Reaching Mutual Understanding in a Society of Humans and Database Systems

Arash Termehchy
School of EECS, Oregon State University, Corvallis, OR
termehca@oregonstate.edu

1. ABSTRACT

Most users do not know the structure and/or content of databases and cannot precisely express their information needs in form of queries. Hence, it is challenging for a DBMS to understand and satisfy users' information needs. Researcher have proposed methods and systems to help users specify their queries more precisely and DBMS understand users' intents more accurately. These systems may allow the user to interact with the database to express his intent more clearly. Current systems, however, mainly focus on improving user satisfaction for a single intent. Nevertheless, many users explore databases to find answers for various information needs over a long period of time. For example, a scientist may use a reference genetic database to find answers to many information needs, some of which may repeat multiple times, over the life time of a research project. For these users, database querying is an interactive, progressive, and continuing process.

This setting extends the problem of answering a single information need in two aspects. First, the DBMS can improve its understanding of how the user expresses his intents *progressively over the course of many sessions and queries*. The DBMS may learn the desired answers for a submitted query from the user's explicit or implicit feedback on the returned results. Types of the user's feedback may include clicking on the relevant answers, the amount of time the user spends on reading the results, or user's eye movements. Second, the user may leverage his experience from previous interactions with the database to formulate his future queries more precisely. As the user submits queries and inspect their results, he may gain a better understanding of the database structure and content, which potentially impact how he formulates his future queries. Researchers have observed that users show reinforcement learning behavior when interacting with a DBMS over an extended period of time [1]. Ideally, we would like the user and DBMS to develop gradually a perfect or near-perfect mutual understanding over the course of several queries, where the DBMS returns all or a majority of the desired answers to all or most user's queries.

We model this long-term interaction and collaboration as a cooperative game between *two active and potentially rational agents: the user and the DBMS*. The common goal of the user and the DBMS is to reach a mutual understanding on expressing intents in form of queries. We define *the language of user* as a mapping from the set of his intents to queries, which reflects how he expresses his intents. The *language of the DBMS* is a mapping between queries and results, e.g., sets of tuples. The user informs the DBMS of his intents by submitting queries. The DBMS returns some results for the query and user provides some implicit or explicit feedback on the returned results. The user's feedback reflects the degree by which the returned answers satisfy the intent behind the query. We use standard effectiveness measures, such as precision, to measure the degree of intent satisfaction. Both players receive some reward based on the amount of user satisfaction from the returned results. After each interaction, the DBMS and user may update their languages based on the reward they have received. Our framework is independent of the query language. The queries may be (ill-specified) SQL, keyword, or natural language queries.

We are implementing a query processing system to update DBMS language based on concepts in a reinforcement learning. Current query answering algorithms do not use any randomness in selecting answers for an input query. In other words, they return only the answers, which they deem relevant to the query. As the DBMS query answering algorithm is not perfect, this approach does not allow the DBMS and user to explore other potentially relevant results for the query. Hence, it preserves the incorrect perceptions of DBMS during the interaction without allowing user to rectify them. To alleviate this problem, our algorithm may randomly return some results for the query. Of course, if the DBMS aims at only showing random tuples to the user, it may deliver relatively ineffective answers in the short term. We discuss how our algorithm establishes a trade-off between exploration, i.e., collecting feedback on more tuples, and exploitation, i.e, showing tuples that DBMS deem more relevant to the intent behind the query. Interactive and exploratory query workloads generally require fast response time. We discuss how our algorithms computes and updates relevant features to achieve short response time over large databases. We discuss the effectiveness of our system where the user updates his languages during the interaction.

We discuss our observations on the equilibria and eventual stable states of the game, which model the limits in establishing a mutual understanding. As a user may have to explore multiple databases in an extended period of time, we explain how to extend our system to support reaching mutual understanding between a user and several DBMSs where each DBMS manages a different database and explain the architecture of our query processing system in this setting. Finally, we discuss how our framework can be extended to discover mappings in the context of peer-to-peer data integrations.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

2. REFERENCES

[1] Y. Cen et al. Reinforcement learning in information searching. *Information Research*, 18(1), 2013.