Advances in Science and Technology Research Journal
Volume 9, No. 25, March 2015, pages 1–6
DOI: 10.12913/22998624/1917

Research Article

REDUCING TRANSFER COSTS OF FRAGMENTS ALLOCATION IN REPLICATED DISTRIBUTED DATABASE USING GENETIC ALGORITHMS

Navid Khlilzadeh Sourati[1], Farhad Ramezni[2]

[1] Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran, e-mail: mr.khlilzadeh@gmail.com

[2] Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran, e-mail: f.ramezani@gmail.com

Received: 2015.01.26
Accepted: 2015.02.10
Published: 2015.03.01

ABSTRACT

Distributed databases were developed in order to respond to the needs of distributed computing. Unlike traditional database systems, distributed database systems are a set of nodes that are connected with each other by network and each of nodes has its own database, but they are available by other systems. Thus, each node can have access to all data on entire network. The main objective of allocated algorithms is to attribute fragments to various nodes in order to reduce the shipping cost. Thus, firstly fragments of nodes must be accessible by all nodes in each period, secondly, the transmission cost of fragments to nodes must be reduced and thirdly, the cost of updating all components of nodes must be optimized, that results in increased reliability and availability of network. In this study, more efficient hybrid algorithm can be produced combining genetic algorithms and previous algorithms.

Keywords: distributed database, genetic algorithms, communication costs, GA, data segmentation, Fitness, Crossover, node, fragment data.

INTRODUCTION

Advances in networking and database technology in recent decades has led to development of distributed database systems. Data assignment is used in distributed database in order to achieve the objectives. The first objective is to minimize the total cost of transmission for processing and the second objective is to unify implementation strategy. The primary concerns of distributed database systems are fragmentation and allocation of fragments in main database. Data fragment unit can be a file; in this case, allocation subject is file allocation problem that is NP degree which requires fast heuristics in order to produce effective solutions. In addition, the optimal allocation of database fragments are strongly dependent on query execution strategy that have been implemented by distributed database. Fragments allocation problem has been done in many ways, including repetitive and non-repetitive distributed database, in this article, we have discussed this approach combined with genetic algorithm.

RELATED WORKS

Fragments allocation solution can be divided into two categories including static and dynamic and articles related to static method are briefly examined and its advantages and disadvantages are discussed.

STATIC ALLOCATION ALGORITHM

In 2002, Quang Cook & Goode Berg et al., presented a genetic algorithm; fragments can be distributed by this method among sites so that it results in transmission cost reduction. These papers evaluate update costs in order to reduce transmission costs when allocating fragments of

two basic parameters named fragments transmission cost reduction.

Transmission cost reduction

Node that requests a fragment must send its request to a node holding the fragments that do not lead to increased shipping costs.

Update cost

Since the fragments are provided for several sites in each period, then updating the fragment after writing operation on each fragment will be necessary that must be done automatically by the system. Meybodi et al. [2010] presented the genetic algorithm and two considered like previous method factors of reducing the transmission costs as well as updating factor as a parameter for fitness function; another parameter called machine- based learning separated system from other systems.

We will discuss on combining the genetic algorithm with algorithm in distributed database discuss, all sites are formed in a set called $F = \{S1, S2, ..., Sn\}$. Each distributed database is made of an array ArrSizeNode [], each *Si* is determined by its capacity which is the sum of all fragments size S_i = {Fragment 1 + Fragment 2 + …. + Fragment n}.

REQUIREMENT MATRIX

Each fragment may be required for at least one of sites in the near future. Each site need for each fragment will be determined by a matrix called requirement matrix, where *Rij* represents the site *i* need for fragment *j* which does not have this fragment in its local database, then it must put this demand in requirement matrix so that distributed system becomes aware of this practice. For example, the node number 5 has the demand for fragment number 25. In general, this requirement will be displayed by means of an actual amount that is weight, but another way is to use a binary value. Then, row 5 and column 25 will change requirement matrix amount from 0 to 1 (Figure 1).

Transmission Cost Matrix

This matrix contains the cost of fragments transmission from one node to other nodes. Generation function of random numbers is used in order to determine the weight and random value of this function will be greater than 50 and less than 100, what is determined as below.

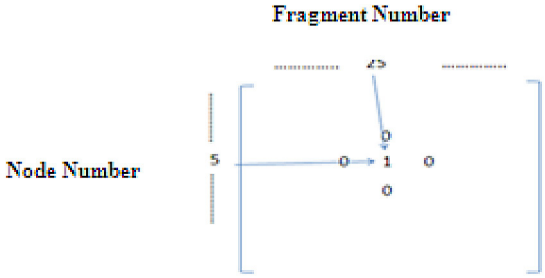

Fig. 1. Requirement Matrix

Rand.Next (50, 100);

It must be noted that according to vast communications of World Wide Web each node can communicate with other nodes that follow from the protocols of distributed system. The cost of transmission from one node to target node does not differ, so in this case we can reduce transmission cost of matrix, and this means that a matrix instead of having rows and columns of the size equal to number of nodes, the matrix can be outlined up triangular or lower triangular.

It must be noted that according to vast communications of World Wide Web each node can communicate with other nodes that follow from the protocols of distributed system. The cost of transmission from one node to target node does not differ, so in this case we can reduce transmission cost of matrix, and this means that a matrix instead of having rows and columns of the size equals to the number of nodes, the matrix can be outlined up triangular or lower triangular.

As shown in the Figure 2, the transmission cost will be 67 in order to transfer fragment from node 1 to node 2. According to the definitions given in previous sections, evaluation formula in order to allocate fragments will be formed of three relationships:

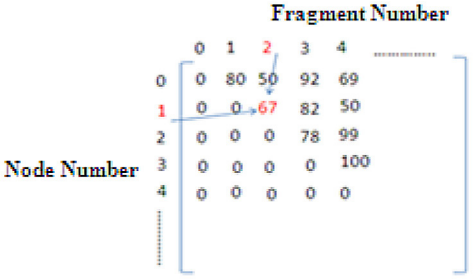

Fig. 2. Transmission Cost Matrix

1) The size of fragment does not exceed the capacity of site.

$$\sum_{j=1}^{n} r_{i,j} s_j \leq c_i \qquad \forall i | 1 \leq i \leq m$$

2) The transmission cost will be optimal.

$$\sum_{i=1}^{m} \sum_{j=1}^{n} r_{i,j} t_{i,p_j}$$

3) The node that will do transmission with the lowest cost of transmission and update.

$$\sum_{i=1}^{i=s} \sum_{j=1}^{j=f} \left[(\min(Transmission(i,k)))required(i,j) + \right.$$
$$\left. + \sum_{l} update(i,l) required(i,j) \right]$$

Chromosomes view in genetic algorithms

1) The function of initial population

In this function, the number of rows is equal to the number of chromosomes and number of columns is equal to number of fragments and the number of genes within chromosomes will be equal to the number of nodes that have used these fragments as well as we have considered the initial population for each generation as 50 (Figure 4).

2) Combinational function

In the combinational function, according to conventional methods of function in this paper, two parents one point method is used for this algorithm, and combinational rate is considered equal to 0.7 (Figure 5).

3) The mutation function

The mutation is a one parent one point method, however, in mutation method random numbers between 0 and 1, are produced using generation function; if this number is equal to 1 it indicates that add a node to nodes having this fragment, but we must not forget one thing and that is whether the node that is going to be owner of this fragment has had it previously or not, and if it is true replace new node, otherwise select

Number of sites including fragment 1	Number of sites including fragment 2	Number of sites including fragment 2	...	Number of sites including fragment n-1	Number of sites including fragment n-2	Number of sites including fragment n

Fig. 3. The structure of a chromosome

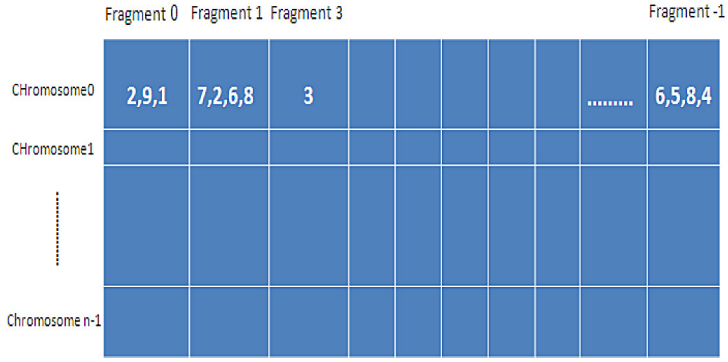

Fig. 4. Generation population

```
for (int i = NP; i < NP + NC; i++)
{
int Split = 0, CHromosomeFirst = 0, CHromosomeSecond = 0;
   Split = Rand.Next(0, NumberFragment);
   CHromosomeFirst = Rand.Next(0, NumberFragment);
   CHromosomeSecond = Rand.Next(0, NumberFragment);
 while (CHromosomeFirst == CHromosomeSecond)
       {
      CHromosomeSecond = Rand.Next(0,NumberFragment);
       }
  crossover(Split, CHromosomeFirst, CHromosomeSecond, i);
}
```

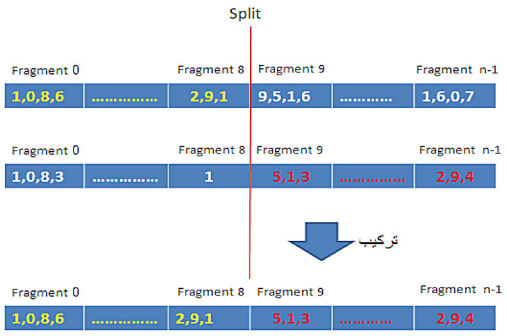

Fig. 5. Operation CrossOver

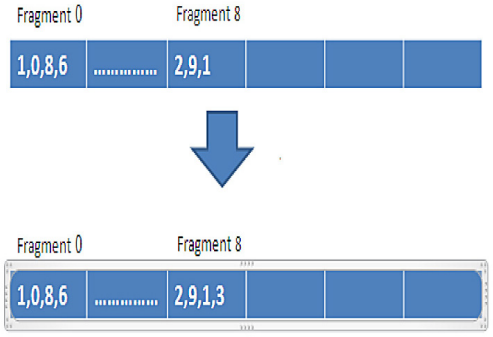

Fig. 6. Operation Mutation

the same node; in below figure, adding node 3 to list of nodes having fragment 8.

But if the generated random value is equal to 0, this indicates that mutation operation will remove a node among the current nodes, then a problem will occur; for solving this problem, it will firstly be checked whether the number of nodes that have this fragment is more than one or not, if this is true, remove the node, otherwise if elimination is done, certainly the availability of distributed system will be disappeared and the system will fail in the near future. It is better to choose another gene from the chromosome and this will be repeated until the problem is resolved and desired result is reached. The mutation rate is considered equal to 0.3. In figure below, removing node 5 from list of nodes having fragment 9 (Figure 7).

Fitness function

In fitness function according to the parameters established in the previous method and were tested, that is transmission cost and update cost, two other parameters have been added it in order to increase efficiency in selection of optimal chromosome. If the node that has the desired fragment fails for any reason, we can restore the node fragments that are provided for other sites. As you know, hardware fragments are not put together in distributed systems, so that they can be repaired, so the fragment and the site will be out of control and availability of the distributed system will be in crisis, and the whole systems may fail; for solving this problem we will use an counter for counting the number of genes in chromosomes that face the problem.

Another idea that was discussed in this paper is when the information is fragmented by a system, it is better to number fragments by the same number which have inter dependency to each other. At this time, nodes that demand these fragments when assigning the fragments, they are asked whether dependent fragments are sent to this fragment or not. If the node accepts the demand, the fragment will be sent to node with related parts. Now, this practice helps all the fragments to be sent in a package to the destination and reduces the cost of transmission. Since the transaction of fragment may need fragments related to the main fragment which reduces the cost of resending fragment. This means, the node that demands fragment will obtain fragments in a package instead of searching twice nodes and paying costs.

Selection function

Choosing the best chromosome is done by tournament from population of chromosomes. Tournament calculates each generation chromosomes according to the main parameters, such as minimum transmission cost, the cost of update, the number of available fragments of a node, and chromosome that can do allocation operation with the lowest cost will be selected as optimal chromosome.

```
for (int i = NP + NC; i < NP + NC + NM; i++)
{
  int Split = 0, Chromosome = 0;
  Split = Rand.Next(0, Number Fragment);
   CHromosome = Rand.Next(0, NP);
   Mutation(CHromosome, Split, i);
   ArrayRequiredUpdate[50, NumberUpdate++]
=  CHromosome.ToString();
 }
```

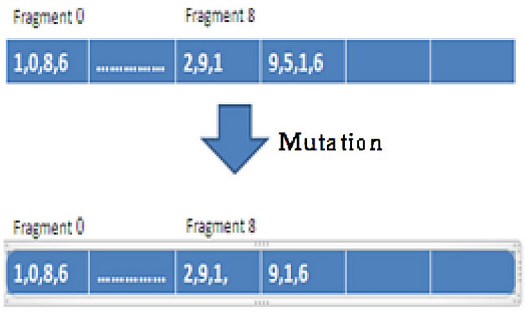

Fig. 7. Operation Mutation

SIMULATION RESULTS

Tests are shown for two proposed measurement factors separately on simulation software and finally, applying these two parameters the transmission cost will be evaluated in the previous and proposed method.

Applying the first measurement parameter of fragments that belong to a node. As the results show, in the proposed GA-F algorithms, a number of fragments that are available for a node are declining and directly effects on availability and reliability (Figure 8).

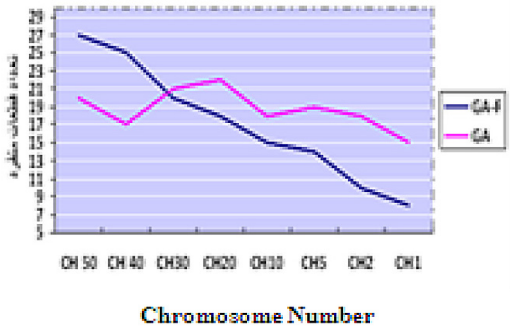

Fig. 8. First measurement parameter

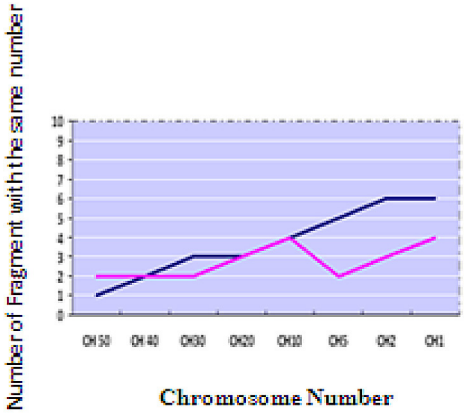

Fig. 9. Second measurement parameter

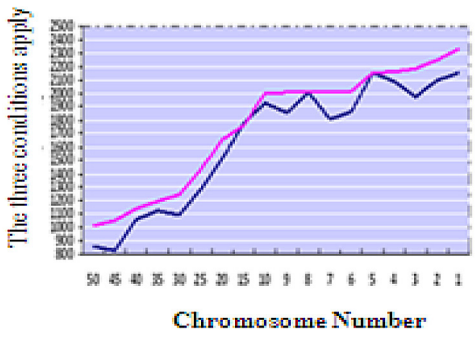

Fig. 10. Evaluating cost of transmission in proposed and previous method

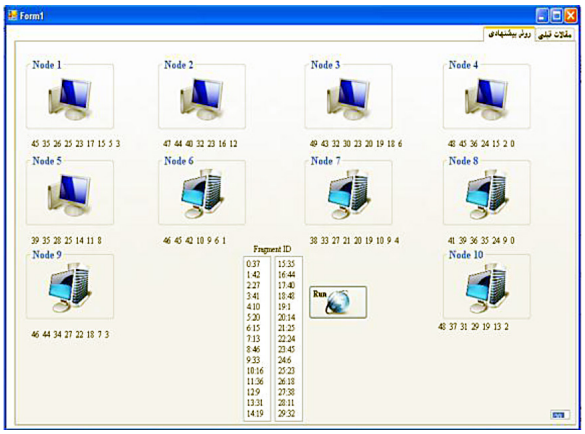

Fig. 11. A view of implemented program

Applying the second parameter of measurement

Number of fragments that are provided for node with determined ID number (Fig. 9, 10, 11).

Recommendations and future works

This paper provides complete descriptions on different methods of reducing transmission cost when assigning fragments of duplicate distributed database using genetic algorithm. In addition, our proposed method could affect the optimality of GA. For future work, we intend to offer our partners to research on the following:

- combining cellular automata with genetic algorithm in order to increase the efficiency,
- examining non-randomized and intelligent methods for initial population of GA,
- using clustering method for this system and combining it with GA.

CONCLUSION

In this paper, we examine the transmission cost reduction when allocating fragments of duplicate distributed database using a genetic algorithm, in addition to previous methods that have implemented this algorithm; we have decided to try advantages of each method and add more effective measurement parameters so that generated output become more effective. Also, we changed some states of genetic algorithm which were specified as hypotheses. As a result, the output of proposed method will show that if the allocation of fragments is done reasonably at the basic steps, they can be very effective in reducing the cost of transmission. We must note that the needed cost for doing this must not be so high.

REFERENCES

1. Hu Y., Chen J., Fragment allocation in distributed database design. Journal of Information Science and Engineering 17, 2001, 491–506.
2. Dokeroglu T., Cosar A., Dynamic programming with ant colony optimization meta heuristic for the optimization of distributed database queries. [In:] Proceedings of the 26th International Symposium on Computer and Information Sciences (ISCIS), London 2011.
3. Lee Z., Su S., Lee C., A heuristic genetic algorithm for solving resource allocation problems.Knowl. Inf. Syst. 5 (4), 2003, 503–511.
4. Schwartz R.A., Kraus S., Negotiation on data allocation in multi-agent environments. Autonomous Agents and Multi-Agent Systems, 5 (2), 2002, 123–172.
5. Chin A.G., Incremental data allocation and reallocation in distributed database systems. Journal of Database Management, 12 (1), 2001, 35–45.
6. Huang Y.F., Chen J.H., Fragment allocation in distributed database design. Journal of Information Science and Engineering, 17 (3), 2001, 491–506.
7. Morgan H.L., Levin K.D., Optimal program and data locations in computer networks. Communications of the ACM, 20 (5), 1977, 315–322.
8. Jin Hyun Son, Myoung Ho Kim, An adaptable vertical partitioning method in distributed systems. The Journal of Systems and Software. Elsevier 2003.
9. Shemshaki M., Shahhoseini H.S., Energy efficient clustering algorithm with multi-hop transmission. IEEE, Scalable Computing and Communications; Eighth International Conference on Embedded Computing, 2009, 459–462.
10. Wai Gen Yee, Donahoo M.J., Shamkant B., Navathe, A framework for server data fragment grouping to improve server scalability in intermittently synchronized databases. CIKM 2000.
11. Chun-Hung Cheng, Wing-Kin Lee, Kam-Fai Wong, A genetic algorithm-based clustering approach for database partitioning. IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews, 32 (3), 2002.
12. Srinivas M., Patnaik L.M., Genetic Algorithms: A Survey. IEEE Computer, 2002, 17–26.
13. An introduction to genetic algorithms. Kanpur Genetic Algorithms Laboratory (KanGAL). Sadhana, 24 (4-5), 1999, 293–315.
14. Basseda R., Fragment allocation in distributed database systems. Database Research Group 2006.
15. Basseda R., Data allocation in distributed database systems. Technical Report No. DBRG. RB-ST. A50715, 2005.
16. Ulus T., Uysal M., Heuristic approach to dynamic data allocation in distributed database systems. Pakistan Journal of Information and Technology 2 (3), 2003, 231–239.
17. Baseda S., Tasharofi M.R., Near neighborhood allocation: A novel dynamic data allocation algorithm in DDB, CSICC, 2006.
18. Safari A.M., Meybodi M.R., Clustering of software systems using new hybrid algorithms. Proc. Int. Conf. on Computer and Information Technology (CIT 2009), Xiamen, China, 2009, 20–25.
19. Oommen B.J., Ma D.C.Y., Deterministic learning automata solutions to the equi partitioning problem. IEEE Trans. on Computers, Vol. 37, 1998, 2–13.
20. Ahmed I., Karlapalem K., Kowok Y.K., Evolutionary algorithms for allocating data in distributed database systems. International Journal of Distributed and Parallel Databases, 11 (1), 2002, 5–32.
21. Chu W.W., Optimal file allocation in a multiple computer system. IEEE Transactions on Computers, C-18 (10), 1969, 885–889.
22. Morgan H.L., Levin K.D., Optimal program and data locations in computer networks. Communications of the ACM, 20 (5), 1977, 315–322.
23. Chu W.W. 1969. Optimal file allocation in a multiple computer system. IEEE Transactions on Computers, C-18 (10), 885–889.
24. Ishfaq Ahmad, Yu-Kwong Kwok, Siu-Kai So, Distributed and parallel databases. Kluwer Academic Publishers, 11, 2002, 5–32,
25. Srinivas M., Patnaik L.M., Genetic algorithms: A survey. Computer, 27 (6), 1994, 17–26.
26. Goldberg D.E. 1989. Genetic algorithms in search, optimization and machine learning. Addison-Wesley: Reading, MA.
27. Hurley S., Taskgraph mapping using a genetic algorithm: A comparison of fitness functions. Parallel Computing, 19, 1993, 1313–1317.
28. Mahfoud S.W., Goldberg D.E., Parallel recombinative simulated annealing: Agenetic algorithm. Parallel Computing, 21, 1995, 1–28.
29. Jing L., Michael K.N., Huang J.Z., Knowledge-based vector space model for text clustering. 2009.
30. McClean S., Scotney B., Shapcott M., Using domain knowledge to learn from heterogeneous distributed databases. Springer-Verlag, Berlin Heidelberg 2004.
31. Xiuxia Yu, Yinghong Dong, Li Yue, A study of optimized algorithm for distributed database half-join query and knowledge engineering. Springer-Verlag, Berlin Heidelberg 2012.
32. Moghaddam H., Mamaghani S., Mahi M., Meybodi M., A novel evolutionary algorithm for solving static data allocation problem in distributed database systems. IEEE 2010.