Using this QCP, a user can reach the top-ranked interpretations quickly, but has to undertake a lot of interactions to find the less probable interpretations. But IQ^P always aims to create the optimal QCP.

IV. ESTIMATING QUERY PROBABILITY

To support efficient query construction, it is important to have an accurate assessment of the probability of whether a structured query (or a query construction option) interprets a user's keyword query correctly. In this section, I introduce a probabilistic model, which enables IQ^P to compute these probabilities. Given a keyword query, IQP is uncertain about the exact Informational need .Given a keyword query K, let the structured query Q be a complete interpretation of K, i.e., Q : K. Then, P(Q|K) represents the conditional probability that, given K, Q is the user intended complete interpretation of K. Analogously, given a query construction option (a partial interpretation) O of K, P(Q|K) represents the conditional probability that, given K, O subsumes the user intended complete interpretation of K. represented by this query. I quantify this uncertainty using probability. P(Q|K) corresponds to P (leaf) in (1). It is a crucial parameter used by IQP in creating query construction plans. If a keyword query K has been used repeatedly in a database, we can directly estimate P(Q|K) using the previous interpretations of K in a database's query log. However, in a large database, it is unlikely to find sufficient number of records for a particular keyword query.

V. QUERY CONSTRUCTION ALGORITHM

The proposed algorithm is used to create a plan that imposes as little effort on the user as possible, i.e., a minimum query construction plan. IQ^P uses a greedy algorithm to construct a near-optimal QCP. IQ^P generates query interpretations by expanding the query hierarchy in a bottom-up fashion. Instead of fully expanding the query hierarchy, the greedy algorithm stops when the size of the top level of the query hierarchy reaches a certain threshold (denoted by T). Then, it searches for the best query construction option (denoted by best_r) within the current query hierarchy and presents the option to the user. If the user accepts the option, the algorithm keeps the part of the top level subsumed by this option and discards the rest. If the user rejects an option, the algorithm discards the part of the top level subsumed by this option. In either case, the algorithm would reduce the size of the top level of the current query hierarchy. The algorithm continues presenting query construction options to the user, until the size of the top level falls below the threshold T.

ALGORITHM

```
Proc greedy_tree (HQ, TQ, T)
Input:
HQ := Initial Query Hierarchy;
TQ := Top Level of HQ;
T := Threshold;
Output:
Query C := Final Structured Query;
Program:
while (true)
if |TQ| < T, then
if HQ can be expanded, then
expand HQ;
else if |TQ| = 1, then //let TQ = {c}
return c;
end if;
end if;

partial_query best_r := null;
float best_gain := +∞ ;
for each R in HQ, do
if IG(TQ | R) < best_gain, then
best_gain := IG(TQ | R);
best_r := R;
end if;
end for;
```

present best_r to user;
if best_r is accepted, then
Sub(best_r) := all queries subsumed by
best_r;
TQ := TQ ^ Sub(best_r);
else if best_r is rejected, then
Sub(best_r) := all queries subsumed by
best_r;
TQ := TQ Sub(best_r);
end if;
end while;
End Proc;

This greedy algorithm tries to find a query construction option that can reveal as much information as possible about the intended structured query.

VI. CONCLUSION

In this paper, I presented IQ^P – a novel system, which enables construction of structured queries from keywords. I presented a conceptual framework for the incremental query construction as well as a probabilistic model, which enables consistent assessment of the probability of a query interpretation. I presented an algorithm for generating optimal query construction plan, which enables the user to obtain the intended structured query with a minimal number of interactions. IQ^P is highly helpful when user intended structured queries cannot be found within the top-ranked results.

REFERENCES

[1] Agrawal.S, Chaudhuri.S and Das.G (2002), 'BXplorer: A System for Keyword- Based Search over Relational databases', In ICDE.

[2] Amer-Yahia.S, Botev.C, Dorre .J and Shanmugasundaram.J (2006), 'XQuery Full-Text extensions explained'. In IBM Systems Journal,pages 335.352.

[3] Chakaravarthy.V.T , Gupta.H, Roy. and Mohania .M (2006), 'Efficiently Linking Text Documents With Relevant Structured Information', VLDB Endowment, ACM 1-59593-385-9/06/09, 2006.

[4] Hristidis.V and Papakonstantinou.Y (2002), 'DISCOVER: Keyword search in relational databases', In VLDB.

[5] Hristidis.V, Koudas .N, Papakonstantinou.Y, and Srivastava.D (2006), 'Keyword Proximity Search in XML Trees', IEEE Transactions on Knowledge and Data Engineering, 18(4).

[6] Lei, Y., Uren, V., Motta. E. (2006), 'Semsearch: A search engine for the semantic web' In: Proceedings of the 15th International Conference on Knowledge Engineering and Knowledge Management (EKAW).

[7] Mansuri. I. and Sarawagi.S (2006), 'Integrating unstructured data into relational databases'. In ICDE.

[8] Simitsis.A and Ioannidis.Y.E (2009), 'DBMSs Should Talk Back Too', in CIDR.

[9] Yu. C and Jagadish H. V. (2006), 'Schema Summarization' In Proceedings of VLDB.