CROWD SOURCING SYSTEMS USING EFFICIENT QUERY OPTIMIZATION

1PODETI SRINIVAS GOUD  2 MR.N.NAVEEN KUMAR

1M. Tech Student, Department of CSE, School of Information Technology, JNTUH, Village Kukatpally, JNTUH, Mandal Balanagar, District Rangareddy, T.S, India.

2Assistant Professor, Department of CSE, School of Information Technology, JNTUH, Village Kukatpally, JNTUH, Mandal Balanagar, District Rangareddy, T.S, India.

ABSTRACT: Optimization of the query is the biggest issue now days for crowd sourcing system. Crowd sourcing is source for the experts to resolve the problem and freely sharing the solution with everyone also hiding the complexities and to relief the user from burden of managing the crowd. The user has to submit an SQL query and the system takes the responsible for compilation the query, generating the execution plan and evaluating within the crowd sourcing market. The Relational-Database-Management-Systems, query optimization is produce query interfaces that are important for crowd sourcing. The recommended system, a payment-based query optimization approach for crowd-sourcing systems. The value and latency consider in query optimize objective for recommended system and generates query plans that give a good balance between the payment 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 like the real crowd.

INTRODUCTION

Crowd sourcing and Query Optimization:

Crowd-sourcing is a new method of utilizing the facility of the crowd in comes that usually need a large number of people, and once the prices of their completion by traditional ways, in-house or by outsourcing, is not cost effective. Crowd-sourcing is channeling the experts need to solve a problem so freely sharing the solution with everybody. Crowd-sourcing is a rising paradigm which is based on harnessing the ability of crowd in solving issues. Crowd-sourcing is a form of out sourcing; although it usually does not need a formal contraction which is found in outsourcing tasks to an external organization specialized in this task to perform. Crowd-sourcing is also meant to achieve a wider range
of people, which can generally be required to get a solution correctly and simply.

**Query Optimization:**

Query optimization is a perform of many relative database management systems. The question optimizer attempts to determine the most efficient way to execute a given question by considering the doable question plans. Generally, the query optimizer cannot be accessed directly by users: once queries are submitted to database server, and parsed by the parser, they’re then passed to the query optimizer wherever optimization happens. A query is a request for data from a database. Queries results are generated by accessing relevant database data and manipulating it in a method that yields the requested information. Since database structures are complicated, in most cases, and especially for not-very-simple queries, the needed knowledge for a question can be collected from a database by accessing it in several ways that, through different data-structures, and in several orders. Every different method typically needs different interval. Query optimization realize the best question arrange in terms of estimated financial price.

Recent crowd-sourcing systems, like Crowd-DB, Quark and Dec, give an SQL-like command language as a declarative interface to the group. An SQL-like declarative interface is intended to encapsulate the complexities of dealing with the group and provide the crowd-sourcing system an interface that is familiar to most database users. A declarative querying improves the usage of the system, it requires the system to have the capability to optimize and provide a “near optimal” query execution plan for every question. Since a declarative crowd-sourcing query will be evaluated in some ways, the preferred of execution plan has a significant impact on overall performance, which includes the amount of queries being asked, the types of the questions and the monetary cost incurred. It is thus important to design an economical crowd-sourcing question optimizer that is able to consider all potentially smart question plans and select the “best” plan based on a value model and optimization objectives.

![Architecture of Query Optimization](image)

**Fig 1: Architecture of Query Optimization**

Crowd-sourcing is considered to be distributed and the crowd might be inexperienced within the task. The definition does not elaborate on the categories of crowd sourced tasks and the characteristics of the crowd-sourcing platform and what facilities it should give. These two definitions view crowd-sourcing from two different views, with little or no options in
common. Some queries cannot be answered by machines only. Processing such queries needs human input for providing information that is missing from the database, for activity computationally difficult functions, and for matching, ranking, or aggregate results supported fuzzy criteria. Crowd-DB uses human input via crowd-sourcing to process queries that neither information systems nor search engines will adequately answer. It uses SQL each as a language for complex queries and as the way to model data. While Crowd-DB leverages several aspects of traditional database systems, there are vital differences. Conceptually, a significant modification is that the traditional closed-world assumption for question process does not hold for human input. From an implementation perspective, human-oriented query operators are needed to require, integrate and cleanse crowd sourced information. Furthermore, performance and price rely on a number of new factors including worker affinity, training, fatigue, motivation and location.

Crowd-sourcing has created a variety of opportunities for many difficult issues by leveraging human intelligence. For example, applications similar to image tagging, tongue process, and semantic-based information retrieval will exploit crowd-based human computation to supplement existing machine algorithms. Naturally, human employees in crowd-sourcing solve issues based on their data, experience, and perception. it is thus not clear that issues will be better solved by crowd-sourcing than solving only using traditional machine-based ways. Therefore, a cost sensitive measure technique is required.

1. RELATED WORK

Mainly the query optimization is used three kinds of queries

- **Selection Queries:**

The selection query is used to pick data from a database. The result is hold on in a result table, known as the result-set. It will apply one or additional human recognized condition over the ordered pair in a single relation. A selection query applies one or additional human recognized selection conditions over the ordered pair during a single relation. Choice query has several applications in real crowd-sourcing scenarios, like filtering data and finding certain things.

Example:

```sql
SELECT R3.image
FROM IMAGE R3
WHERE build = "Volvo" and elegance = "Sedan"
AND color = "black" AND quality = "high"
```

Here, is example of Finding high-quality images of black Volvo sedan, where selection conditions (e.g., make =“Volvo”) area unit evaluated exploitation crowd-sourcing and the
image m1 satisfying all the conditions is came as a result.

- **Join Queries:**

An SQL join query is used to mix rows from two or more tables, based on a common field between them. The most common style of join is: SQL INNER join (simple join). An SQL INNER is a part of come all rows from multiple tables wherever they be a part of condition is met. A type of the various SQL JOINs area unit follows:

- **INNER JOIN:** Returns all rows once there is at least one match in each tables
- **LEFT JOIN:** come all rows from the left table, and the matched rows from the correct table
- **RIGHT JOIN:** come all rows from the correct table, and the matched rows from the left table
- **FULL JOIN:** come all rows once there is a match in one of the tables

One typical application of be a part of question is crowd-sourcing entity resolution, that identifies pairs of records representing a similar real-world entity. Other applications embrace subjective classification and schema matching.

Example:

```sql
SELECT R2._, R3.image
FROM AUTOMOBILE R2, IMAGE R3
WHERE R2.make = R3.make
AND R2.model = R3.model
Join Filter 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.

- **Complex Selection-Join Queries:**

The category of query optimization system is used complicated question. This can contain each choices and joins. These queries will facilitate users’ specific additional complex crowd-sourcing necessities. Q1 is an example of the complicated query that finds black cars with high-quality images and “positive” reviews. For the case where the latency constraint is not imposed, we are able to optimize the query arrange equally to traditional databases: apply some heuristic rules, such as pushing down choices and determining the join ordering, and then invoke the above-mentioned techniques for optimizing choices and joins.

### 2. FRAMEWORK

From the mentioned literature survey it is clear that there are existing systems that job on query optimization where data-sets or databases aren't any thus difficult. There square measure systems that works on the question execution plans although data-sets have some problematic values. Although there is good question optimizer, they are unable to deal in declarative crowd sourcing area. During this setting once user fire some question then existing system are
The question arranges is then executed by CROWD-SOURCING executor to get human intelligence tasks (or HITs) and transfer these HITs on crowd sourcing platforms. Supported the HIT answers collected from the crowd, executor executes the query and returns the generated results to the user.

3. EXPERIMENTAL RESULTS

In addition to this applications and algorithm of the idea of crowd sourcing system several results have investigated per the performance side. These works are often classified into user participation, quality management. during this section, we first evaluate the effectiveness of our planned improvement schemes for the crowd-powered choice, join and complex queries in a simulated crowd sourcing surroundings, and then examine the latency model and query optimization via experiments on the real crowd on Amazon Mechanical Turk (AMT). We develop efficient and effective improvement algorithms for choose, join and complex queries. Our experiment on each simulated and real crowd demonstrates the effectiveness of our question optimizer and validates our price model and latency model. This section evaluates our optimization approach for choice queries. We tend to initial consider the objective of cost reduction wherever no budget constraint is imposed. We vary the number of choice conditions during a choice question from two to six, and at random generate ten queries for every
selection condition setting and report the common price.

1) Financial cost: The monetary price of question strategy \( q \), represented by price \( (Q) \), is that the rewards obtained for execution all crowd-sourcing operators within the question setup. The value of associate degree operator depends on the price given to crowd for each query produced by the operator.

2) Accuracy: Crowd-sourcing could yield relatively low-quality results or even noise, if there are spammers or cruel workers. Thus, accuracy is occupied as another necessary performance metric to measure the standard of crowd-sourcing results. In our CROWD-OP system, we have a tendency to tend to address the accuracy issue by using block.

3) Latency: As crowd sourcing takes time, latency is clearly introduced to enumerate the quickness of question analysis. However, it is non-trivial to calculate and enhance latency.

4. CONCLUSION

Query optimizer for crowd-sourcing are discussed well. The efficient and effective optimization algorithm develop for choose, join, complex question in the present situation, simulated and real crowd demonstrate the effectiveness of our query optimizer and take review of query optimization objective and generates question plans that provide an honest balance between the cost and latency.

REFERENCES


