Hindawi Publishing Corporation
Chinese Journal of Engineering
Volume 2014, Article ID 785321, 9 pages
http://dx.doi.org/10.1155/2014/785321

Research Article

Nonreplicated Static Data Allocation in Distributed Databases Using Biogeography-Based Optimization

Arjan Singh,[1] Karanjeet Singh Kahlon,[2] and Rajinder Singh Virk[2]

[1] *Department of Mathematics, Punjabi University, Patiala, Punjab 147001, India*
[2] *Department of Computer Science & Engineering, Guru Nanak Dev University, Amritsar, Punjab 143005, India*

Correspondence should be addressed to Arjan Singh; arjanpu@gmail.com

Received 1 November 2013; Accepted 16 December 2013; Published 23 February 2014

Academic Editors: H. Hu and H. H. Iu

Allocation of data is one of the key design issues of distributed database. A major cost of query execution in a distributed database system is the data transfer cost from one site to another site. The allocation of fragments among the different sites over the network plays an important role in performance of the distributed database system. The main objective of a data allocation in distributed database is to place the data fragments at different sites in such a way, so that the total data transfer cost can be minimized while executing a set of queries. In this paper, a new biogeography-based optimization (BBO) algorithm has been used to allocate the fragments during the design of distributed database system. The goal of this paper is to design a fragments allocation algorithm, so that the total data transmission cost can be minimized. To show the performance of proposed algorithm, results of biogeography-based optimization algorithm for data allocation are compared with genetic algorithm.

1. Introduction

Distributed database technology is one of the most important developments of the past two decades in the field of database systems. Distributed database system technology is the union of two separate branches of computer science: database system and computer network [1]. Distributed database technology has become an integral part of most of the business organization due to its decentralized nature. Distributed databases have eliminated many of the shortcomings of the centralized databases and fit more naturally in the decentralized structures of many organizations [2]. Distributed database can be defined as a collection of logically interrelated data distributed over the sites of a computer network [1]. Distributed database system has many advantages over centralized database system [1, 2]:

(i) reduced communication overhead,
(ii) improved performance,
(iii) reliability,
(iv) availability,
(v) expandability.

The design of centralized database has two main issues: designing the conceptual schema and designing the physical database. But the design of distributed databases adds two more issues: designing the fragmentation of global relations and allocation of fragments over network [2]. All these issues complicate the design of distributed database. The problem of fragmenting the database is a difficult one in itself and a number of different techniques have been proposed for fragmenting the database by different research works. This study concentrates only on data/fragments allocation problem.

Fragment allocation can further be divided into two different categories: replicated/redundant and nonreplicated/nonredundant [1, 2]. In a nonreplicated/nonredundant allocation exactly one copy of each fragment will exist across all the sites, while under a replicated/redundant allocation, fragments are replicated over multiple sites. Data in distributed database system is allocated according to two different types of access patterns: static and dynamic [1]. In a static environment, the access probabilities of application running on different site to fragments never change but in a dynamic environment these probabilities change over time.

Chu [3] was the first to develop a model to minimize overall operating costs under the constraints of response time and storage capacity with fixed number of copies of each file. Casey [4] further investigates Chu's allocation model and relaxes the assumption of fixed number of copies. Casey [4] has given stress on the difference between updates and retrieval. Eswaran [5] proved that Casey's formulation was NP-complete, so finding optimal solution is not computationally feasible. Ceri et al. [6] considered the problem of file allocation for typical distributed database applications with a simple model of transaction execution. Ceri et al. [6] proposed a nonreplicated allocation of data and suggested that once the optimal nonreplicated solution has been found for nonreplicated environment, then replication can be handled easily by applying a greedy algorithm.

Apers [7] proved that the fragment allocation problem in distributed database is altogether different from the file allocation problem. Apers proposed a method for data allocation so that total data transfer cost during the execution of a set of transaction can be minimized. Sarathy et al. [8] have given nonlinear integer programming formulation for fragments allocation. Tamhankar and Ram [9] had given an integrated method of fragmentation and allocation together. Corcoran and Hale [10] and March and Rho [11, 12] presented a genetic algorithm-based approach to allocate operations to nodes. Loukopoulos and Ahmad [13] have given genetic algorithms for static and adaptive distributed data replication. Ahmad et al. [14] compared genetic algorithm, a simulated evolution algorithm, mean field annealing algorithm and neighborhood search algorithm for data allocation in distributed database design. Ahmad et al. showed that when efficiency and solution quality are equally important then genetic algorithm is an attractive solution [14]. Menon [15] has presented an integer programming formulation for the nonredundant version of the fragment allocation problem. Hababeh et al. [16] have given a high-performance computing method for data allocation in distributed database system using cluster based approach for network sites. Rahmani et al. [17] have tried to improve the Hababeh et al. allocation by incorporating genetic algorithm. More recently, Abdalla [18] has given a synchronized design technique for efficient data distribution.

In all of the above approaches, data allocation has been proposed based on the static data access patterns. Brunstroml et al. [19] proposed an optimization algorithm for nonreplicated dynamic allocation of fragments in distributed database systems. Ulus and Uysal [20, 21], Singh and Kahlon [22], and Abdallaha et al. [23] have given threshold algorithm, TTC algorithm, and POE algorithm, respectively, to further improve the performance of optimization algorithm given by Brunstroml et al. [19]. Wolfson et al. [24] introduced a model for adaptive replicated data allocation for data redistribution.

In this paper, a new algorithm for nonreplicated static allocation of fragments during distributed database design using biogeography-based optimization technique is introduced. To show the performance of the proposed algorithm, results are compared with the genetic algorithm of Ahmad et al. [14]. The new proposed algorithm is giving quality solutions within a shorter time.

Table 1: Description of various notations.

Symbol	Meaning
S	The set of all sites
n	The number of sites in the network
S_i	The ith site
Q	The set of all queries
q	The number of distributed database queries
Q_j	The jth query
F	The set of all fragments
m	The number of data fragments in the database system
F_k	The kth fragment
FR_{ij}	The execution frequency of the jth query at ith site
RF_{jk}	The retrieval frequency to the kth fragment by jth query
R_{jk}	Percentage of kth fragment needed for retrieval by jth query
UF_{jk}	The update frequency to the kth fragment by jth query
U_{jk}	Percentage of kth fragment needed to be updated by jth query
$\mathrm{CC}_{i,i'}$	The communication cost from site S_i to site $S_{i'}$
Size (F_k)	Size of the kth fragment
RC	Retrieval cost
UC	Update cost
TC	The total data transfer cost

2. The Data Allocation Model

2.1. Fragment Allocation Problem. Assume a distributed database system consisting of sites $S = \{S_1, S_2, \ldots, S_n\}$ on which a set of queries $Q = \{q_1, q_2, \ldots, q_q\}$ is running. Each site has its own processing power, memory, and local database system and all the sites are connected by a communication link network. Let $F = \{F_1, F_2, \ldots, F_m\}$ be the set of fragments after partitioning all global relations during fragmentation phase of distributed database design. The allocation problem involves finding the optimal placement of the fragments (F) to the sites (S). The optimality can be defined with respect to two measures, minimal cost and performance [1].

2.2. The Cost Model. Table 1 gives the description of various notations used to draw the cost model of data allocation. There are primarily two types of costs associated with execution of a query. The first type is the cost of retrieval of fragments to process a query and the second type is the cost to update fragments to process that query. The formula to calculate total cost of data transfer is given as follows.

Table 2: Minimum and average cost achieved by GA and BBO for 4 sites.

Number of fragments	GA		BBO	
	Minimum cost	Average cost	Minimum cost	Average cost
4	409010	409010	409010	409010
6	693420	693420	693420	693470
8	921810	921810	92810	923487
10	1.4887e + 6	1.4887e + 6	1.4887e + 6	1.4905e + 6
12	1.8345e + 6	1.8361e + 6	1.8345e + 6	1.8374e + 6
16	2.5841e + 6	2.5869e + 6	2.5841e + 6	2.5887e + 6
20	2.7861e + 6	2.7905e + 6	2.7855e + 6	2.7912e + 6
24	3.2696e + 6	3.2881e + 6	3.2622e + 6	3.2983e + 6

Table 3: Minimum and average running time of GA and BBO for 4 sites.

Number of fragments	GA		BBO	
	Minimum time	Average time	Minimum time	Average time
4	0.5645	0.5919	0.4315	0.4559
6	0.6375	0.6399	0.4873	0.4985
8	0.6569	0.6687	0.5344	0.5391
10	0.7336	0.7381	0.5694	0.5714
12	0.7590	0.7882	0.6016	0.6049
16	0.8798	0.8827	0.6685	0.6705
20	0.9811	0.9953	0.7307	0.7328
24	1.0147	1.0687	0.9005	0.9162

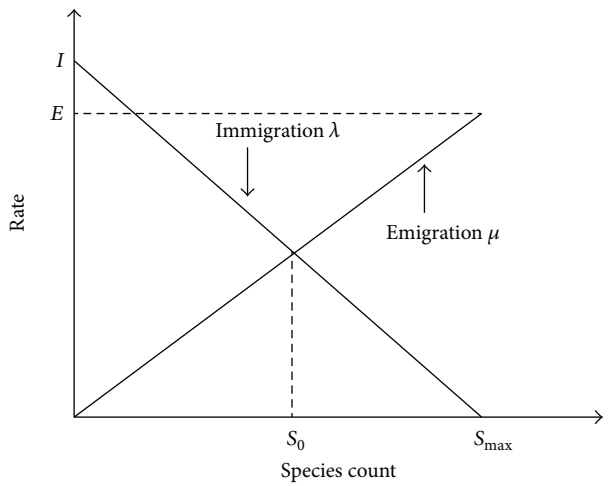

Figure 1: Species model of a single habitat [26].

Total data transfer cost (TC) = retrieval cost (RC) + update cost (UC):

$$\mathrm{TC} = \sum_{i=1}^{n}\sum_{j=1}^{q}\mathrm{FR}_{ij} * \left(\mathrm{RC}_j + \mathrm{UC}_j\right),$$

$$\mathrm{RC}_j = \sum_{k=1}^{m}\mathrm{RF}_{jk} * \left(\mathrm{CC}_{i,A_k}\right) * \left\{\frac{R_{jk}}{100} * \mathrm{Size}\left(F_k\right)\right\},$$

$$\mathrm{UC}_j = \sum_{k=1}^{m}\mathrm{UF}_{jk} * \left(\mathrm{CC}_{i,A_k}\right) * \left\{\frac{U_{jk}}{100} * \mathrm{Size}\left(F_k\right)\right\}. \quad (1)$$

CC_{i,A_k} is the communication cost associated with ith site and the site containing the kth fragment, where $A_k = i'$ means kth fragment is allocated at site i' and $\mathrm{CC}_{i,A_k} = 0$ if $i = i'$.

2.3. Cost Function. The main objective of the study is to generate a fragment allocation schema which can minimized the total data transmission cost during the execution of database queries. So, the cost function that has to be minimized is given below:

$$\min\left\{\sum_{i=1}^{n}\sum_{j=1}^{q}\mathrm{FR}_{ij} * \left\{\left\{\sum_{k=1}^{m}\mathrm{RF}_{jk} * \left(\mathrm{CC}_{i,A_k}\right) * \left\{\frac{R_{jk}}{100} * \mathrm{Size}\left(F_k\right)\right\}\right\} + \left\{\sum_{k=1}^{m}\mathrm{UF}_{jk} * \left(\mathrm{CC}_{i,A_k}\right) * \left\{\frac{U_{jk}}{100} * \mathrm{Size}\left(F_k\right)\right\}\right\}\right\}\right\}. \quad (2)$$

Table 4: Minimum and average cost achieved by GA and BBO for 8 sites.

Number of fragments	GA		BBO	
	Minimum cost	Average cost	Minimum cost	Average cost
4	1.3798*e* + 6	1.3840*e* + 6	1.3798*e* + 6	1.3808*e* + 6
6	2.1302*e* + 6	2.1315*e* + 6	2.1302*e* + 6	2.1320*e* + 6
8	2.8273*e* + 6	2.8397*e* + 6	2.8273*e* + 6	2.8408*e* + 6
10	4.5312*e* + 6	4.5454*e* + 6	4.5312*e* + 6	4.5649*e* + 6
12	5.5677*e* + 6	5.5823*e* + 6	5.5672*e* + 6	5.5848*e* + 6
16	8.4488*e* + 6	8.4828*e* + 6	8.4134*e* + 6	8.4973*e* + 6
20	8.7929*e* + 6	8.8256*e* + 6	8.7836*e* + 6	8.8487*e* + 6
24	1.0573*e* + 7	1.0738*e* + 7	1.0537*e* + 7	1.0832*e* + 7

Table 5: Minimum and average running time of GA and BBO for 8 sites.

Number of fragments	GA		BBO	
	Minimum time	Average time	Minimum time	Average time
4	0.6833	0.7037	0.5630	0.5684
6	0.7678	0.7833	0.6057	0.6346
8	0.8364	0.8967	0.6601	0.6996
10	0.9532	1.0090	0.7254	0.7673
12	0.9972	1.1073	0.7871	0.8179
16	2.1414	2.2961	0.9204	0.9681
20	2.6198	2.7222	1.0682	1.0809
24	3.0563	3.1742	1.3687	1.3931

3. The Biogeography-Based Optimization Algorithm for Data Allocation

3.1. Biogeography-Based Optimization. The biogeography-based optimization (BBO) is a newly developed population-based evolutionary technique. Biogeography-based optimization (BBO) is based on theory of biogeography. Biogeography is the study of geographical distribution of species. Simon [25] developed the biogeography-based optimization (BBO). BBO is primarily based on "The Theory of Island Biogeography" given by MacArthur and Wilson [26]. MacArthur and Wilson [26] have given a mathematical model of biogeography. The mathematical model describes that the rate of change in the number of species on an island highly depends on the stability between the immigration of new species onto the island and the emigration of established species [26].

The BBO algorithm works on a population called habitats (or islands). Each habitat represents a possible solution to the problem in hand. Each solution feature of a habitat is called a suitability index variable (SIV) of that habitat. The fitness of each habitat is represented by its habitat suitability index (HSI). HSI is a metric that determines the goodness of a candidate solution. Habitats with a high HSI tend to have a large number of species, while those with a low HSI have a small number of species. Habitats with a high HSI have many species that emigrate to nearby habitats. Habitats with a high HSI have a low species immigration rate (λ) and a high species emigration rate (μ). Habitats with a low HSI have a high species immigration rate (λ) because of their thin populations. This immigration of new species to low HSI habitats may raise the HSI of the habitat, because the suitability of a habitat is proportional to its biological diversity. However, if a habitat's HSI remains low, then the species that reside there will tend to be vanished. This will further open the way for additional immigration. So the low HSI habitats are more dynamic in their species distribution than high HSI habitats. Figure 1 shows the relationships between fitness of habitats (number of species), immigration rate (λ), and emigration rate (μ) [25].

The immigration rate (λ) and emigration rate (μ) are functions of the number of species in the habitat. They can be calculated as follows [25]:

$$\lambda_k = I\left(1 - \frac{K}{n}\right),$$
$$\mu_k = E\left(\frac{K}{n}\right), \tag{3}$$

where I is the maximum possible immigration rate; E is the maximum possible emigration rate; K is the number of species of the kth individual; n is the maximum number of species.

3.2. BBO Algorithm. In biogeography-based optimization, there are two operators: migration and mutation [25]. A population of candidate solution can be represented by different design variables. Each design variable for a particular population member is considered as suitability index (SIV).

Migration is a probabilistic operator that improves the quality of a habitat. The immigration and emigration of each

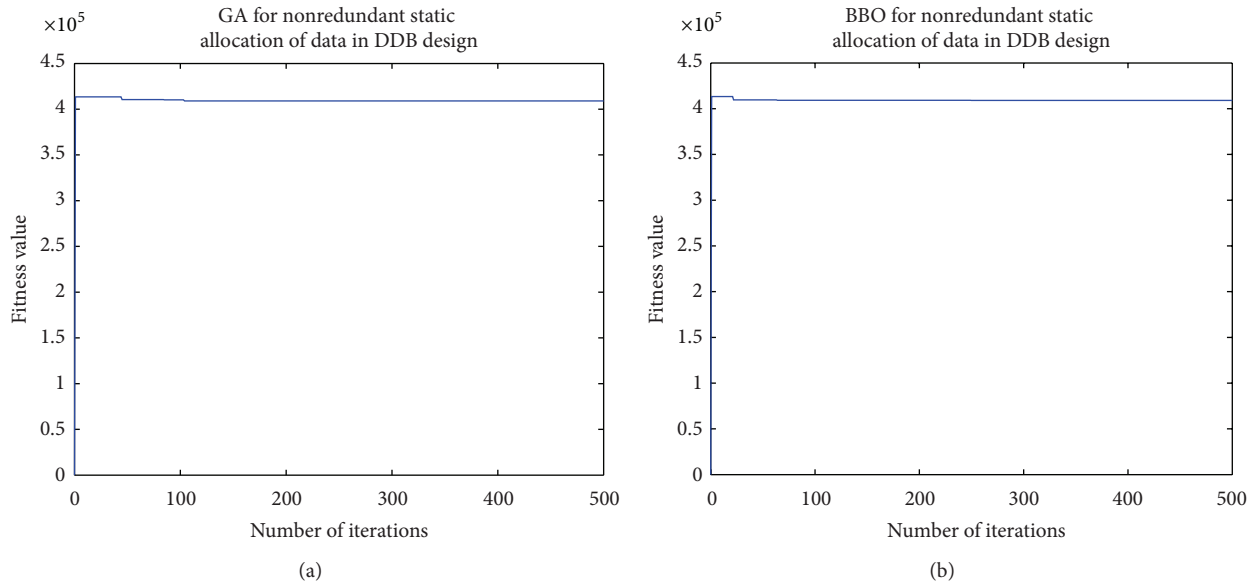

Figure 2: Convergence of GA and BBO for 4 fragments and 4 sites.

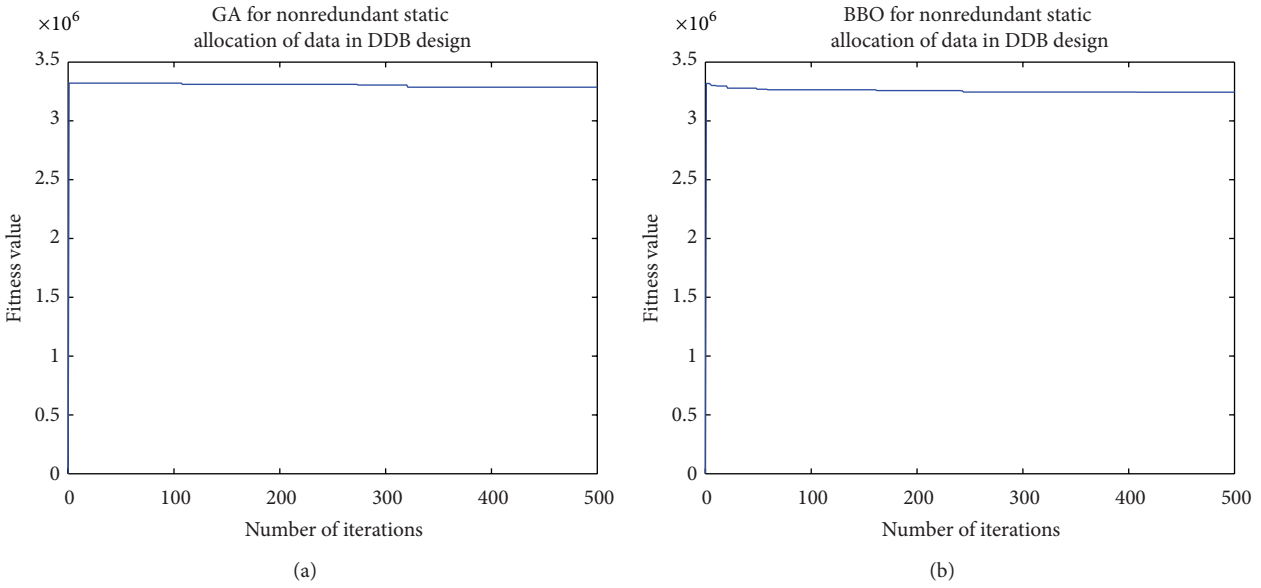

Figure 3: Convergence of GA and BBO for 24 fragments and 4 sites.

solution are used to probabilistically share the information between habitats. For each habitat H_i, its immigration rate λ_i is used to probabilistically make a decision whether to immigrate or not. If immigration is selected, then the emigrating habitat H_e is selected probabilistically based on the emigration rate μ_e. Migration is represented as [25]

$$H_i\,(\mathrm{SIV}) \longleftarrow H_e\,(\mathrm{SIV})\,. \tag{4}$$

Mutation is a probabilistic operator that randomly modifies a habitat's SIV. A randomly generated SIV replaces a selected SIV in the solution H_i according to a mutation probability, which is predefined. The main reason of mutation is to increase diversity of the population. Mutation is useful for both poor solution and good solution. For low HSI solutions, mutation gives them an opportunity of enhancing the quality of solutions, and for high HSI solutions, mutation is capable of making them better [25].

The biogeography-based optimization algorithm is given (see Algorithm 1) [25, 27].

3.3. Encoding of Habitat. In the proposed BBO algorithm for data allocation, the allocation of each fragment to different sites over the communication network is encoded in a binary representation. For example, if a data fragment is assigned to site 2, then its assignment value is 10. The assignment values of all the data fragments are concatenated to form

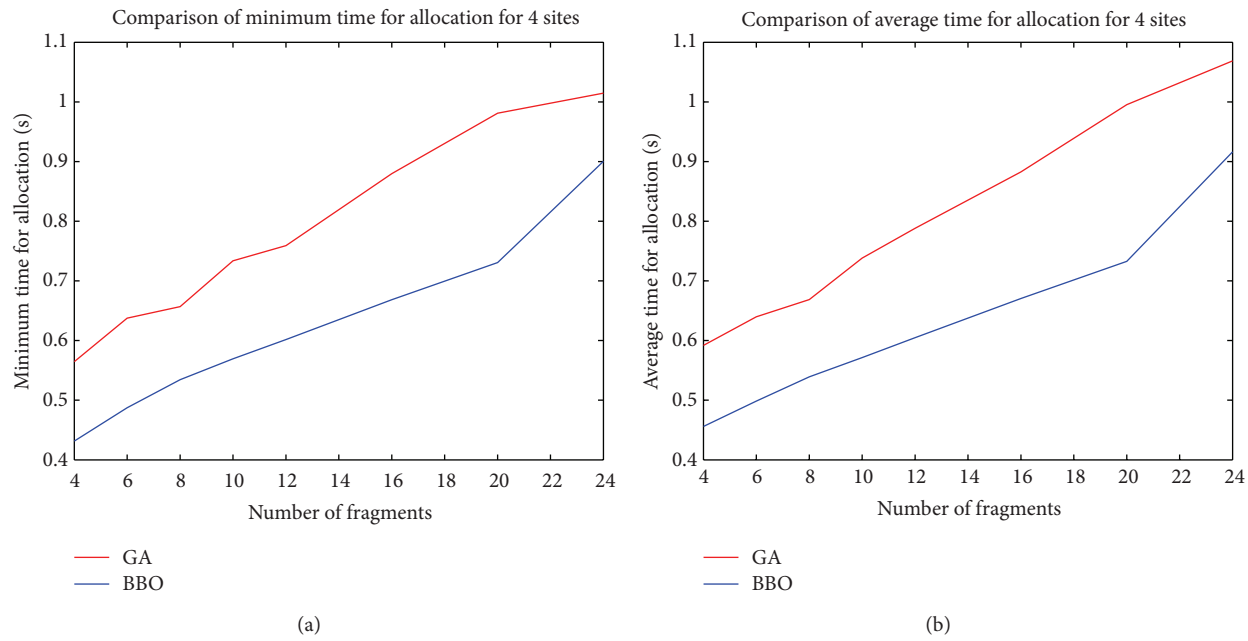

Figure 4: Minimum and average running time of GA and BBO for 4 sites.

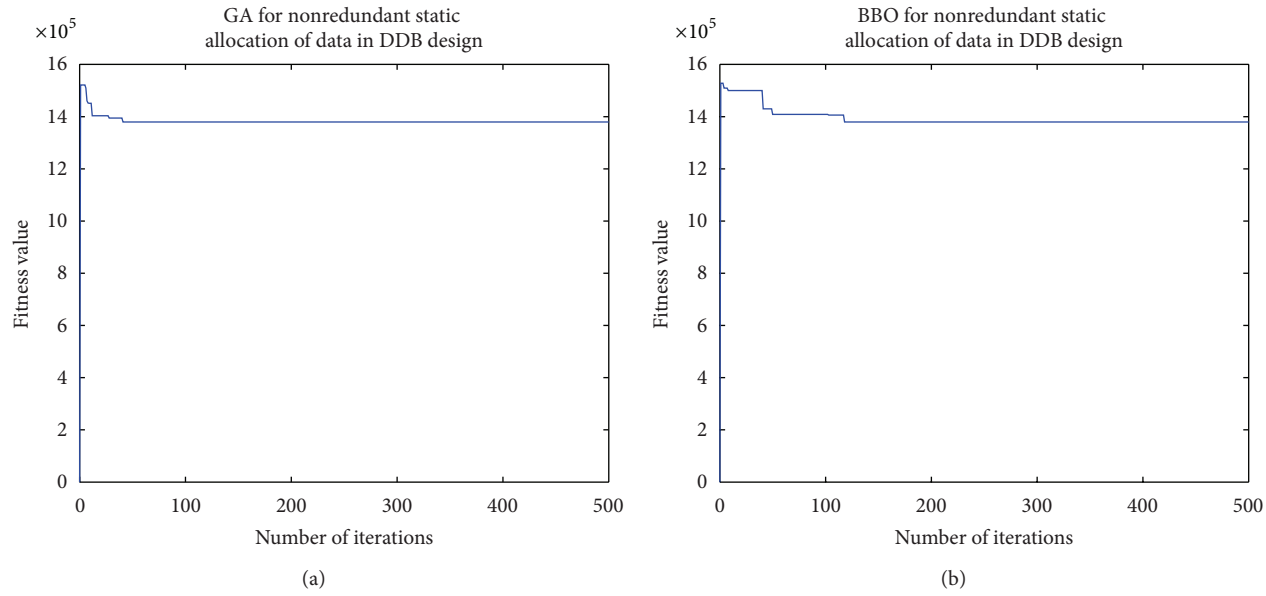

Figure 5: Convergence of GA and BBO for 4 fragments and 8 sites.

a binary string. Each binary string represents a potential solution (habitat) to the fragments allocation problem.

For example, in the case of 3 sites and 5 fragments, a habitat is represented by 5 sets of 2 bit each, one set for each fragment. If fragment 1 is allocated to site 3; fragment 2 is allocated to site 1; fragment 3 is allocated to site 3; fragment 4 is allocated to site 2; fragment 5 is allocated to site 3, then the habitat for these allocated fragments will be [11 01 11 10 11]. Habitat suitability index (HIS) is the total cost of data fragments allocation.

4. Results

To check the performance of the proposed BBO algorithm, it is compared with GA of Ahmad et al. [14]. Different experiments are conducted with number of fragments ranging from 4 to 24 and number of sites fixed as 4 and 8. All the experiments are done on 2.8 GHz Intel Core i5 processor with 4 GB RAM and 64 bit Microsoft Windows 7 as an operating system. BBO and GA algorithms are implemented in the MATLAB 2010 programming environment.

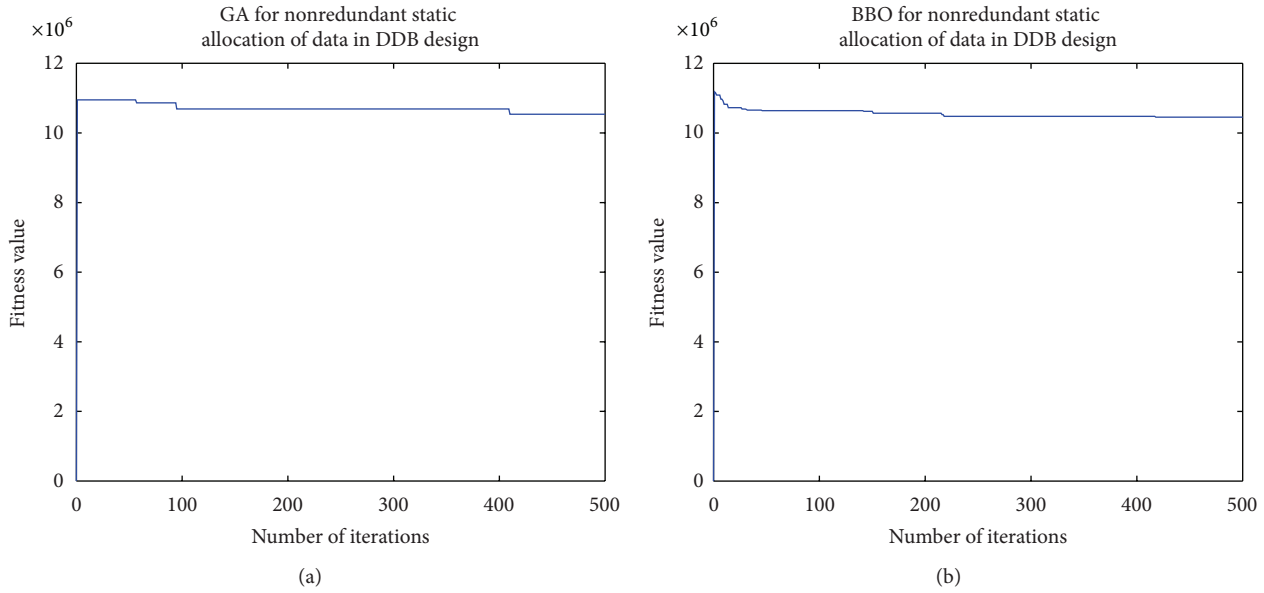

Figure 6: Convergence of GA and BBO for 24 fragments and 8 sites.

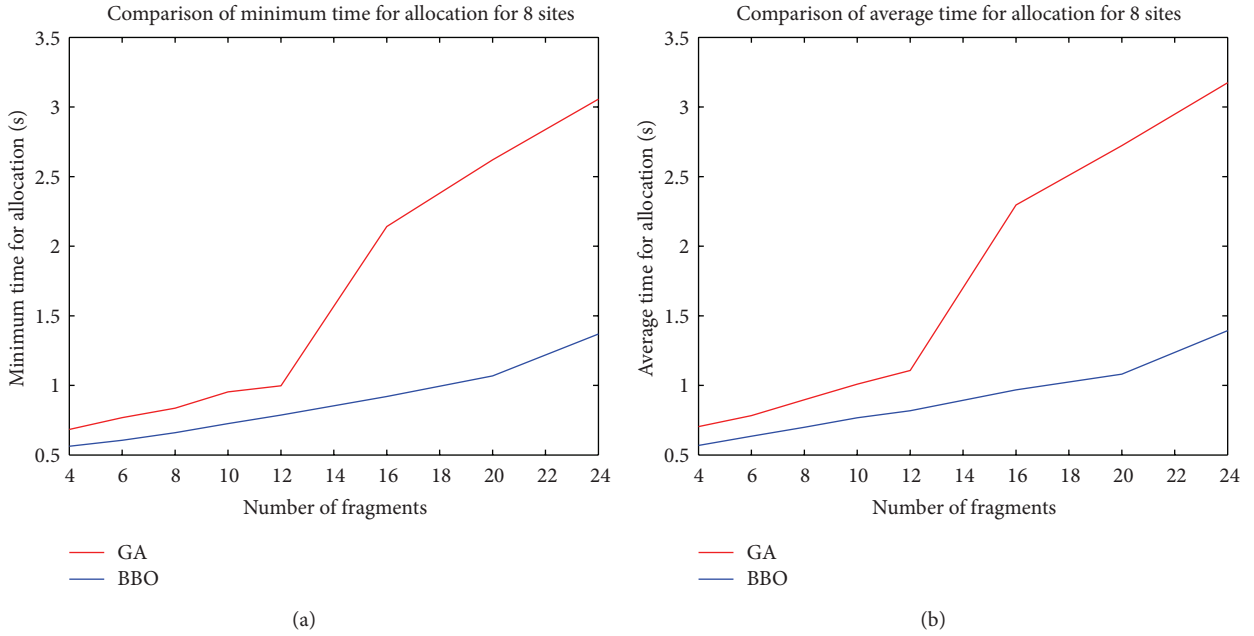

Figure 7: Minimum and average running time of GA and BBO for 8 sites.

The communication network topology, communication cost between sites, the size of fragments, numbers of queries, execution frequency of each query at different site, retrieval frequency of different fragments, and update frequency of different fragments are randomly generated from uniform distributions for each experiment [8, 14, 15]. Both algorithms are tested on the same data set for each experiment and the values of other parameters are given as follows:

maximum number of iterations = 500,

population size = 20,

mutation rate = 0.15,

maximum immigration rate (I) = 1,

maximum emigration rate (E) = 1,

elitism parameter = 2.

Tables 2 and 3 summarize the experimental results obtained from BBO and GA for 4 sites and the number of fragments ranging from 4 to 24. Tables 4 and 5 show the experimental results obtained from BBO and GA for 8 sites and the number of fragments ranging from 4 to 24. These

Step 1: Initialize Population Size, Maximum Number of Iterations (NI), Maximum Immigration rate (I), Maximum Emigration rate (E), Mutation rate, and Elitism Parameter;
Step 2: Generate a random set of habitats based on the size of the population. Each habitat corresponds to a potential solution to the given problem;
Step 3: Evaluate habitats and compute corresponding HSI value of each habitat;
Step 4: For $i = 1$ to NI
Step 5: Calculate the immigration rate (λ) and emigration rate (μ) for each habitat according to HSI of each habitat;
/* Start of Migration */
Step 6: Select non-elite habitat H_i with probability $\propto \lambda_i$ for immigration;
Step 7: if H_i is selected then select H_j with probability $\propto \mu_j$ for emigration;
Step 8: if H_j is selected then randomly select a SIV from H_j;
Step 9: H_i (SIV)$\leftarrow$ H_j (SIV);
Step 10: End if
Step 11: End if
/* End of Migration */
/* Start of Mutation */
Step 12: Select an SIV in H_i with probability based on the mutation rate;
Step 13: if H_i (SIV) is selected then replace H_i (SIV) with a randomly generated SIV;
Step 14: End if
/* End of Mutation */
Step 15: Re-evaluate habitats and compute corresponding HSI value of each habitat;
Step 16: End for

Algorithm 1

results are obtained after running both the algorithms 20 times independently for each experiment.

From Tables 2 and 4, it is clearly evident that the minimum cost achieved by proposed BBO algorithm for allocation of data fragments is less than the minimum cost achieved by GA. In all the experiments, BBO algorithm for fragments allocation is providing allocation schema better than GA based fragments allocation. From Table 3, Figure 4, Table 5, and Figure 7, it is also clear that BBO algorithm for fragment allocation is faster than GA based fragment allocation. But the average cost of fragment allocation for BBO algorithm is more than GA in some cases as shown in Tables 2 and 4. Figures 2, 3, 5, and 6 show the convergence of GA and BBO. In most of the cases convergence rate of BBO is fast as compared to GA. In overall the proposed BBO algorithm for fragment allocation is providing quality solutions in less time.

5. Conclusion

This paper presents a new biogeography-based optimization technique for nonreplicated static allocation of data fragments during the design of distributed database. To evaluate the performance of proposed algorithm, results are compared with GA. From the results, it is clearly evident that the proposed technique for data fragment allocation is providing quality solutions in quick time. The proposed algorithm significantly minimize the data transfer cost during the execution of a set of queries. However, in some cases the average cost of allocation for BBO is more than GA, but for fast running time and quality solution, BBO can be introduced as a capable algorithm for fragment allocation during distributed database design.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] M. Ozsu and P. Valduriez, *Principles of Distributed Database Systems*, Prentice Hall, 2nd edition, 1999.

[2] S. Ceri and G. Pelagatti, *Distribution Databases: Principles Systems*, McGraw-Hill.

[3] C. W. Chu, "Optimal file allocation in multiple computer systems," *IEEE Transactions on Computers*, vol. C-18, no. 10, pp. 885–889, 1969.

[4] R. G. Casey, "Allocation of copies of a file in an information network," in *Proceedings of the Spring Joint Computer Conference (AFIPS '72)*, vol. 40, pp. 617–625, 1972.

[5] K. P. Eswaran, "Placement of records in a file and file allocation in a computer network," in *Proceedings of IFIP Congress*, North-Holland, 1974.

[6] S. Ceri, S. Navathe, and G. Wiederhold, "Distribution design of logical database schemas," *IEEE Transactions on Software Engineering*, vol. 9, no. 4, pp. 487–504, 1983.

[7] P. M. G. Apers, "Data allocation in distributed databases," *ACM Transactions on Database Systems*, vol. 13, no. 3, pp. 263–304, 1988.

[8] R. Sarathy, B. Shetty, and A. Sen, "A constrained nonlinear 0-1 program for data allocation," *European Journal of Operational Research*, vol. 102, no. 3, pp. 626–647, 1997.

[9] A. M. Tamhankar and S. Ram, "Database fragmentation and allocation: an integrated methodology and case study," *IEEE Transactions on Systems, Man, and Cybernetics A*, vol. 28, no. 3, pp. 288–305, 1998.

[10] A. Corcoran and J. Hale, "A genetic algorithm for fragment allocation in a distributed database system," in *Proceedings of the ACM Symposium on Applied Computing*, pp. 247–250, 1994.

[11] 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.

[12] S. March and S. Rho, "Characterization and Analysis of a nested genetic algorithm for distributed database design," *Seoul Journal of Business*, vol. 2, no. 1, pp. 85–120.

[13] 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.

[14] I. Ahmad, K. Karlapalem, Y.-K. Kwok, and S.-K. So, "Evolutionary algorithms for allocating data in distributed database systems," *Distributed and Parallel Databases*, vol. 11, no. 1, pp. 5–32, 2002.

[15] S. Menon, "Allocating fragments in distributed databases," *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 7, pp. 577–585, 2005.

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

[17] S. Rahmani, V. Torkzaban, and A. T. Haghighat, "A new method of genetic algorithm for data allocation in distributed database systems," in *Proceedings of the 1st International Workshop on Education Technology and Computer Science (ETCS '09)*, pp. 1037–1041, March 2009.

[18] H. I. Abdalla, "A synchronized design technique for efficient data distribution," *Computers in Human Behavior*, vol. 30, pp. 427–435, 2014.

[19] A. Brunstroml, S. T. Leutenegger, and R. Simhal, "Experimental evaluation of dynamic data allocation strategies in a distributed database with changing workload," in *Proceedings of the 4th International Conference on Information and Knowledge Management*, 1995.

[20] T. Ulus and M. Uysal, "Heuristic approach to dynamic data allocation in distributed database systems," *Pakistan Journal of Information and Technology*, vol. 2, no. 3, pp. 231–239, 2003.

[21] M. Uysal and T. Ulus, "A threshold based dynamic data allocation algorithm: a Markov chain model approach," *Journal of Applied Sciences*, vol. 7, no. 2, pp. 165–174, 2007.

[22] A. Singh and K. S. Kahlon, "Non-replicated dynamic data allocation in distributed database system," *International Journal of Computer Science and Network Security*, vol. 9, no. 9, 2009.

[23] H. I. Abdallaha, A. A. Amer, and H. Mathkour, "Performance optimality enhancement algorithm in DDBS (POEA)," *Computers in Human Behavior*, vol. 30, pp. 419–426, 2014.

[24] O. Wolfson, S. Jajodia, and Y. Huang, "An adaptive data replication algorithm," *ACM Transactions on Database Systems*, vol. 22, no. 2, pp. 255–314, 1997.

[25] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[26] R. MacArthur and E. Wilson, *The Theory of Biogeography*, Princeton University Press, Princeton, NJ, USA, 1967.

[27] H. Ma, "An analysis of the equilibrium of migration models for biogeography-based optimization," *Information Sciences*, vol. 180, no. 18, pp. 3444–3464, 2010.