An Enhanced Query Processing Algorithm for Distributed Database Systems

Njoku Donatus O., Nwokorie Chioma E., Madu Fortunatus U.

Abstract— The growth of the web and the unlimited use of database management systems have brought to the forefront and integrated interconnection of diverse and large numbers of information sources known as distributed databases. The major problem in such an environment is the time taken to responds to query and the cost of transmission across the distributed site. This paper aimed at developing an enhanced query processing algorithm for distributed database systems, the proposed architecture and algorithm, uses the Iterative Dichotomizer 3 (ID3) as query optimizer that forms a decision tree, which decides the rules, patterns and the best execution plan to execute query at the barest minimum time. The applications of this work in organization, will automate the administration of operations, by providing user interactive platform for all stations to make queries and update their records. The object oriented analysis and design methodology was used in the study and the application was developed using MySQL, PHP and XAMP sever. The result shows that due to train dataset in the design of the database, the execution time of the algorithm in search, updating and querying of databases information in distributed sites, was less when compare to system search engine running time.

Keywords— Iterative Dichotomizer, Query Optimizer, Database, Algorithm, Decision Tree

1 Introduction

A distributed database system is the combination of two different technologies used for data processing: Database Systems and Computer Networks. The main component of a database is the data which is basically collection of facts about something. Distributed database (DDB) is a collection of multiple, logically interrelated databases distributed over a computer network [4]. Retrieval of data from different sites in a DDB is known as distributed query processing.

In query processing, the database users generally specify what data are required rather than specifying the procedure to retrieve the required data. Thus, an important aspect of query processing is query optimization. In query optimization, the optimizer of the database system finds a good way to execute the queries. Query processing is more complex and difficult in distributed environment in comparison to centralized environment as large numbers of parameters affect the performance of [2] distributed queries. Relations may be fragmented and/or replicated, and considering many sites to access, query response time may become very high. The distributed query optimization has several problems related to the cost model, larger set of queries, optimization cost, and optimization interval [9].

The goal of Distributed Query Optimization is to execute queries efficiently in order to minimize the response time and the total communication cost associated with a query.

Therefore, it seems logical to look at potential benefits in relation to their costs or timing constraints. The three major activities in the processing of distributed database system include; database is fragmented; complex mechanism is used to allocate the database fragment to the different sites and the execution of task takes place.

It is believed that an effective database fragmentation improves the performance of the database. No doubt fragmentation increases the complexity of physical database design but it significantly impact performance and manageability. A query normally has many possible execution strategies, and choosing a suitable one for processing a query is known as query optimization and is expressed by using a high-level language such as SQL (Structured Query Language) in relational data model. The main function of a relational query processor is to transform a high-level query into an equivalent lower-level query (relational algebra), and the transformation must achieve both correctness and efficiency. Execution strategy for the given query is implemented by lower-level query [1]. Since data is geographically distributed in distributed relational database system, the processing of a distributed query is composed of the following phases, which include: local processing phase; reduction phase, and final processing phase.

The local processing phase basically involves local processing such as selections and projections. The reduction phase uses a sequence of reducers (i.e, semi-joins and joins) to reduce the size of relations. The final processing phase sends all resulting relations to the assembly site where the final result of the query is constructed. The machine learning algorithms of [11] Iterative Dichotomizer 3 (ID3) is use as an optimizer to process a distributed query which would searching all relations directly to the assembly site database, by minimizing the time and overall costs required for the applications to run in the network.

2 Related work

In [8], query optimization and processing is one of the key technologies in a distributed database system. It generally uses semi-join operation to improve the time response performance of query and reduce communication cost. Query optimization and processing adopt reasonable algorithms and precisely reduce the transmission of information as far as possible, which increase the response time performance of the query, and reduce system overhead. The cost is different for different query processing method, which means that the query optimization and processing of distributed database become more and more important. According to [4] in their work opined that implementation of distributed database were hindered with lots of challenges perused in past researches, few of such include unreliable network technology, high cost of Comput-

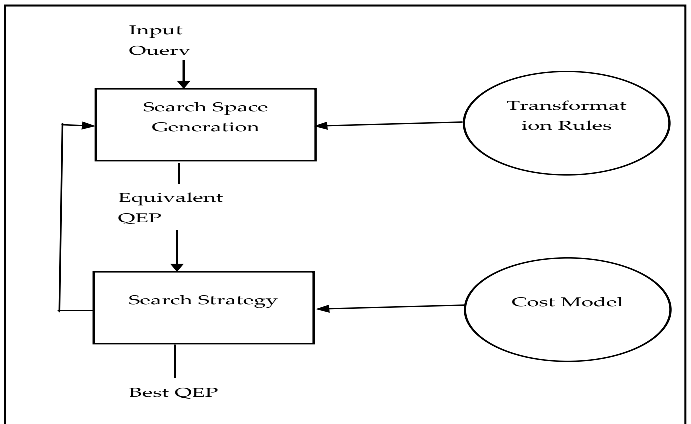

Fig. 1: Query Optimization Process Model

ers, and insecurity among users and also some of the challenges encountered include; problems of distribution of resources, search and updating of resources. And the query strategy optimization is more important between them [12]. There will be several strategies in the same query due to that the data are stored in different sites. Below is the query optimization process model.

The optimization engine then performs various analyses on the query data, generating a number of valid evaluation plans. In [8], [12] there, it determines the most appropriate evaluation plan to execute. After the evaluation plan has been selected, it is passed into the DBMS' query-execution engine (also referred to as the runtime database processor), [1] where the plan is executed and the results are returned.

2 .1.1 Distributed Query Processing Techniques

A database query is an instructing command to DBMS to update or retrieve specific data to and fro the physically stored medium. The actual updating and retrieval of data is performed through various "low-level" operations. Examples of such operations for a relational DBMS can be relational algebra operations such as project, join, select and Cartesian product, etc.

2.1.2 PHASES OF QUERY PROCESS TECHNIQUES

There are three phases that a query passes through during the DBMS' processing of that query:

1. Parsing and translation
2. Optimization
3. Evaluation

Most queries submitted to a DBMS are in a high level language such as SQL (Structured Query Language). During the parsing and translation stage, the human readable form of the query is translated into forms usable by the DBMS.

After parsing and translation into a relational algebra expression, the query is then transformed into a form, usually a query tree or graph that can be handled by the optimization engine. The optimization engine then performs various analyses on the query data, generating a number of valid evaluation plans. From [1] it determines the most appropriate evaluation plan to execute. After the evaluation plan has been selected, it is passed into the DBMS' query-execution engine (also referred to as the runtime database processor), where the plan is executed and the results are returned.

2.1.3 QUERY PROCESSING IN RELATIONAL DATABASE SYSTEMS

The conventional method of processing a query in a relational DBMS is to parse the SQL statement and produce a relational calculus-like logical representation of the query, and then to invoke the query optimizer, which generates a query plan [1]. The query plan is fed into an execution engine that directly executes it, typically with little or no runtime decision-making.

The conventional method of processing a query in a relational DBMS is to parse the SQL statement and produce a relational calculus-like logical representation of the query, and then to invoke the query optimizer, which generates a query plan. The query plan is fed into an execution engine that directly executes it, typically with little or no runtime decision-making [1].

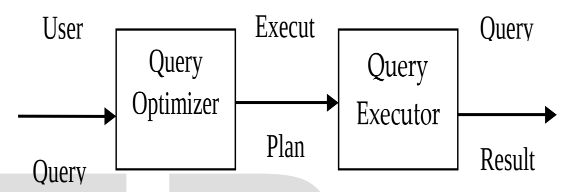

Fig. 2: Relational Query Processor

In many cases the query plan also includes low-level"physical" operations like sorting, network shipping, etc. that do not affect the logical representation of the data. Certain query processors consider only restricted types of queries, rather than SQL. A common example of this is select project-join or SPJ queries: an SPJ query essentially represents a single SQL SELECT-FROM-WHERE block with no aggregation or subqueries.

On the [8]whose work are on "A Dynamic Query Optimization Approach for Autonomous Distributed Database Systems", has discussed that Query processing in a distributed database system requires the transmission of data between sites using communication networks. Distributed query processing is an important factor in the overall performance of a distributed database system. In distributed query optimization, complexity and cost increases with increasing number of relations in the query. Cost is the sum of local cost (I/O cost and CPU cost at each site) and the cost of transferring data between sites. [7] works on Query Execution and Maintenance Costs in a Dynamic Distributed Federated Database, proposed that the cost of query evaluation in a Dynamic Distributed Federated Databases (DDFD) depends on the topology connecting the database nodes together [9]. Different topologies provide opportunities to adopt a variety of query optimization strategies and topology also influences the efficiency of these strategies. In [1] the distributed database system, the query optimization includes two parts; the query strategy optimization and local processing optimization

2.2 MATHEMATICAL COST MODEL FOR QUERY EX-

ECUTION COST

One of the problems in query optimization is to accurately estimate the costs of alternative query execution plans. Optimizers cost query plans using a mathematical model of query execution costs [8] that relies heavily on estimates of the cardinality, or number of tuples, flowing through each edge in a query plan.

Cost Function

An optimizer cost model [12] includes cost functions to predict the cost of operators, and formulas to evaluate the sizes of results.
Cost function (in terms of time) = I/O cost + CPU cost + Comm. Cost. (1)

Total cost = CPU cost + I/O cost + communication cost (2)
CPU cost = unit instruction cost ∗ no. of instructions (3)
I/O cost = unit disk I/O cost ∗ no. of disk I/Os (4)
Communication cost = message initiation + transmission (5)

Response Time:

This the elapsed time between the initiation of query and the completion of a query. Response time = CPU time + I/O time + Communication Time (6)

CPU time = Unit Instruction Time *no. of Sequential Instructions (7)

I/O time = Unit I/O time *no. of Sequential I/Os (8)
Communication time = unit msgs initiation time * no. of sequential msgs + unit transmission time * no. of sequential byte (9)

The goal of this paper is to review the works of [2], [4], and propose a model for querying a Distributed Database System (DDBMS) using Iterative Dichotomizer 3 (ID3) algorithm is the search of data in query respond time. The propose architecture for the distributed database system is show in fig. 3

Proposed Algorithm

A modified ID3 algorithm was developed to generate the decision tree, which creates simple and efficient tree with the smallest depth. Unlike binary tree (such as used for Huffman encoding where there is just a left and right node), ID3 decision tree can have multiple children and siblings. It makes use of two concepts when generating a tree from top-down and these concepts includes: Entropy and Information Gain.
Entropy is the measurement of uncertainty where the higher the entropy, then the higher the uncertainty [6]. The ID3 algorithm is used to train a certain dataset S to generate a decision tree which is stored in memory. At runtime, this decision tree is used to classify new unseen test cases by working down the decision tree [10] using the values of this test case to arrive at a terminal node that tells what classes the test case belong.
Entropy H(S) is a measure of the amount of uncertainty in the (data) set S i.e. entropy characterizes the (data) set.

$$H(S) = -\sum_{x \varepsilon X} p(x) \log_2 p(x) \quad (10)$$

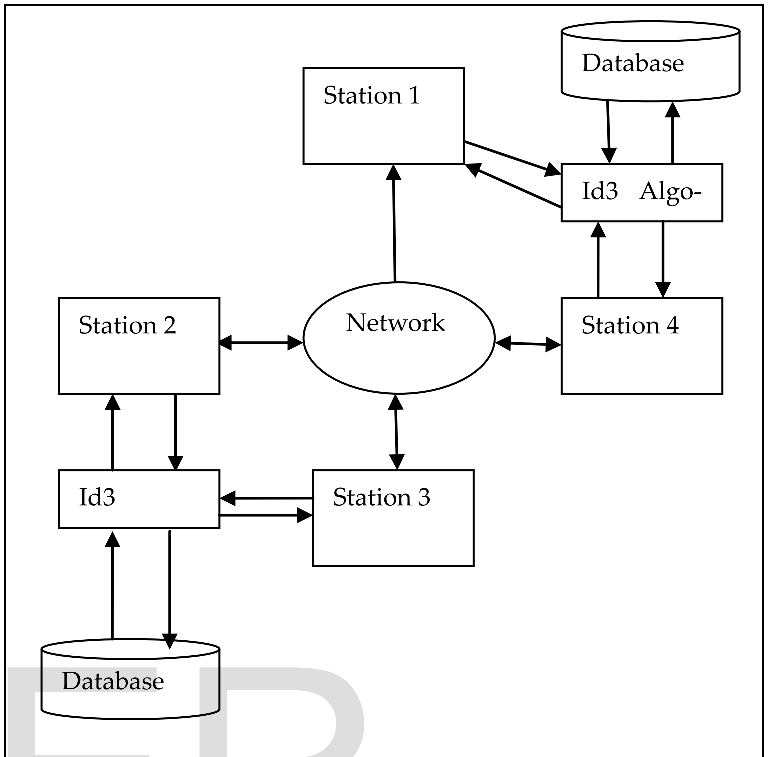

Fig. 3: Architecture of Distributed Database System

From equation (10), S- The current (data) set for which entropy is being calculated (changes every iteration of the ID3 algorithm) x- Set of classes in S, where:
p(x): The proportion of the number of elements in class x to the number of elements in set S
When H(S) = 0, the set S is perfectly classified (i.e. all elements in S are of the same class).
Information gain IG (A) is the measure of the difference in entropy from before to after the set S is split on an attribute A. In other words, how much uncertainty in S was reduced after splitting set S on attribute A.

$$IG(A,S) = H(S) - \sum_{t \varepsilon T} p(t) H(t) \quad (11)$$

H(S) - Entropy of set S
T- The subsets created from splitting set attribute A
P(t) - The proportion of the number of elements in t to the number of elements in set S
H(t)- Entropy of subset t

3 CLASSIFICATION EVALUATION OF TRAINING

DATASETS

Table 1: Data Collection for Calculating entropy and information gain

Attributes	Instances	No of Instances	Gender F	Gender M
Under 16	gambling	1	0	1
	Unlawful assembly	4	1	3
16-20	Stealing with violence,	7	2	5
	Army Robbery/Murder, Breaking of Burglary	4	0	4
	gambling	6	1	5
	Unlawful assembly	7	1	6
21-25	Cheating	15	2	13
	Impersonation	8	1	7
	Homicide; suicide infanticide	6	1	5
	Forgery and Personation	13	2	11
	Stealing with violence,	37	4	33
	Abduction	16	1	15
	Army Robbery/Murder, Breaking of Burglary	38	2	36
	gambling	24	3	21
	Abduction	10	4	6
	Unlawful assembly	34	3	31
	Corruption, Bribe and Abuse of office	19	8	11
	Frauds by Trustees	2	0	2
	Sedition & undesirable Publication	2	0	2
	Rescues, Obstructing Officers of Court	1	0	1
	Offences Relating To Worship	2	0	2
26-50	Cheating	2	0	2
	Homicide; suicide infanticide	2	0	2
	Stealing with violence,	3	0	3
	Army Robbery/Murder, Breaking of Burglary	2	0	2
	Abduction	3	3	0
	Frauds by Trustees	2	0	2
51 and above	Homicide; suicide infanticide	1	0	1
	Army Robbery/Murder, Breaking of Burglary	2	0	2
	Abduction	1	1	0
	Corruption, Bribe and Abuse of office	3	1	2
		277	41	236

The attributes which will be used as the root nodes [10] is calculated using information gain equation (11) and it is measured based on the entropy function from information theory. Calculating Entropy H(S) before splitting, using equation (10) then,

$$H(S) = -\frac{41}{277}\log_2\left(\frac{41}{277}\right) = 0.604849 \quad (12)$$

The entropy after splitting using the attributes (Female) and their instances from table 1:

A_1: Under 16(Gambling, Unlawful assembly)

For Gambling (F= 0, male=1)

Entropy H ($A_{1\ \text{Gambling}}$)

$$= -\frac{0}{1} * \log_2 \frac{0}{1} - \frac{1}{1} * \log_2 \frac{1}{1} = 0.0000 \quad (13)$$

For Unlawful assembly (F= 1, male=3)

Entropy H($A_{1\text{Unlawfulassembly}}$)

$$= -\frac{1}{4} * \log_2 \frac{1}{4} - \frac{3}{4}\log_2 \frac{3}{4} = 0.811278 \quad (14)$$

Entropy H(A_1) = Sum of total no instances in an Attributes/sum total of the given set S

$$S = \frac{1}{277} * (0.0000) + \frac{4}{277}(0.811278) = 0.011715 \quad (15)$$

Information Gain (IG) is calculated using the formula in equation (11) we have:

Gain (A_1, S) =

H(S) - Entropy H(A_1) = 0.604849-0.011715 = 0.593134 (16)

A_2: 16-20(Stealing with violence, Army Robbery/Murder and Breaking of Burglary, gambling, Unlawful assembly)

For Stealing with violence (F= 2, male=5)

EntropyH ($A_{2\ \text{Stealing with violence}}$)

$$= -\frac{2}{7} * \log_2\left(\frac{2}{7}\right) - \frac{5}{7} * \log_2\left(\frac{5}{7}\right) = 0.764205 \quad (17)$$

For Army Robbery/Murder and Breaking of Burglary (F= 2, male=5)

Entropy H ($A_{2\ \text{Army Robbery/Murder and Breaking of Burglary}}$)

$$= -\frac{0}{4} * \log_2 \frac{0}{4} - \frac{4}{4} * \log_2 \frac{4}{4} = 0.0000 \quad (18)$$

For gambling (F= 1, male=5)

Entropy H ($A_{2\ \text{gambling}}$)

$$= -\frac{1}{6} * \log_2 \frac{1}{6} - \frac{5}{6}\log_2 \frac{5}{6} = 0.650022 \quad (19)$$

For Unlawful assembly (F= 1, male=6)

Entropy H ($A_{2\ \text{Unlawful assembly}}$)

$$= -\frac{1}{7} * \log_2 \frac{1}{7} - \frac{6}{7}\log_2 \frac{6}{7} = 0.591673 \quad (20)$$

Entropy H (A_2) = Sum of total no instances in an Attributes/sum total of the given set

$$S = \frac{7}{277} * (0.74205) + \frac{4}{277}(0) + \frac{6}{277}(0.650022) + \frac{7}{277}(0.5916737) = 0.048374 \quad (21)$$

The information Gain of the calculated values for the parameters are summarized in table 2.

Table 2: Summary of Information Gain

H(S)	0.604849
Gain (A1 Under 16, S)	0.593134
Gain (A2: 16-20) , S)	0.556505
Gain (A3: Aga 21-25) , S)	0.390188
Gain (A4: age 26-50 , S)	0.604849
Gain (A5: 51 and above, S)	0.594904

Attribute age range 26-50 has the highest information gain and it is therefore chosen as the root node of the decision tree in developing the application in the database system drawn thus:

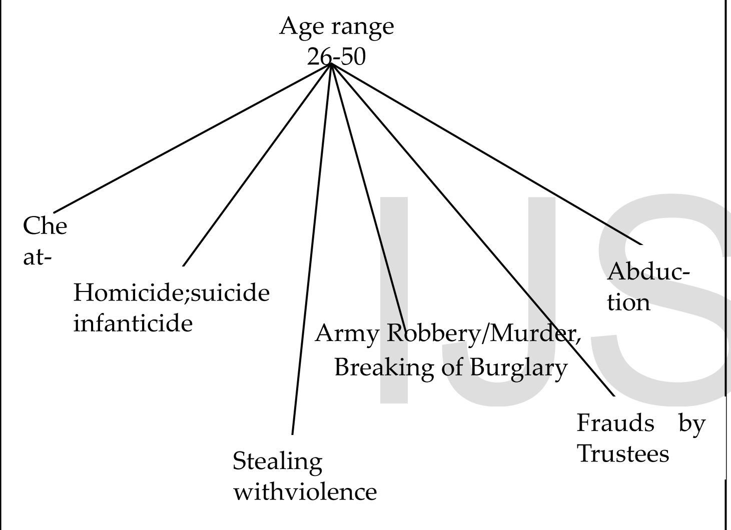
Age range 26-50
Cheat-
Homicide;suicide infanticide
Stealing withviolence
Army Robbery/Murder, Breaking of Burglary
Frauds by Trustees
Abduction

Fig 5: Decision root for Query database

The decision tree in fig. 5 above represents the natural way of presenting a decision-making process as used in this paper to optimized query time, because they are simple and easy for anyone to understand how the pattern will follow.

Pseudo Code for ID3 Algorithm

```
Function ID3 (I, 0, T)   {
/*I is the set of input attributes
*O is the output attribute
*T is a set of training data
**function ID3 returns a decision tree*/
if (T is empty)
{
return a single node with the value “wrong”;
                    }
if (all records in T have the same value for O) {
Return a single node with that value;
             }
if (I is empty)
  {
return a single node with the value of the most frequent value of O in T;
      }
/* case where we can't return a single node */
Compute the information gain for each attribute in I relative to T;
let X be the attribute with largest Gain(X, T) of the attributes in I;
Let {x_j| j = 1,2, .., m} be the values of X;
Let {T_j| j = 1,2, .., m} be the subsets of T when T is partitioned according to the value of X;
Return a tree with the root node labelled X and arcs labelled X1, X2, X3....Xm, where the arcs go to the trees ID3(I-{X}, O, T_1), ID3(I-{X}, O, T_2)...
ID3 (I-{X}, O, T_m);
         }
```

Interactive Dichotomizer 3 (ID3) is a supervised ML which induces general function from specific training data set as learning agent and test then into categories[11] (attributes and instances). The learning agents ID3 must search through the hypothesis space and locates the best hypothesis when given the test sets. According to [5] opined that machine learning as a process which causes systems to improve with experience.

4 SYSTEM IMPLEMENTATION

The results show in the table 2 from the propose algorithms, the analysis shows the performance of the query processing by minimizing communication time of the system when query is issued. This result obtain in table 2 which the entropy relationship and the various information again. This result was used to design the implementation of proposed system application. The result of the query is represented in figure 6, which the entropy relationship and the various information again. This result was used to design the implementation of proposed system application.

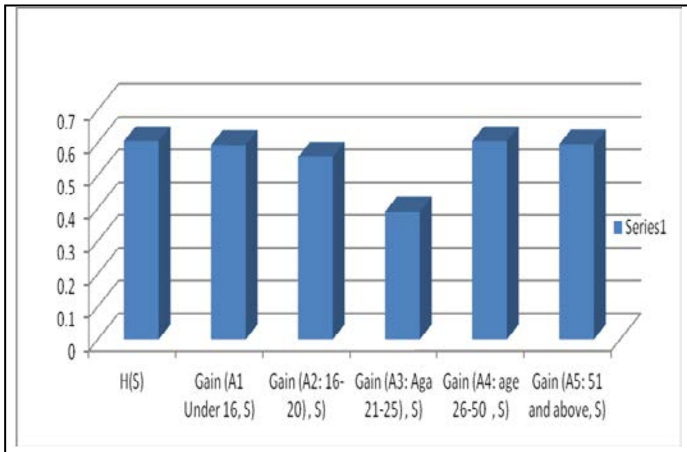

Fig. 6: Information Gain for the attribute determine Optimization of the Query Node

ID3 takes the attributes and instance in table 1 as input, where each example consists of a collection of attributes [11], togeth-

er with an outcome (or class) and induces a decision tree, [10] where each node is a test on an attribute, each branch is the outcome of that test and at the end are leaf nodes indicating the class to which the example, when following that path, belongs.
The result of the implementation shows that query time execution and response time of ID3 and search engine results in a distributed database as show in table 3

Table 3: Query Execution Time of ID3 and Search Engine on Offences

Offence	*ID3 query Response Time (sec) x10^{-3}*	*Search Engine query Response Time (sec) x10^{-3}*
Offence1	5	8
Offence2	8	13
Offence3	10	15
Offence4	6	10
Offence5	7	10

In table 3, the results shown comparatively can be represented in fig. 7, indicate that the ID3 algorithm results that follows the sets of rules, pattern and decision developed perform query on related offences in the database with minimal time comparing with the search engine in the distributed database result.

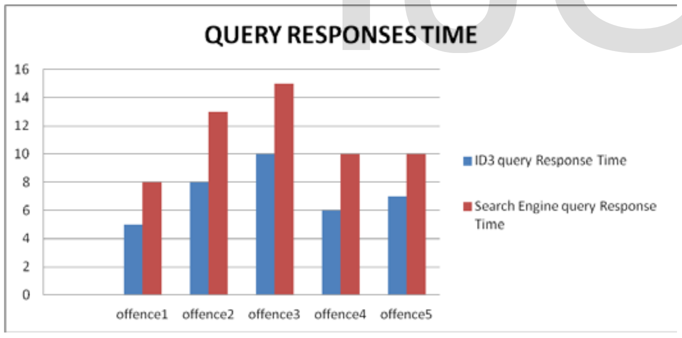

Fig. 7: Resultof Query Time of ID3 and Search Engine on offences

Again, the analysis implementation of the ID3 rules developed in querying the distributed database using the Age range of crime committed inmates in table 1 with the search engine of the database the results shown in table 4:

Table 4: Query Execution Time and Search Engine on Age Range

AGE RANGE	ID3 query Response Time (sec) x10^{-3}	Search Engine query Response Time (sec) x10^{-3}
Under 16	2	5
Age 16-20	5	10
Age 21-25	7	12
Age 26-50	1	3
51 & above	4	7

Fig. 8 shows the ID3 query response time is minimal comparing that of the search engine and the result is showed:

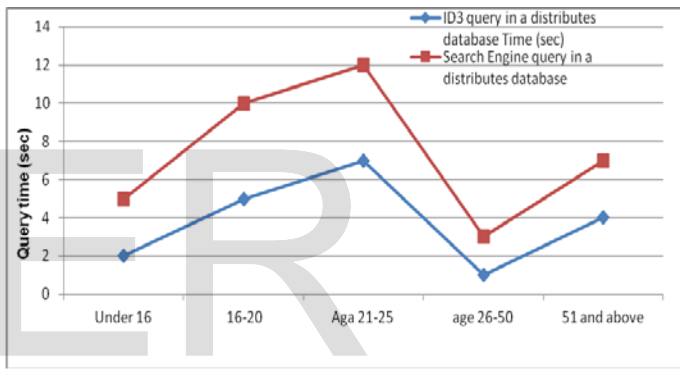

Fig. 8: Result of Query Time of ID3 and search Age

The application of this work is being developed using object oriented. Due to PHP and MYSQL are object-oriented programming have been a great success in designing application. The Net-Beans IDE, MYSQL-based, PHP is an open-source development environment which is available for most modern platforms.

5 CONCLUSION

This Paper on enhanced query processing algorithm is important for researchers and developer in design and implementation of distributed database system the application has been design and evaluates the cost of making queries, execution and corresponding response time is minimizing in the distributed database environments using the ID3 algorithm, and also, due to the complexity of the modern database management systems the execution time for process queries across distributed sites determine its accuracy in estimations and predictions for performance characteristics of the functionality of the databases. In this paper propose a solution for query execution and responses time and optimization in distributed database systems done by means of modified an enhanced ID3 algorithm.

ACKNOWLEDGMENT

The authors wish to thank E. N. Erumaka, O. F. Uzoh, E. O. Nwachukwu, on their contribution and insight in the course of this work. The staff and management of the department of Computer Science, Federal University of Technology, Owerri (FUTO) and most specially E. J. Njoku for her encouragement and patience.

REFERENCES

[1] J. Kunal, P. Viki and B.B. Meshram, " Query Processing Strategies in Distributed Database" Journal of Engineering, Computers & Applied Sciences (JEC&AS) ISSN No: 2319-5606 Volume 2, No.7, V.J.T.I., Mumbai, pp 72-76, 2013. .

[2] G.R. Abhijeet, and O. Bamnote "Query Processing In Distributed Database" International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 2, Badnera (M.S), India, pp 53-55, 2013

[3] K. T. Anand, and M. Tripathi, "A Framework of Distributed Database Management Systems in the Modern Enterprise and the Uncertainties removal" International Journal of Advanced Research in Computer Science and Software Engineering Volume 2, Issue 4, ISSN: 2277 pp 128, Jhansi ,India, 2012

[4] S. A. Idowu, S. O. Maitanmi "Transactions- Distributed Database Systems: Issues and Challenges" International Journal of Advances in Computer Science and Communication Engineering (IJACSCE) Vol. 2 Issue I ISSN 2347-6788, pp 24-26, 2014, Ilisan Remo, Ogun State, Nigeria

[5] E.O. Nwachukwu, C. Ugwu and E.E. Williams "Introduction to Artificial Intelligence and Expert System" MunaGenesis Concept Nig,. Owerri pp 25-39, 54(56), 2012

[6] C. E. Shannon "Prediction and Entropy of Printed English".(Retrieved04/23/2010). http://languagelog.ldc.upenn.edu/myl/Shannon.pdf

[7] P. Stone "Query Execution and Maintenance Costs in a Dynamic Distributed Federated Database."pp 56,74 (2012),

[8] W. Chihping, and S. C. Ming, "On the Complexity of Distributed Query Optimization", IEEE Transactions on Knowledge and Data Engineering, Volume 8, 1996

[9] P. Stone, G. Dantressangle, A. Bent, A. Mowshowitz, and B. Szymanski, Relational algebra-coarseg rained query cost models for DDFDs. In Proceedings of the Fourth Annual Conference of ITA . 2010, pp, 14-17

[10] J. R. Quinlan "Induction of Decision Trees, Machine Learning: , 1986.: 81-106,

[11] P. Wei, C. Juhua and Z. Haiping, "An Implementation of ID3 – DecisionTree Learning Algorithm" 2012. http://web.arch.usyd.edu.au/~wpeng/DecisionTree2.pdf

[12] K. Ridhi "Cost Estimates & Optimization of Queries Distributed Databases" International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 Vol. 2 Issue, Punjab, India, pp 3275-3277, 2013

- *Njoku Donatus O., Department of Computer Science, Federal University of Technology, Oweri, Imo State, Nigeria , +2348036824158. E-mail: njokudonatus1@gmail.com*
- *Nwokorie Chioma E., Department of Computer Science, Federal University, Owerri, Imo State, Nigeria, +2348035081310, E-mail: cenwokorie@gmail.com*
- *Madu Fortunatus U., Department of Computer Science, Federal Polytechnic, Nekede, Owerri, Imo State, Nigeria, +2348036779086,E-mail: madufortune@yahoo.com*