International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 02 Issue: 09 | Dec-2015 www.irjet.net p-ISSN: 2395-0072

A Review Paper on Query Optimization for Crowdsourcing Systems

Rohini Pingle
M.E. Computer Engineering, Gokhale Education Society's, R. H. Sapat College of Engineering, Management Studies and Research, Savitribai Phule University, Pune. Nashik, Maharashtra, India.
Prof. Mrs. Rucha Samant
Assistant Professor at Gokhale Education Society's, R. H. Sapat College of Engineering, Management Studies and Research, Savitribai Phule University, Pune. Nashik, Maharashtra, India.

Abstract - ***Optimization of the query is the biggest problem now days for crowdsourcing system. Crowdsourcing is source for the experts to solve the problem and freely sharing the answer with everyone also hiding the complexities and to relief the user from burden of dealing with the crowd. The user has to submit an SQL query and the system takes the responsible for compiling the query, generating the execution plan and evaluating in the crowdsourcing market. The relational database systems, query optimization is providing query interfaces which are important for crowdsourcing. The propose system, a cost-based query optimization approach for crowdsourcing systems. The cost and latency consider in query optimization objective for proposed system and generates query plans that give a good balance between the cost and latency. The first stage develops efficient algorithms for optimizing Selection queries, join queries, and complex selection-join queries and the second stage validate our approach through extensive experiments by simulation as well as with the real crowd.***

Key Words: *Crowdsourcing, query optimization, human intelligence tasks (HIT).*

1. INTRODUCTION

1.1 Crowdsourcing and Query Optimization:

Crowdsourcing is a new way of utilizing the power of the crowd in projects which usually require a large number of people, and when the costs of their completion by traditional ways, in-house or by outsourcing, is not cost-effective. crowdsourcing is channeling the experts desire to solve a problem and then freely sharing the answer with everyone. Crowdsourcing is an emerging paradigm which is based on harnessing the power of crowd in solving problems. Crowdsourcing is a form of outsourcing, although it typically does not require a formal contraction which is found in outsourcing tasks to an external organization specialized in that task to perform. Crowdsourcing is also meant to reach a wider range of people, which may sometimes be required to get a solution correctly and efficiently.

Query Optimization:

Query optimization is a function of many relational database management systems. The query optimizer attempts to determine the most efficient way to execute a given query by considering the possible query plans. Generally, the query optimizer cannot be accessed directly by users: once queries are submitted to database server, and parsed by the parser, they are then passed to the query optimizer where optimization occurs. A query is a request for information from a database. Queries results are generated by accessing relevant database data and manipulating it in a way that yields the requested information. Since database structures are complex, in most cases, and especially for not-very-simple queries, the needed data for a query can be collected from a database by accessing it in different ways, through different data-structures, and in different orders. Each different way typically requires different processing time. Query optimization find the best query plan in terms of estimated monetary cost.

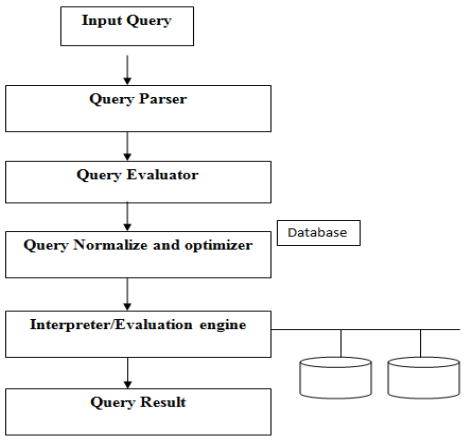

Fig 1: Architecture of Query Optimization

Recent crowdsourcing systems, such as CrowdDB, Qurk and Dec, provide an SQL-like query language as a declarative interface to the crowd. An SQL-like declarative interface is designed to encapsulate the complexities of dealing with the crowd and provide the crowdsourcing system an interface that is familiar to most database users. Consequently, for a given query, a declarative system must first compile the query, generate an execution plan, post human intelligence tasks (HITs) to the crowd according to the plan, collect the answers, handle errors and resolve the inconsistencies in the answers.

A declarative querying improves the usability of the system, it requires the system to have the capability to optimize and provide a "near optimal" query execution plan for each query. Since a declarative crowdsourcing query can be evaluated in many ways, the choice of execution plan has a significant impact on overall performance, which includes the number of questions being asked, the types/difficulties of the questions and the monetary cost incurred. It is therefore important to design an efficient crowdsourcing query optimizer that is able to consider all potentially good query plans and select the "best" plan based on a cost model and optimization objectives.

Crowdsourcing is considered to be distributed and the crowd might be inexperienced in the task. The definition does not elaborate on the types of crowd sourced tasks and the characteristics of the crowdsourcing platform and what facilities it should provide. These two definitions view crowdsourcing from two different perspectives, with little or no features in common.

Some queries cannot be answered by machines only. Processing such queries requires human input for providing information that is missing from the database, for performing computationally difficult functions, and for matching, ranking, or aggregating results based on fuzzy criteria. CrowdDB uses human input via crowdsourcing to process queries that neither database systems nor search engines can adequately answer. It uses SQL both as a language for posing complex queries and as a way to model data. While CrowdDB leverages many aspects of traditional database systems, there are also important differences. Conceptually, a major change is that the traditional closed-world assumption for query processing does not hold for human input. From an implementation perspective, human-oriented query operators are needed to solicit, integrate and cleanse crowd sourced data. Furthermore, performance and cost depend on a number of new factors including worker affinity, training, fatigue, motivation and location.

Crowdsourcing has created a variety of opportunities for many challenging problems by leveraging human intelligence. For example, applications such as image tagging, natural language processing, and semantic-based information retrieval can exploit crowd-based human computation to supplement existing computational algorithms. Naturally, human workers in crowdsourcing solve problems based on their knowledge, experience, and perception. It is therefore not clear which problems can be better solved by crowdsourcing than solving solely using traditional machine-based methods. Therefore, a cost sensitive quantitative analysis method is needed.

2. CHALLENGES OF CROWDSOURCING

Following are some challenges for crowdsourcing are

i) Burdensome concept review process to deal with a large amount of crowd-sourced design concepts.
ii) Insufficient consideration in integrating design knowledge and principles into existing data processing methods for crowdsourcing.
iii) Lack of a quantitative decision support process to identify better concepts.

3. RELETED WORK

Mainly the query optimization are used three types of queries

Selection Queries:

The selection query is used to select data from a database. The result is stored in a result table, called the result-set. It will applies one or more human recognized condition over the tuples in a single relation[9].

A selection query applies one or more human-recognized selection conditions over the tuples in a single relation. Selection query has many applications in real crowdsourcing scenarios, such as filtering data [10] and finding certain items.

Example:

```
SELECT R3.image
FROM IMAGE R3
WHERE make = "Volvo" AND style = "Sedan"
AND color = "black" AND quality = "high"
```

Here, is example of Finding high-quality images of black Volvo sedans, where selection conditions (e.g., make = "Volvo") are evaluated using crowdsourcing and the image m1 satisfying all the conditions is returned as a result [9].

Join Queries:

An SQL JOIN query is used to combine rows from two or more tables, based on a common field between them. The most common type of join is: SQL INNER JOIN (simple join**)**. An SQL INNER JOIN return all rows from multiple tables where the join condition is met.
Types of the different SQL JOINs are follows:

- **INNER JOIN**: Returns all rows when there is at least one match in BOTH tables
- **LEFT JOIN**: Return all rows from the left table, and the matched rows from the right table
- **RIGHT JOIN**: Return all rows from the right table, and the matched rows from the left table
- **FULL JOIN**: Return all rows when there is a match in ONE of the tables

One typical application of join query is crowdsourcing entity resolution, which identifies pairs of records representing the same real-world entity. Other applications include subjective classification (e.g., sentimental analysis) [6] and schema matching [2].
Example:

```
SELECT R2._, R3.image
FROM AUTOMOBILE R2, IMAGE R3
WHERE R2.make = R3.make
AND R2.model = R3.model
JoinFilter R2.style = R3.style
```

Here, is a join query Q3 over the relations is to link the automobile records in R2 with the images in R3, which is presented [9].

Complex Selection-Join Queries:

The another category of Query optimization system is used complex query. This will containing both selections and joins. These queries can help users express more complex crowdsourcing requirements. Q1 is an example of the complex query, which finds black cars with high-quality images and "positive" reviews[9].

For the case where the latency constraint is not imposed, we can optimize the query plan similarly to traditional databases: apply some heuristic rules, such as pushing down selections and determining the join ordering, and then invoke the above- mentioned techniques for optimizing selections and joins.

4. LITERATURE REVIEW

Davidson, Khanna, Milo,Roy[1] , state that to evaluating top-k and group-by queries using the crowd to answer either type or value questions. Given two data elements, the answer to a type question is "yes" if the elements have the same type and therefore belong to the same group or cluster; the answer to a value question orders the two data elements. The assumption here is that there is an underlying ground truth, but that the answers returned by the crowd may sometimes be erroneous. They formalize the problems of top-k and group-by in the crowd-sourced setting, and give efficient algorithms that are guaranteed to achieve good results with high probability.

Ju Fan, Meiyu Lu, Beng Chin Ooi, Tan, Zhang[2], They proposed system a two-pronged approach for web table matching that effectively addresses the above difficulties. First, they propose a concept-based approach that maps each column of a web table to the best concept, in a well-developed knowledge base, that represents it. This approach overcomes the problem that sometimes values of two web table columns may be disjoint, even though the columns are related, due to incompleteness in the column values. Second, They develop a hybrid machine crowd sourcing framework that leverages human intelligence to discern the concepts for "difficult" columns. The overall framework assigns the most "beneficial" column to- concept matching tasks to the crowd under a given budget and utilizes the crowdsourcing result to help our algorithm infer the best matches for the rest of the columns.

Franklin, Kossmann, Tim Kraska, Ramesh, Reynold Xin [3]**,** proposes a system was to deploy CrowdDB uses human input via crowd sourcing to process queries that neither database systems nor search engines can adequately answer. It uses SQL both as a language for posing complex queries and as a way to model data. While CrowdDB leverages many aspects of traditional database systems, there are also important differences. Conceptually, a major change is that the traditional closed-world assumption for query processing does not hold for human input. From an implementation perspective human-oriented query operators are needed to solicit, integrate and cleanse crowdsourced data. Furthermore, performance and cost depend on a number of new factors including worker affinity, training fatigue, motivation and location.

Chien-Ju Ho, Jabbari and Vaughan[6], state that Crowdsourcing markets have gained popularity as a tool for inexpensively collecting data from diverse populations of workers. Classification tasks, in which workers provide labels for in- stances are among the

most common tasks posted, but due to human error and the prevalence of spam, the labels collected are often noisy. This problem is typically addressed by collecting labels for each instance from multiple workers and combining them in a clever way, but the question of how to choose which tasks to assign to each worker is often overlooked. They investigate the problem of task assignment and label inference for heterogeneous classification tasks. By applying online primal-dual techniques, derive a provably near-optimal adaptive assignment algorithm. They show that adaptively assigning workers to tasks can lead to more accurate predictions at a lower cost when the available workers are diverse.

Parameswaran, GarciaMolina,Park, Polyzotis , Ramesh , Widom [10], discusses a set of data items and filtering them based on a set of properties that can be verified by humans. This problem I commonplace in crowdsourcing applications, and yet, to our knowledge, no one has considered the formal optimization of this problem. This paper develop deterministic and probabilistic algorithms to optimize the expected cost (i.e., number of questions) and expected error. They focus on one of these fundamental building blocks, an algorithm to filter a set of data items.

Marcus,Karger,Madden,Miller,[8], proposed a several techniques for using workers on a crowdsourcing platform like Amazon's Mechanical Turk to estimate the fraction of items in a dataset that satisfy some property or predicate (e. and do this without explicitly iterating through every item in the dataset. This is important in crowdsourced query optimization to support predicate ordering and in query evaluation, when performing a GROUP BY operation with a COUNT or AVG aggregate. They compare sampling item labels, a traditional approach, to showing workers a collection of items and asking them to estimate how many satisfy some predicate. Additionally, they develop techniques to eliminate spammers and colluding attackers trying to skew selectivity estimates when using this count estimation approach and find that for images, counting can be much more effective than sampled labeling, reducing the amount of work necessary to arrive at an estimate that is within 1% of the true fraction by up to an order of magnitude, with lower worker latency.

5. CONCLUSIONS

Different approaches of query optimization for crowdsourcing have been discussed in detail. The efficient and effective optimization algorithm develop for select, join, complex query. In the present scenario, simulated and real crowd demonstrate the effectiveness of our query optimizer and take review of query optimization objective and generates query plans that provide a good balance between the cost and latency.

REFERENCES

[1] S. B. Davidson, S. Khanna, T. Milo, and S.Roy, "Using the crowd for top-k and group-by queries," in Proc. 16th Int. Conf. Database Theory, 2013, pp. 225–236..

[2] J. Fan, M. Lu, B. C. Ooi, W.-C. Tan, and M. Zhang,"A hybrid machine-crowdsourcing system for matching web tables," in Proc .IEEE 30th Int. Conf. Data Eng., 2014, pp. 976–987.

[3] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin, "CrowdDB: Answering queries with crowdsourcing," in Proc.ACM SIGMOD Int. Conf. Manage. Data, 2011, pp. 61–72.

[4] J. Gao, X. Liu, B. C. Ooi, H. Wang, and G. Chen, "An online cost sensitive decision-making method in crowdsourcing systems," in Proc. ACM SIGMOD Int. Conf. Manage. Data, pp. 217–228, 2013.

[5] Y. Gao and A. G. Parameswaran, "Finish them!: Pricing algorithms for human computation," Proc. VLDB Endowment, vol. 7, no. 14, pp. 1965–1976, 2014.

[6] C.-J. Ho, S. Jabbari, and J. W. Vaughan, "Adaptive task assignment for crowdsourced classification," in Proc. 30th Int. Conf. Mach. Language, 2013, vol. 1, pp. 534–542.

[7] X. Liu, M. Lu, B. C. Ooi, Y. Shen, S. Wu, and M. Zhang, "CDAS: A crowdsourcing data analytics system," Proc. VLDB Endowment, vol. 5, no. 10, pp. 1040–1051, 2012.

[8] A. Marcus, D. R. Karger, S. Madden,R. Miller, and S. Oh, "Counting with the crowd," Proc. VLDB Endowment, vol. 6, no. 2, pp. 109–120, 2012.

[9] "CrowdOp: Query Optimization for Declarative Crowdsourcing Systems", ieee transactions on knowledge and data engineering, VOL. 27, NO. 8, AUGUST 2015.

[10] A. G. Parameswaran, H. Garcia-Molina, H. Park, N. Polyzotis, A. Ramesh, and J. Widom, "CrowdScreen: Algorithms for filtering data with humans," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2012, pp. 361–372.

[11] H. Park and J. Widom, "Query optimization over crowdsourced data," Proc. VLDB Endowment, vol. 6, no. 10, pp. 781–792, 2013.