International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 9 (2020), pp. 2461-2473
 https://dx.doi.org/10.37624/IJERT/13.9.2020.2461-2473

SBBO Based Replicated Data Allocation Approach for Distributed Database Design

Arjan Singh

Punjabi University, Patiala, Punjab, India

ORCID: 0000-0003-1877-9211

Abstract

In the current scenario, when the demand of cloud based services and IoT devices is increasing day by day, designing distributed databases have become more challenging task. The performance of any distributed database is heavily dependent on its proper design. Moreover, data allocation is an important part of distributed database design after data fragmentation. A new replicated data allocation approach has been proposed in this paper. Simplified biogeography-based optimization (SBBO) is used for developing the proposed approach. SBBO based approach helps in minimizing total processing cost of a query and also increasing the overall performance of the system. Results obtained from the SBBO based approach are evaluated against the performance of BBO and GA based approaches. The SBBO based approach provides quality solutions as compared to other two algorithms.

Keywords: Distributed Databases, Replicated Allocation, Data Allocation, Evolutionary Algorithms, Meta-heuristic Techniques

1. INTRODUCTION

Distributed database system is based on technology in which an integrated database is built on computer network(s) instead of using a single machine for the purpose of data distribution [13]. During the last three decades, distributed database technology has emerged as one of the most significant development in the field of database systems. In the current scenario, when the cloud based services and IoT devices are increasing day by day, the role of distributed database has become more important. Moreover, all the major database software vendors nowadays support distributed database. Therefore, designing distribution database is an important area of research.

Distributed database system is having following edge over conventional centralized database system [13,31]:

- *Local Autonomy*: Data is allocated to location closer to the users who use it most frequently. With this approach, the data will be in the control of the local users and this will give them local autonomy in terms of allowing them to manage, establish and implement local procedure with respect to the use of data.
- *Reduced Communication Overhead*: In distributed database environment, most of the data is available locally. This local availability of data results in decreasing data movement while query execution.
- *Improved Reliability and Availability*: Data in the distributed database approach can be replicated which means that the same data is available at more than one site of the communication network. In case of the failure of one particular node or the disruption in single particular link result into making one or many nodes unreachable but the entire system will not breakdown in this approach. However, the performance of the distributed database can degrade but gracefully due to such failures.
- *Improved Performance*: Local availability of data in distributed database reduces query response time as well as improves system throughput.
- *Expandability*: Increase of database size is much easier to handle in the distributed database environment than the centralized database environment. New autonomous nodes can be easily included in the network and this will not affect the working of existing nodes. This freedom gives the permission to an establishment to spread out comparatively easy.
- *Economical*: Technology advancements have considerably reduced development cost of computer system. It allows the organizations to use separate workstations or PCs for different branches or divisions of the organization. It is also cost effective to add new workstations or PCs to the existing system rather updating a mainframe computer.

Fragmentation and allocation of data are two fundamental design issues in the design of a distributed database [31]. Fragmentation helps in decomposing relation into fragments, where each fragment is considered as a single entity. Decomposition allows multiple queries to execute simultaneously. Fragments are the fundamental logical entities for allocation [13]. Fragmentation also helps in reduction of irrelevant data access and increases data localization [31]. Fragmentation of database has following three types [13,31,28]:

- *Horizontal Fragmentation*: Horizontal fragmentation is formulated by specifying a predicate which carries out restrictions on the tuples in the relation. Horizontal fragmentation separates a global relation into different fragments in a horizontal manner by clubbing rows to generate subsets of tuples.
- *Vertical Fragmentation*: Vertical fragmentation separates a global relation into different fragments in a vertical manner. Vertical fragmentation of a relation retains only few attributes of the relation satisfying a condition on attributes of the relation.
- *Mixed or Hybrid Fragmentation*: A scenario in which a vertical fragmentation might follow a horizontal fragmentation or this can happen other way round is called as hybrid or mixed fragmentation.

Allocation is the process to determine the optimal distribution of each fragment over the communication network [31]. There are two alternatives for allocation of fragments: Replicated/Redundant and Non-replicated / Non-redundant [13,31].

- *Replicated/Redundant*: In a replicated/redundant allocation, same copy of the fragments is owed by multiple sites. The fragments replication helps in improving the system reliability and efficacy of read only queries. But the consistency of the data has to be maintained by the system otherwise the execution of update queries might result into the inconsistency of data. Replication of database may be further categorizes into two types: fully replicated database and partially replicated database. In fully replicated database, the allocation of all the fragments is done at each site. On the other hand, in partially replicated database, the replicas of a single fragment are place at multiple sites.
- *Non-replicated/Non-redundant*: In a non-replicated/non-redundant allocation precisely single image of every fragment is allocated all over the network. All the access to a particular data are diverted to the site containing the data. Non-replicated allocation is less reliable and supports less parallelism as compare to replicated allocation.

It is clearly evident from the above discussion that distributed database design is a problem which needs to address the two key issues: fragmentation of the global database and allocation of those fragments in replicated or non-replicated way. In this paper, allocation of the fragments or data will be investigated for partially replicated point of view assuming that database is already fragmented.

2. RELATED WORK

Allocation problem is first studied from file allocation point of view. Chu [16] was first to investigate the file allocation problem. Chu [16] has given a model based on fixed number of replicas which optimizes total cost of storage and transmission. Casey [10] has extended the work of Chu [16] and proposed a new model by giving more stress on the process of updation and retrieval of the data fragments and relaxing the fixed number of replicas. Eswaran [20] proves NP-Completeness of file allocation problem and suggested heuristics rather than deterministic techniques to solve the file allocation problem. Chen and Akoka [14] developed an optimization model using bounded branch and bound integer programming technique for a distributed information system. Ceri et al. [12] developed optimization model using decomposition heuristics for data allocation in a linear 0-1 programming problem form. Horizontal fragmentation is used as an input for the allocation model. Ceri et al. [12] proposed a greedy approach for handling replicated allocation of data after optimal solution has been found for non-replicated allocation. Wong and Katz [39] suggested local sufficiency as a measure of parallelism in a distributed database. They have suggested three different approaches for replication of fragments, each having different blend of cost and benefits.

Apers [6] proved that the data allocation in distributed database environment is NP-hard problem and different than the problem of allocation of files. Heuristics and optimization algorithms are proposed for non-replicated data allocation. Optimization algorithms outperformed heuristic algorithm as concluded from the results. Chiu and Raghavenda [15] presented an allocation model for enhancing system reliability with use of triple module redundancy (TMR) scheme. Blankinship et al. [9] proposed an iterative heuristic technique for allocation of data as well as for optimization of query. Ram and Marsten [33] have given a model of allocation of data in distributed databases by including “WRITE LOCKS ALL-READ LOCKS ONE” concurrency control method. Lin et al [25] proposed heuristic approach for minimizing total cost of communication. Proposed approach considers physical

network and transaction processing strategies for the allocation of data. Corcoran and Hale [18] presented a genetic algorithm to allocate fragments. Objective function which has be minimized concentrates only on total transmission cost and transmission cost considers only the retrieval frequency of the different sites to all the fragments and moreover replication of fragments is not taken into consideration.

Lin and Orlowska [26] suggested that once the data allocation problem is converted into an integer linear program than the probability of finding a polynomial time bounded solution is quite high. Daudpota [19] proposed a heuristic algorithm named TGTF for the transformation of global relation into fragments and their allocation in replicated manner. Tamhankar and Ram [37] proposed concurrency control mechanism based methodology to fragment and allocate replicated data. Barney and Low [8] extended the work of Tamhankar and Ram [37] for object-oriented databases by including the process workload estimation. Barker and Bhar [7] proposed a non-redundant cost model based on graphical optimization for allocation. Huang and Chen [22] developed a model using the behavior of transactions in distributed databases. Two heuristic algorithms are proposed to minimize total communication cost.

Ahmad et al. [5] proposed three different evolutionary algorithms and search based heuristic for non-redundant data allocation. Genetic algorithm has outperformed other three approaches in terms of solution quality as well as efficiency point of view. Karlapalem et al. [23] also empirically evaluated the performance of these four algorithms for allocation of data in distributed multimedia databases. Loukopoulos and Ahmad [27] proposed greedy heuristic based approach and GA for data allocation. GA based method is giving better performance than greedy heuristic based approach. Hababeh et al. [21] proposed clustering based approach to allocate replicated data for high performance computing. Adl and Rankoohi [2] proposed three different versions of ACO based heuristic methods for data allocation. Mamaghani et al. [29] proposed a hybrid evolutionary approach for data allocation. Hybrid approach is combination of object migration learning automata and genetic algorithm. Tosun et al. [38] proposed genetic algorithm, simulation annealing algorithm and fast ant colony algorithm for non-replicated fragment allocation. Singh et al. [35] proposed biogeography-based optimization for replicated data allocation in Distributed Databases. Amer et al. [2] proposed heuristic approach to fragment and allocate data. Rahimi et al. [34] proposed Bond Energy Algorithm (BEA) based method for fragmenting and allocating data simultaneously. Castro-Medina [11] proposed a heuristic based algorithm to fragment and allocate replicated data in cloud environment. Abdalla and Artoli [1] also proposed cluster based heuristic technique to fragment and allocate data to minimize the total transmission cost. Choudhary and Jha [17] developed a genetic algorithm based virtual scheduling model for job allocation in real time distributed database system. Amer [4] developed a heuristic based on K-means clustering to vertically fragmentation and allocate data in the relational database context.

From above discussion following observations have been made:

- Data allocation problem in distributed database design is NP-hard
- Most of the researchers have proposed either heuristic algorithms or optimization algorithms to solve it

Moreover, heuristic approaches do not always provide optimal solutions. Therefore, meta-heuristic algorithmic approach is the good option to discover out optimal or near optimal solutions. Ahmad et al. [5], Corcoran and Hale [18], Karlapalem et al. [23], Loukopoulos and Ahmad [27], Mamaghani et al. [29], March and Rho [30], Rahmani et al. [32], Adl and Rankoohi [2], Singh et al. [35], Rahimi et al. [34], Choudhary and Jha [17], and Tosun et al. [38] have used different meta-heuristic algorithms in different ways to solve data allocation problem in distributed databases. Corcoran and Hale [18] and Ahmad et al. [5] have suggested genetic algorithm as an attractive way out for efficient and quality solution for data allocation.

Furthermore, according to No Free Lunch (NFL) Theorem [40], there is no single meta-heuristic technique that can be used to provide solutions for all kind of optimization problems. Each meta-heuristic technique has its own advantages and disadvantages. All the meta-heuristics techniques provide near to optimal solution instead of exact solution to a given optimization problems. All above mentioned factors have motivated us to explore new approaches. Therefore, a new meta-heuristic optimization technique named as Simplified Biogeography-Based Optimization (SBBO) is explored to find optimal data allocation in distributed database environment. SBBO has capability to produce quality solution to optimization problems than that of other meta-heuristic technique as reported in the literature.

3. COST MODEL FOR DATA ALLOCATION

The inputs to the data allocation problem are [24]:

- $F = \{F_1, F_2, \ldots\ldots, F_m\}$ is the set of all fragments and size is represented by $Size(F_k)$, where $1 \leq k \leq m$.
- $N = \{N_1, N_2, \ldots\ldots\ldots, N_n\}$ be the set of n sites, which are part of distributed database system and $CC = [CC_{ij}]$ represent the cost of data movement from site N_i to site N_j

International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 9 (2020), pp. 2461-2473
 https://dx.doi.org/10.37624/IJERT/13.9.2020.2461-2473

- Q = $\{q_1,q_2,\ldots,q_q\}$ be the set of queries initiated by different sites of distributed database system and FR_{ij} is the execution frequency of the j^{th} query at i^{th} site. RF_{jk} and UF_{jk} are the retrieval and update frequencies to the k^{th} fragment by j^{th} query respectively. R_{jk} and U_{jk} are the average percentage of k^{th} fragment needed for retrieval and to be updated by j^{th} query respectively.
- The cost of storing a unit data (USC_i) at the site N_i and the storage capacity (SC_i) of the site N_i.

On the basis of above mention inputs, the cost functions for replicated allocation of data is given bellow.

Total Cost of Allocation = Retrieval Cost (RC) + Update Cost (UC) + Total Storage Cost (SC)

Let *RA* is a matrix of size n × m representing an arbitrary replicated allocation of fragments and $RA_{ik} = 1$ if F_k is allocated to N_i otherwise $RA_{ik} = 0$.

The above said allocation (*RA*) is restricted under following constraints:

1. Each fragment is allocated to at least one site i.e.

$$\sum_{i=1}^{n} RA_{ik} \geq 1 \forall 1 \leq k \leq m \quad (1)$$

2. The total data stored at each site should not be more than the storage capacity of the site i.e.

$$\sum_{k=1}^{m} USC_i * Size(F_k) * RA_{ik} \leq SC_i \forall 1 \leq i \leq n \quad (2)$$

The data transmission overhead for retrievals and updates are different from each others. In case of retrieval, the communication takes place between the site originating the query and the site having the desired data fragment with minimum transmission cost from the originating site. But in case of update queries, it is necessary to update the fragment at all the sites where replicas exists so that consistency of the data can be preserved. The data transmission due to retrievals and updates are given below [22]:

$$RC = \sum_{i=1}^{n}\sum_{j=1}^{q} FR_{ij} * \sum_{k=1}^{m} RF_{jk} * \left[\min_{\forall N_i \in N}(CC_{o(i),i})\right] * \left[\frac{R_{jk}}{100} * Size(F_k)\right] \quad (3)$$

$$UC = \sum_{i=1}^{n}\sum_{j=1}^{q} FR_{ij} * \sum_{k=1}^{m} UF_{jk} * \left[\sum_{\forall N_i \in N} CC_{o(i),i}\right] * \left[\frac{U_{jk}}{100} * Size(F_k)\right] \quad (4)$$

where $CC_{o(i),i}$ is the communication cost associated between the site ($N_{o(i)}$) originating the query and the site (N_i) containing the fragment F_k. $CC_{o(i),i} = 0 \ \forall \ o(i) = i$.

$$SC = \sum_{i=1}^{n}\sum_{k=1}^{m} USC_i * Size(F_k) * RA_{ik} \quad (5)$$

The cost function for replicated allocation (*RA*) is given below:

$$C(RC) = \left\{ \begin{array}{l} \left\{ \sum_{i=1}^{n}\sum_{j=1}^{q} FR_{ij} * \sum_{k=1}^{m} RF_{jk} * \left[\min_{\forall N_i \in N}(CC_{o(i),i})\right] * \left[\frac{R_{jk}}{100} * Size(F_k)\right] \right\} + \\ \left\{ UC = \sum_{i=1}^{n}\sum_{j=1}^{q} FR_{ij} * \sum_{k=1}^{m} UF_{jk} * \left[\sum_{\forall N_i \in N} CC_{o(i),i}\right] * \left[\frac{U_{jk}}{100} * Size(F_k)\right] \right\} + \\ \left\{ \sum_{i=1}^{n}\sum_{k=1}^{m} USC_i * Size(F_k) * RA_{ik} \right\} \end{array} \right\} \quad (6)$$

International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 9 (2020), pp. 2461-2473
 https://dx.doi.org/10.37624/IJERT/13.9.2020.2461-2473

A proper allocation of data fragments is that allocation which minimizes the total cost during the execution of database queries under storage capacity constraint. The goal of data allocation is to find an optimal allocation P under the storage capacity constraint such that:

$$C(P) \leq C(RA) \ \forall \ RA$$

4. DATA ALLOCATION FRAMEWORK USING SBBO

Simplified biogeography-based optimization (SBBO) is a simplified version of the BBO [36]. Simplified biogeography-based optimization (SBBO) always uses the best solution (habitat) from the population as the emigration habitat and any other randomly chosen solution (habitat) from the population is selected as the immigration habitat [36]. The immigrating habitat is selected from a uniform probability distribution. The migration curves of the SBBO are shown in Fig. 1. The fittest solution (habitat) has a 100% probability of emigration and a zero probability of immigration [36].

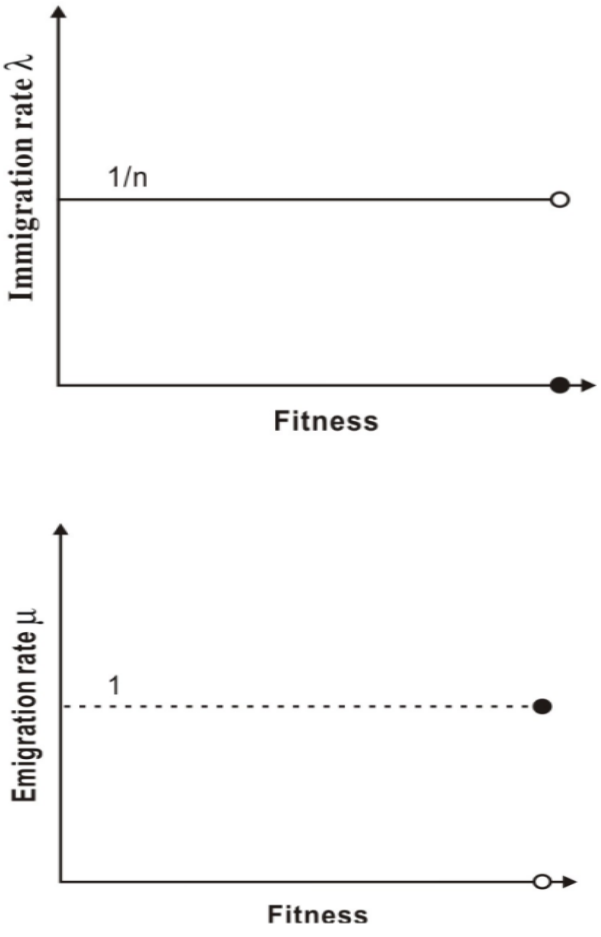

Fig. 1. Migration Curves of SBBO in a Population of n Habitats [36]

Each habitat represents a candidate solution i.e. data allocation schema. A set of Suitability Index Variable (SIV) is used to define the habitat. Each SIV consists of a set of n bits. The numbers of SIVs in a habitat depend on number of fragments. The i^{th} individual habitat (H_i) of the population can be defined as follows:

$$H_i = [SIV_1, SIV_2, SIV_3, \ldots\ldots\ldots\ldots, SIV_m]$$

SIV_k is a bit structure representing the decision variables X_{kj}. The decision variable X_{kj} has value 1 if the fragment F_k is allocated to a site S_j; otherwise 0.

Data allocation framework using SBBO based approach is given in Fig. 2.

5. EXPERIMENTAL RESULTS

The performance evaluation of the proposed SBBO based approach is done against BBO [35] and G.A. [5] based approaches for data allocation. The simulation of all the approaches is done in MATLAB 2010 on a computer having Intel(R) Core(TM) i5 processor @ 2.8 GHz and 4GB RAM to validate the proposed approach. All three algorithms are applied to the patterns shown in Table 1. Various parameters randomly generated from uniform distributions for each experiment [5] are given below:

- Communication cost range between 1 and 10
- Number of queries range between 5 and 20
- Size of each fragment range between 10 and 500
- Execution frequency of each query at different site range between 0 and 50
- Retrieval frequency of different fragments range between 0 and 20
- Average percentage of the fragment needed for retrieval range between 0 and 5
- Update frequency of different fragments range between 0 and 10
- Average percentage of the fragment needed to be updated range between 0 and 3
- Storage Capacity of each site is set between 200 and 800

For each experiment, same data set is used to check the performance of all the three approach and each algorithm is executed independently for 20 times. The values of other parameters related to SBBO BBO and GA are given below:

- Numbers of iterations are taken as 500 and 1000
- Population size is taken as 10 and 20
- Mutation Rate is taken as 0.10 and 0.15
- Maximum Immigration rate (I) = 1
- Maximum Emigration rate (E) = 1
- Elitism Parameter = 2

Experimental work is divided into following two categories:

1. Numbers of iterations, population size and mutation rate are taken as 500, 10 and 0.15 respectively when the number of sites are 6, 7 and 8
2. Numbers of iterations, population size and mutation rate are taken as 1000, 20 and 0.10 respectively when the number of sites are 12 and 14

International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 9 (2020), pp. 2461-2473
 https://dx.doi.org/10.37624/IJERT/13.9.2020.2461-2473

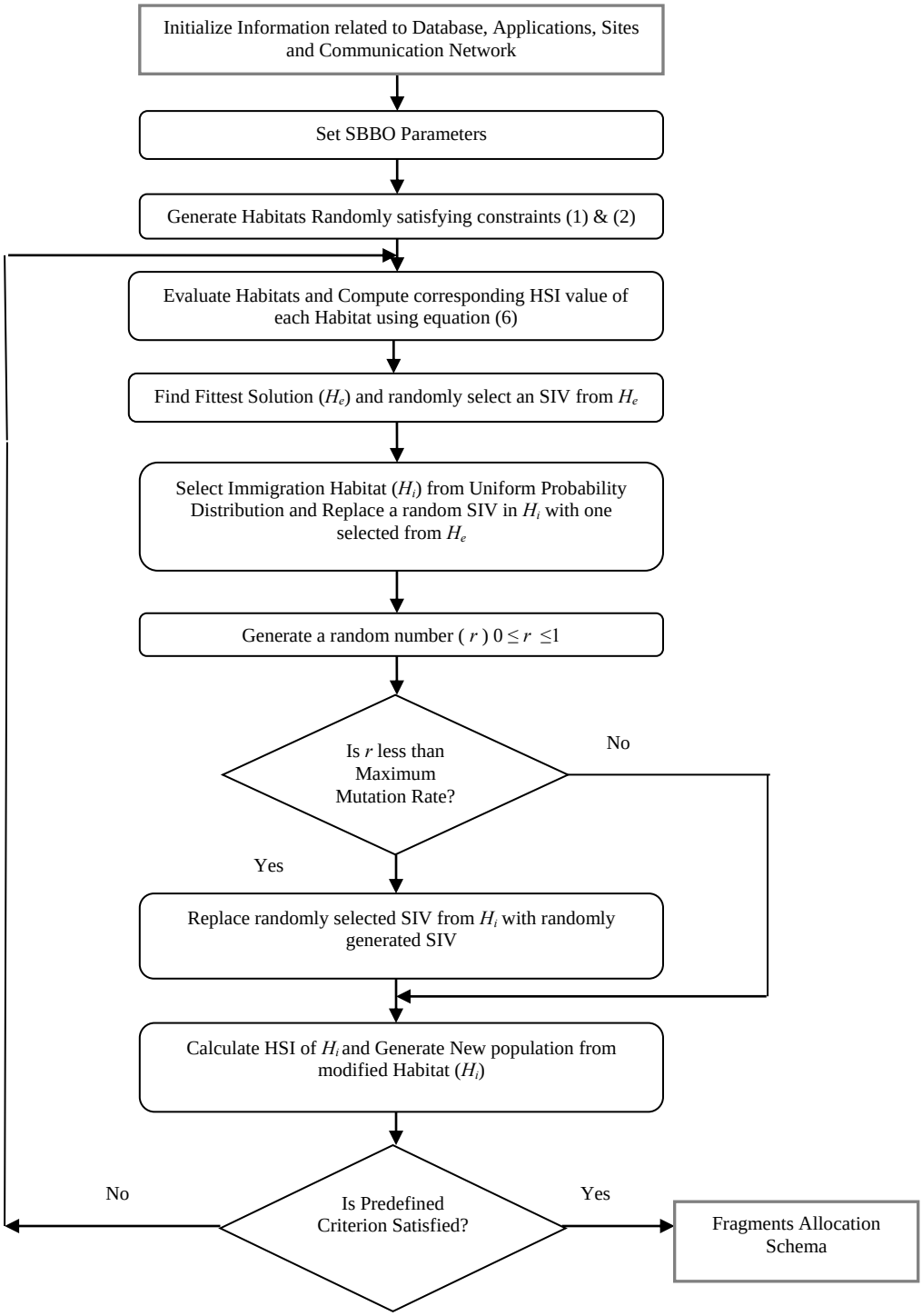

Fig. 2: Data Allocation using SBBO

Table 1: Test Cases

Number of Sites	Number of Fragments
6	4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20
7	4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20
8	4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20
12	4, 8, 9, 10, 12, 14, 16, 18, 20
14	4, 8, 9, 10, 12, 14, 16, 18, 20

Table 2, Table 3, Table 4, Table 5 and Table 6 illustrate the allocation cost achieved by all the three approaches for 6, 7, 8, 12 and 14 sites respectively for allocating the fragments ranging from 4 to 20. These results show that SBBO based approach is generating better solutions then that of provided by BBO and GA based approaches. Results also demonstrate that the performance of SBBO based approach improves further as number of sites and number of fragments increases.

The convergence graphs as well as performance characteristic of the proposed approach for fragment allocation are displayed in Fig. 3, Fig. 4, Fig. 5, Fig. 6, Fig. 7, Fig. 8, Fig. 9, Fig. 10, Fig. 11 and Fig. 12. These convergence graphs also demonstrate that the convergence rate of SBBO based approach is better than other two approaches and it improves further as number of sites and number of fragments increases.

From above discussion, one can easily conclude that the proposed SBBO based approach offers more promising solutions than other two approaches. Therefore, SBBO based approach can be successfully applied for data allocation problem in distributed database design.

6. CONCLUSION

A new SBBO based approach has been proposed for allocation of data during the distributed database design process in this paper. The performance of proposed approach has been evaluated against the performance of biogeography-based optimization (BBO) based and genetic algorithm (GA) based approaches.

Experimental results have shown that the proposed SBBO based approach is offering more promising results and has better convergence rate than other approaches. Therefore, SBBO based approach can be successfully applied for data allocation problem in distributed database design. This will help in decreasing the communication cost and speedup the query execution process. It will also help in decreasing the traffic on communication network and improve the overall performance of the system.

Table 2. Allocation cost achieved by all the three approaches for 6 sites

Number of Fragments	G.A.		BBO		SBBO	
	Minimum Cost	Average Cost	Minimum Cost	Average Cost	Minimum Cost	Average Cost
4	7.2737e+5	7.3563e+5	7.2737e+5	**7.3199e+5**	7.2737e+5	**7.3088e+5**
5	7.5335e+5	7.5819e+5	7.5335e+5	7.5877e+5	7.5335e+5	**7.5530e+5**
6	1.2254e+6	1.2414e+6	**1.2214e+6**	**1.2351e+6**	**1.2214e+6**	**1.2332e+6**
7	1.3752e+6	1.4008e+6	**1.3543e+6**	**1.3880e+6**	**1.3350e+6**	**1.3594e+6**
8	1.4327e+6	1.4880e+6	**1.4160e+6**	**1.4639e+6**	**1.3932e+6**	**1.4384e+6**
9	2.0081e+6	2.0806e+6	**1.8908e+6**	**2.0401e+6**	**1.8770e+6**	**1.97073e+6**
10	2.4136e+6	2.5352e+6	**2.3107e+6**	**2.4926e+6**	**2.2897e+6**	**2.3994e+6**
12	2.9859e+6	3.0608e+6	**2.9298e+6**	3.0754e+6	**2.8487e+6**	**2.9551e+6**
14	3.4549e+6	3.6400e+6	**3.4316e+6**	**3.5897e+6**	**3.4015e+6**	**3.5683e+6**
16	4.2686e+6	4.4202e+6	**4.2651e+6**	4.4466e+6	**4.2202e+6**	**4.4080e+6**
18	4.7394e+6	4.9634e+6	**4.7282e+6**	4.9824e+6	**4.7257e+6**	**4.9109e+6**
20	6.5028e+6	6.7092e+6	**6.4081e+6**	6.7315e+6	**6.3014e+6**	**6.6083e+6**

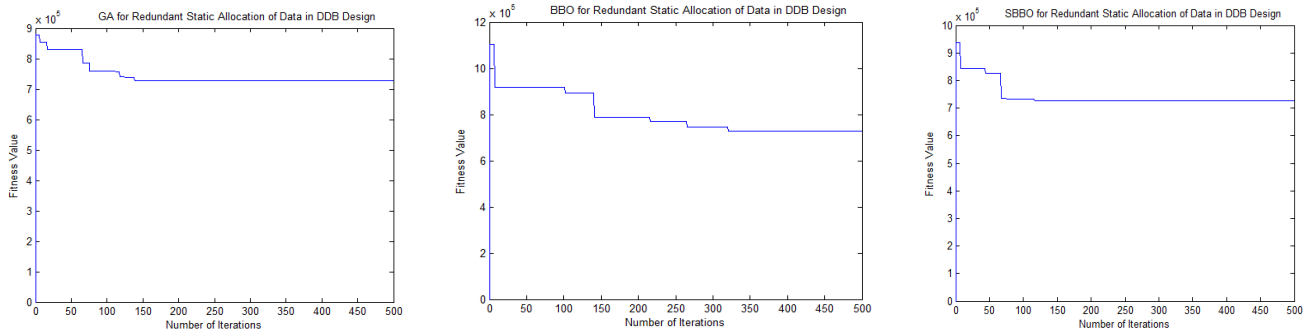

Fig. 3: Convergence graphs for 6 sites and 4 Fragments

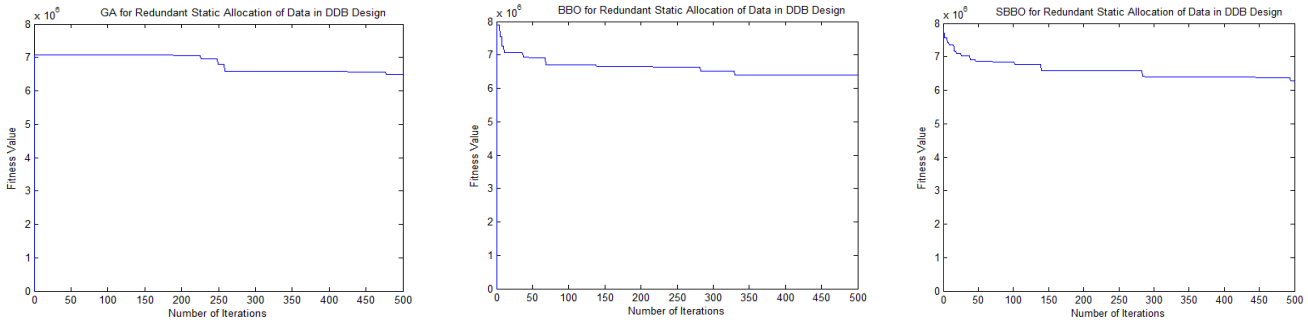

Figure 4: Convergence graphs for 6 sites and 20 Fragments

Table 3. Allocation cost achieved by all the three approaches for 7 sites

Number of Fragments	G.A.		BBO		SBBO	
	Minimum Cost	Average Cost	Minimum Cost	Average Cost	Minimum Cost	Average Cost
4	9.7616e+5	9.8112e+5	9.7616e+5	**9.8059e+5**	9.7616e+5	**9.7870e+5**
5	1.0765e+6	1.0854e+6	**1.0751e+6**	1.0883e+6	**1.0746e+6**	**1.0838e+6**
6	1.7058e+6	1.7215e+6	**1.7056e+6**	**1.7185e+6**	**1.6986e+6**	**1.7094e+6**
7	1.8723e+6	1.9113e+6	**1.8701e+6**	**1.8880e+6**	**1.8545e+6**	**1.8794e+6**
8	1.9475e+6	2.0131e+6	**1.9452e+6**	2.0253e+6	**1.9326e+6**	**1.9815e+6**
9	2.6986e+6	2.8307e+6	**2.6747e+6**	2.8516e+6	**2.5790e+6**	**2.7711e+6**
10	3.6871e+6	3.9151e+6	**3.4432e+6**	**3.7726e+6**	**3.3460e+6**	**3.5827e+6**
12	3.9503e+6	4.1030e+6	**3.9387e+6**	4.1358e+6	**3.8817e+6**	**4.0871e+6**
14	5.0187e+6	5.0697e+6	5.0483e+6	5.1004e+6	**5.0132e+6**	**5.0603e+6**
16	6.0267e+6	6.1958e+6	6.0293e+6	6.2176e+6	**5.9963e+6**	**6.1409e+6**
18	6.6436e+6	6.7300e+6	**6.6349e+6**	6.7872e+6	**6.5972e+6**	**6.7081e+6**
20	7.6735e+6	7.8523e+6	**7.6716e+6**	7.8996e+6	**7.6659e+6**	**7.7956e+6**

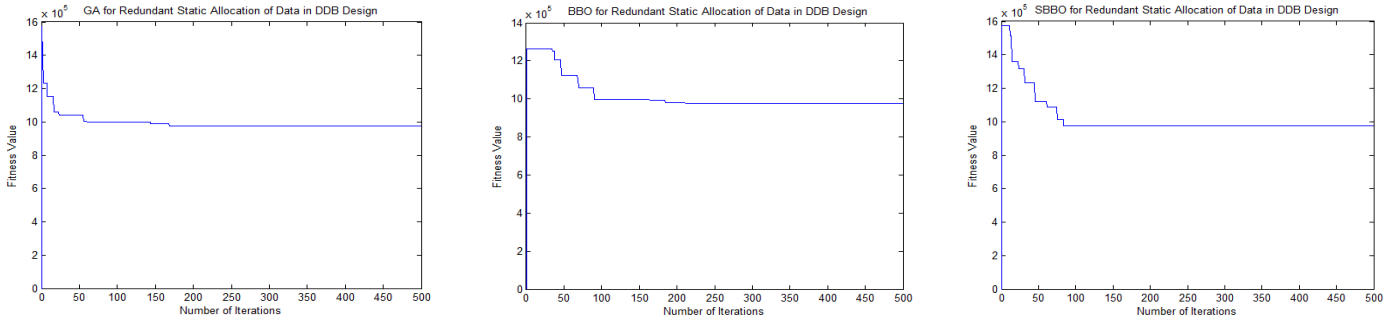

Fig. 5: Convergence graphs for 7 sites and 4 Fragments

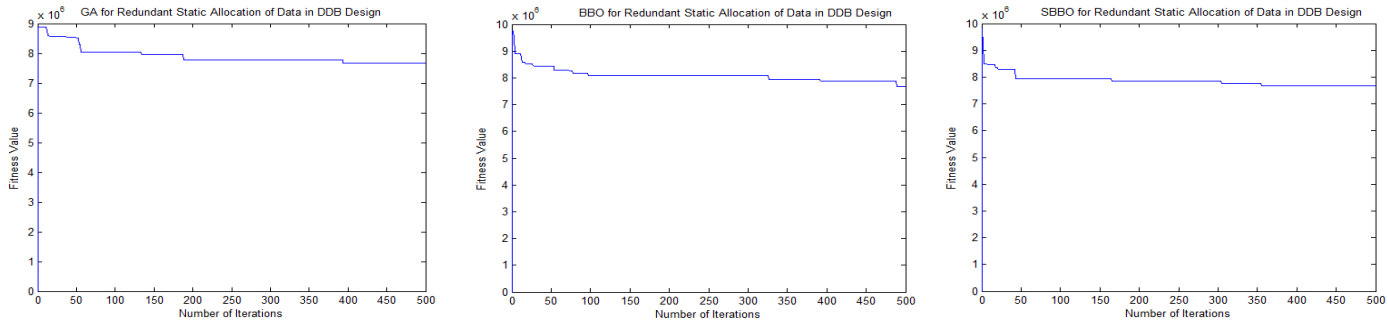

Fig. 6: Convergence graphs for 7 sites and 20 Fragments

International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 9 (2020), pp. 2461-2473
 https://dx.doi.org/10.37624/IJERT/13.9.2020.2461-2473

Table 4. Allocation cost achieved by all the three approaches for 8 sites

Number of Fragments	G.A.		BBO		SBBO	
	Minimum Cost	Average Cost	Minimum Cost	Average Cost	Minimum Cost	Average Cost
4	1.6212e+6	1.6806e+6	1.6212e+6	1.6822e+6	1.6212e+6	**1.6759e+6**
5	1.7166e+6	1.7569e+6	**1.7139e+6**	1.7628e+6	**1.7000e+6**	**1.7465e+6**
6	2.6576e+6	2.7357e+6	2.6672e+6	2.7535e+6	**2.6450e+6**	**2.6954e+6**
7	2.9235e+6	3.0269e+6	**2.9095e+6**	**3.0099e+6**	**2.7492e+6**	**2.9142e+6**
8	3.5366e+6	3.6302e+6	**3.4731e+6**	3.6872e+6	**3.2628e+6**	**3.5142e+6**
9	4.2718e+6	4.4001e+6	**4.2588e+6**	4.4492e+6	**3.9419e+6**	**4.1903e+6**
10	4.9561e+6	5.1758e+6	5.0021e+6	5.2164e+6	**4.8574e+6**	**5.0720e+6**
12	6.1124e+6	6.3220e+6	6.1318e+6	6.3413e+6	**5.9676e+6**	**6.1629e+6**
14	7.6211e+6	7.7931e+6	7.6255e+6	7.8005e+6	**7.6126e+6**	**7.7789e+6**
16	**9.1000e+6**	9.4762e+6	9.3007e+6	9.4931e+6	9.1076e+6	**9.4347e+6**
18	9.9486e+6	1.0244e+7	9.9634e+6	1.0466e+7	**9.9444e+6**	**1.0151e+7**
20	1.1854e+7	1.2003e+7	1.1928e+7	1.2237e+7	**1.1772e+7**	**1.1882e+7**

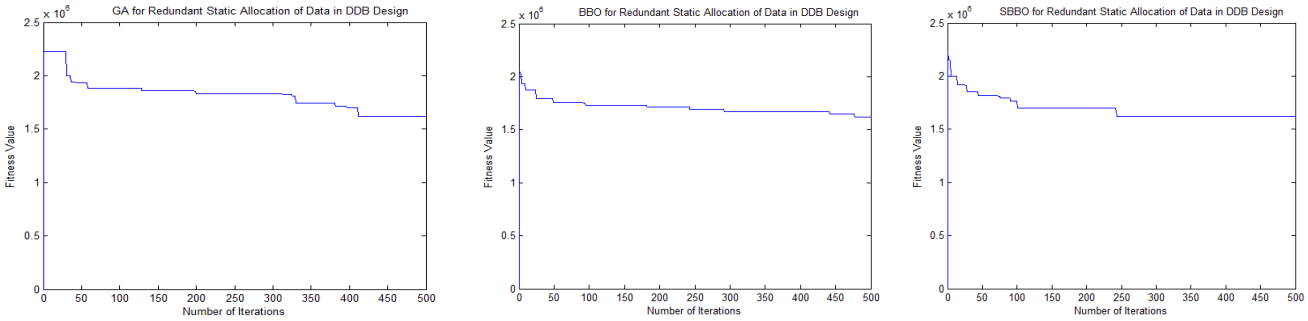

Fig. 7: Convergence graphs for 8 sites and 4 Fragments

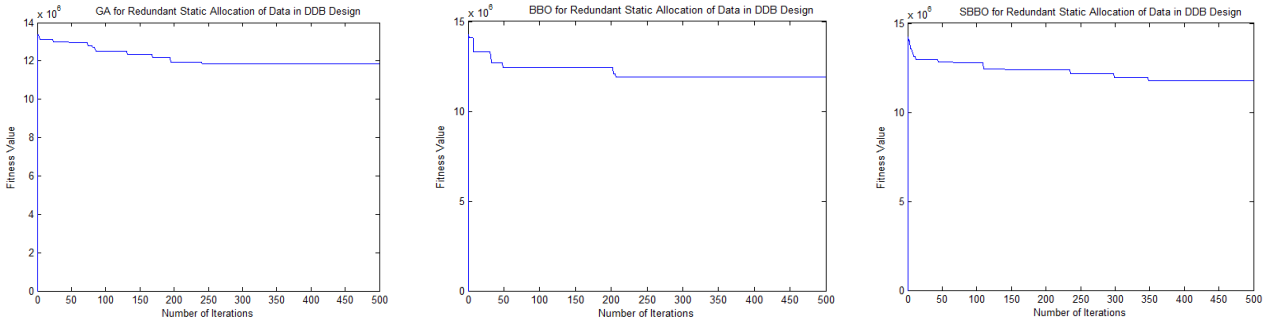

Fig. 8: Convergence graphs for 8 sites and 20 Fragments

International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 9 (2020), pp. 2461-2473
 https://dx.doi.org/10.37624/IJERT/13.9.2020.2461-2473

Table 5. Allocation cost achieved by all the three approaches for 12 sites

Number of Fragments	G.A.		BBO		SBBO	
	Minimum Cost	Average Cost	Minimum Cost	Average Cost	Minimum Cost	Average Cost
4	1.9570e+6	2.0679e+6	**1.8977e+6**	2.0681e+6	**1.8594e+6**	**1.9887e+6**
8	3.9996e+6	4.2786e+6	**3.8973e+6**	4.3320e+6	**3.2742e+6**	**4.0925e+6**
9	6. 0051e+6	6.3693e+6	**5.9196e+6**	6.4161e+6	**4.8250e+6**	**6.0013e+6**
10	6.8506e+6	7.3222e+6	**6.7800e+6**	7.3445e+6	**5.8350e+6**	**6.6693e+6**
12	8.4735e+6	8.8095e+6	**8.4168e+6**	8.8442e+6	**8.1785e+6**	**8.6708e+6**
14	1.0347e+7	1.0663e+7	1.0436e+7	1.0798e+7	**1.0287e+7**	**1.0634e+7**
16	1.3032e+7	1.3447e+7	1.3099e+7	1.3619e+7	**1.2855e+7**	**1.3280e+7**
18	1.4288e+7	**1.4766e+7**	1.4305e+7	1.5053e+7	**1.4107e+7**	1.4836e+7
20	1.6242e+7	**1.7079e+7**	1.6294e+7	1.7316e+7	**1.5654e+7**	1.7108e+7

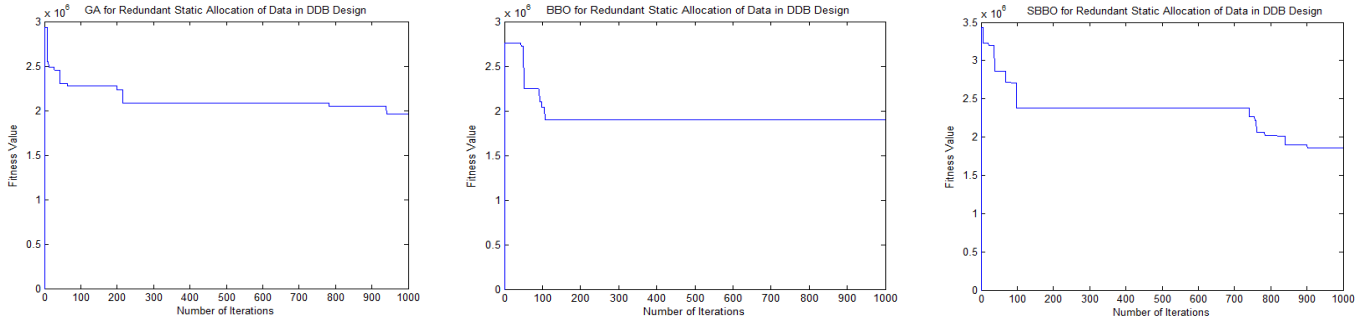

Fig. 9: Convergence graphs for 12 sites and 4 Fragments

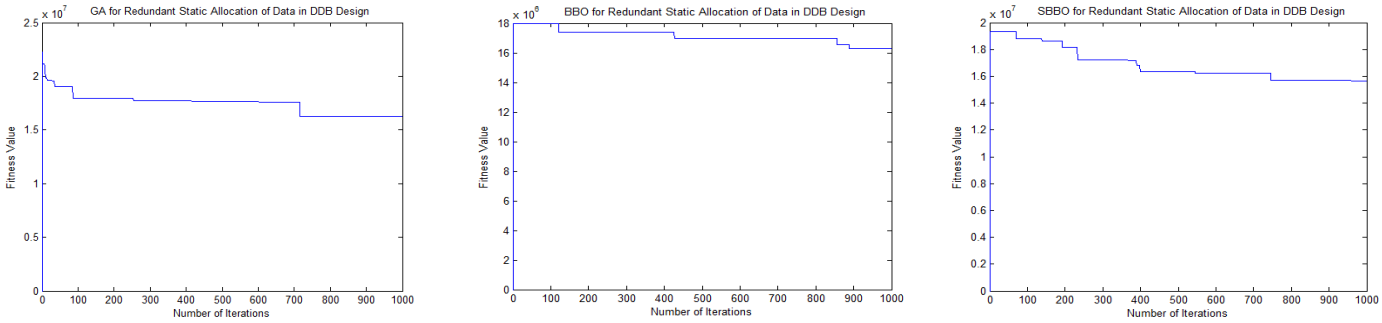

Fig. 10: Convergence graphs for 12 sites and 20 Fragments

Table 6. Allocation cost achieved by all the three approaches for 14 sites

Number of Fragments	G.A.		BBO		SBBO	
	Minimum Cost	Average Cost	Minimum Cost	Average Cost	Minimum Cost	Average Cost
4	2.4698e+6	2.7673e+6	**2.3818e+6**	**2.7526e+6**	**2.3818e+6**	**2.6691e+6**
8	5.8562e+6	6.0699e+6	**5.2686e+6**	**6.0402e+6**	**4.9287e+6**	**5.8388e+6**
9	8.5592e+6	8.7910e+6	**7.7782e+6**	**8.6359e+6**	**5.0922e+6**	**7.6359e+6**
10	1.0003e+7	1.0434e+7	**9.9692e+6**	**1.0302e+7**	**7.3564e+6**	**9.7117e+6**
12	1.1487e+7	1.2057e+7	**1.0861e+6**	**1.1924e+7**	**1.0137e+7**	**1.1789e+7**
14	1.4712e+7	1.5404e+7	1.4801e+7	1.5509e+7	**1.4520e+7**	**1.5319e+7**
16	1.7378e+7	1.8592e+7	**1.7299e+7**	1.8737e+7	**1.7043e+7**	**1.8430e+7**
18	1.9702e+7	**2.0430e+7**	1.9914e+7	2.0608e+7	**1.9528e+7**	2.0557e+7
20	2.2750e+7	2.5640e+7	2.4076 e+7	2.6364e+7	**2.1937e+7**	**2.5316e+7**

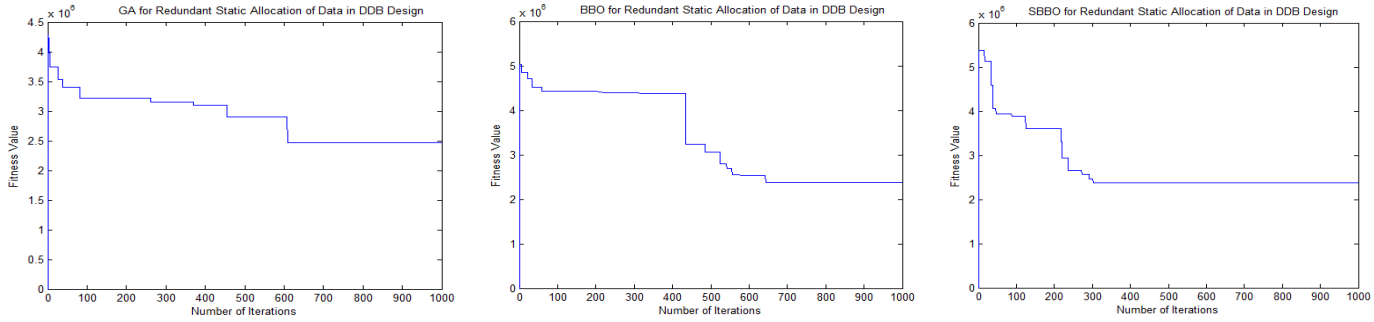

Fig. 11: Convergence graphs for 14 sites and 4 Fragments

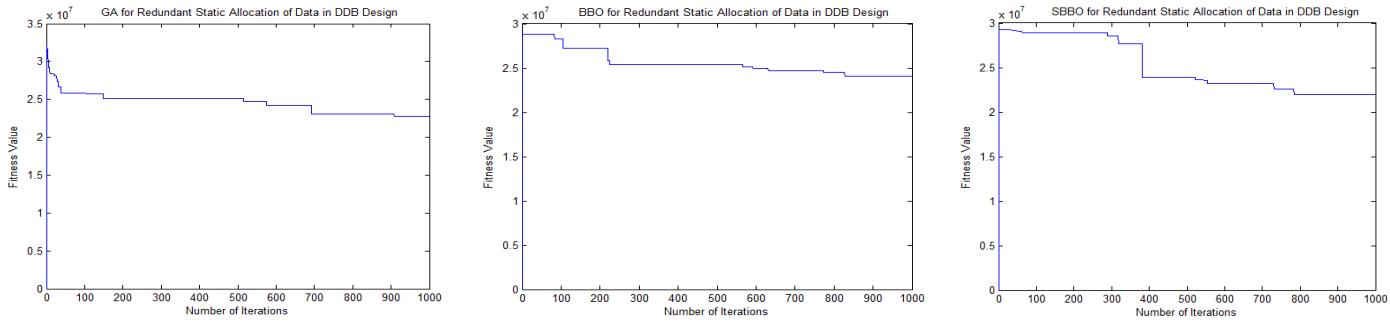

Figure 12: Convergence graphs for 14 sites and 20 Fragments

REFERENCES

[1] H. Abdalla and A.M. Artoli, "Towards an Efficient Data Fragmentation, Allocation, and Clustering Approach in a Distributed Environment", *Information,* 2019, *10*, 112.

[2] R.K. Adl and S.M.T.R. Rankoohi, "*A New Ant Colony Optimization Based Algorithm for Data Allocation Problem in Distributed Databases*", Knowledge and Information Systems, Vol. 20, No. 3, pp 349-373, 2009.

[3] A. A. Amer, A.A. Sewisy and T.M.A. Elgendy, "An optimized approach for simultaneous horizontal data fragmentation and allocation in Distributed Database Systems (DDBSs)", Heliyon, Vol.3, Issue 12, 2017, pp. 1-42

[4] A.A. Amer, "On K-means clustering-based approach for DDBSs design", Journal of Big Data, 7, 31 (2020). https://doi.org/10.1186/s40537-020-00306-9

[5] I. Ahmad, K. Karlapalem, Y.K. Kwok and S.K. So, "*Evolutionary Algorithms for Allocating Data in Distributed Database Systems*", Distributed Parallel Databases, Vol. 11, No. 1, pp. 5–32, 2002.

[6] P.M.G. Apers, "*Data Allocation in Distributed Database Systems*", ACM Transaction on Database Systems, Vol. 13, No. 3, pp. 263-304, 1988.

[7] K. Barker and S. Bhar , "*A Graphical Approach to Allocating Class Fragments in Distributed Objectbase Systems*", Distributed and Parallel Databases, Vol. 10, No. 3, pp 207-239, 2001.

[8] H.T. Barney and G.C. Low, "*Object Allocation with Replication in Distributed Systems*", World Academy of Science, Engineering and Technology, Vol. 24, pp. 496-504, 2008.

[9] R. Blankinship, A.R. Hevner and S.B. Yao, "*An Iterative Method for Distributed Database Optimization*", Data & Knowledge Engineering, Vol. 21, No. 1, pp. 1–30, 1996

[10] R. G. Casey, "*Allocation of Copies of a File in an Information Network*", In Proceedings of AFIPC 1972 SJCC, Vol. 40, pp. 617-625, 16-18 May 1972.

[11] F. Castro-Medina, L. Rodríguez-Mazahua, A. López-Chau, I. Machorro-Cano and M. A. Abud-Figueroa, "Design of a Horizontal Data Fragmentation, Allocation and Replication Method in the Cloud," 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE), Vancouver, BC, Canada, 2019, pp. 614-621, doi: 10.1109/COASE.2019.8842934.

[12] S. Ceri, S. Navathe, and G. Weiderhold, "*Distribution Design of Logical Database Schemas*," IEEE Transactions on Software Engineering, Vol. 9, pp. 487-563, 1983.

[13] S. Ceri and G. Pelagatti, "*Distributed Databases: Principles Systems*", McGraw-Hill International Edition, 1985.

[14] P.P.-S. Chen and J. Akoka, "*Optimal Design of Distributed Information Systems", IEEE Transactions on Computers*, Vol. C-29, No. 12, pp. 1068-1080, 1980.

[15] G. Chiu and C.S. Raghavendra, "*A Model for Optimal Database Allocation in Distributed Computing Systems*", In Proceedings of IEEE INFOCOM 1990, Vol. 3, pp. 827-833, 3-7 June 1990.

[16] W.W. Chu, "*Optimal File Allocation in Multiple Computer Systems*", IEEE Transactions on Computers, Vol. C-18, No. 10, pp. 885-889, 1969.

[17] S.R. Choudhary and C.K. Jha, "Task Allocation in Distributed Real Time Database Systems in IoT", 4th International Conference on Internet of Things and Connected Technologies (ICIoTCT), 2019, pp. 54-68.

[18] A. Corcoran and J. Hale, "*A Genetic Algorithm for Fragment Allocation in a Distributed Database System*", In Proceedings of ACM Symp. Applied Computing, pp. 247-250, 1994.

[19] N.H. Daudpota, "*Five Steps to Construct a Model of Data Allocation for Distributed Database Systems*", Journal of Intelligent Information Systems, Vol. 11, No. 2, pp. 153-168, 1998.

[20] K.P. Eswaran, "*Placement of Records in a File and File Allocation in a Computer Network*", In Proceedings of the IFIP Congress on Information Processing, pp. 304-307, 1974.

[21] I.O. Hababeh, M. Ramachandran and N. Bowring, "*A high-performance computing method for data allocation in distributed database systems*", The Journal of Supercomputing, Vol. 39, No. 1, pp. 3-18, 2007.

[22] Y.-F. Huang and J.-H. Chen, "*Fragment Allocation in Distributed Database Design*", Journal of Information Science and Engineering, Vol. 17, pp. 491- 506, 2001.

[23] K. Karlapalem, I. Ahmad, S.-K. So and Y. Kwok, "*Empirical Evaluation of data allocation algorithms for distributed multimedia database systems*", In Proceedings of The Twenty-First Annual International Computer Software and Applications Conference (COMPSAC '97), pp. 296-301, 13-15 August 1997.

[24] K. Karlapalem and N.M. Pun, "*Query-driven data allocation algorithms for distributed database systems*", In Proceedings of 8th International Conference on Database and Expert Systems Applications (DEXA'97), pp. 347-356, September 1997.

[25] X. Lin, M. Orlowska and Y. Zhang, "*On Data Allocation with the Minimum Overall Communication Costs In Distributed Database Design*" In Proceedings of ICCI'93: 5th International Conference on Computing and Information, pp. 539-544, 27-29 May 1993.

[26] X. Lin and M. Orlowska, "*An Integer linear Programming Approach to data Allocation with the Minimum Total Communication Cost in Distributed Database Systems*", Information Sciences, Vol. 85, No. 1-3, pp. 1–10, 1995.

[27] T. Loukopoulos and I. Ahmad, "*Static and Adaptive Distributed Data Replication using Genetic Algorithms*", Journal of Parallel and Distributed Computing, Vol. 64, No. 11, pp. 1270-1285, 2004.

[28] D. Nashat and A.A. Amer, "A Comprehensive Taxonomy of Fragmentation and Allocation Techniques in Distributed Database Design" ACM Comput. Surv. 51, 1, Article 12 (January 2018), 25 pages.

[29] A.S. Mamaghani, M. Mahi, M.R. Meybodi, and M.H. Moghaddam, "*A Novel Evolutionary Algorithm for Solving Static Data Allocation Problem in Distributed Database Systems*", In Proceedings of 2nd International Conference on Network Applications Protocols and Services (NETAPPS), pp. 14-19, 22-23 September 2010.

[30] S.T. March and S. Rho, "*Allocating Data and Operations to Nodes in Distributed Database Design*", IEEE Transactions on Knowledge and Data Engineering, Vol. 7, No. 2, pp. 305-317, 1995.

[31] M. Ozsu and P. Valduriez, "*Principles of Distributed Database Systems*", Prentice Hall, 2nd Edition, 2002.

[32] S. Rahmani, V. Torkzaban and A.T. Haghighat, "*A New Method of Genetic Algorithm for Data allocation in Distributed Database Systems*", In Proceedings of IEEE 1st International Workshop on Education Technology and Computer Science, pp. 1037-1041, 2009.

[33] S. Ram and R. Marsten, "*A Model for Database Allocation Incorporating a Concurrency Control Mechanism*", IEEE Transactions on Knowledge and Data Engineering, Vol. 3, No. 3, pp. 389-395, 1991.

[34] H. Rahimi, F. Parand and D. Riahi, "Hierarchical simultaneous vertical fragmentation and allocation using modified Bond Energy Algorithm in distributed databases", Applied Computing and Informatics Volume 14, Issue 2, July 2018, Pages 127-133

[35] Arjan Singh, Karanjeet Singh Kahlon and Rajinder Singh Virk, "Replicated Static Data Allocation in Distributed Databases Using Biogeography-Based Optimization", In Proceedings of Fifth International Conference on Advances in Communication, Network and Computing – CNC 2014, Feb 21-22, 2014, Chennai, India.

[36] D. Simon, "*A Probabilistic Analysis of a Simplified Biogeography-Based Optimization Algorithm*", Evolutionary Computation, Vol. 19, No. 2, pp. 167-188, 2011.

[37] A.M. Tamhankar and S. Ram, "*Database Fragmentation and Allocation: An Integrated Methodology and Case Study*", IEEE Transactions on Systems, Man and Cybernetics-Part A: System and Humans, Vol. 28, No. 3, pp. 288-305, 1998.

[38] U. Tosun, T. Dokeroglu and A. Cosar, "*Heuristic Algorithms for Fragment Allocation in a Distributed*

Database System", Book Chapter, Computer and Information Sciences III, Springer, pp. 401-408, 2013.

[39] E. Wong, R.H. Katz, "*Distributing a Database for Parallelism*" In Proceedings of the 1983 ACM SIGMOD International Conference on Management of Data (SIGMOD '83), pp. 23-29, 1983.

[40] D.H. Wolpert and W.G. Macready, "No free lunch theorems for optimization", IEEE Transactions on Evolutionary Computing, 1997;1:67–82.