Asia-pacific Journal of Multimedia Services Convergent with Art, Humanities, and Sociology
Vol.6, No.11, November (2016), pp. 595-619
http://dx.doi.org/10.14257/AJMAHS.2016.11.59

Symbiotic Organisms Search Response to Distributed Database Queries

Atinderpal Singh[1], Krishan Kumar[2],Rajinder Singh Virk[3], Hye-jin Kim[4]

ABSTRACT

This paper has prime focus on the query optimization problem in distributed database environment. Query Processing is the major concept involved in distributed system. Data that is dispersed geographically at different locations is too gathered at resulting user site. There are many different execution actions that lead to same query result but differ in the cost incurred in query processing. We have to consider the execution strategy that yields optimal result. Communication cost is the major cost incurred when we talk about query processing in distributed system. Symbiotic Organisms Search Algorithm (Meta-heuristic) has been applied for query optimization in distributed environment. Result obtained using this meta-heuristic approach is compared with several optimization approaches. Results reveal the better performance of the algorithm for solving the hard problem of query processing in distributed database environment.

Keywords: Distributed Database, Query, Symbolic organism search, Execution strategy, Business data

I. INTRODUCTION

Distributed database system has emerged to provide optimal solutions to the information processing problems related to the organisations that are dispersed geographically. A distributed database is a collection of several, logically interrelated databases disseminated over several sites. Query optimizer is considered to be the important portion of distributed system. It considers the user query and make search for entire execution approach for the requisite query and result in optimal cost plan. The main component of a Database System is the data and is defined in literature as the collection of facts about something. This 'something' may be the business data in case of a business corporation, strategic data in case of a military's database and experimental data in a scientific experiment etc [1-20]. Data that constitutes the database

Received (September 21, 2016), Review Result (October 6, 2016)
Accepted (October 13, 2016), Published (November 30, 2016)

[1]Department of Computer Science and Engineering, Guru Nanak Dev University, Amritsar, Punjab.
email: saini_amrit@live.com
[2]Department of Computer Science and Engineering, Guru Nanak Dev University, Amritsar, Punjab.
email: Er.krishankumar4014@gmail.com
[3]Department of Computer Science and Engineering, Guru Nanak Dev University, Amritsar, Punjab.
email: tovirk@yahoo.com
[4](Corresponding Author) Business Administration Research Institute, Sungshin W. University,
2, Bomun-ro 34da-gil, Seongbuk-gu, Seoul, Korea,
email: hyejinaa@daum.net

ISSN: 2383-5281 AJMAHS
Copyright ⓒ 2016 HSST

has to be correlated and stored on different sites of a computer network to be a part of distributed database. Distributed Database is a very complex and costly technology, so mostly employed by big businesses, or governmental organizations [21-23].

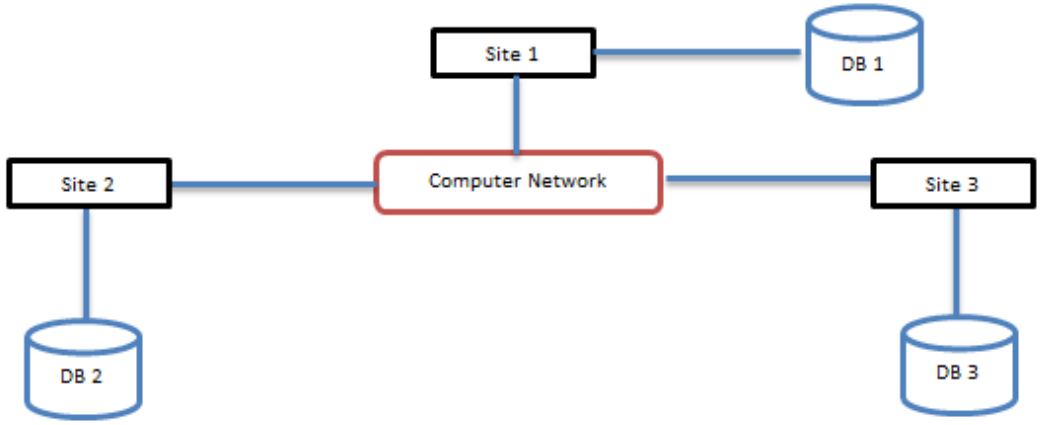


Fig 1.1 Distributed Database Systems

Optimization problem become more complex with the increase in complexity of distributed queries. The various operations on the relations over various sites include the major operations like selection, projection and join. These operations are to be dealt strictly to have optimal results. For small problems that involve small queries many algorithms are available to provide solutions but the queries that involve multiple sites, for them optimization problem become challenging and the various currently available algorithms may not cope with complexity.

The query [24, 25] often considered to be a logical data is referenced by a relation in relational distributed database system. The processing of query involves allocation of various sub-query operations to multiple network sites and result is obtained by joining various relations over various network sites or servers in distributed database system. The query optimization is based on cost model that reveals the various cost incurred in processing and answering a query. Number of algorithms have been practically utilised to solve above mentioned query processing concept. In order to validate the research carried out by us we have compared SOS for [23] distributed queries against Genetic Approach [26] and Exhaustive algorithm. This algorithm has positive aspects over other metaheuristic techniques because it does not require specific algorithm parameters.

II. STRATUMS OF DISTRIBUTED QUERY HANDLING

Copyright ⓒ 2016 HSST

The problem of distributed query processing has been labelled [27] in detail by decomposing it into various sub problems as shown below in the diagram.

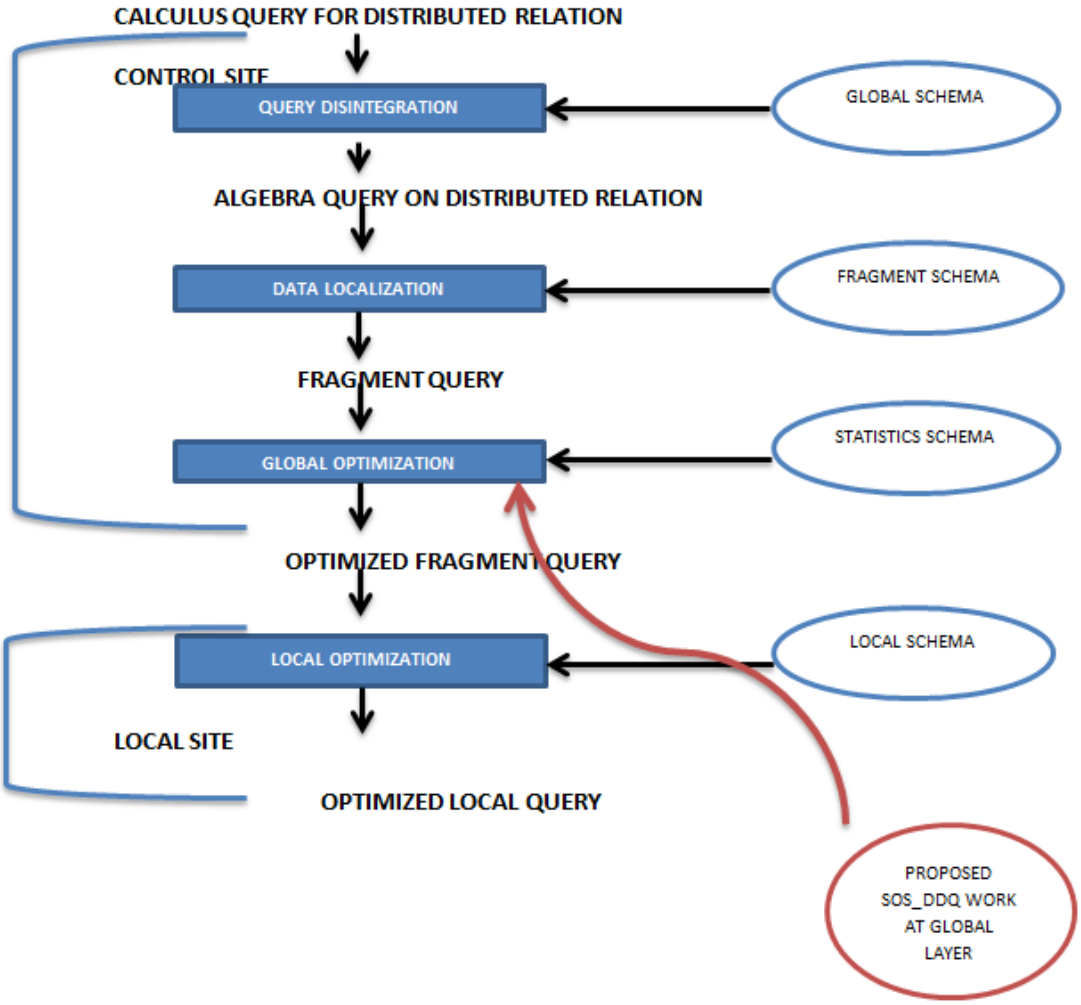


Fig 2.1 Generic Stratums of Query Handling.

III. LITERATURE SURVEY

Distributed database systems design and query optimization has been a dynamic area of research for database community, due to the complex and NP-Hard nature of the general problem [27].

Apers et al. Described how Processing schedules are generated for distributed query

Copyright ⓒ 2016 HSST

optimization. Virtual data sites are merged into actual network sites to find an optimal data allocation plan, by minimizing intermediate relation to relation transmission costs. Finally the query result is sent to originating site. The drawback of the approach was intractability in case of large size problems [28, 29].

Barker et al. Come up with an innovative idea of applying restricted growth to chromosome string. Group Oriented Restricted Growth String - GA to exclude redundant chromosomes from further GA process. They applied them for m-way partitioning of relations. They designed two new crossover operators and four new mutation operators. They further proved the effectiveness of using " Binary Merge Crossover Operator" and " Merge and Jump Mutation Operators" for large scale partitioning problems [30].

Carlo et al. has anticipated a technique for estimating the size of relational query outcomes. The method is constructed on the estimations of the attribute individual values. In particular, the proficiency of critical method to estimate selectivity factors of relational operations is considered. They also presented some experimental results on real databases which show the likely performance of analytic approach [31].

Chiu et al. Authors studied the use of semi-joins for chain queries for a distributed database query. A powerful and efficient dynamic programming algorithm was developed that translated a chain query into sequence of semi joins. It has a computing complexity of the order n3,herendenotesthenumberofrelationsreferencedbythequery.Finallytheyextendtheirworktooptimizealar gerclassofqueriescalledtree queries [32, 33].

Cosar et al. Proposed a new Genetic Algorithm for distributed database query optimization. They also investigated the effect of increasing number of nodes and relations on the performance of GA [34].

Cui et al. Gave a multi-objective genetic algorithm for distributed database management. They formulated a multi objective combinatorial optimization problem based on Dominance and Optimality. Multiple criteria are developed with a goal to provide trade-off optimal performances for web services [35].

Deshpande A et al. Studied the problem of query optimization in federated database

Copyright ⓒ 2016 HSST

systems and highlighted the need of decoupling various aspects of query processing. They implemented it on „Cohera" federated database system and demonstrated superiority of 2PO algorithm in case of pre known physical design of database [36].

Douglas W et al developed a methodology to assign relations and determine the join sites simultaneously. It decomposes queries into relational algebra operations and then makes site assignments based on a linear integer programming technique to minimize the intersystem communication. It describes procedures for balancing resource utilization across systems. Further it uses a heuristic technique to minimize average response time [37].

Faiza N et al. offered a statistical method for simulating method for calculating the cardinality of the resulting relation acquired by relational operator by using a sample based estimation that execute the query to be optimized on small sample of real database and then used the results to determine the cost. All the database statics maintained in the system are initialized when database is loaded. Chen's formulation is used to obtain the tuples in intermediate relation [38].

Gavish et al. extended the work of Chiu by proposing certain properties of a tree query to check the usefulness of a join sequence or a non optimality of one. Then they extend this work by imposing more restrictions on tree queries to convert them into star queries [39].

Ghaemi et al. in their paper on evolutionary query optimization for heterogeneous distributed database systems, discuss a multiagent based architecture and use of genetic algorithms. They demonstrate the superiority of the GA over Dynamic Programming methods in case of large scale problems [39].

Hevner et al. made the first successful attempt at using semi joins in a distributed query process. After performing the local processing part of simple queries, each relation contains one common joining attribute. Authors generalized the algorithm for equi-join queries. They introduced the concept of Selectivity for join operations, as number of domain values currently appearing in the joining columns divided by the total domain values. They assumed that selectivity of one joining domain does not affect selectivity of other joining domain. A heuristic algorithm with improved exhaustive search was proposed for general queries. This approach also suffered from scaling problems, in the way that if number of relation joins and number of

Copyright ⓒ 2016 HSST

sites involved move into double digit figure, algorithm computing time increases to exponential growths and quickly goes intractable [39].

Huang et al. Proposed a simple and comprehensive model for Fragment Allocation in Distributed Database Design that reflects transaction behaviour [40].

Jiunn W et al. in their work represented the genetic approach based on appropriate data structure to reduce the cost for distributed query processing. Simulation was performed for the distributed database environment and the experiment result revealed that genetic showed up as better way in computation effort and quality of solution subjected against other various approaches [41].

Kossmann D et al. present architecture for distributed query processing and some techniques to reduce communication costs, to exploit intra query parallelism [41].

Li et al. Presented a tree based GA with new coding method of genetic parameters with tree structure based on position and value. Improved Crossover and Mutation operators are devised to claim improved stochastic coding rules of genetic algorithms [42].

Li et al. proposed a design of a distributed query optimization algorithm. It is based on multirelation semi joins, processed at a buffer zone of distributed database, so as to reduce the communication time of intermediate results generated [43].

Lin Z et al. Proposed first a data allocation algorithm with respect to a simple strategy to process transactions, and then secondly gave a dynamic data allocation algorithm, guaranteeing to produce a locally optimal data allocation [44].

Martin L et al. demonstrate the cost effectiveness of four algorithms, Branch & Bound, Greedy, Local Search and Simulated Annealing for site selection during the optimization of compiled queries in a large replicated distributed database system. They conclude that enumerative algorithms are best suited for simple queries and recommend a local search algorithm for complex queries [45].

Pund et al. describe the basic concepts of query processing and query optimisation in the

Copyright ⓒ 2016 HSST

relational database domain. Further they differentiate three different types of query processing algorithms via Deterministic, Genetic, and Randomized Algorithm [46].

Rahmani et al. proposed an innovative model by clustering sites based on the cost of communication between sites, to allocate data on nodes of a Distributed Database. Authors claim significant reduction in the data redundancy in fragment allocation and network traffic [47].

Sakti et al. presented a semi join reducer cover set based technique for optimizing join queries in distributed databases. The technique converts semi join program into a partial order graph which allows concurrent processing of semi join programs [48].

Segev et al. proposed a mathematical model for a special set of queries and two ways join queries later implemented by Segev. They prove the problem to be NP Complete and propose a heuristic solution by partitioning data horizontally and no use of semi joins [49].

SOS algorithm has been implemented, validated and tested for number of unconstrained mathematical problems and engineering design problems. Hence we deployed this algorithm for distributed queries and it showed better results as compared to other optimizations approaches.

IV. SOS ALGORITHM

Symbiotic Organism Search is inspired by natural interaction plans of organisms that are incorporated by them for sustaining life in the ecosystem. It is a powerful meta-heuristic algorithm that can be easily applied to engineering problems. Though large number of optimization methods has been developed but still some fail to solve real world engineering problems [48, 49].

SOS illustrates the interactive behaviour among various organisms. According to nature's aspect and law it is hard to find the organisms living alone all way. They sustain in ecosystem while being dependent on each other for primary needs. This way of living and interaction among organisms is called symbiosis i.e. to live together. The relationships that are common among distinct species organisms are Mutualism, Commensalism and Parasitism.

Like other nature inspired algorithms SOS simulates the relationship between paired organisms of distinct species for a search that results in fittest organism. In order to get

Copyright ⓒ 2016 HSST

optimal Solution SOS has its inception with an initial population size as ecosystem. In the beginning phase organisms are generated randomly that represent their result to particular problem. Each organism has fitness value that reveals their chance to achieve desired goal. The new generation of organisms are obtained by initializing the biological interaction among organisms. The type of interaction among organism defines the main goal of each phase. Each organism interacts with other in all three phases.

SOS applies avaricious strategy at the end of each communicating phase to have finest organism in ecosystem. It considers two major Parameters.

- Population Size or Ecosystem Size.
- Maximum Generations or Iterations.

It provides solution to wide variety of problems and is more robust than other computing algorithms and it does not require parameter tuning, hence it avoids risk that may result in compromised performance.

The general outline of the algorithm is given below and the individual phase is explained later in the paper.

3.1. General Outline of Symbiotic Organisms Search Algorithm:

```
{
    Initial population (Ecosystem);
    While (The termination criteria is not met)
        {
        Mutualism Phase;
        Commensalism Phase;
        Parasitism Phase;
        }
    Output the best Organism;
}
```

Fig.1 General Outline of SOS Algorithm

3.2. Detailed representation of Various Phases:

A) Ecosystem Initialization

B) Evaluate the Best Organism (Z_{best})

C) Mutualism Phase

D) Commensalism Phase

Copyright ⓒ 2016 HSST

```
        {
            Number of Organisms (eco_size);
            Initial Ecosystem;
            Termination Criteria;
            Num_iter=0;
            Num_fit_eval=0;
            Max_iter;
            Max_fit_eval;
        }
{
    Select one organism arbitrarily, Z_i where Z_i ≠ Z_i;
    Determine Mutual relationship vector (MV) and benefit factor (B_Fac) ;
    MV=(Z_i + Z_j)/2;
    B_Fac1= any arbitrary number 1 or 2;
    B_Fac2= any arbitrary number 1 or 2;
    Modify Organism Z_i and Z_j based upon there mutual relationship;
    Z_i modified = Z_i + random(0,1) * (Z_best - MV *B_Fac1);
    Z_j modified = Z_j + random(0,1) * (Z_best - MV *B_Fac2);
    Calculate fitness value of modified organisms.
    Num_fit_eval=Num_fit_eval+2;

        If amended organisms fitter than previous
            {
              Accept the amended organisms;
            }
        Else
            {
              Reject the amended organisms and keep previous;
            }
}
        {
          Select one organism arbitrarily, Z_i where Z_i ≠ Z_i;
          Modify Organism Z_i with the help of Z_j;
          Z_i modified = Z_i + arbitrary (-1,1) * (Z_best - Z_j);
          Calculate fitness value of reformed organism.
              If reformed organisms fitter than previous
                  {
                    Accept the reformed organism (Z_i modified);
                  }
              Else
                  {
                    Reject the reformed organism (Z_i modified ) and keep previous
(Z_i) ;
                  }
        }
```

E) Parasitism Phase

IV. DATABASE DESIGN AND OBJECTIVE FORMULATION

4.1. Database Design

Copyright ⓒ 2016 HSST

```
{
  Select one organism arbitrarily, Z_i where Z_i ≠ Z_i;
  Create parasite (PV) from organism Z_i;
  Calculate fitness value of altered organism.
    If altered organisms fitter than previous
      {
        Accept the altered organism (Z_i modified);
      }
    Else
      {
        Reject the altered organism (Z_i modified) and keep previous (Z_i);
      }
}
```

The Design of distributed database is simulated considering a set 'S' of network sites, a set 'R' of relations(tables) stored at various sites and a Set 'Q' as set of transactions. A transaction query (q) for information retrieval, is broken into a set of sub queries on the 'R' set of relations [28, 41, 29].

4.2. Problem Definition

The above stated variables in database design formulate the problem in the following context. Input data file represents the data allocation matrix given by for base relation K stored at sites''. The objective [41, 29] function for the distributed database queries is given below. For the below objective function we have to find:

Given a set of Relations or fragments $R = \{r_1, r_2, \cdots, r_n\}$

A net of sites $S = \{s_1, s_2, \cdots, s_m\}$

A set of sub queries $Q = \{q_1, q_2, \cdots, q_q\}$

A) Input Variables

DAV_{rs}: Data Allocation Scheme that have matrix representation for I/O cost and communication cost.

IF_{rK}^{q} : Matrix that represent the intermediate fragments 'r' used by a sub query 'K' of main query 'q'.

B) Output to be Generated

SDD_{Ks}^{q} **:** An Operation Allocation Scheme Matrix which optimizes objective function.

4.3 Cost Model Formulation:

Copyright ⓒ 2016 HSST

A) ***Data Allocation Variables :***

$DAV_n = 1$ *(If Relation is available over site s).*

$DAV_n - 0$ *(If Relation is not available).*

B) ***Variables for selecting the site where the sub-query allocation will take place.***

SDD^q_{Ks} : *(Represents the sequence of sub query execution that takes place at distinct sites)*

$SDD^q_{Ks} = 1$ *(If sub-query K of main query Q is executed at site s).*

$SDD^q_{Ks} = 0$ *(Else).*

C) ***Variables that are used for join operation***

$SDD_{Kv[M]S} = 1$ *([M] = 1 for performing left previous operation of a Join)*

$SDD_{Kv[M]S} = 1$ *([M] = 2 for performing right previous operation of a Join)*

$SDD_{Kv[M]S} = 0$ *(Else).*

D) ***IF^q_{rK} tells whether the sub query 'K' of query 'q' make reference to intermediate relation or not.***

$IF^q_{rK} = 1$ *(If the intermediate fragment is used by sub query K of the main query q).*

$IF^q_{rK} = 0$ *(Else).*

E) ***Variables For the use of intermediate fragments for Join Operation***

$IF^q_{rKv[M]} = 1$ *(For left previous operation of join where [M] =1 for query K).*

$IF^q_{rKv[M]} = 1$ *(For right previous operation of join where [M] =2 for query K).*

$IF^q_{rKv[M]} = 0$ (Else).

Distribution of relations to sites represented by cost function involving query processing cost and storage cost [25].

$$Total\ Cost = \sum_{\forall qi \in Q} QPi + \sum_{\forall s \in S} \sum_{\forall rj \in R} ST_{jp}$$

Here QP_i is the cost for processing query q_i for any application and ST_{jp} is the cost for storing fragments F_j at site S_p. The total cost for the query is sum of local processing cost and Transmission cost i.e. according to OZSU [29] model. Further we have modified it considering only the retrieval transaction according to our design. Query processing cost is given below.

$$QP_i = LP\_COST + COMM\_COST$$

Copyright ⓒ 2016 HSST

LP_COST represents the local processing cost and the COMM_COST represents the communication cost involved in distributed query processing. Also processing cost includes the access cost and does not include the integrity cost and concurrent update control cost for model with retrieval transactions. The access cost portion of LP_COST is represented as follows.

Fragments access cost for query (FACi)=(uij*URij+rij*RRij)*Xjp*LP_COST .

It represents the total number of access made for all fragments and X_{jp} consider cost for sites where fragments are stored in real.

4.3.1 Local Processing Cost for various Operations in Distributed Query Processing:

A) Processing Cost for Selection and Projection Operations

It involves the input/output cost from secondary memory to primary and CPU processing cost for performing selection and projection at particular site.

$$LP\_COST_K^q = \sum_s SDD_{KS}^q(IO, \sum_r IF_{rK}^q MB_{rK}^q + CPUCOSTs \sum_r IF_{rK}^q MB_{rK}^q)$$

4.3(a)

Here MB_{rK}^q : Represents the memory blocks of relation '**r**' accessed by sub-query'**K**' of main query '**q**'.

IO$_s$: Represents the Input Output Cost Coefficient of a particular site s of S.

CPUCOST$_s$: Represents the CPU Cost coefficient of site s i.e. is a subset of S.

B) Processing Cost for Join Operation:

The general model of OZSU does not incorporate local processing cost for join operation but we have considered it in our design.

Local processing costs for a join may be given as

$$LP\_COST_K^q = \sum_s SDD_{KS}^q(IO, \sum_m \sum_r m_v IF_{rKv[M]}^q MB_{rKv[M]}^q)$$

4.3(b)

+

$$\sum_s SDD_{KS}^q(IO, \prod_r IF_{rK}^q MB_{rK}^q + CPUCOST, \prod_r IF_{rK}^q MB_{rK}^q)$$

4.3(c)

Here m_v is the selectivity factor and it is considered as the ratio of the different values of a field to the domain of that field($0 <= m_v <= 1$)

$MB_{rKv[M]}^q$: Represents the size of an intermediate relation.

Copyright ⓒ 2016 HSST

V[m] : Represents the left and right join operation for M=1 and M=2 respectively
Equation 4.3(b) Represents the I/O costs in storing the intermediate results of previous operations to the site where current join operation is performed.
Equation 4.3(c) represents the CPU & I/O costs for evaluating join operation at current site.

4.4 Communication Cost involved in Distributed Query Processing:

Communication costs are incorporated in case of join operations and final resultant operation only. Since the selections & projections of retrievals on relations are to be performed only at sites which hold a copy of those base relations. Join can be performed at any of all possible sites. Communication Cost represented below:

$$COMM\_COST_K^q : \sum_m \sum_s \sum_u SDD_{Kv[M]S}^q * SDD_{yu}^q COM_{su}\left(\sum_r IF_{rKv[M]}^q MB_{rKv[M]}^q \right)$$

4.4(a)

COM_{su} : is considered as the communication cost coefficient between site s and u from the input data matrix.

COM_{su} = 0 (for s = u) this is the case in which the previous operation and current join operation is performed on same site.

If the final operation in query processing is not performed at the query originating or destination site then a Communication Cost variable is added separately for costs incorporated in sending the final query answer to the query originating/destination site.

4.5 Objective Formulation

The best way to calculate cost of a query execution plan is to minimize the 'Total Cost of Query', which is represented in terms of time units and refers to use of resources such as CPU Cycles, Disk I/O & Communication Channels by a candidate allocation plan. It can be represented by the sum of mathematical equations as formulated & illustrated above.

Thus our Objective Function is to: Minimize the sum [4.3(a) + 4.3(b) + 4.3(c) + 4.4(a)]

Copyright ⓒ 2016 HSST

$$\left(\sum_s SDD^q_{KS}(IO_s \sum_r IF^q_{rK} MB^q_{rK} + CPUCOST_s \sum_r IF^q_{rK} MB^q_{rK})\right) +$$
$$\left(\sum_s SDD^q_{KS}(IO_s \sum_m \sum_r m_v IF^q_{rKv[M]} MB^q_{rKv[M]}\right) +$$
$$\left(\sum_s SDD^q_{KS}(IO_s \prod_r IF^q_{rK} MB^q_{rK} + CPUCOST_s \prod_r IF^q_{rK} MB^q_{rK})\right) +$$
$$\left(\sum_m \sum_s \sum_u SDD^q_{Kv[M]S} * SDD^q_{yu} COM_{su}\left(\sum_r IF^q_{rKv[M]} MB^q_{rKv[M]}\right)\right)$$

4.6. SOS for Implementing Distributed Database Queries

This section gives the various steps involved in implementing SOS for distributed queries [2].

Step 1: Data Distribution Scheme along with fragmentation schemes is given as input to the SOS_DDQ.
Step 2: Ecosystem Initialization.
Step 3: Consider the best (organism) solution, Z_{best}.
Step 4: Mutualism Phase.
Step 5: Commensalism Phase.
Step 6: Parasitism Phase.
Step 7: Move to step 3 if the current Z_i is not the last organism in pool of ecosystem, otherwise move to next step.
Step 8: Stop if the maximum number of generations is reached and print the optimal solution, else move to step 3 and repeat the whole process.

V. EXPERIMENTAL SETUP AND DATABASE STATISTICS

A customized Simulator developed in MATLAB is considered to analyse the performance of different techniques in optimizing distributed queries in distributed environment. Simulator SOS_DDQ takes the required stat as input and produces the desired output. Various parameters (Input/output Cost, CPU cost, Communication Cost) of distributed query are considered for the analysis. The input to the simulator is fed in the form of text file. Intermediate Fragment Size and Block Size of a relation plays inevitable role in performing analysis of distributed queries.

Different benchmark queries are considered for analysis of the performance of SOS_DDQ, GA_SA and Exhaustive approach. The Experiments are performed on Intel(R) core (TM) 2 Duo T6600 @ 2.20 GHz Machine having 3 GB Random Access Memory.

5.1 DATABASE STATISTICS

Copyright ⓒ 2016 HSST

Base Relations are replicated over two different sites for distributed processing. Block structure of concerned relations are utilized for analysis of queries. The below table reveals the availability of relations over several sites. R_i denotes the relations where i= 1 to 7 and S_i denotes the several sites in range of 1 to 10.

Data Allocation (DAV_{rs})	DIFFERENT SITES									
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
R1	1	1	0	0	0	0	0	0	0	0
R2	0	0	0	0	1	1	0	0	0	0
R3	0	0	0	0	0	0	0	1	1	0
R4	0	0	0	0	0	0	0	0	1	1
R5	1	1	0	0	0	0	0	0	0	0
R6	0	0	1	1	0	0	0	0	0	0
R7	0	0	0	0	0	1	1	0	0	0

Fig 5.1 Data Allocation (DAVRS)overSeveralSites

COST COFFECIENTS	DIFFERENT SITES									
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
INPUT/OUTPUT	1	1.1	1.2	1	1.1	1	1.2	1	1.1	1
CPU COST	1.1	1	1	1.1	1	1.2	1	1	1.2	1

Fig 5.2 I/O and CPU Cost Coefficients

COMMUNICATION COST MATRIX	DIFFERENT SITES									
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
S1	0	10	12	13	14	11	12	13	14	11
S2	10	0	11	12	13	14	11	12	13	14
S3	12	11	0	11	12	13	14	11	12	13
S4	13	12	11	0	11	12	13	14	11	12
S5	14	13	12	11	0	11	12	13	14	11
S6	11	14	13	12	11	0	11	12	13	14
S7	12	11	14	13	12	11	0	11	12	13
S8	13	12	11	14	13	12	11	0	11	12
S9	14	13	12	11	14	13	12	11	0	11
S10	11	14	13	12	11	14	13	12	11	0

Fig 5.3 Communication Cost Matrix.

5.2 INPUT DATA FILE REPRESENTATION

5.2.1 JOIN OPERATION FILE

Copyright ⓒ 2016 HSST

Consider join operation file for query in Case 3 described later.

Line 1: 8 10 12 15 17 19 Represents the Left Previous Operations.
Line 2: 9 11 13 16 14 18 Represents the Right Previous Operations.
Line 3: 15 16 17 19 18 20 Represents the Join Operations.
Line 4: 15 17 19 22 24 25 Represents the Left Previous Fragments.
Line 5: 16 18 20 23 21 26 Represents the Right Previous Fragments

5.2.2 FILE FOR COST PARAMETERS, DATA ALLOCATION AND INTERMEDIATE FRAGMENT SIZE.

File for Case 3. (Example)

Line 1: 4 Represents the resultant site.
26 Represents the Fragments.
20 Represents the Operations.
7 Represents the Base Relations.
10 Represents the number of sites.
6 Represents the number of joins.
7,7 Represents the number of selections and projections respectively.

Line 2: 1 1.1 1.2 1 1.1 1 1.2 1 1.1 1 Represents I/O Cost.
Line 3: 1.1 1 1 1.1 1 1.2 1 1 1.2 1 Represents CPU Cost.
Line 4 – 13 Represents Communication Cost.

0 10 12 13 14 11 12 13 14 11
10 0 11 12 13 14 11 12 13 14
12 11 0 11 12 13 14 11 12 13
13 11 12 0 11 12 13 14 11 12
14 13 12 11 0 11 12 13 14 11
11 14 13 12 11 0 11 12 13 14
12 11 14 13 12 11 0 11 12 13
13 12 11 14 13 12 11 0 11 12
14 13 12 11 14 13 12 11 0 11
11 14 13 12 11 14 13 12 11 0

Line 14 – 20 Represents the Data Allocation over several sites.

1 1 0 0 0 0 0 0 0 0
0 0 0 0 1 1 0 0 0 0
0 0 0 0 0 0 0 1 1 0
0 0 0 0 0 0 0 0 1 1
1 1 0 0 0 0 0 0 0 0
0 0 1 1 0 0 0 0 0 0
0 0 0 0 0 1 1 0 0 0

Copyright ⓒ 2016 HSST

Line 21: Represents the Fragment Size 2000 2000 2000 2000 2000 2000 2000 1400 1400 1400 1400 1400 1400 1400 1400 1260 1260 1260 1260 1260 1260 1260 600 600 600 600 600 600

5.3 EXPERIMENTAL QUERIES

Case 1: In this case geographically dispersed environment is considered where five relations are distributed over 5 sites. The distributed query given below performs 5 selections, 5 projections and 4 join operations.

$(\pi_{p1}(\sigma_{S1})R_1)$:X: $(\pi_{p2}(\sigma_{S2})R_2)$:X: $(\pi_{p3}(\sigma_{S3})R_3)$:X: $(\pi_{p4}(\sigma_{S4})R_4)$:X: $(\pi_{p5}(\sigma_{S5})R_5)$.

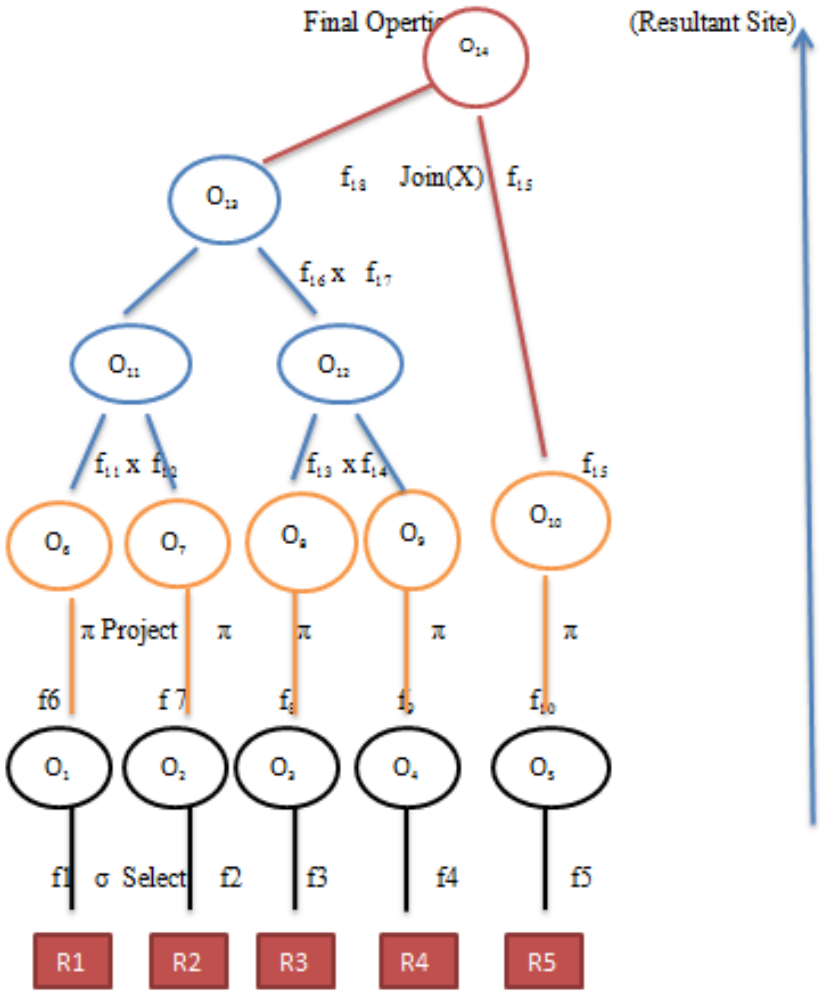


Fig 5.3.1 Distributed Query with Five Base relations

Copyright ⓒ 2016 HSST

Number of Intermediate Fragments: 18

Number of Operations: 14

Case 2: In this case geographically dispersed environment is considered where six relations are distributed over 10 sites. The distributed query given below performs 6 selections, 6 projections and 5 join operations.

$(\pi_{p1}(\sigma_{S1})R_1)$:X: $(\pi_{p2}(\sigma_{S2})R_2)$:X: $(\pi_{p3}(\sigma_{S3})R_3)$:X: $(\pi_{p4}(\sigma_{S4})R_4)$:X: $(\pi_{p5}(\sigma_{S5})R_5)$:X: $(\pi_{p6}(\sigma_{S6})R_6)$.

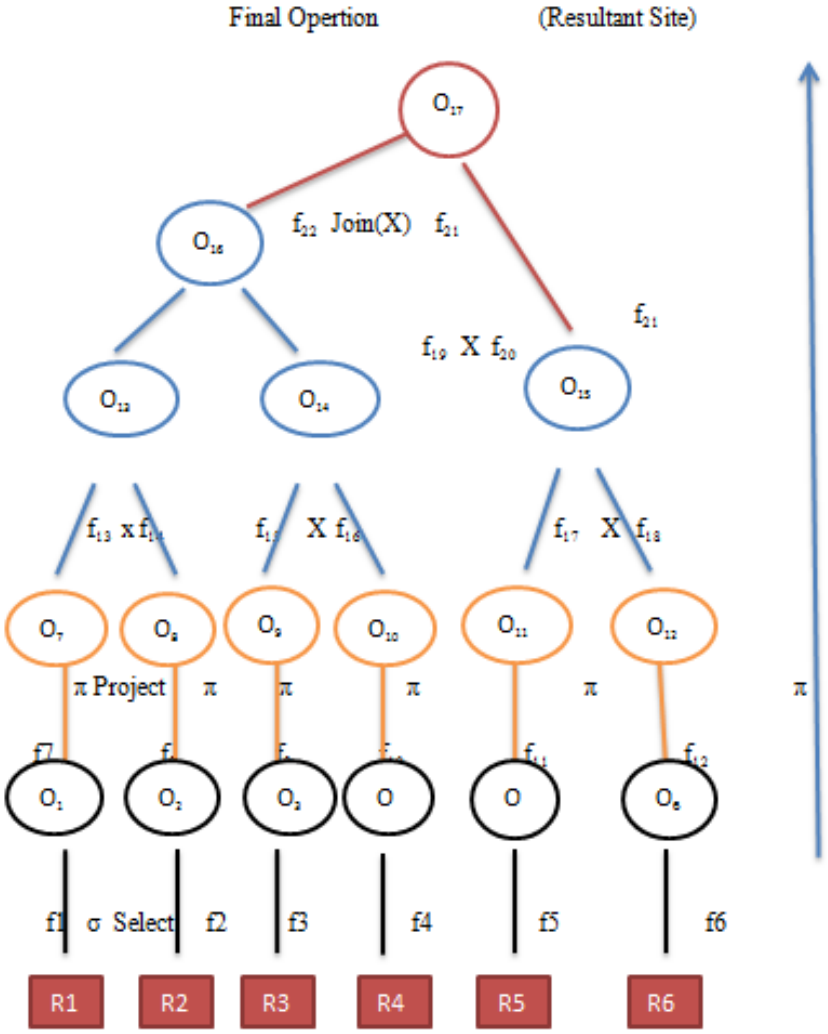


Fig 5.3.2 Distributed Query with Six Base relations

Copyright ⓒ 2016 HSST

Number of Intermediate Fragments: 22

Number of Operations: 17

Case 3: In this case geographically dispersed environment is considered where seven relations are distributed over 10 sites. The distributed query given below performs 7 selections, 7 projections and 6 join operations.

$(\pi_{p1}(\sigma_{S1})R_1)$:X: $(\pi_{p2}(\sigma_{S2})R_2)$:X: $(\pi_{p3}(\sigma_{S3})R_3)$:X: $(\pi_{p4}(\sigma_{S4})R_4)$:X: $(\pi_{p5}(\sigma_{S5})R_5)$:X: $(\pi_{p6}(\sigma_{S6})R_6)$:X: $(\pi_{p7}(\sigma_{S7})R_7)$

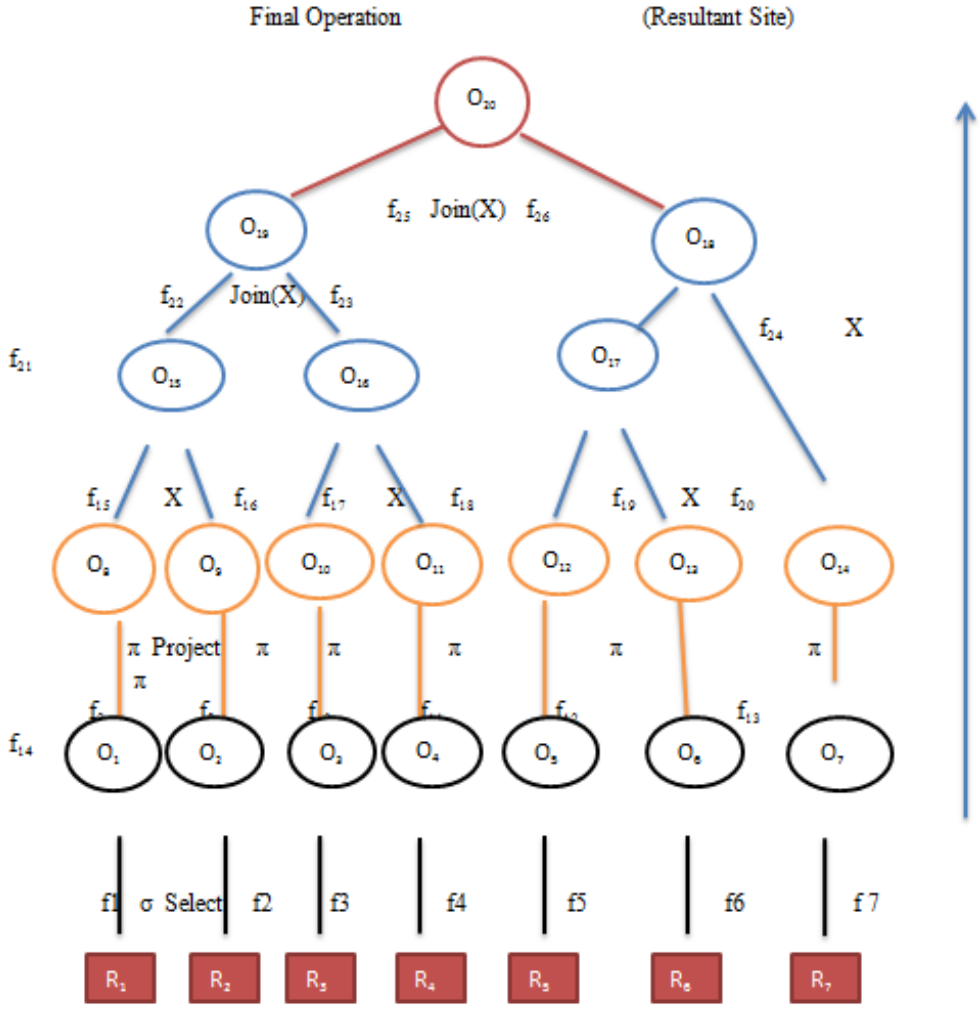


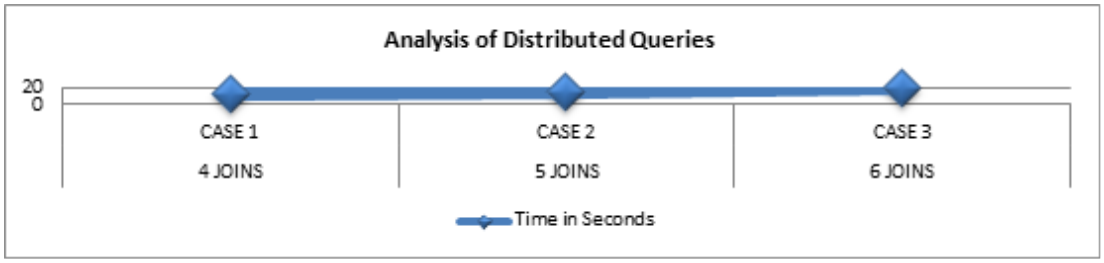
Fig 5.3.3 Distributed Query with Seven Base relations

Number of Intermediate Fragments: 26

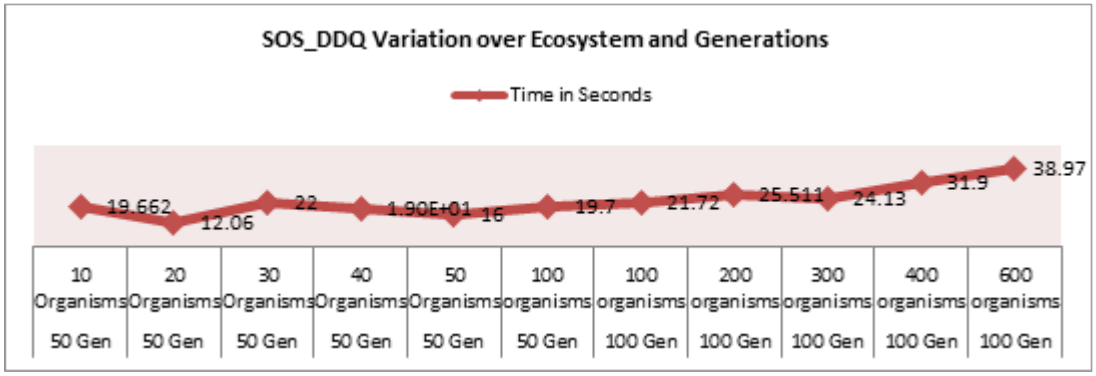
Number of Operations: 20

Copyright ⓒ 2016 HSST

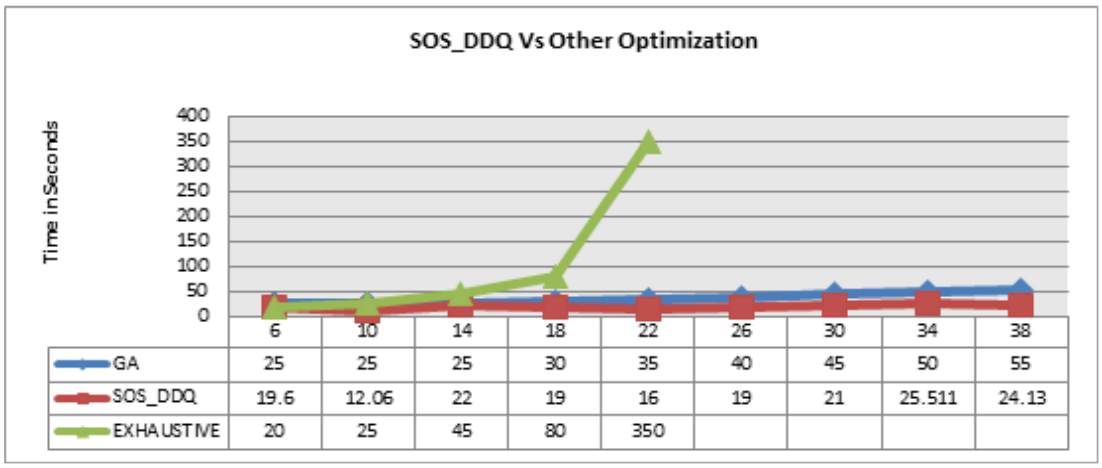
5.4 GRAPHICAL RESULT REPRESENTATION



5.4.1 SOS_DDQ Analysis of Different Queries.



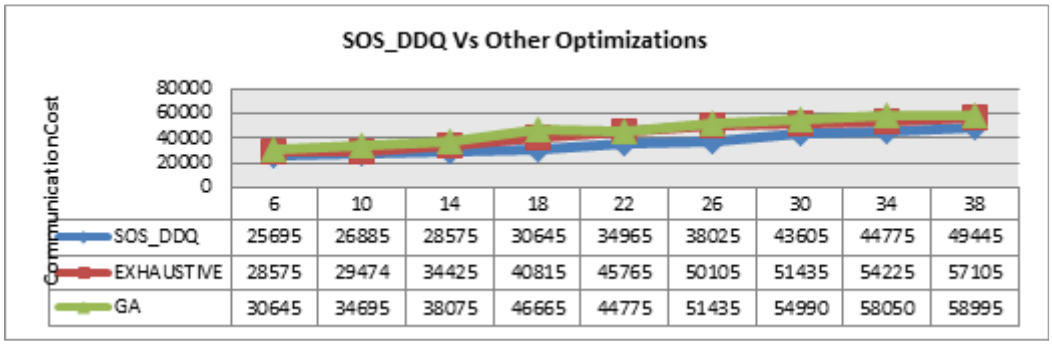
5.4.2 SOS_DDQ performance over different organisms and generations.



5.4.3 SOS_DDQ Time performance against other Optimization

.

Copyright ⓒ 2016 HSST



	6	10	14	18	22	26	30	34	38
SOS_DDQ	25695	26885	28575	30645	34965	38025	43605	44775	49445
EXHAUSTIVE	28575	29474	34425	40815	45765	50105	51435	54225	57105
GA	30645	34695	38075	46665	44775	51435	54990	58050	58995

5.4.4 SOS_DDQ Communication Cost against other Optimization.

VI. CONCLUSION AND FUTURE SCOPE

This study and the work completed by us compared the symbiotic organism's algorithm (SOS) performance against genetic algorithm and exhaustive approach for distributed queries. Various parameters for distributed environment have been evaluated and it showed up the fine results of the above approach against other optimization techniques. SOS_DDQ (Symbiotic Organisms Search for Distributed Database Queries) simulation was performed in MATLAB.

Earlier SOS has been validated for many benchmark functions and engineering design mathematical problems. Hence we deployed it to distributed environment by incorporating its function in relevance to query optimization. The analysis of our study reveals the positive and negative aspects of different algorithms. SOS performed consistently better than other algorithms when compared over various parameters. The result conveys that Exhaustive approach provides fine and optimal results for adequate queries but it can't be applied to large distributed relations since it may take hours or many days to provide reasonable solution. Genetic approach provides results in reduced time but it may not always provide optimum query execution.

Symbiotic Search algorithm performance surpasses others in distributed optimization. Analysis was carried out over 100 generations (SOS_DDQ) and ecosystem range from 10 to 600. Each organism has a fitness value associated which represents the total cost involved in in distributed query answering. Communication cost is the major cost involved in distributed environment. Therefore we have plotted the communication cost against other approaches in results section. Results evaluated in our work (SOS_DDQ) demonstrated that this algorithm was able to achieve better results with fewer evaluation functions than algorithms tested in previous

Copyright ⓒ 2016 HSST

research. Concurrency control and Security concepts can be incorporated in future work. Some heuristic can be applied in interactive phases of algorithm to further optimize the complex update transactions.

REFERENCES

[1] Hyun-Jung Lee, Seung-Eun Jeong, Jae-In Shin, Kab-Seung Kou, "Common Criteria Requirement of Data Leakage Protection System", Journal of Security Engineering, Vol.11, No.2 (2014), pp.65-78, http://dx.doi.org/10.14257/jse.2014.02.06

[2] Nae-Hyun Kim, "Residential Humidifying Element Made of Cellulose and PET Composite", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 4 (2016), pp.13-18

[3] Dae-Young Ko, Sung-June Baek, Aaron Park, Jin-Bong Kim, Yong-Su Nah, “Noisy OTDR Data Event Detection Analysis for the Real Time Optical Fiber Link Monitoring”, Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 4 (2016), pp.122-128

[4] Hyun-Joo Kim, “Design and Implementation of an Efficient Web Services Data Processing Using Hadoop-Based Big Data Processing Technique”, Journal of The Korea Academia-Industrial cooperation Society, Vol. 16, No. 1 (2015) pp.726-734

[5] Donghyun Kim, Minho Kim, "Study of Overlap Block-Layer Compositing Technology in Digital Watermarking and XML Data Encryption", Journal of Security Engineering, Vol.11, No.3 (2014), pp.207-220, http://dx.doi.org/10.14257/jse.2014.06.02

[6] Sunyong Kim, "Optimum Service Life Management Based on Probabilistic Life-Cycle Cost-Benefit Analysis", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 4 (2016), pp.19-25

[7] Jong-Hee Lee, Junho Bang, Hyun-Jun Chun, Beom-Geun Seo, In-Ho Ryu, "ICT based Wireless Power Transmission System Development", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 5 (2016), pp.67-73

[8] Suwook Ha, Seungyun Lee, Kangchan Lee, "Development of Reference Architecture Based on Big Data Ecosystem", Journal of Security Engineering, Vol.11, No.6 (2014), pp.511-522, http://dx.doi.org/10.14257/jse.2014.12.06

[9] Seung-il Choi, “Studies on Representative Body Sizes and 3D Body Scan Data of Korean Adolescents”, Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 2 (2016), pp.227-232

[10] Eunju Kim, Jong-Woong Park, Sung-Han Sim, "Development of Wireless Smart Sensing Framework for Structural Health Monitoring of High-speed Railway Bridges", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 5 (2016), pp.1-9

[11] Mi-Rye Kim, In-Ho Cho, "Design of Congestion Standardization System Based on IoT", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 5 (2016), pp.74-79

[12] Sangho Lee, Haengrae Cho, Aekyung Moon, "Data Suppression Scheme for Corona based Wireless Sensor Networks", Journal of Security Engineering, Vol.11, No.4 (2014), pp.339-354, http://dx.doi.org/10.14257/jse.2014.08.06

Copyright ⓒ 2016 HSST

[13] Yong-Jeon Lee, Seong-Rak Rim, "A scheme of Docker-based Version Control for Open Source Project", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 2 (2016), pp.8-14

[14] Myung-Kyu Park, Si-Woo Park, "A study on the properties of the carbon long-fiber-reinforced thermoplastic composite material using LFT-D method", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 5 (2016), pp.80-85

[15] Seung-Ho Kim, Jae-Beom Park, Dong-Hyun Tae, Seung-Jong Kim, Joong-Ho Song, Dae-Seok Rho, "A Study on the Algorithm for Single Phase Control of IGBT PWM Rectifier", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 4 (2016), pp.26-33

[16] Phan Woo Park, Woo Yeol Kim, "A Study on Government 3.0 based Method for Implementing Open API Service of Education Data", Journal of Security Engineering, Vol.12, No.1 (2015), pp.97-108, http://dx.doi.org/10.14257/jse.2015.02.09

[17] Jeong-Min Park, Hye-Ryeon Yong, Hyun-Seok Hwang, "The Influence of Cross Promotion in Mobile games on Consumer's Attitude toward Game adoption", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 2 (2016), pp.112-122

[18] Young-Choon Kim, Young-Man Kim, Sung-Gil Kim, Hong-Bae Kim, Moon-Taek Cho, "Development of the Mechenical System and Vision Algorithm for the External Appearance Test Using Vision Image Processing", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 2 (2016), pp.203-208

[19] Geun-Hyung Kim, Young-Kuk Kim, Seung Hwan Park, "A Framework for Quality Improvement in Weapon System using Post-Logistics Support Data", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 5 (2016), pp.680-687

[20] Seung Ho Baek, Min Ji Lee, Kyoungsoon Shin, "Short-term changes of phytoplankton communities after nutrient addition and establishment of stable mass culture condition to prepare the type approval test of USCG Phase-II in mesocosm enclosure", Journal of The Korea Academia-Industrial cooperation Society, Vol. 17, No. 4 (2016), pp.34-42

[21] Ceri, Pelagatti, "Distributed Databases Principles & Systems" McGraw Hill International Editions, 1985.

[22] Kossmann D., "The State of Art in Distributed Query Optimization", ACM Computing Surveys, 32(4):422-469.

[23] Min Y., Daddy P., "Symbiotic Organisms Search: A new Metaheuristic Optimization Algorithm", Computer and Structures 139, 98-112, Elsevier, 2014.

[24] Ghaemi R.,Amin M., "Evolutionary Query optimization for Heterogeneous Distributed Database Systems", in Proceedings of World Academy of Science, Engineering and Technology, vol.33, sep-2008.

[25] Hevnern R., "Query Processing in Distributed Databases", Ph.D. Dissertation, Prude Univ, 1979.

[26] Goldberg D., "Genetic Algorithm in Search, Optimization and Learning", Pearson Education 2nd Edition.

[27] Tamer O., Patric V., "Principles of Distributed Database Systems", Springer, 2010

[28] Apers P., "Data Allocation in Distributed Databases", 13(3):263-304, ACM TODS, 1988.

[29] Atinderpal S, Manreet S, Rajinder S, "Symbiotic Organisms Search Framework for Distributed Database

Copyright ⓒ 2016 HSST

Optimization" in International Journal of Engineering technology and Computer Research Volume 3, Issue 3 page no: 188-193 ISSN 2348-2117,2015.

[30] Barker K., Jun D., "Genetic Algorithm based approach to database vertical partition", J Intel Inf System (26):167-183, Springer, 2006.

[31] Carlo D, Ezio L. Filippo T., "Analytic-based Estimation of Query Result Sizes", 2005.

[32] Chiu D., "Optimal query interpretation for distributed databases" Ph.D. Thesis, Harvard Univ", 1979.

[33] Chiu D., Ho Y., "A methodology for interpreting tree queries into optimal semi-join expressions", Proc. SIGMOD Conf. Jun, 1980.

[34] Cosar, Sevinc, "An Evolutionary Genetic Algorithm for optimization of Distributed Database Queries", The Computer Journal, Oxford University Press, British Computer Society-2011.

[35] Cui X., Lin C., "A multi-objective genetic algorithm for Distributed Database Management", In Proceedings of 5th world congress on Intelligent Control and Automation, Hangzhou China.pp.117-121,IEEE,2004.

[36] Deshpande A., Hellerstein J., "Decoupled Query Optimization for Federated Database Systems", A Project Report, University of California Berkeley, April 2001.

[37] Douglas W, Cornell, Philip Y., "On Optimal Site Assignment for Relations in the Distributed Database Environment", Transactions on Software Engineering, vol 15, no. -8, IEEE, 1989.

[38] Faiza N., Yahiya S., "Cardinality estimation of distributed join queries"IEEE, 2002.

[39] Gavish B., Segev A., "Query Optimization in Computer Systems. Management of Distributed Data Processing", North-Holland Publishing Company, pp.233-252, 1982.

[40] Huang., Chen., "Fragment Allocation in Distributed Database Design", Journal of Information Sciences and Engineering -2001.

[41] Jiunm W., Jorng H., Yiming H., Baw L., "A genetic Algorithm for set Set query Optimization in distributed System", IEEE,1996.

[42] Li H., Luo B., "A Tree based Genetic Algorithm for Distributed Database", Proceedings of International Conference, On Automation and Logistics, Qingdao China pp.2614-2618, IEEE, 2008.

[43] Li., Gao. Yao., " Study of Query of Distributed Database Based on Relation Semi Join", Proceedings of International Conference, On Computer Design and Applications, ICCDA,IEEE,2010.

[44] Lin O., Zhang., "Data Allocation with Minimum Overall Communication costs in Distributed Database Design", IEEE,1993.

[45] Martin L., Russel. "An Evaluation Of Site Selection Algorithm For Distributed",1990.

[46] Pund A., Jadhao S., Thakre P., "A Role of Query Optimization in Relational Databases", International Journal of Scientific & Engineering Research,(2):1-5,2011.

[47] Rahmani, Torkzaban, Haghighat., "A new method of Genetic Algorithm for Data Allocation in Distributed Database Systems", In proceedings of the IEEE-International Workshop on Education Technology and Computer Science-2009.

Copyright ⓒ 2016 HSST

[48] Sakti, Vineyard D., "Optimizing Join Queries in Distributed Databases" IEEE Transactions on Software Engineering,vol14,no.9,Sep,1988.

[49] Segev A., Gavish., "Optimization of Join Operations in Horizontally Partitioned Database Systems" ,ACM TODS, vol.11,no.1,pp.48-80,1986.

Copyright ⓒ 2016 HSST