Human-Database Interaction: A Holistic Approach

Mohamed Sarwat, Venkata Vamsikrishna Meduri
Arizona State University,
699 S. Mill Ave, Tempe, AZ 85281
msarwat@asu.edu,vmeduri@asu.edu

I. Background

There has been an increasing interest into blurring the line between human-interaction and database systems. Several research papers tackled the Human-Database Interaction (HDI) challenge, yet none of them provides a holistic HDI approach. In this talk, we briefly describe dbTinder, a database engine that bridges the gap between the conversation approach humans use to interact and the $Query \rightarrow Answer$ approach used in classic database systems. To achieve that, dbTinder turns HDI into an intent discovery process where the system pro-actively converses with the user to guide her towards potentially $interesting$ tuples in the database.

II. The DBTinder Approach

There have been efforts in the recent past [1], [2] that cater to the SQL-agnostic user who prefers to query databases using natural language. With an advent of touch-based interfaces [3], [4], the end user tends to replace keyword-based querying with swipes and drop-down option selections. dbTinder considers the following interaction modes to converse with the human (see Figure 1):

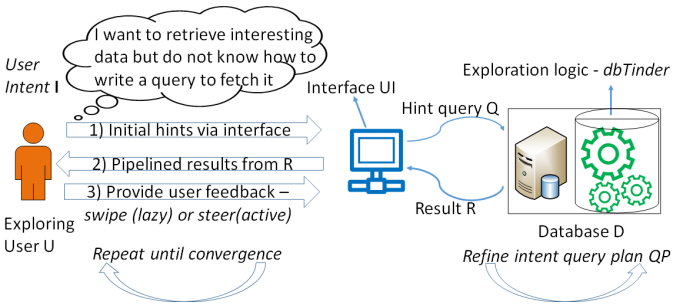

Fig. 1. Human-Database Interaction (HDI) modeled as a conversation

1. Lazy HDI (Swipe Left / Right): The user is presented with a batch of $interesting$ tuple suggestions from the answers to the initial hint query. The human can then either $swipe$ left or right on tuples denoting a binary labeling of interesting or not interesting (inspired by the Tinder dating app). The system then leverages the swipes to determine what the next batch of tuples are. Such HDI approach is considered lazy since the human relies solely on swiping left / right to converse with the system.

2. Active HDI (Steer Query Plan): The human can explicitly suggest an alternative "hint" query by picking from the options of ⟨attribute,value⟩ combinations for steering operators such as narrow/widen indicating the addition/removal of a selection predicate from the initial query. Unlike query steering or active learning systems that can incur a significant user wait time, dbTinder preserves interactivity by directly modifying the query execution plan thereby avoiding repeated query parsing and optimization. Approximate Query Processing [5] systems enhance user interactivity by sample based execution, yet they are confined to aggregate queries and may not cater to human goal, because the sample has no statistical guarantee of containing the specific tuple(s) the human is looking for.

3. Hybrid HDI (Swipe & Steer): The human can alternate between swiping left/right and steering during the HDI session.

dbTinder is also equipped with a module that $learns$ what tuples are $interesting$, presents them earlier in the conversation, and hence helps the user converge to her goal faster. Contrary to the existing systems, dbTinder adopts a holistic approach that considers the following learning strategies:

I. Learning from the Data: Similar to SeeDB [6] and Dice [7], dbTinder learns from interestingness heuristics within the data. However, dbTinder creates a candidate pool of statistically significant query plans based on the precomputed histograms stored by the query optimizer. The user feedback allows dbTinder to further filter out the less interesting plans.

II. Learning from the Conversation with the Human: The system models the user intent to be reached as a goal and the feedback through swipes as rewards / penalties and employs a reinforcement learning approach to identify the optimal sequence of tuples to be retrieved (or the underlying query plans to be created) to maximize the reward. This is complementary to the active learning based exploration that can also learn from user feedback, e.g. AIDE [8]. While there have been interesting formulations to capture user intent through search trees [9], they assume that some of the user preferences are specified apriori through objective functions to be optimized in the solutions to the search tree. They also offer limited functionality such as query relaxation as against several steering operators supported by dbTinder, for which the search trees can be relatively much more huge. A cooperative game formulation of data exploration [10] models a dual learning problem for the user and the database separately, contrary to which we rely on a unified learning model for user intent.

III. Learning from the Crowd Historical Interactions: A historical interaction session if available can be represented as a vector of query operators from the entire conversation. Based on the similarity of the current HDI session to the historical HDI sessions [11], [12], an exploratory operator can

be included into the intent plan for the current user.

dbTinder, built inside the core kernel of a database engine, can co-optimize the conversation dynamics close to the data and thereby achieve interactive performance, leading to more fluid conversation. In the talk, we will highlight how dbTinder re-organizes the data based upon the three learning strategies, introduces HDI-aware access methods, and modifies the physical query plan to cope with the aforementioned HDI modes.

REFERENCES

[1] F. Li and H. V. Jagadish, "Constructing an interactive natural language interface for relational databases," *PVLDB*, vol. 8, no. 1, pp. 73–84, 2014.

[2] V. Zhong, C. Xiong, and R. Socher, "Seq2sql: Generating structured queries from natural language using reinforcement learning," *CoRR*, vol. abs/1709.00103, 2017. [Online]. Available: http://arxiv.org/abs/1709.00103

[3] E. Liarou and S. Idreos, "dbTouch in action database kernels for touch-based data exploration," in *ICDE*, 2014.

[4] L. Jiang and A. Nandi, "SnapToQuery: Providing interactive feedback during exploratory query specification," *PVLDB*, vol. 8, no. 11, pp. 1250–1261, 2015.

[5] A. Galakatos, A. Crotty, E. Zgraggen, C. Binnig, and T. Kraska, "Revisiting reuse for approximate query processing," *PVLDB*, vol. 10, no. 10, pp. 1142–1153, 2017.

[6] M. Vartak, S. Rahman, S. Madden, A. G. Parameswaran, and N. Polyzotis, "SeeDB: efficient data-driven visualization recommendations to support visual analytics," *PVLDB*, vol. 8, no. 13, pp. 2182–2193, 2015.

[7] N. Kamat, P. Jayachandran, K. Tunga, and A. Nandi, "Distributed and interactive cube exploration," in *ICDE*, 2014.

[8] K. Dimitriadou, O. Papaemmanouil, and Y. Diao, "Explore-by-example: An automatic query steering framework for interactive data exploration," in *ACM SIGMOD*, 2014.

[9] D. Mottin, A. Marascu, S. Basu Roy, G. Das, T. Palpanas, and Y. Velegrakis, "IQR: An interactive query relaxation system for the empty-answer problem," in *ACM SIGMOD*, 2014.

[10] B. McCamish, A. Termehchy, and B. Touri, "A game-theoretic approach to data interaction: A progress report," in *HILDA@SIGMOD*, 2017.

[11] J. J. Levandoski, M. Sarwat, M. F. Mokbel, and M. D. Ekstrand, "RecStore: an extensible and adaptive framework for online recommender queries inside the database engine," 2012.

[12] M. Sarwat, R. Moraffah, M. F. Mokbel, and J. L. Avery, "Database System Support for Personalized Recommendation Applications," in *ICDE*, 2017.