IASJ المجلات الاكاديمية العراقية

Tikrit Journal of Pure Science

ISSN: 1813 – 1662 (Print) --- E-ISSN: 2415 – 1726 (Online)

Journal Homepage: http://tjps.tu.edu.iq/index.php/j

Data Allocation in Distributed Database based on CSO

Saadi Hamad Thalij

Dept. of Computer Science, Collage of Computer & Math. , Tikrit University, Tikrit, Iraq

https://doi.org/10.25130/tjps.v27i2.66

ARTICLE INFO.

Article history:

-Received: 20 / 1 / 2022

-Accepted: 7 / 3 / 2022

-Available online: / / 2022

Keywords: Distributed database systems, allocation, communication cost, data allocation problem, quadratic assignment problem, chicken swarm optimization.

Corresponding Author:

Name: Saadi Hamad Thalij

E-mail: saadi.alluhaibi@tu.edu.iq

Tel:

ABSTRACT

Distributed databases (DDBs) provide smart processing of large databases, the problems of fragmentation and allocation are vital design problems in addition to the centralized design. The majority of performance degradation in DDBs is due to the communication cost by query remote access and retrieval of data. This can be optimized through an efficient data allocation approach that will provide flexible retrieval of a query by low cost accessible sites. In this paper, a novel high performance data allocation approach is designed using Chicken Swarm Optimization (CSO) algorithm. Data allocation problem (DAP) is a NP-Hard problem modelled as optimization problem. The proposed data allocation approach initially characterizes the DAP into optimal problem of choosing the appropriate and minimal communication cost provoking sites for the data fragments. Then the CSO algorithm optimally chooses the sites for each of the data fragments without creating much overhead and data route diversions. This enhances the overall distributed database design and subsequently ensures quality replication. The experimental results illustrate that the proposed CSO based intelligent data fragment allocation approach has better performance than most existing approaches and thus signifies the impact of efficient data allocation in DDBs.

1. Introduction

The increase in the volume of data in all developmental fields has paved the way for today's big data era. In this era, the storage of larger databases in a single system or machine is a tedious process. The introduction of the distributed databases has given wider flexibility in managing these larger data by storing them in a distributed manner at different machines even located at different geographical locations [1]. These data are stored distributive in master-slave setup and based on load balancing the data are spread across distributed machines connected through a centralized database design. This data can be accessed by the clients through thread managers at the time of query processing [2]. Query processing initiates the data storage machines through the query requests made by the clients whose requirement for specified data must be considered for placing them in geographically near locations for easy access [3]. This can speed up the processing and also reduce the overall execution costs. However, this process is not a simple one as the number users are more than expected and the size of data is also variable [4]. In order to match both the user demands and storage complexities, efficient allocation of these data fragments must be carried out. This lead to the research on developing efficient and capable data fragments allocation models for the DDBs.

Data fragment allocation or simply, data allocation is the process of selecting the most suitable location for placing a data considering the constraints such as access time, query processing, storage availability and security constraints [5], [6]. Hence the DAP is considered as an optimization problem based on constraints. In DDBs, disk drive speed, parallelism of the queries, network traffic, load balancing of servers are mainly considered for designing while some other parameters also considered rarely. Similar to file allocation problem (FAP), DAP is also basically identical to QAP [7] with the only difference is the logical and semantic relations of the fragments in the DAP [8]. Many allocation models have tried their hands in providing efficient solution to DAP, but with the ever increasing big data era, the DAP also

mutated and is vital in degrading the DDB performance. Intelligent optimization algorithms can be an effective solution for the current state of DAP models [9].

In this paper, an intelligent data allocation approach is developed based on the CSO algorithm [10]. In this approach, the DAP is modelled into optimization problem as like QAP and then specified with two kinds of dependencies between transactions and fragments. Based on these dependencies, the DAP is resolved. The CSO algorithm resolves the DAP problem through effective selection of site for allocating the data fragments. The simulations are performed in Hadoop environment to validate the performance of CSO based data allocation. The remainder of this article is organized as follows: Section 2 presents the description of related research works. Section 3 explains the proposed data allocation approach whose evaluations are presented in section 4. Section 5 makes a conclusion about the data allocation approach presented in this article.

2. Related works

Researches in distributed database have been mainly centered on data fragmentation, allocation and replication. The data provision techniques are widely studied by the research community as a means to improve the availability to each cloud users through best site selection for allocating the data. In any distributed database system, the major role in ensuring availability with QoS depends on efficient data fragment allocation strategy. This section provides a discussion on some of the recent researches on data allocation strategies. Data fragment allocation algorithm shave been developed based on time constraints. Mukherjee 2011, [11] proposed refined dynamic fragment allocation process which integrates the time limitations of archive entrances. The capacity onset and the size of data transferred in successive time breaks also engaged to vigorously rearrange fragments at runtime. This model provides low frequency to the unwanted fragment migrations and data transfer over the network during query executions with maximum throughput.

Li & Wong 2013, [12] proposed the use of time series for the DAP problem in DDBs. Primarily, the DAP is exhibited in accessible DDBs and accomplished short-term load predicting (STLP) using time series to dynamically reallocate data fragments. Load balancing and resource saving is achieved using time series based on effective future workloads estimation and minimized fragments migrations. However, this model has shortcomings for the processing in data of large size. Singh 2016, [13] also presented an empirical evaluation of threshold and time constraint system for non-replicated dynamic data provision in DDBs. This approach has better performance for data allocation. Gu et al 2016, [14] presented data allocation approach with least cost under guaranteed likelihood. Abdalla 2012, [15] presented a novel data re-allocation approach for replicated and non-replicated constrained DDBSs. This approach estimates the cost of reallocation and then utilizes the highest query update cost site for migration decision with minimal cost for communication.

Sun et al 2017, [16] suggested a dynamic non-redundant data allocation approach for distributed database systems, in which fragment update parameters and dynamic cost parameters are specified in order to discover the optimal solution for reallocating redundant data. This non-redundant approach takes into account the subsystem's time to allocate jobs in order to tackle the challenge of determining the source of unreliability in reliability task assignment. Solve the estimated optimal solution of the DAP issue using a cost function that defines the unreliability of task execution and a cost function of unreliability induced by processor latency. This model, on the other hand, has a higher data transfer overhead, which affects its efficiency. Lwin & Naing 2018, [17] also proposed a non-redundant dynamic fragment allocation approach with horizontal partition in DDBs. This approach reallocates fragments based on the access patterns made to each fragments with amount of data volume up to time constraint and threshold value. This method lowers the expense of updating the site as well as the cost of storing information, all while increasing the site's response time. This strategy can also handle the problem of several sites qualifying for fragment reallocation in threshold, optimum approaches.

Chen et al 2018, [18] suggested a data allocation scheme with both static and dynamic nature of data center considered for site selection. First the DAP is modelled as optimization problem and minimized the communication cost model is presented. Then the DAP is transformed into a chunk distribution tree (CDT) construction problem and reduced to graph partitioning problem. This approach resolved the DAP with low overhead but the limitation of this model is that it is developed only for single machine model.

Many researches have focussed on developing algorithms that perform both fragmentation and allocation together like Al-Sayyed et al 2014, [19] who suggested a new approach that performs fragmentation and allocation together depending upon the high performance clustering and transaction execution cost functions. In this approach, the data relations are split into fragments and determine which network sites are best for each of the fragments. Though cost is considered negligible for enhancing the allocation, this approach does not support larger network models.

Optimization algorithms especially metaheuristic algorithms have been recently employed significantly to resolve data allocation problem which is a NP-hard problem. Rahmani et al 2009, [20] proposed the use of genetic algorithm (GA) for data allocation in DDBs. Zhao et al 2011, [21] also utilized genetic

algorithm for data allocation in DDBs. Though efficient, this approach has imitations as there are many improved metaheuristic algorithms better than genetic algorithm. Mamaghani et al 2010, [22] modelled the DAP into NP-complete optimization problem and utilized an object migration learning automaton based approach for data fragments allocation. The execution time is minimized while the stability of this DAP algorithm is also significant. Mamaghani et al 2010, [23] employed two techniques of genetic algorithm and learning automata (GA-LA) synchronically for examining the states space of problem. This approach is efficient in solving DAP; the quality of generated solutions has been accelerated. Tosun et al 2013, [24] have proposed a collection of SA, GA and Fast ACO to solve DAP in DDBs. The practice of these procedures escalates the performance in terms of the implementation times and the superiority of the fragment allocation even for very large quantity of fragments and locations. The ideal used for determining the locations where each fragment will be allotted assigns only one fragment to each location and decides the DAP problem. Nevertheless, there are restrictions in allocating multiple fragments to multiple sites in this approach due to the use of higher memory for operations.

Singh et al 2014, [25] presented a new biogeography-based optimization (BBO) algorithm to improve DDBs allocation procedure. This BBO based model considerably reduced the data transfer cost through the implementation of a set of queries. BBO has been preferred for fast running time and quality solution, as per the authors, as a capable algorithm for fragment allocation during DDB design. However, in some cases the average cost of allocation for BBO is more than Genetic algorithm. Mahi et al, [26] presented particle swarm optimization (PSO) algorithm based data allocation scheme to solve the NP-hard problem of DAP. Initially the DAP problem is modelled into NP-hard optimization problem based on QAP and PSO is employed to resolve the issue. PSO-DAP minimizes the query execute time and transaction cost. Even as the problem's dimensionality develops, the performance of the other methods suffers as the solution space expands exponentially. while PSO-DAP provide better performance. However, the slow convergence of PSO in global optima solution is a major concern as the high dimensional data creates the PSO to iterate in local optimum. Apart from these models in literature, there are many optimization algorithms that perform better than GA, PSO, BBO such as firefly optimization, chicken swarm optimization, etc. From these inferences, this research article focuses on solving DAP problem using CSO.

3. Data allocation for distributed databases

The proposed data allocation scheme is developed using a hierarchical swarm based optimization algorithm of Chicken Swarm Optimization. It allocates the distributed data fragments by modelling the DAP problem into the optimization problem.

3.1. Problem Definition

The DAP problem is defined as the problem of finding the best processor to place the data fragments with low transaction cost and delay. It is modelled based on direct and indirect transaction-fragment dependencies. The dependency between the processors and fragments are referred as the transaction-fragment and processor-transaction dependencies.

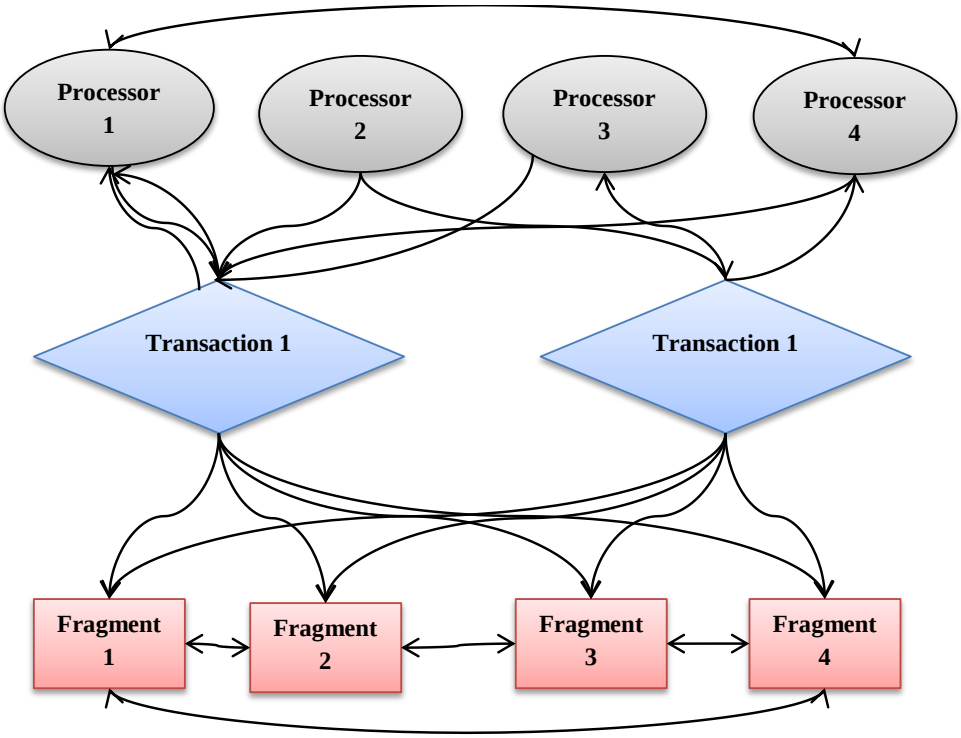

Fig. 1: Dependencies of transactions on fragments and processors on transactions

Figure 1 shows the dependencies of transactions on fragments and processors on transactions on distributed database systems [26]. The cost function is derived as the sum of transaction-fragment dependencies' direct and indirect costs [28]. If there is data transmission from the processor containing *frag* for each execution of *t*, the dependency between *t* and *frag* is called direct. When data must be transferred from a processor other than the transaction's originating processor, the reliance is deemed indirect. The entire cost of distributing data the total cost is the result of adding two costs together. The first and second costs are the same.

$Cost\ (\Phi) = Cost\ 1(\Phi) + Cost\ 2(\Phi)$….(1)

Here Φ denotes the *m* element vector wherever Φj specifies the processor to which $frag$j is allocated. $Cost\ 1$ is denoted by the volume of processor-fragment dependencies which can be expressed by the product of two matrices 1) Matrix that stores processor fragment dependencies (*stfr)* and 2) Matrix that stores the unit communication cost among the processors (*uc)*. The cost of loading a fragment in processor pi is denoted by a partial cost matrix $pcost1n \times m$. The unit partial cost matrix is expressed as

$pcost1_{ij} = \sum_{q=1}^{n} \mathrm{uc}_{iq} \times \mathrm{stfr}_{qj}$….(2)

Based on these parameters, the $Cost\ 1$ can be computed by evaluating the unit $pcost1_{ij}$ for each i and j.

$Cost\ 1(\Phi) = \sum_{j=1}^{m} pcost1_{ij}$….(3)

Similarly, for the computation of $Cost\ 2$, the inter-fragment dependency matrix (*ifdm)* is utilized. The *ifdm* matrix representing the inter-fragment dependency is the multiplication of the matrix *qfr* with the matrix *q*. It is given by

$ifdm = qfr_{l\times m\times n} \times q_{l\times m\times n}$ ….(4)

Where matrix *qfr* represents the execution frequencies of the transactions and *q* denotes the indirect transaction fragment dependency. Based on this matrix, $Cost\ 2$ is derived.

$Cost\ 2(\Phi) = \sum_{j1=1}^{m} \sum_{j2=1}^{m} \mathrm{ifdm}_{j1j2} \times \mathrm{uc}_{\Phi_{j1}\Phi_{j2}}$….(5)

Hence the DAP can be modelled as optimization problem by combining these cost values.

$Cost\ (\Phi) = \sum_{j=1}^{m} pcost1_{ij} + \sum_{j1=1}^{m} \sum_{j2=1}^{m} \mathrm{ifdm}_{j1j2} \times \mathrm{uc}_{\Phi_{j1}\Phi_{j2}}$ ...(6)

3.2. Chicken Swarm Optimization algorithm for DAP solution

CSO algorithm has been developed based on the behaviour of the chicken flocks by Meng et al, [10]. The hierarchy will be in the order of head rooster, other roosters, and hens with their chicks [28]. According to the literature study, it has been found that PSO has already been utilized for solving DAP [26] since PSO has a smaller amount of control parameters, features of speed convergence and lower consuming time, robustness against to solution space of the optimization problems. However, there are many optimization algorithms especially swarm optimization models that are way better than PSO. CSO has been found to be more effective for the DAP solution due to the hierarchical order of the problem formation. In the initialization phase, considered in a virtual search space, there is an X-sized chicken swarm, with each rooster functioning as the agent of its cluster. The agent is in charge of tracking the chicken's location and movement. Assume that the

numbers RX, HX, CX, and MX, respectively, reflect the number of roosters, hens, chicks, and mother hens.

The best fitness chickens will compete for the head while those with the worst fitness will be at the bottom of the food table. All X virtual chickens, depicted by their positions $x_{i,j}^t$ $(i \in [1,\dots,X], j \in [1,\dots,D])$ at time step t, search for food in a D-dimensional space. The location and movement of the rooster can be updated based on [10].

$$x_{i,j}^{t+1} = x_{i,j}^t * \left(1 + Randn(0,\sigma^2)\right) \dots\dots (7)$$

$$\sigma^2 = \begin{cases} 1, & if\ f_i \le f_k, \\ \exp\left(\frac{(f_k - f_i)}{|f_i| + \varepsilon}\right), & otherwise, \end{cases} \quad k \in [1,X], k \ne i \ \dots(8)$$

The Gaussian distribution Randn$(0,\sigma^2)$ has a mean of 0 and a standard deviationσ^2. The smallest constant in the computer is ε, which is utilized to avoid zero-division-error. A rooster's index, k, is chosen at random from the group of roosters, and f is the fitness value of the corresponding x.

The dominant hens' location and movement can be updated as

$$x_{i,j}^{t+1} = x_{i,j}^t + S1 * Rand * \left(x_{r1,j}^t - x_{i,j}^t\right) + S2 * Rand * (x_{r2,j}^t - x_{i,j}^t) \dots (9)$$

$$S1 = \exp(f_i - f_{r1}) / (abs(f_i) + \varepsilon)) \dots\dots (10)$$

$$S2 = \exp(f_{r2} - f_i) \dots\dots\dots (11)$$

Rand is a uniform random number ranging from 0 to 1. r1 [1... X] is the index of the rooster who is the *i*-th hen's group-mate, and r2 [1,...,X] is the index of the chicken (rooster or hen) who is randomly selected from the swarm. r1≠r2.

The location and movement of the chicks around the mother hens can be updated as

$$x_{i,j}^{t+1} = x_{i,j}^t + FL * (x_{m,j}^t - x_{i,j}^t) \dots\dots (12)$$

The position of i-th chick's mother (m [1, X]) is represented by. is a parameter, which indicates that the chick would track its mother to forage for food. Considering the individual differences, the FL for each chick would be randomly selected between [0, 2]. The optimization model is evaluated using the convergence metric and the result demonstrates that CSO-based clustering has greater convergence.

Initially, each of the processors or processors is determined where the fragments are needed to be positioned. For minimizing the total cost of transaction, fragments are positioned at the best processors. The process of CSO into DAP problem is given in the following.

Step 1: Initialize N processors, data fragments and memory resources

Step 2: Set initial iterations and start the CSO

Step 3: Evaluate the fitness for each agent using $f_{ij} = Cost\ (\Phi)_{ij}$

Step 4: Rank processors based fitness in hierarchical order

Step 5: Cluster the sorted processor set into groups (server, sub-server or client)

Step 6: Update the position and status of each processor

Step 7: Allocate data fragments based on sorted processor list

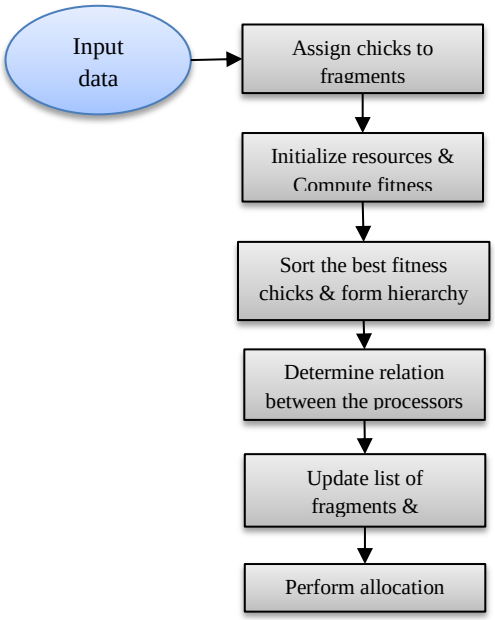

Fig. 2: CSO based DAP approach

The process of the proposed CSO for DAP solution is given in Figure 2. To resolve DAP, the fragment placement must be optimally determined. In the proposed CSO based DAP model, the processors are initialized as chickens and the parameters are defined. The capacity of each processor is analysed prior to applying CSO to ensure there is no mistakes in the fitness value calculation. The fitness for each processor is the cost function which is computed based on Eq.12 and mapped into CSO objective and update functions. Based on these fitness values, the N processors are sorted in the most dominating order. Each processor has to take either the roles of server, sub-server or clients which means the worst fitness processors are designated as clients. Based on this order, the clusters are formed with only one server, more than one sub-server and many clients. This clustering process act as determinant in the locations for placing the data fragments. At each iteration, the best location is determined by the food location expressions in CSO. Based on this, the location for placing these fragment allocations are finalized and the order is updated. This solution acts as the DAP resolution until the next iteration. At the maximum iterations, the best location for each fragment is finalized with minimum transaction cost.

4. Performance evaluation

4.1. Experimental Environment

The experiments are performed on the Hadoop cluster environment consisting of 16 dedicated machines (1 master and 15 slaves). This structure is further assigned into 3 sub-servers and 12 clients among the slave nodes. The master node (mother server) acts as both the namenode as well as datanode. Different

experiments were conducted with 4 to 24 fragments over fixed number of sites as 4 and 8. The experiments were conducted on Intel Core i5 processor with 4GB RAM and 64bit Windows 10 operating system where java 1.8 was installed on Hadoop framework.

The communication topology, fragments size, number of queries, the cost of communication between the sites, execution time and frequency, data retrieval frequency, number of update frequency of fragments in different sites and the initial processing speed are arbitrarily created for simplicity from the uniform distributions of the datasets in the experiments. The initial number of iterations starts from 1 and the maximum iterations is set as 500. The population size of the CSO is set as 20 with its learning rate set at 0.25. The control parameter value is set as 2 to regulate the search process. The dataset utilized are extracted from three major sources, Twitter, Facebook and YouTube containing various types of files namely text, audio and video files with varying sizes from 500KB to 500MB. A total workload of 275 queries that includes a set of 150 simple elements selection of nine attributes in the utilized dataset of Facebook, Twitter and YouTube is established.

4.2. Performance Metrics

Average Cost: It includes the processing cost, communication cost and system operating costs. It is incurred by a model to ensure the sufficient utilization of a system without over-exploitation or resource wastage.

Execution time: It includes the time taken for processing the fragments and selection of best location for allocating those fragments.

4.3. Comparison results and discussion

The proposed CSO-based data allocation method's performance is compared with the existing models namely GA [20], GA-LA [23], BBO [25] and PSO [26]. The comparisons of these models are obtained for varying number of fragments and varying number of sites. Table 1 shows the execution time and average costs of the implemented models for 4 sites with number of fragments ranging from 8, 16 and 24 fragments.

Table 1: Performance comparison of Data allocation models for 4 sites

Methods	Average cost ($)			Execution time (seconds)		
	8	**16**	**24**	**8**	**16**	**24**
GA	5.98	9.23	13.5	9.42	15.67	23.45
GA-LA	6.11	11.4	14.97	9.67	15.18	22.33
BBO	5.25	8.96	12.67	7.86	12.11	20.47
PSO	4.72	7.67	11.48	7.11	11.5	19.16
CSO	3.91	6.12	9.89	6.85	9.97	17.38

The comparative results of average cost and execution time are plotted in Figures 3 and 4, respectively for the scenario containing 4 sites.

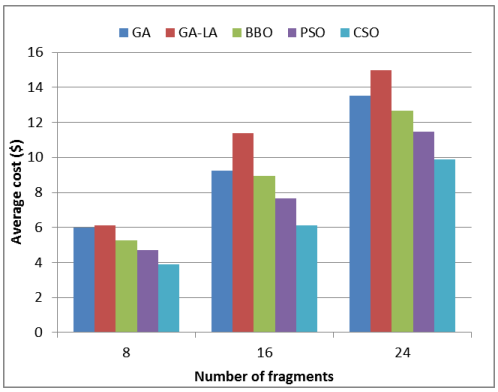

Fig. 3: Average cost for 4 sites

Figure 3 shows the average cost comparison of the proposed CSO against the existing models for 4 sites. The plot of system cost is against the number of fragments and it shows that the proposed CSO based method has reduced the average cost without increasing any liabilities or complexities. The utilization of hierarchical optimization of the CSO has limited the cost under justifiable levels. Even when the number of fragments increases, the gradually linearly increased average cost is comparatively less in CSO. For a maximum of 24 fragments, CSO has average cost of 9.89$ which is 13.85%, 21.94%, 33.93% and 26.74% lesser cost than incurred by the PSO, BBO, GA-LA and GA based allocation models, respectively.

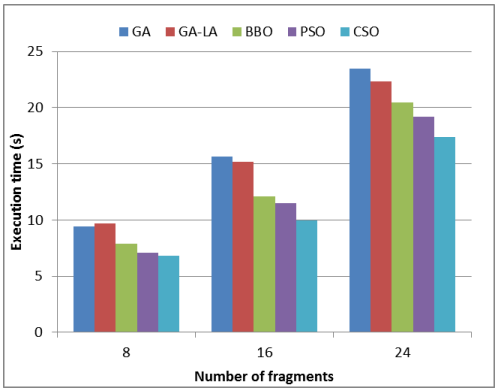

Fig. 4: Execution time for 4 sites

Figure 4 shows the execution time comparison of the proposed CSO against the existing PSO, BBO, GA-LA and GA models for 4 sites. The execution time for completing this data allocation process is evaluated from the beginning of data allocation (i.e. the end of fragmentation) to the end of allocation of all fragments. The plot shows that the proposed CSO based method has reduced the execution time for different number of fragments and provided faster convergence due to the minimum iterations of CSO. For a maximum of 24 fragments, CSO has average cost of 17.38 seconds which is 9.3%, 15.1%, 22.17% and 25.88% lesser execution time than incurred by the PSO, BBO, GA-LA and GA based allocation models, respectively.

Table 2 shows the execution time and average costs of the implemented models for 8 sites with number of fragments ranging from 8, 16 and 24 fragments.

Table 2: Performance comparison of Data allocation models for 8 sites

Methods	Average cost ($)			Execution time (seconds)		
	8	16	24	8	16	24
GA	8.97	13.845	21.25	14.13	23.505	31.175
GA-LA	9.165	17.1	21.455	14.505	22.77	31.495
BBO	7.875	13.44	17.005	11.79	18.165	29.705
PSO	7.08	11.505	15.22	10.665	17.25	27.74
CSO	5.865	9.18	13.835	10.275	14.955	24.07

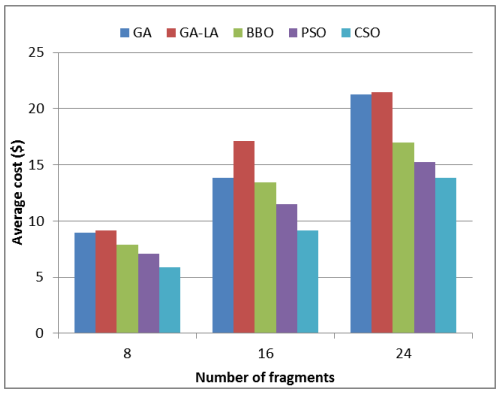

Fig. 5: Average cost for 8 sites

Figure 5 shows the average cost comparison of the proposed CSO against the existing models for 8 sites. Even when the number of fragments increases, the gradually linearly increased average cost is comparatively less in CSO. For a maximum of 24 fragments, CSO has average cost of 14.835$ which is 9.09%, 18.64%, 35.52% and 34.89% lesser cost than incurred by the PSO, BBO, GA-LA and GA based allocation models, respectively.

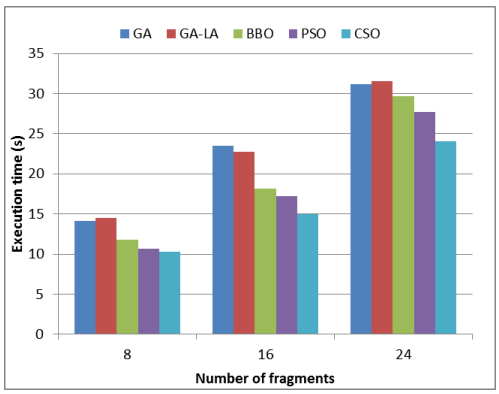

Fig. 6: Execution time for 8 sites

Figure 6 shows the execution time comparison of the proposed CSO against the existing PSO, BBO, GA-LA and GA models for 8 sites. For a maximum of 24 fragments, CSO has average cost of 24.07 seconds which is 13.23%, 18.97%, 23.58% and 22.79% lesser execution time than incurred by the PSO, BBO, GA-LA and GA based allocation models, respectively. The evaluation results prove that the presented data allocation model using CSO algorithm is highly efficient than the other models and suitable for different data allocation applications.

5. Conclusion

Data fragment allocation is a vital problem in distributed database design. Introduction of optimization based allocation algorithms have improved the performance of DAP solution. Based on this ideology, CSO based DAP solution is proposed to develop an intelligent data allocation approach for DDBs. The DAP problem has been modeled as an optimization problem based on QAP concept and cost based optimization of CSO is employed. The experimental results show that the proposed CSO based data allocation approach outperforms other existing schemes with 26.74% lesser average cost and 25.88% less execution time for 4 sites while achieved 34.89% less average cost and 22.79% lesser execution time for 8 sites, thus proving its efficiency in data allocation problem solving of DDB design issues. In future, other advanced metaheuristic algorithms will be tested for solving DAPs. The utilization of other additional parameters in defining the fitness of the processors is also being considered for future work.

References

[1] Özsu, M. T., & Valduriez, P. (2011). "*Principles of distributed database systems*." Springer Science & Business Media.

[2] Elmasri, R., & Navathe, S.(2010). "*Fundamentals of database systems*." Addison-Wesley Publishing Company.

[3] Coronel, C., & Morris, S. (2016). "*Database systems: design, implementation, & management*." Cengage Learning.

[4] Virk, R. S., & Singh, D. G. (2011). "Optimizing access strategies for a distributed database design using genetic fragmentation." *IJCSNS International Journal of Computer Science and Network Security*, *11*(6).

[5] Hababeh, I., Bowring, N., & Ramachandran, M. (2005). "A method for fragment allocation design in the distributed database systems." In *The Sixth Annual UAE University Research Conference*.

[6] Basseda, R., Tasharofi, S., & Rahgozar, M. (2006). "Near neighborhood allocation (nna): A novel dynamic data allocation algorithm in ddb." In *proceedings of 11th Computer Society of Iran Computer Conference (CSICC2006), Tehran*.

[7] Loiola, E. M., de Abreu, N. M. M., Boaventura-Netto, P.O., Hahn, P., & Querido, T.(2007). "A survey for the quadratic assignment problem. " *European journal of operational research*, *176*(2), 657-690.

[8] Cela, E. (2013). "*The quadratic assignment problem: theory and algorithms"* (Vol. 1). Springer Science & Business Media.
[9] Tosun, U. (2014). "Distributed database design using evolutionary algorithms." *Journal of Communications and Networks*, *16*(4), 430-435.
[10] Meng, X., Liu, Y., Gao, X., & Zhang, H. (2014). "A new bio-inspired algorithm: chicken swarm optimization." In *International conference in swarm intelligence* (pp. 86-94). Springer, Cham.
[11] Mukherjee, N. (2011). "Synthesis of non-replicated dynamic fragment allocation algorithm in distributed database systems." *ACEEE Int. J. on Information Technology*, *1*(01).
[12] Li, S. P., & Wong, M. H. (2013). "Data allocation in scalable distributed database systems based on time series forecasting." In *2013 IEEE International Congress on Big Data (BigData Congress)*, (pp. 17-24). IEEE.
[13] Singh, A. (2016). "Empirical Evaluation of Threshold and Time Constraint Algorithm for Non-replicated Dynamic Data Allocation in Distributed Database Systems." In *Proceedings of the International Congress on Information and Communication Technology* (pp. 131-138). Springer, Singapore.
[14] Gu, S., Zhuge, Q., Yi, J., Hu, J., & Sha, E. H. M. (2016). "Data allocation with minimum cost under guaranteed probability for multiple types of memories." *Journal of Signal Processing Systems*, *84*(1), 151-162.
[15] Abdalla, H. I. (2012). "A new data re-allocation model for distributed database systems. "*International Journal of Database Theory and Application*, *5*(2), 45-60.
[16] Sun, Q., Deng, B., Fu, L., & Sun, J. (2017). "Non-redundant Distributed Database Allocation Technology Research." In *2017 International Conference on Computing Intelligence and Information System (CIIS)*, (pp. 155-159). IEEE.
[17] Lwin, N. K. Z., & Naing, T. M. (2018). "Non-Redundant Dynamic Fragment Allocation with Horizontal Partition in Distributed Database System." In *2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)* (Vol. 3, pp. 300-305). IEEE.
[18] Chen, W., Paik, I., Li, Z., & Yen, N. Y. (2018). "A cost minimization data allocation algorithm for dynamic datacenter resizing." *Journal of Parallel and Distributed Computing*, *118*, 280-295.
[19] Al-Sayyed, R. M., Al Zaghoul, F. A., Suleiman, D., Itriq, M., & Hababeh, I. (2014). "A new approach for database fragmentation and allocation to improve the distributed database management system performance." *Journal of Software Engineering and Applications*, *7*(11), 891.
[20] Rahmani, S., Torkzaban, V., & Haghighat, A. T. (2009). "A new method of genetic algorithm for data allocation in distributed database systems." In *First International Workshop on Education Technology and Computer Science, 2009. ETCS'09.* (Vol. 1, pp. 1037-1041). IEEE.
[21] Zhao, C., Gao, J., & Ma, X. (2011). "A Genetic Algorithm for Data Allocation of Distributed Database System in Delay-Sensitive Network. " *Journal of Information &Computational Science*, *8*(15), 3737-3745.
[22] Mamaghani, A. S., Mahi, M., & Meybodi, M. R. (2010). "A learning automaton based approach for data fragments allocation in distributed database systems." In *2010 IEEE 10th International Conference on Computer and Information Technology (CIT)*, (pp. 8-12). IEEE.
[23] Mamaghani, A. S., Mahi, M., Meybodi, M. R., & Moghaddam, M. H. (2010). "A novel evolutionary algorithm for solving static data allocation problem in distributed database systems." In *2010 Second International Conference on Network Applications, Protocols and Services*(pp. 14-19). IEEE.
[24] Tosun, U., Dokeroglu, T., & Cosar, A. (2013). "Heuristic algorithms for fragment allocation in a distributed database system." In *Computer and Information Sciences III* (pp. 401-408). Springer, London.
[25] Singh, A., Kahlon, K. S., & Virk, R. S. (2014). "Nonreplicated static data allocation in distributed databases using biogeography-based optimization.' *Chinese Journal of Engineering*, *2014*.
[26] Mahi, M., Baykan, O. K., & Kodaz, H. (2018). "A new approach based on particle swarm optimization algorithm for solving data allocation problem." *Applied Soft Computing*, *62*, 571-578.
[27] Grillo, R. (2014). 'Chicken Behavior: An Overview of Recent Science."
[28] Smith, C.L., & Zielinski, S.L. (2014). "The Startling Intelligence of the Common Chicken." Scientific American 310(2)..

توزيع البيانات في قواعد البيانات الموزعة معتمدا على خوارزمية التحسين في سرب الدجاج

سعدي حمد ثلج

قسم علوم الحاسوب ، كلية علوم الحاسوب والرياضيات ، جامعة تكريت ، تكريت ، العراق

الملخص

توفر قواعد البيانات الموزعة معالجة ذكية لقواعد البيانات الكبيرة ,وتعد مشكلتي تجزئة البيانات وتوزيعها على المواقع من مشاكل التصميم الحيوية بالإضافة الى التصميم المركزي.

يرجع تدهور الاداء لقواعد البيانات الموزعة الى تكلفة الاتصال للوصول الى البيانات عن بعد واستعلام البيانات واسترجاعها.

حيث يمكن تحسين ذلك من خلال نهج فعال لتخصيص البيانات الذي يوفر استرجاعا مرنا لاستعلام البيانات عن طريق مواقع مخصصه ومنخفضة التكلفة ويمكن الوصول اليها بسهولة. في هذا البحث , تم تصميم أسلوب جديد لتخصيص البيانات عالي الاداء باستخدام خوارزمية تحسين تمسى (سرب الدجاج) (Chicken Swarm Optimization (CSO) algorithm) , أن مشكلة تخصيص البيانات Data Allocation Problem (DAP) هي مشكلة hard polynomial-time non-deterministic (NP-Hard) وهي من المشاكل الاكثر صعوبة في تحديد المواقع لمشكلة التحسين. نهج تخصيص البيانات المقترح في البداية يميز DAP في المشكلة المثلى لاختيار المواقع المناسبة والحد الادنى من تكلفة الاتصال استفزاز لأجزاء البيانات ثم تختار خوارزمية(CSO) المواقع على النحو الامثل لكل جزء من اجزاء البيانات دون انشاء الكثير من التكلفة وتحويل مسار البيانات. هذا يعزز التصميم الشامل لقاعدة البيانات الموزعة ويضمن بالتالي تكرار الجودة.

توضح النتائج التجريبية ان النهج المقترح لتخصيص اجزاء البيانات الذكية المستند الىCSO يتمتع بأداء افضل من معظم الاساليب الحالية , وبالتالي يشير الى تأثير التخصيص الفعال للبيانات في قواعد البيانات الموزعة.