ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2016

Optimizing Analytical Queries on Probabilistic Databases with Unmerged Duplicates Using MapReduce

Kavita K. Beldar[1], Prof. M.D.Gayakwad[2], Prof. M. K. Beldar[3]

Research Scholar, Dept. of I.T., BVDUCOE, Pune, India[1]

Assistant Professor, Dept. of I.T., BVDUCOE, Pune, India[2]

Assistant Professor, Dept. of Mech. Engg., BVDUCOE, Pune, India[3]

ABSTRACT: In this paper, we present the first known approach for efficiently handling complex analytical queries over probabilistic databases with unmerged duplicates. Our technical indexing structure is for efficient access to the entity resolution information. The novel techniques are proposed for the efficient evaluation of complex probabilistic queries that can retrieve analytical data. Also used JOIN for summarized information over a large collection of possible resolution worlds. Generally an existent database holds the data whose correctness is unsure. In order to work with such data, there is a need to quantify the reliability of the data. This is achieving by using probabilistic databases. A probabilistic database is an unsure database in which the probable worlds have associated possibilities. Probabilistic Linkage is the process of combining multiple databases into one extensive database for analysis or linking multiple events. We proposed an indexing structure which reduces the complexity of the computations required when processing quires. We are applying map reduce to large probabilities dataset which will mapped over similar data. The proposed MapReduce algorithm gives the better time performance for query evaluation.

KEYWORDS: Probabilistic Databases, Unmerged Duplicates, MapReduce algorithm, indexing.

I. INTRODUCTION

Database system plays an important role every organization. All the organizations or manufacturing companies are totally depending on the correctness of the database. The database may contain the data from various organizations and the records can be duplicate or similar records. This type of database is called uncertain or unsure or probabilistic or dirty database.

Many of the real world databases hold data whose accurateness is unsure. Work on to this type of data there is necessitating computing the veracity of the data. Unsure databases in which the probable worlds have related possibilities are called a probabilistic database. Handling queries efficiently and understanding huge set of unsure data is the most important test in probabilistic databases. This thesis demonstrates that it is possible to effectively manage large, imprecise databases using a generic approach based on probability theory.

A MapReduce algorithm is developed for performing query efficiently on large probabilistic databases. Here, the performance of time on different scenarios such that on 50000 and 250000 database size is checked. Queries on reduced data set merge with MapReduce and without MapReduce on 50000 as well as 25000 size database are performed. Same queries are run on Hadoop framework and compared the performance time on these different platforms. Our technique throughout a wide-ranging evaluation by real-life databases of online shopping records of Customer and their orders are corroborated

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2016

II. PROBABILISTIC DATABASES EXAMPLE

Following table shows the example of probabilistic database. Table 2.1 shows the customer details with its different attributes. The database contains the L_name Lorys 10 time's which are the duplicates records. To perform queries on such type of databases is a complex task. The record r1, r5, r9 contains the same records that is F_name Mats and L_name Lorys.

Table 2.1. Customer table

Sr.no	Emp_id	F_name	L_name	Gender	Location	Year
r1	101	Mats	Lorys	F	USA	2006
r2	102	Yakkov	Lorys	M	DUB	2005
r3	103	John	Lorys	F	IND	2009
r4	104	John	Lorys	M	JAP	2001
r5	105	Mats	Lorys	F	DUB	2010
r6	106	Yakkov	Lorys	F	ENG	2005
r7	107	John	Lorys	F	IND	2013
r8	108	Yakkov	Lorys	M	DUB	2005
r9	109	Mats	Lorys	F	PAK	2014
r10	101	Smith	Lorys	M	IND	2003

Table 2.2 shows the order table of the Customer persons. It shows that how many items purchased by each customer and with its total amount.

Table 2.2. Order Table

emp_id	emp_no	Items	Amount
r1	101	4	270
r2	102	2	122
r3	103	3	234
r4	104	6	5000
r5	105	1	455
r6	106	2	678
r7	107	3	123
r8	108	4	234
r9	109	1	150
r10	110	2	456

Here, in table 2.3 we calculated the probabilities of table 2.1.Here we compare each row with other rows for checking the similarity between two rows and columns. The record r1 is similar like r5 and r9. Here we calculate the probability using the technique jaccard similarity. Here P and Q are the two different set of records.
P={Mats, Lorys, f, USA, 2006}, Q={Mats, Lorys, f, IND, 2001}
JS= $(P\cap Q)/(P\cup Q)$
JS= $(r1\cap r5)/(r1\cup r5)$
JS= 3/5
JS=0.6
That is, the records in r1 are similar to the records in r2. The following table 2.3 shows the all probability results.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2016

Table 2.3.Probability table

Emp_no	Emp_id 1	Emp_id 2	Probability *P*
1	101	105	0.6
2	101	109	0.6
3	102	106	0.6
4	102	108	0.4
5	103	104	0.4
6	103	107	0.4
7	110	110	1

III.Related Work

In this section, the related works that include the overview of the entity linkages with uncertainty and ranking queries on probabilistic databases, MapReduce techniques, and Top-k ranking queries is introduced.

In 2010 Ekaterini Ioannou, Wolfgang Nejdl proposed solution which supports arbitrarily complex SQL queries with" uncertain" predicates [6]. Main focus is query evaluation on probabilistic databases. This system describes an optimization algorithm that can compute efficiently most queries. As data complexity of some queries is #P complete, this implies that queries do not admit so approximation algorithm and a Monte-Carlo simulation algorithm are used.

In 2011 Ming Hua, Jian Pei authors propose a novel framework for entity linkage with uncertainty [5]. The framework introduces a series of novelties: (i) it performs merges at run time based not only on existing linkages but also on the given query; (ii) it allows results that may contain structures not explicitly represented in the data, but generated as a result of a reasoning on the linkages; and (iii) enables an evaluation of the query conditions that spans across linked structures, offering a functionality not currently supported by any traditional probabilistic databases.

In 2012 Dan Olteanu, Hongkai Wen author proposed solution for leverages data and workload statistics and correlations. [4]. Here the ranking functions can be customized for different applications. Our solution is principled, comprehensive, and efficient. However, investigating unspecified attributes is particularly tricky since we need to determine what the user's preferences for these unspecified attributes. Proposed solution gives idea that the ranking function of a tuple depends on two factors: (a) a global score which captures the global importance of unspecified attribute values, and (b) a conditional score which captures the strengths of dependencies (or correlations) between specified and unspecified attribute values.

In 2013 authors proposed author investigated the problem of ranking query answers in probabilistic databases [3]. It gives a dichotomy for ranking in case of conjunctive queries without repeating relation symbols: it is either in polynomial time or #P-hard. The key observation is that there are queries for which probability computation is #P-hard, yet ranking can be computed in polynomial time. This is possible whenever probability computation for distinct answers has a common factor that is hard to compute but irrelevant for ranking.

IV.Proposed System

Modules of architecture are explained below in detail.

A. *Probabilistic database*

An input or our system is a large probabilistic database, which contains unmerged similar records. The small example probabilistic database is shown in table 2.1. Here the L_name "Lorys" occurs 10 times in the database. It gives the 10 similar results for single query. There is need to find the solution for handling these types of multifunction query.The databases are taken from the online shopping customer's database which stores the data in sequence from various system resources. Here we are going to consider two databases having different sizes. The main purpose behind taking two databases is the time performance.

B. *Construction of indexing structure*

Here we create the indexes on probabilistic database, which retrieves the results very fastlly. First it sorts the all records in alphabetically order then generates the index numbers. Also the database divides in to 2 different factors. That is records as male and records as female. Indexing calculates the factors by dividing the database into two factors i.e. factor f1 for male and factor f2 for female. The database is grouped into two sizes that are all records by male factor and all records by female factors. It gives fast query processing. Indexing algorithm decreases the complexity of query processing. It provides access to the database which holds unmerged duplicate records. Indexing provides easy creation of possible worlds with fast retrieving the probabilities. It provides random look ups and easy accessed ordered records in large databases. The indexing is done on L_name column, so the L_name is quickly accessed in large probabilistic databases. Following Fig. 4.1 shows the proposed system architecture. The proposed system architecture uses indexing structure and MapReduce technique for executing multifaceted queries on huge a probabilistic database.

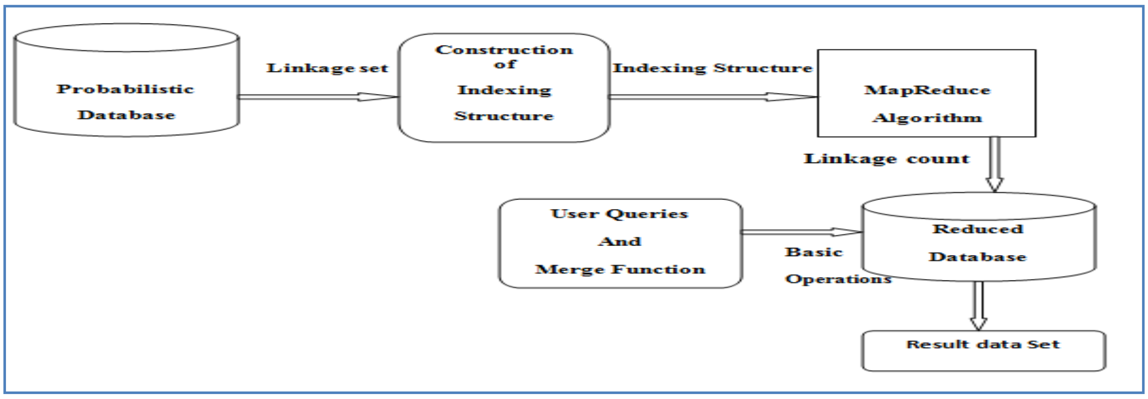

Figure 4.1 System architecture using MapReduce algorithm

C. MapReduce

For parallel programs and working on large amount of data in parallel the MapReduce model is widely used. A MapReduce programming is a worldview for performing operations on big databases in distributed situations. The map function divides the documents into single words and for every word in the document it generates<key, value> pairs. For every words in the documents uses the function map (name, document) emit (word, count). The MapReduce algorithm is divided in to all other smaller tasks which are given below that is input phase to final results.

1. Input phase: in the input phase we took a large database and translate this database in to a input file format.
2. Splitting: in splitting phase the database is divide into two different formats. Here the all F_name are splits into smaller subsets.
3. Mapping: in mapping phase the splitted data is converted into a <key, value> pair format. It gives the results zero or one key value pairs. After mapping the f_ name
   The results will be in the format <Mats, 3>, <John, 3> <Smith, 1>
4. Shuffling: In this step all the records are sorted by alphabetically and then merged together. The <key value > pairs are grouped together at the merging step. That is for example: <Mats, list<3>>.The sorting step takes input from merging step and sorts all key-value pairs by using keys. The output of this step is sorted key-value pairs.
5. Reducer: The reducer function takes group of key-value paired data as input and run reducer function on every of them. The data is merged, cleaned, aggregated in different ways; it necessitates broad range for processing. After execution is completed it gives the result zero otherwise more than one key value pairs.
6. Final result: The final result combines all the above steps together and generates the results. The total number of F_name are counted and grouped together. How many times a word occurred in a database is calculated at the reducer step.

D. Reduced dataset

The reduced dataset is nothing but the results of the MapReduce algorithm, which is shown in table 4.1. In a reduced dataset the duplicate records are grouped in to same link. For example the F_name "Mats" is linked into a single link, which occurs three times in database at different emp_id. The searching process increases automatically because the records are arranged by indexing and then in Mapper and reducer format

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2016

Table 4.1 Result of MapReduce Algorithm

F_name	Mapper					Reducer
	Emp_id	L_name	Gender	location	Year	
John	103	Lorys	M	USA	2004	3
	104	Lorys	F	USA	2008	
	107	Lorys	F	IND	2010	
Mats	101	Lorys	F	JAP	2006	3
	105	Lorys	F	IND	2003	
	109	Lorys	F	DUB	2004	
Yakkov	102	Lorys	M	USA	2003	3
	106	Lorys	M	DUB	2009	
	108	Lorys	F	USA	2012	
Smith	110	Lorys	F	IND	2005	1

E. *Basic operations*

Here we perform some basic operations on reduced dataset. That operation means aggregation and top-k queries. We are executing queries database which having the size 50000.
Query 1: select *from Customer where L_name=”Lorys”
It gives all the records whose L_name is Lorys. The result of query 1 is same as shown in above table 2.1.

F. *Retrieving by groups*

Here we retrieve the queries by using group by clause. The condition is given by on customer’s location and grouped by columns.
Select column1, column2
From Customer
where location =”USA”
Group by column1, column2
Query 2: SELECT * FROM Customer Where location =”USA”

Table 4.2. Results of USA locations

101	Mats	Lorys	F	USA	2006
102	Yakkov	Lorys	M	USA	2003
107	John	Lorys	F	USA	2008
109	Mats	Lorys	F	USA	2006

G. *Retrieving of Factors*

Here we create two different factors that are factor as male and factor as female, which is done by using the indexing structure.
Query 3→SELECT *FROM Customer where gender=”male”

Table 4.3. Records as a male

F_name	L_name	Gender	Location	year
Yakkov	Lorys	M	USA	2003
John	Lorys	M	IND	2004
Yakkov	Lorys	M	DUB	2009

Query 4→SELECT *FROM Customer where gender=”female”

Table 4.4. Result of all female records

F_name	L_name	Gender	Location	year
Mats	Lorys	F	USA	2006
John	Lorys	F	JAP	2008
Mats	Lorys	F	IND	2001
John	Lorys	F	USA	2010
Yakkov	Lorys	F	DUB	2012
Smith	Lorys	F	IND	2005

H. Top-k Query

This query retrieves the top three highest probabilities from the table 2.3
Query 5→ SELECT Top 3* FROM Customer

Table 4.5. Results of top-3 records

Sr no	Emp_id 2	Emp_id 2	Probability
1	110	110	1
2	101	105	0.6
3	102	108	0.4

The entire above query 1 to query 5 are executed on the second database having size 250000. The execution time performance is compared on both databases which is shown in table 5.1 and table 5.2.

V. PSEUDO CODE

```
MapReduce Algorithm:
Input: A probabilistic database D, Indexing structure, Views v
Output: list (F_name, count)
//It map’s all duplicate F_name together with its all details

1: Class mapper
2: Method map (view v, table t)
3: for all F_name F ∈ table t do
4: If (F_name ==F_name)
5:Count++
6: Else
7:Emit (F_name f, count n) //
8: Class reducer
9: Method reducer (F_name f, counts (c1, c2, c3,….,cn)) do
10: Sum <- 0
11: For all count c ∈ counts [c1, c2 …cn] do
12: Sum <- sum+c
13: Emit (F_name, count, sum)
```

VI. RESULTS

The following table shows the time performance of each query.Here , same queries are perofrmed on two different sizes of data sets that is 50,000 and 250000. Also did the comparision between them with respect to time that is data set size 50,000 merge with MapReduce and without MapReduce and data set size 250000 merge with MapReduce and without MapReduce. With the data set size 250000 the time performance is better than data set size 50,000. Here the time performance is meauserd in milliseconds.

Table 5.1 Time Performance on 50000 dataset

Query number	Times in millisecond	
	With MapReduce	Without MapReduce
Q1	2988	3865
Q2	1763	2369
Q3	1671	2586
Q4	1512	2310
Q5	1245	1709
Q6	1463	2786
Q7	1375	2336
Q8	950	1763
Q9	1157	1839
Q10	1354	2137

The following table 5.2 shows the time performance on the second database having 250000 sizes. With MapReduce gives better time performance as compared without MapReduce algorithm**.**

Table 5.2 Time Performance on 250000 dataset

Query number	Times in Milliseconds	
	With MapReduce	Without MapReduce
Q1	2967	4627
Q2	3163	4323
Q3	3229	4358
Q4	3061	4152
Q5	2765	4531
Q6	2378	3687
Q7	2134	3793
Q8	1834	3265
Q9	1957	2846
Q10	1288	2546

Fig.5.1 shows the graphical representation of table 5.1 and fig 5.2 shows the graphical representation of table 5.2. The graph shows the different time performance level of each query in milliseconds is which shown on Y axis and the size of data entries shows on X axis.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2016

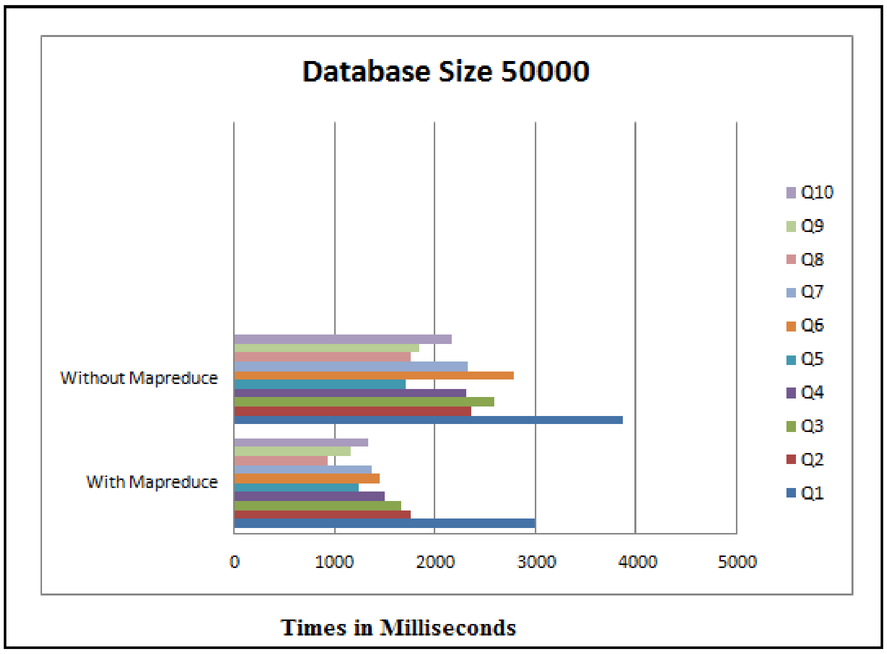

Figure 5.1 Query processing time vs. Database size 50000

Figure 5.2 shows the graph of time performance on database size 250000; we execute the same queries on both dataset. The time performance is differing on both datasets. The second dataset gives higher time as compare to the first database.

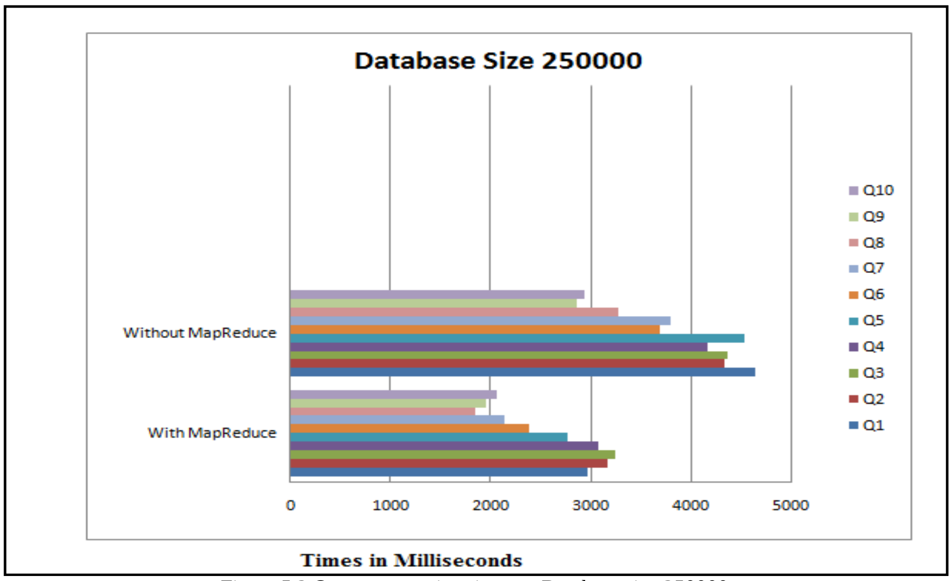

Figure 5.2 Query processing time vs. Database size 250000

VII. Conclusion and Future Work

In this paper we are able to address the resolution problem through a generic framework for processing complex queries over unmerged duplicates. We consider two different databases with duplicated instances. First the indexing algorithm is introduced which provides for quick access of databases with its possible entities. Also it retrieves the query results fast. Second the MapReduce algorithm is introduced, which manages massive amount of probabilistic database easily and provide efficient way for query processing. Experimental evaluation is done on two different real databases contains 50000 and 250000 duplicate records respectively. The comparison between time performances is done on both with MapReduce and without MapReduce algorithm. Using MapReduce algorithm the time performance is better than without MapReduce algorithm. In this we are able to minimize the time using Map Reduce with indexing that provides efficient access to the possible entity merges and their probabilities.

In future we will try to find the solution for index balancing for huge linkage factor of attributes and how to increase the time performance when the database size will increase more than 250000.

REFERENCES

Journal Article

1. Kavita K. Beldar, M. D. Gayakwad, Debnath Bhattacharyya, Tai-hoon Kim, "A Comparative Analysis on Contingence Structured Data Methodologies", *Volume 10, No.5 International Journal of Software Engineering and its Application under ISSN 1738-9984.*
2. Youzhong Ma, Xiaofeng Meng, "Set similarity join on massive probabilistic data using MapReduce", *Springer Science + Business Media New York 2013, 3 December 2014.*
3. Maximilian Dylla, Iris Miliaraki, Martin Theobald, "Top-k Query Processing in Probabilistic Databases with Non-Materialized Views", *University of Antwerp, 2013.*
4. Dan Olteanu Hongkai Wen, "Ranking Query Answers in Probabilistic Databases: Complexity and Efficient Algorithms", *EPSRC EP/G069557/1 FRESNEL project, 2012.*
5. Ming Hua, Jian Pei, Wenjie Zhang, Xuemin Lin, "Ranking Queries on Uncertain Data: A Probabilistic Threshold Approach*", SIGMOD'08, June 9–12, 2011.*
6. Ekaterini Ioannou, Wolfgang Nejdl, Claudia Niederee, "On the Fly Entity Aware Query Processing in the Presence of Linkage" *36th Inter. Conf. on Very Large Data Bases, Sept 1317, 2010.*
7. Ekaterini Ioannou, Minos Garofalakis, "Query Analytics over Probabilistic Databases with Unmerged Duplicates", *IEEE Trans. Knowl. Data Eng., Vol.27 No.8 August 2015.*

BIOGRAPHY

Miss. Kavita Kantilal Beldar is a Post Graduate Research Scholar in the Information Technology Department, Bharati Vidyapeeth Deemed University College of Engineering Pune Maharashtra, India. She received Bachelor of Engineering Degree in 2014 from Solapur University, Maharashtra, India. She has published research article in the Scopus indexed journals. Her research interests are Data Mining, Information Retrieval, and Network Security etc.