ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 5, Issue 7, July 2016

Parallel System Used By Query Optimization for Crowdsourcing

Rohini Pingle, Rucha Samant

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. A query can have different alternative execution plans and the difference in crowdsourcing cost between the best and the worst plans may be several orders of magnitude. 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 system develops efficient algorithms for optimizing Selection queries, join queries, and complex selection-join queries using parallel system.

Index Terms*— Crowdsourcing, query optimization, human intelligence tasks (HIT).

I. INTRODUCTION

Crowdsourcing is a new approach of utilize the power of the crowd in projects which usually require a large number of people, and when the costs of their completion by usual ways, in-house or by outsourcing, is not cost-effective. Crowd sourcing is channeling the experts desire to solve a problem and then freely sharing the answer with every one[9].Crowd sourcing is also meant to reach a wider range of people, which may sometimes be required to get a solution correctly and efficiently.

Query optimization is a role of many relational database management systems. The query optimizer attempt to determine the most professional way to perform a given query by considering the possible query plans. The query optimizer cannot be access 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 related database data and manipulate it in a way that yields the requested data. Since database structures are difficult, in most cases, and especially for complicated queries, the desirable data for a query can be composed from a database by accessing it in different ways, through different data-structures, and in different orders. Each different way typically requires different dealing out time. Query optimization find the best query plan in terms of predictable economic cost.

A 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 a lot of ways, the choice of execution plan has a vital 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 good query plans and select the "best" plan based on a cost model and optimization objectives [2].

The objective of proposed system is to evaluate the effectiveness of query optimization schemes for the crowd powered selection, join and complex queries in a simulated crowd sourcing environment using parallel system. System should give able to split the join query and run that query on two or more system parallely[9].

The remainder of the paper is organized as follows. Section I introduces the Crowd sourcing, query optimization, Section II define related work, in Sections III System Architecture, Section IV Mathematical model, In Section V Implementation, Algorithm, Section VI defines System flow section VII presents the experiments result. Section VIII concludes the paper.

II. RELATED WORK

A considerable amount of research on Query optimization for crowd sourcing has been done over the past few years. Some important techniques are discussed here. Various techniques have been proposed and research has been done in the field of query optimization. Also many advanced methods have been

***Rohini Pingle**, Dept of Computer Engg, GES's COE Nasik,India.*
***Prof. Rucha Samant**, Dept of Computer Engg, GES's COE Nasik Nasik,India.*

introduced for query optimization and crowd sourcing

Chien-Ju Ho, Shahin Jabbari and Jennifer Wortman Vaughan[6], state that Crowdsourcing markets have gained attractiveness as a tool for reasonably collecting data from diverse populations of workers. classify tasks, in which workers provide labels for in- stances are among the most common tasks posted, but due to human error and the rate of spam, the labels collected are often noisy. They examine the problem of task assignment and label deduction for diverse 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 accessible workers are diverse.

Adam Marcus David Karger Samuel Madden Robert Miller Sewoong Oh [8], proposed a several techniques for using workers on a crowdsourcing stage like Amazon's Mechanical Turk. This is important in crowdsourced query optimization to support predicate ordering and in query estimate, when performing a GROUP BY operation with a COUNT or AVG aggregate. They develop techniques to remove spammers and collude attackers trying to slant selectivity estimates when using 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.

Aditya Parameswaran , Hector GarciaMolina , Hyunjung Park , Neoklis Polyzotis , Aditya Ramesh , Jennifer Widom [10], discusses a set of data items and filtering them based on a set of properties that can be confirmed by humans. This problem I common place 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 basic building blocks, an algorithm to filter a set of data items.

Adam Marcus ,Eugene Wu ,David Karger Samuel Madden ,Robert Miller[12] state that focus on how to use humans to compare items for sorting and joining data, two of the most common operations in DBMSs. They describe basic query interface and the user interface of the tasks and post to MTurk. They propose a number of optimizations, including task batching, replacing pairwise comparisons with numerical ratings, and pre-filtering tables before joining them, which dramatically reduce the overall cost of running sorts and joins on the crowd.

Jiannan Wang, Guoliang Li, Tim Kraskay, Michael J. Franklin z, Jianhua Feng[14], define that the crowdsourced join query which aims to use humans to find all pairs of similar objects from two collections. As a human-only solution is exclusive, They accept a hybrid human-machine approach which first uses machines to produce a candidate set of matching pairs, and then asks humans to label the pairs in the candidate set as either matching or non-matching And also present a cross transitive-relations and crowdsourcing labeling structure which aims to crowdsource the minimum number of pairs to label all the candidate pairs.They propose a heuristic labeling order and develop a parallel labeling algorithm to efficiently crowdsource the pairs order.

III. Design of Proposed System

Crowd sourcing is designed to hide the complexities and relieve the user of the burden of dealing with the crowd. So proposed system is based on query optimization approach for crowdsourcing system.

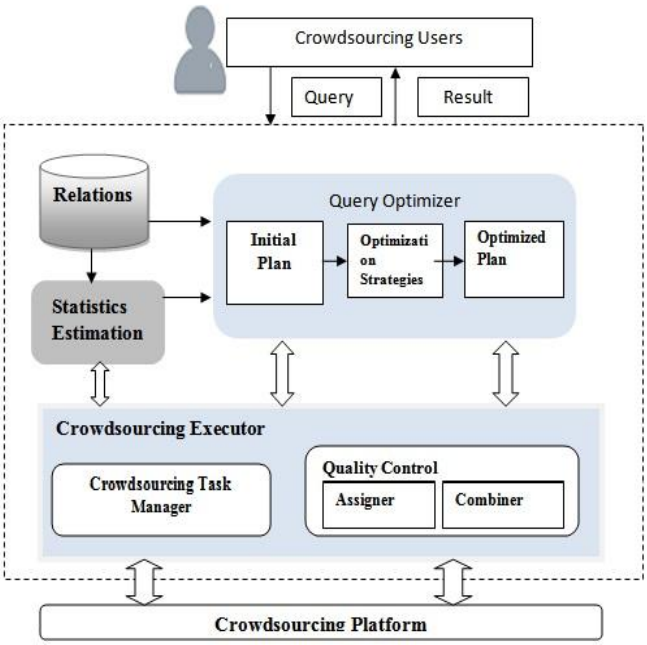


Fig.1:Query Optimization for Crowdsourcing System Architecture

The architecture of query processing is illustrated in Fig. 1.

- Crowd sourcing User: An SQL query is issued by a crowd sourcing user.

- Query Optimizer: Firstly processed on query and parses the query and produces an optimized query plan. In query plan, initial query plan will be define using select query and join query. The join query will be split. Then Optimization strategies are used for join query and final optimized plan generated.

- Crowdsourcing Executor: The query plan is then executed by Crowdsourcing Executor to produce human intelligence tasks or HITs and publish these HITs on crowdsourcing platforms, Based on the HIT answers composed from the crowd, Crowdsourcing Executor evaluates the query and returns the obtained results to the user.

- Quality Control: It consists of assigner and combiner. Assigner is assign job of query for different tables. Combiner is used for combine the all query result. Quality control also used Predicator and a verifier for predicate the accuracy and select the best answer.

IV. MATHEMATICAL MODELLING

- S= {I,O, F,U}

Where,

- **I: Input:{ Q, C }** Where,

  Q= No. of queries.

  C= Cost

- **O: Output: { Q,Op,Qr }** Where,

  Q= Query

  Qp= Generated Query Plan

  Qr= Query Result

- **F: Functions**:{ Cq, Jq, Sq }

  Cq= Perform complex query optimization.

  Jq= Perform join query optimization.

  Sq= Perform selection query optimization.

- **U**: defines end user who wants to find Query result.

V. IMPLEMENTATION

The relational database systems, query optimization is providing query interfaces which are important for crowdsourcing. The propose system, query optimization for crowdsourcing system gives parallel system approch for crowd. 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**.**

Input:

- No. of queries.
- Cost

Output:

- Q= Query
- Op= Generate Query Plan

1. 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] .

- Input: C: Selection conditions ,L: Latency constraint
- Output: P: A query plan

Example:

```
SELECT R3.image
FROM IMAGE R3
WHERE make = “Toyota” AND style = “Sedan”
AND color = “White” AND quality = “high”[16]
```

Here, is example of Finding high-quality images of white Toyota sedans, where selection conditions (e.g., make = “Toyota”) are evaluated using crowd sourcing and the image m1 satisfying all the conditions is returned as a result[9].

2. 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: INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN.

Input: T 1, T 2: tuple sets to join ,QJ: join query,
L: latency constraint;
Output: P: A query plan

```
1. P← GENPLAN ()
2. return P
```

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[16]
```

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 s[9].

3. Complex Selection-Join Queries: The another group of Query optimization system is used complex query. This will containing both selections and joins. These queries can help users express more difficult crowd sourcing 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, It 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.

VI. OPTIMIZATION FRAMEWORK ALGORITHM

Input: Query Q, Cost C

Output: Query, Optimized plan, Query Result

Step 1: Initialize database and tables, load tables.
Step 2: Initialize C =nil
Step 3: Calculate Latency Min (Lmin)
Step 4: Execute Query SELECT
Step 5: Calculate Latency Max (Lmax)
Step 6: Compute Query cost Lmax – Lmin
Step 7: Do Step 3 to 6 for JOIN and COMPLEX
Step 8: Compare Latency

VII. RESULTS

In this section, First evaluate the query optimization schemes for the crowd power selection, join and complex queries in a crowdsourcing atmosphere .

The proposed system used a synthetic dataset Auto1 with a size of attributes. To evaluate query optimization approaches under many queries with unstable number of conditions, domain data sizes ,etc.Auto1 data set are use the specification of 205 cars from UCI Automobile dataset to generate a relation Vehicles[17].

Simulated crowdsourcing environment has the knowledge of the complete database of Auto1. When a HIT arrives, it searches the complete database and returns the correct answer to the CROWDSOURCING EXECUTOR.

Fig1. Data Provider form shows Automobile Attributes like ID, make, model, style, color ,quality ,image, review, sentiment. We can submit these record in to the database.

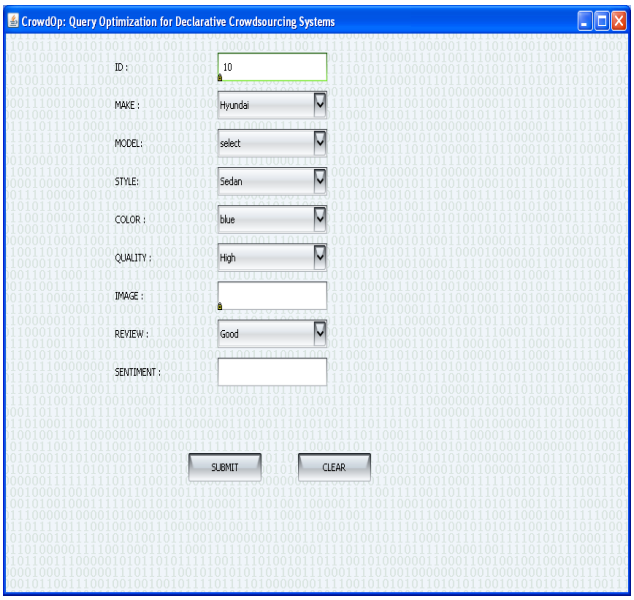


Fig 1. Form1 Data Provider

After Providing data that will be stored in a automobile database , the query result will be execute by query executor by using select ,join, complex operation. The Select operation shown in below fig.2.It will connect with the tables by using select operation.

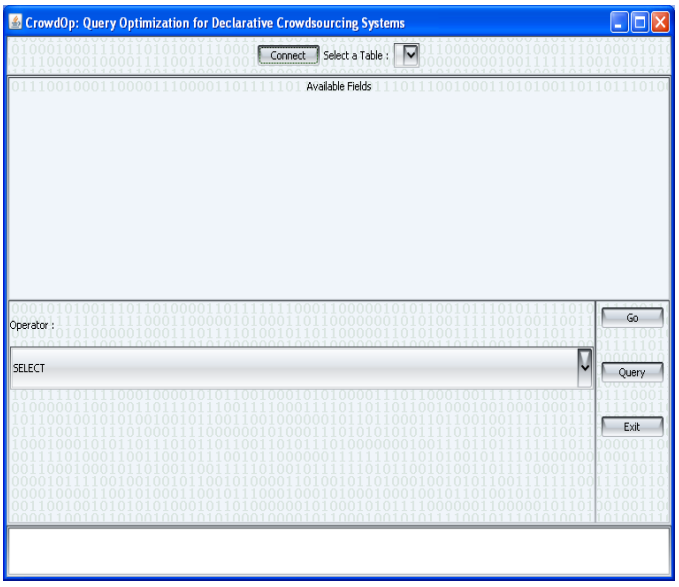


Fig.2. Query Executer

In fig.3 Select operation can select the table from the database. It will show the attributes like review, make, model, sentiment etc.

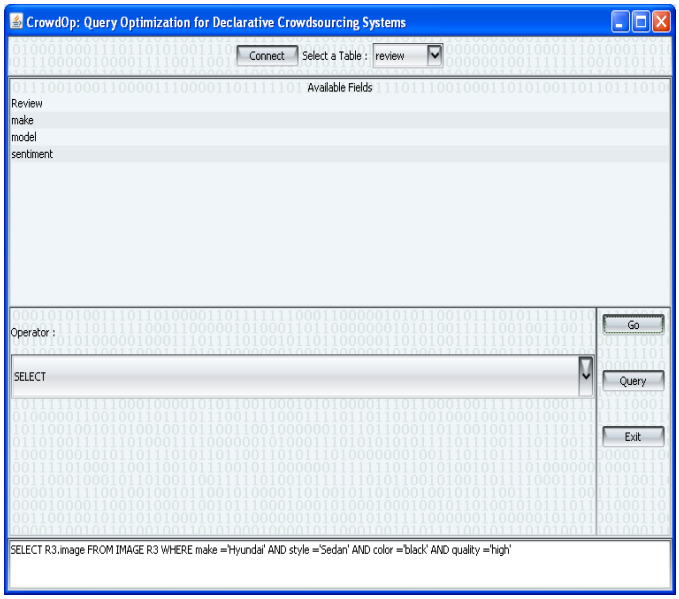
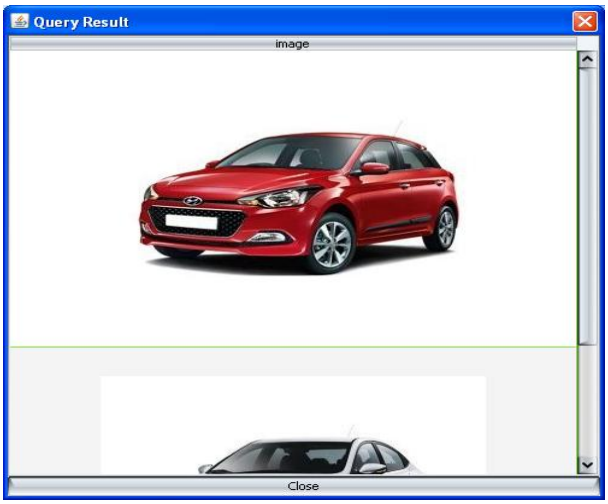


Fig3 .Select Query Execution

In the fig.4, final selection query result will be displayed.



VIII. CONCLUSION

Different approaches of query optimization for crowdsourcing have been discussed in detail. The efficient and effective optimization algorithm build up for select, join, complex query. In the present scenario, simulated crowd demonstrate the effectiveness of our query optimizer and confirm cost model and latency model. Cost and latency in query optimization purpose and generates query plans that provide a good balance between the cost and latency. sThe future plan is to extend the proposed system for more advance SQL operation and run on big data in a cloud environment.

ACKNOWLEDGMENT

I take this opportunity to thank Prof. N.V.Alone , Head of the Computer Engineering department, for his encouragement and guidance. I also want to thank my guide Prof. R. C Samant for her continuous help and generous assistance. She helped me in a broad range of issues from giving me direction, helping to find solutions to problems, outlining requirements and always having the time to see me. I also extend sincere thanks to all the staff members for their valuable assistance. Last but not least I am very thankful to my class-mates for all their valuable suggestions and support.

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] Ju Fan, Meihui Zhang, Stanley Kok, Meiyu Lu, and Beng Chin Ooi, "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.

[12] A. Marcus, E. Wu, D. R. Karger, S. Madden, and R. C. Miller, "Human-powered sorts and joins," Proc. VLDB Endowment, vol. 5, no. 1, pp. 13–24, 2011.

[13] Rohini Pingle, Rucha Samant, "A Review Paper on Query Optimization for Crowdsourcing Systems" in International Research Journal of Engineering and Technology (IRJET) Volume: 02 Issue: 09 |Dec-2015

[14] Jiannan Wang, Guoliang Li, Tim Kraskay, Michael J. Franklin z, Jianhua Feng," Leveraging Transitive Relations for Crowdsourced Joins", SIGMOD'13, June 22–27, 2013, New York, New York, USA.

[15] J. M. Hellerstein and M. Stonebraker, "Predicate migration: Optimizing queries with expensive predicates" , in Proc. ACM SIGMOD Int. Conf. Manage.Data,1993.

[16] Mark Mcllroy,"SQL Essentials" reference book,Blue sky Technology,2009.

[17] https://archive.ics.uci.edu/ml/datasets/Automobile