# A Signaling Game Approach to Databases Querying

Ben McCamish<sup>1</sup>, Arash Termehchy<sup>1</sup>, Behrouz Touri<sup>2</sup>, and Eduardo Cotilla-Sanchez<sup>1</sup>

<sup>1</sup> School of Electrical Engineering and Computer Science, Oregon State University <sup>2</sup> Department of Computer Science, University of Colorado, Boulder

## 1 Introduction

Most users do not know the structure or content of databases and cannot precisely express their queries. Hence, it is challenging for database query interfaces to understand and satisfy users' information needs, i.e., intents. Ideally, we would like the user and query interface to establish a *mutual understanding*: the query interface understands how the user expresses her intents and/or the user learns to formulate her queries precisely. Researchers have proposed methods to achieve such a mutual understanding between users and databases [1, 3]. Current methods, however, mainly improve the mutual understanding of a user and a database for a single information need. Nevertheless, many users explore a database to find answers for various information needs potentially over a long period of time. Moreover, current approaches assume that the way a user expresses her intents remains generally intact over her course of interaction with the database to express their future intents more precisely.

We propose a novel framework that models database querying as a game between two active and potentially rational agents: the user and query interface. The common goal of the players is to reach a mutual understanding on expressing intents in the form of queries. The players may reach this goal through communication: the user informs database of her intents by submitting queries, the database returns some results for the queries, and user provides some feedback on how much the returned results match her intents, e.g., by clicking on some desired answer(s). Moreover, the user may modify her query to better reflect her intent after exploring some answers. Both players receive some reward based on the degree by which the returned answers satisfy the intents behind queries. This interaction may reach some stable states, i.e., equilibria, where neither the database receives any reward by changing the way it answers queries nor the user has any incentive to change the way she expresses her intents. One obvious and desirable equilibrium is the state in which database precisely pinpoints the intent behind each query. Nevertheless, we show that the game has other equilibria, which are less desirable. We also propose a query answering algorithm for the database side, which lead to a non-decreasing reward for the user and database over the course of their interaction. We believe that our framework provides the basic tools to identify the equilibria of the interactions between users and databases in different settings and design methods to lead the interactions to more desirable equilibria.

### 2 Framework

**Intent:** An *intent* represents an information need sought after by a user. We assume that each intent is a query in a fixed query language, e.g., conjunctive queries. The set of possible intents is infinite. In practice a user has only a finite number of information needs in a finite period of time. Hence, we assume the number of intents for a particular user is finite. We index each intent over database instance I by  $1 \le i \le m$ .

**Query:** Because the user is not able to formulate her information need e, she submits query  $s \neq e$  to the database instead. Of course, the user still expects that the query interface returns the answers of intent e for s. A user in practice submits a finite number of queries in a finite time period. Hence, we assume that the set of all queries submitted by a user is finite. We index each query over the database instance I by  $1 \leq j \leq n$ .

**Result:** Given a query s over a database instance I, the query interface generally returns a set of tuples as the response to s. An obvious choice is to return s(I). Because the query interface knows that the input query may not precisely specify the user's intent, it also considers alternative answers to satisfy the information need behind the query [1]. **Strategies:** A user strategy, P, is a  $m \times n$  row-stochastic matrix from the set of intents to queries. Each strategy of the database Q is a  $n \times o$  row-stochastic matrix from queries to results. Each pair (P, Q) is called a *strategy profile*.

**Rewards:** This interaction between the database and user can be modeled as a signaling game, with identical interests, played between the user and the database. Given that the query interface returns result l for intent e, we need some standard metrics to measure how effectively l answers e. We use standard effectiveness metric *precision* to measure the user satisfaction given a returned set of tuples.  $l_i$  represents the result tuple set returned after querying the database with intent  $e_i$ . The precision of a result l for intent e over database I, denoted as p(e, l), is the fraction of its tuples that are in e(I). Our framework can be extended for other effectiveness metrics. We define the payoff as

$$u(P,Q) = \sum_{i=1}^{m} \pi_i \sum_{j=1}^{n} P_{ij} \sum_{\ell=1}^{o} Q_{j\ell} \ p(e_i, l_\ell).$$
(1)

This function is an expected payoff of the interaction between the user and query interface when the user maps a (random) intent  $e_i$  to a query  $s_j$  with probability  $P_{ij}$  and the database maps the query  $s_j$  to result  $l_\ell$  with probability  $Q_{j\ell}$ .  $\pi$  is the prior probability over intents and p is precision.

**Example 1:** Consider a database about universities with schema U(name, abbreviation, state, ranking). Table 1(a) and (b) show a user's intents and the queries she submits to the database to express these intents, respectively. Table 1 (c) and (d) illustrate two possible strategy profiles for these sets of intents and queries. For instance, given the strategy profile shown in Table 1 (c), if the user wanted to find the ranking for Michigan State University, i.e., intent  $e_2$ , she submits query  $s_1$  that precisely represent this information need. If the database receives query  $s_1$ , it returns the tuples in the database that satisfy  $s_1$ , i.e.,  $l_2$ . However, if the user wants to find the rankings of the Mississippi State University and Missouri State University, she submits query  $s_2$ , the query interface returns

the rankings of Mississippi State University and Missouri State University with equal probability.

Table 1: Intents, queries, and strategies for exploring the database about universities.(a) Intents(b) Queries

Intent#	Intent	Query# Query	
$e_1$	$ans(y) \leftarrow U(x, `MSU', `MS', y)$	$s_1 \qquad ans(y) \leftarrow U(x, MSU',$	$\overline{MI', y}$
$e_2$	$ans(y) \leftarrow U(x, `MSU', `MI', y)$	$s_2 \qquad ans(y) \leftarrow U(x, MSU, z)$	,y)
$e_3$	$ans(y) \leftarrow U(x, `MSU', `MO', y)$		
	(c) A strategy profile	(d) Another strategy profile	
	$\begin{bmatrix} s_1 & s_2 \\ e_1 & 0 & 1 \end{bmatrix} \begin{bmatrix} l_1 & l_2 & l_3 \\ \hline & & & l_1 & l_2 \end{bmatrix}$	$\begin{bmatrix} s_1 & s_2 \\ e_1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} l_1 & l_2 & l_3 \\ e_1 & e_1 & e_2 \end{bmatrix}$	

0 1 0

0 1

0 1

0 1 0

 $s_2|0.5|0|0.5$ 

## 3 Equilibria of The Game

0

1

A Nash equilibrium for a game is a strategy profile where the query interface and user will not do better by unilaterally deviating from their strategies. A Nash equilibrium represents a stable state in the interaction between the user and database.

**Definition 1.** Strategy profile (P,Q) is a Nash equilibrium iff  $u(P,Q) \ge u(P',Q)$  for all  $P' \ne P$  and  $u(P,Q) \ge u(P,Q')$  for all  $Q' \ne Q$ .

If the intents in Table 1 have equal prior probabilities, the strategy profile in Table 1 (c) is a Nash equilibrium. None of the players achieve a better payoff by unilaterally changing their strategies.

The strategy profile in Table 1 (c) provides the highest payoff for the user and database system given the intents and queries in Table 1 (a) and Table 1 (b). However, some Nash equilibria may not provide high payoffs. For instance, Table 1 (d) depicts another strategy profile for the set of intents and queries in Table 1 (a) and Table 1 (b). In this strategy profile, the user has a little knowledge about the database content and expresses al off her intents using a single query  $s_2$ , which asks for the ranking of universities whose abbreviations are MSU. Given query  $s_2$ , the query interface always returns the ranking of Michigan State University. Obviously, the query interface always returns the non-relevant answers for the intents of finding the rankings of Mississippi State University and Missouri State University. If all intents have equal prior probabilities, this strategy profile is a Nash equilibrium. For example, the user will not get a higher payoff by increasing her knowledge about the database and using query  $s_1$  to express intent  $e_2$ . Clearly, the payoff of this strategy profile is less than the one of the strategy profile in Table 1 (c). Nevertheless, the user and the query interface do not have any incentive to leave this undesirable stable state once reached and will very likely stay in this state.

A Strict Nash Equilibrium is a strategy profile where neither party can unilaterally change their strategies and receive the same or better payoff.

**Definition 2.** Strategy profile (P, Q) is a Strict Nash Equilibrium iff u(P, Q) > u(P', Q) for all  $P' \neq P$  and u(P, Q) > u(P, Q') for all  $Q' \neq Q$ .

If we remove row  $e_3$  and column  $l_3$  from the user and database strategy in Table 1 (c) respectively and change  $s_2, l_1$  for the database strategy to 1, we are left with a strict Nash. A strict Nash equilibrium is more stable than a Nash equilibrium.

### 4 An Adaptation Mechanism

After figuring out the desirable states of the game, one may want to design algorithms for the database side which lead the game to such states. In many relevant applications, the user's learning is happening in a much slower time-scale compared to the learning of the database. So, one can assume that the user's strategy is fixed compared to the time-scale of the database adaptation. As in [2], we use Roth-Erev reinforcement learning mechanism for the database adaption as follows.

- a. Let R(0) > 0 be an  $n \times o$  initial reward matrix whose entries are strictly positive.
- b. Let Q(0) be the initial database strategy with  $Q_{j\ell}(0) = \frac{R_{j\ell}(0)}{\sum_{\ell=1}^{o} R_{j\ell}(0)} > 0$  for all  $j \in [n]$  and  $\ell \in [o]$ .
- c. For iterations  $t = 1, 2, \ldots$ , do
  - i. If the user's query at time t is s(t), return a result L(t):

$$P(L(t) = i' \mid s(t)) = Q_{s(t)i'}(t).$$

ii. User gives a reward  $r_{ii'}$  given that i is the intent of the user at time t. Then, set

$$R_{j\ell}(t+1) = \begin{cases} R_{j\ell}(t) + r_{i\ell} \text{ if } j = s(t) \text{ and } \ell = i' \\ R_{j\ell}(t) \text{ otherwise} \end{cases}$$
(2)

iii. Update the database strategy by

$$Q_{ji}(t+1) = \frac{R_{ji}(t+1)}{\sum_{\ell=1}^{o} R_{j\ell}(t+1)},$$
(3)

for all  $j \in [n]$  and  $i \in [o]$ .

Let u(P, Q(t)) denote the payoff in interaction t according to in Equation (1). A random process  $\{X(t)\}$  is a submartingale if it is absolutely integrable (i.e.  $E(|X(t)|) < \infty$ for all t) and  $E(X(t+1) | \mathcal{F}_t) \ge X(t)$ , where  $\mathcal{F}_t$  is the  $\sigma$ -algebra generated by  $X_1, \ldots, X_t$ . In other words, the expected value of X(t+1) given the past, is not strictly less than the value of X(t).

#### **Theorem 1.** The sequence of u(P, Q(t)) is a submartingale.

Hence, the proposed reinforcement learning rule *stochastically* improves the efficiency of communication between the database and user. It is an interesting future work to explore whether this algorithm will converge to a globally optimal payoff.

## References

- 1. Fagin, R., Kimelfeld, B., Li, Y., Raghavan, S., Vaithyanathan, S.: Understanding Queries in a Search Database System. In: PODS (2010)
- 2. Hu, Y., Skyrms, B., Tarrès, P.: Reinforcement learning in signaling game. arXiv preprint arXiv:1103.5818 (2011)
- 3. Khoussainova, N., Kwon, Y., Balazinska, M., Suciu, D.: Snipsuggest: Context-aware autocompletion for sql. PVLDB 4(1) (2010)