QueRIE: System for Personalized Query Recommendation

Narayan Bhasaheb Vikhe\(^1\)  
Department of Computer Engineering,  
Sir Visvesvaraya Institute of Technology, Nashik - Maharashtra

Prof. Mrs. M.M. Naoghare\(^2\)  
Department of Computer Engineering,  
Sir Visvesvaraya Institute of Technology, Nashik - Maharashtra

Abstract

In this article, we have discussed about Query languages are computer languages used to make queries into databases and information systems. The difference is that a database QueRIE language attempts to give factual answers to factual questions, while an information retrieval query language attempts to find documents containing information that is relevant to an area of inquiry. Database management systems (DBMSs) are computer software applications that interact with the user, other applications, and the database itself to capture and analyze data. The data are typically organized to model aspects of reality in a way that supports processes requiring information. Database management systems are often classified according to the database model that they support.

1. Introduction

Database systems provide the critical infrastructure to access and analyze large volumes of data in a variety of applications. Prominent examples include large-scale data warehouses that support business-intelligence tools, systems for ad-hoc analytics over big data, and services for scientific-data exploration, such as the Genome browser 1 or SkyServer 2, which allow scientists to QueRIE large databases of scientific data over a web-enabled interface. Relational database users employ a QueRIE interface (typically, a web-based client) to issue a series of SQL queries that aim to analyze the data and mine it for interesting information. First-time users may not have the necessary knowledge to
know where to start their exploration. Other times, users may simply overlook queries that retrieve important information. In this work we describe a framework to assist non-expert users by providing personalized QueRIE recommendations.

QueRIE is built on a simple premise that is inspired by Web recommender systems: If users A and B have posed similar queries, then the other queries of B may be of interest to user A and vice versa. In other words, we can recommend the queries of user B in order to help user A in their exploration of the database. A database is an organized collection of data. A general-purpose DBMS is designed to allow the definition, creation, querying, update, and administration of databases. Well-known DBMSs include MySQL, Oracle, PostgreSQL and so many. A database is not generally portable across different DBMSs, but different DBMSs can interoperate by using standards such as SQL and ODBC or JDBC to allow a single application to work with more than one DBMS for example, modelling the availability of rooms in hotels in a way that supports finding a hotel with vacancies. DAS in addition had the purposes of immigration control including issuing visas Such as fragment/splattered structures, random sampling, index hash table and Advanced Cryptographic Techniques (ACT) to avoid the information loss and corrupt the Meta data, etc

2. Justification Of Problem

The tuple-based approach constructs large (and relatively dense) summaries and, most importantly, requires real-time calculations of the similarities between the session summary $S_0$ of the current user and these of past users. The fragment-based approach clearly captures information at a coarser level of detail, and hence it is expected to miss interesting correlations between users.

3 Review of Literature

- Hive - A petabyte scale data warehouse using HADOOP (A. Thusoo et al.): says that HADOOP is a popular open-source map-reduce implementation which is being used in companies like Yahoo, Facebook etc. to store and process extremely large data sets on commodity hardware. However, the map-reduce programming model is very low level and requires developers to write custom programs which are hard to maintain and reuse. Hive, an open-source data warehousing solution built on top of HADOOP. Hive supports queries expressed in a SQL-like declarative language - HiveQL, which are compiled into map-reduce jobs that are executed using HADOOP.
- QueRIE: A recommender system supporting interactive database exploration (S. Mittal, J. S. V. Varman): mentioned that the demonstration presents QueRIE, a recommender system that supports
interactive database exploration. This system aims at assisting non-expert users of scientific databases by generating personalized query recommendations. Drawing inspiration from Web recommender systems, QueRIE tracks the querying behavior of each user and identifies potentially “interesting” parts of the database related to the corresponding data analysis task by locating those database parts that were accessed by similar users in the past. It then generates and recommends the queries that cover those parts to the user.

• Amazon.com recommendations: Item-to-item collaborative filtering (G. Linden, B. Smith, and J. York) wrote that we can use recommendation algorithms to personalize the online store for each customer. The store radically changes based on customer interests, showing programming titles to a software engineer and baby toys to a new mother. There are three common approaches to solving the recommendation problem: traditional collaborative filtering, cluster models, and search-based methods. Here, we compare these methods with our algorithm, which we call item-to-item collaborative filtering. Unlike traditional collaborative filtering, our algorithm's online computation scales independently of the number of customers and number of items in the product catalog. Our algorithm produces recommendations in real-time, scales to massive data sets, and generates high quality recommendations.

• Personalized queries under a generalized preference model (G. Koutrika and Y. Ioannidis) find the solution for problem that we face regularly that we present a preference model that combines expressivity and concision. In addition, we provide efficient algorithms for the selection of preferences related to a QueRIE, and an algorithm for the progressive generation of personalized results, which are ranked based on user interest. Several classes of ranking functions are provided for this purpose. We present results of experiments both synthetic and with real users (a) demonstrating the efficiency of our algorithms, (b) showing the benefits of QueRIE personalization, and (c) providing insight as to the appropriateness of the proposed ranking functions.

4 Methodology

The proposed methodology includes the following steps:

1. **Kullback–Leibler divergence Theorem:**

   ie For discrete probability distributions \( P \) and \( Q \), the KL divergence of \( Q \) from \( P \) is defined to be

   \[
   D_{KL}(P||Q) = \sum_i P(i) \ln \frac{P(i)}{Q(i)}.
   \]

   In words, it is the expectation of the logarithmic difference between the probabilities \( P \) and \( Q \), where the expectation is taken using the probabilities \( P \). The KL divergence is only defined if \( P \) and \( Q \) both sum to 1 and if \( Q(i) = 0 \) implies \( P(i) = 0 \) for all \( i \) (absolute continuity). If the quantity \( 0 \ln(0) \) appears in the formula, it is interpreted as zero because

   \[
   \lim_{x \to 0} x \ln(x) = 0
   \]

2. **Agglomerative Clustering Algorithm:**

   **Agglomerative:** This is a "bottom up" approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.
**Divisive**: This is a "top down" approach: all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy.

![Diagram](image)

**Figure 2: Agglomerative & Divisive System**

The methodology is based on KL divergence, which is a special case of a broader class of divergences called f-divergences. It was originally introduced by Solomon Kullback and Richard Leibler in 1951 as the directed divergence between two distributions. It can be derived from a Bregman divergence. Agglomerative Clustering Algorithm forms clusters in a bottom-up manner.

### 5 Understanding Database System

A database is an organized collection of data. Formally, "database" refers to the data themselves and supporting data structures. Databases are created to operate large quantities of information by inputting, storing, retrieving and managing that information. Databases are set up so that one set of software programs provides all users with access to all the data.

![Database System](image)

**Fig. Database system**

Working on different types & from different places database To ensure that the distributive databases are up to date and current, there are two processes: replication and duplication. Replication involves using specialized software that looks for changes in the distributive database. It may be stored in multiple computers located in the same physical location, or may be dispersed over a network of interconnected computers.
6 Conclusion & future Work

There are many interesting directions we would like to explore in the future. We would like to measure the impact the QueRIE relaxation process has in the quality of recommendations. Exploring a sequence-based approach is another interesting direction for future work, but it requires a careful reconsideration of several aspects of our framework. For instance, pure sequence information may not be sufficient to discover user similarities. Instead, we may have to consider the relative changes between queries in the sequence, e.g., that selection predicates becomes more selective as queries progress, in order to properly detect similarities. We also plan to focus on relational databases that have a form-based interface. While the fragment-based approach seems as a
straightforward selection for such environments, new challenges related to the formulation of session similarity, the synthesis of recommendations and their presentation arise. Finally, as we aim at developing a more generic and scalable system, we are currently working on integrating alternative techniques for generating recommendations, such as matrix factorization methods.

References

[16] C. W. Bachmann (November 1973), The Programmer as Navigator, CACM (Turing Award Lecture 1973)
[22] Ken North, "Sets, Data Models and Data Independence", Dr. Dobb's, 10 March 2010
[23] www.google.com