استخدام خوارزمية الحشرات الضوئية مع عوامل الخوارزمية الجينية للبحث عن البيانات المخزنة في قواعد البيانات الموزعة

م. إناس عدي د. م. علي دياب

(الإيداع: 24 نيسان 2024، القبول 11 تموز 2024)

الملخص:

مع تقدم شبكات الكمبيوتر وتزايد عدد مصادر البيانات وكمية البيانات بسرعة كبيرة في السنوات الأخيرة أدت اللامركزية في قواعد البيانات إلى تطوير قاعدة البيانات الموزعة على أجهزة متعددة حيث يكون توزيع قاعدة البيانات شفافًا للمستخدمين، يفرض توزيع البيانات هذا تحديًا على معالجة استفسارات المستخدم فإن الاستراتيجية ضرورية لإنتاج خطط استعلام مثالية في أنظمة قواعد البيانات الموزعة، في هذا البحث قمنا باستخدام خوارزمية (FAGA) وهي عبارة عن دمج لخوارزمية اليراعة المضيئة (FA)مع عوامل الخوارزمية الجينية (GA) وهي الاختيار والطفرة والتقاطع في مرحلة وضع اليراع من معيار(FA) تم اختبار أداء النهج المقترح على قاعدة البيانات وتمت مقارنتها بخوارزمية مستعمرة النحل الاصطناعي(ABC) والخوارزمية الجينية (GA) و خوارزمية اليراع المضيئة (FA) من حيث الكلفة مقابل عدد التكرارات والعلاقات والاستعلامات وتبين من خلال النتائج والمقارنة أنها قادرة على انشاء خطط الاستعلام الموزعة وبكلفة أقل نسبيًا لمعالجة استعلام موزع

تمت محاكاة عملية البحث عن البيانات ضمن قواعد البيانات الموزعة باستخدام بيئة MATLAB.

الكلمات المفتاحية: قواعد البيانات الموزعة، خوارزميات البحث،الخوارزمية الجينية(GA) ،خوارزمية اليراع المضيئة(FA) ، خوارزمية مستعمرة النحل الاصطناعي(ABC)، استخدام خوارزمية اليراع المضيئة مع عوامل الخوارزمية الجينية(FAGA).

* ماجستير علوم ويب – اختصاص شبكات- كلية الهندسة المعلوماتية - الجامعة الافتراضية السورية.

** أستاذ – قسم التحكم الآلي والحواسيب- كلية الهندسة الميكانيكية والكهربائية – جامعة البعث.

Using The Firefly Algorithm With Genetic Algorithm Operators To Search For Data Stored In Distributed Databases.

Eng.Inas Adi * Dr.Eng.Ali Diab**

(Received: 25 January 2024,Accepted: 23 July 2024)

Abstract:

Distributed query processing entails accessing data from multiple sites. keeping in mind that the distribution of the database should be transparent to user .In addition to the usual disk IO and CPU costs, the cost due to transmission of data between different sites, referred to as the site-to-site communication cost, also exists. This cost, being the major cost, needs to be reduced in order to improve the response time for distributed queries. One way to reduce this communication cost is by devising a distributed query processing strategy that involves fewer number of sites for answering the distributed queries.

In this paper, a distributed query plan generation (DQPG) algorithm based FAGA algorithm, which is a combination between the luminous Firefly Algorithm(FA) and the Genetic Algorithm (GA), which generates distributed query plans that involves less number of sites and have higher relation concentration in the participating sites, is presented. Additionally, the experimental comparison of the FAGA algorithm with the GA,FA and artificial bee colony (ABC) algorithms in terms of cost rate algorithm exhibits that the former is able to generate comparatively better quality top-K query plans for a given distributed query.

Keywords: Distributed databases, search algorithms, genetic algorithm, firefly algorithm, artificial bee colony algorithm, mix between firefly algorithm and operator genetic algorithm.

*** Master of Web Science - Networks - Faculty of Informatics Engineering - Syrian Virtual University.**

**** Professor - Department of Automatic Control and Computers - Faculty of Mechanical and Electrical Engineering - Al-Baath University.**

1- INTRODUCTIO:

Distributed databases are essential in the current business environment, in which data spread across multiple geographical locations is connected by a computer network. An effective distributed database management system (DDBMS) is required to manage and retrieve data from such disparate data sources[1]. DDBMS provides a simple, unified interface to the user, which appears like a single database instead of disparate databases. Query processing plays a crucial role in the performance of DDBMS[2]. The query processing problem is far more challenging in a distributed environment, since a distributed query may involve relations that are fragmented and/or replicated across multiple distinct sites leading to the incurring of site-to-site data transmission costs. In order to address this problem, an optimal .Distributed Query Processing (DQP) strategy has to be devised. In DQP, the user queries are analyzed and transformed into a set of data manipulation operations. In DDBMS, first the user query is passed to a query parser, where the user query is parsed and the corresponding relational algebraic expression is generated. Using the query optimizer, an optimal relational algebraic expression, comprising an effective query processing plan, is generated. The user query is broken into sub-queries and the execution plan is devised for all such sub-queries, each of which is processed in parallel at their respective sites. The results of these sub-queries are thereafter integrated to produce the final result for the user query. Thus, the goal of the DQP strategy is to generate query processing plans in a manner that reduces the data transmission cost, which would substantially reduce the total cost of processing a distributed query. This query processing cost comprises of local processing costs and site-to-site data transmission cost.

In DQP, the data transmission cost is more significant than the local processing cost. In order to make distributed query processing more efficient, communication of data between sites needs to be minimized in order to reduce the site-to-site data transmission cost[3].

2- The research objective:

The research aims to develop a mechanism for searching data in distributed large databases stored in a distributed manner in cloud computing to achieve optimal utilization of it.

The developed method should surpass the gaps and flaws of the old methods and consider that the data storage location may change due to changes in the network and cloud environment.

3- Literature review:

The cloud service is defined as an application that the customer obtains via the internet and uses when needed through a browser without the need to install the application on their local device[1][2].
The concept of distributed databases has been defined as the deployment of data across multiple geographically distributed locations and linking them to a computer network instead of using a central database to store the data, with each site processing the data independently and transparently[3][4].
R esearchers used genetic algorithms and their elements of selection, crossover, and mutation to generate query plans with lower cost for searching data within distributed databases[8][7][6][5].
The researchers also used the bee algorithm to search for data stored within distributed databases at a lower cost in order to achieve optimality for the access plan to user queries distributed across network sites[11][10][9].
The researchers discussed the concept of the firefly algorithm and its parameters such as light intensity, attractiveness, distance, and movement[12][16].
The researcher proposed in his research paper the Firefly Algorithm (FA algorithm) to achieve optimality for the access plan for distributed queries at locations in the network [13].
In 2019, the researcher presented the integration of the Firefly Algorithm with the genetic algorithm operators to find the optimal solution [14][15].

4- Distributed database search techniques:

4-1 Genetic Algorithm:

Genetic algorithm (GA) is a search and optimization algorithm that follows the natural evolutionary process according to which living organisms adapt themselves to changes in the environment[3]. GA consists of encoding schemes, fitness function, and selection of parent's, genetic operators (crossover, mutation and inversion) . The fitness value of each chromosome in the population, using the fitness function, is evaluated. The fitter individuals are then selected for crossover and mutation to arrive at the population for the next generation. GA explores the entire solution space to arrive at an optimal set of chromosomes[4].

4-2 Artificial Bee Colony Algorithm:

The artificial bee colony algorithm (BCO) is an optimization algorithm based on the intelligence model of bee swarm foraging behavior . It is known that when the bees find food during the search trip, they return to the hive with a sample of it to tell the rest of the working bees about the location and direction of the food through the bee performing a vibrating dance in a certain direction and a certain number of times to indicate the location of the food. It was suggested by Karabuga in 2005.

4-3 Firefly algorithm (FA):

Firefly algorithm (FA) proposed by Xin-She Yang is metaheuristic algorithm
which is a biologically inspired and is inspired from the flashing behavior of fireflies. This optimization technique is based on the fact that the each firefly attracts to other firefly on the basis of the brightness i.e. firefly with low brightness is attracted toward firefly with more brightness and hence search space is explored efficiently[11].

❖ **A firefly algorithm follows three basic rules:**

1) It considers that all fireflies are unisex. Thus, they attract each other without regarding their sex.

2) Each firefly attracts to other firefly which is proportional to the brightness of individual firefly. So, less bright firefly attracts toward brighter firefly and hence decrease the distance between them.

3)This brightness is a measure of objective function. The objective function is problem dependent function and changes problem to problem[12].

The light intensity of each firefly determine its brightness and hence its attractiveness. Attractiveness of the firefly is calculated using $\beta(r) = \beta 0 e{-}\gamma r^2$.

Where $\beta 0$ measures the attractiveness at $r = 0$ and is usually selected as 1.

γ represents light absorption coefficient. ri,j is the Cartesian distance between two fireflies i and j at location xi and xj respectively in the space.

The movement of the firefly i in the space which is attracted toward another firefly j is defined by using $Xi = xi + \beta 0 e{-}\gamma r^2 + \alpha\ (\text{rand} - 1/2)$

Where α is the randomization parameter in interval [0, 1] and rand is random number generator with numbers uniformly distributed in range [0, 1].

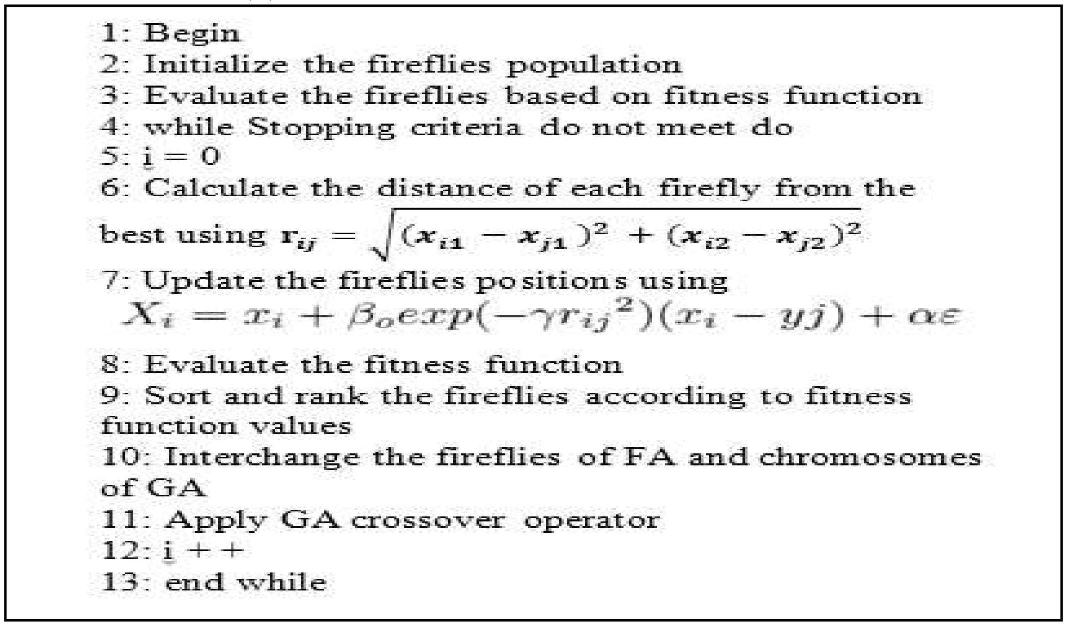
Parameter γ controls the variation in attractiveness and define convergence.

In most of cases, its values lie in range [0.01, 100].

4-4 Using improved firefly algorithm based on genetic algorithm crossover operator:

The firefly algorithm is advantageous over other optimization algorithms due to its flexibility, simplicity, robustness and easy implementation but a major drawback associated with the standard FA applied for solving different optimization problems is poor exploitation capability when the randomization factor is taken large during firefly changing position. This poor exploitation may lead to skip the most optimal solution even present in the vicinities of the current solution which results in poor local convergence rate that ultimately degrades the solution quality. To overcome this problem, the crossover operator of genetic algorithm (GA) is incorporated into firefly position changing stage that results in better exploitation capability which improves the local convergence rate resulting in better solution quality[13][14].

The following figure(1) illustrates the fake code for the proposed approach:



```
1: Begin
2: Initialize the fireflies population
3: Evaluate the fireflies based on fitness function
4: while Stopping criteria do not meet do
5: i = 0
6: Calculate the distance of each firefly from the
best using $r_{ij} = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2}$
7: Update the fireflies positions using
   $X_i = x_i + \beta_o exp(-\gamma r_{ij}^2)(x_i - yj) + \alpha\varepsilon$
8: Evaluate the fitness function
9: Sort and rank the fireflies according to fitness
function values
10: Interchange the fireflies of FA and chromosomes
of GA
11: Apply GA crossover operator
12: i ++
13: end while
```

Figure 1:. Proposed Algorithm FAGA

❖ **Proposed Algorithm:**

1: Parameters Initialization.

[RSM: Relation Site Matrix, Nr: Number of Relation, N_f: Total Number Of Fireflies , $(F_1, F_2, F_3 \ldots F_n)$: Firefly Population, $\text{Max}_{iteration}$: Max Number Of Iterations, TKQP: Number Of Top-K Query Plans, (γ) Gamma: Air Absorption Coefficient, (α) Alpha: Randomization Parameter , (α) Alpha: Randomization Parameter, pc: Crossover, pm:Mutation]

2- Generating initial solutions randomly using the FA criterion.

3-The solutions created randomly are evaluated based on the fitness function, then the best value is selected.

4- The iteration phase begins with i=1.

5- Calculating the distance between the best obtained value and the rest of the fireflies, then evaluating them using the fitness function.

6- Based on the calculated fitness values of the solutions, they are sorted and all fireflies' positions are changed.

7-FA fireflies are replaced with GA chromosomes.

8- Then, applying selection, mutation, crossover factors, and evaluating the resulting solutions using the fitness function, and the process repeats until the maximum specified number of iterations.

5-Results and Discussion:

The BCO algorithm, the GA algorithm, FA algorithm and FAGA algorithm were implemented in MATLAB 7.7 in a Windows 7 environment. The fore algorithms were compared by conducting experiments on an Intel-based 2 GHz PC having 1 GB RAM. The comparisons were carried out on parameters like number of iterations, average QPC (AQPC) and top-K query plans.

The line graphs were plotted to algorithm on AQPC against the number of iterations for selecting the top-ten query plans. These graphs, for the number of relations n = 6 and 8 and 12, are shown in Figures (2–15) respectively. Graphs were plotted showing the average QPC value of top-10 plans .

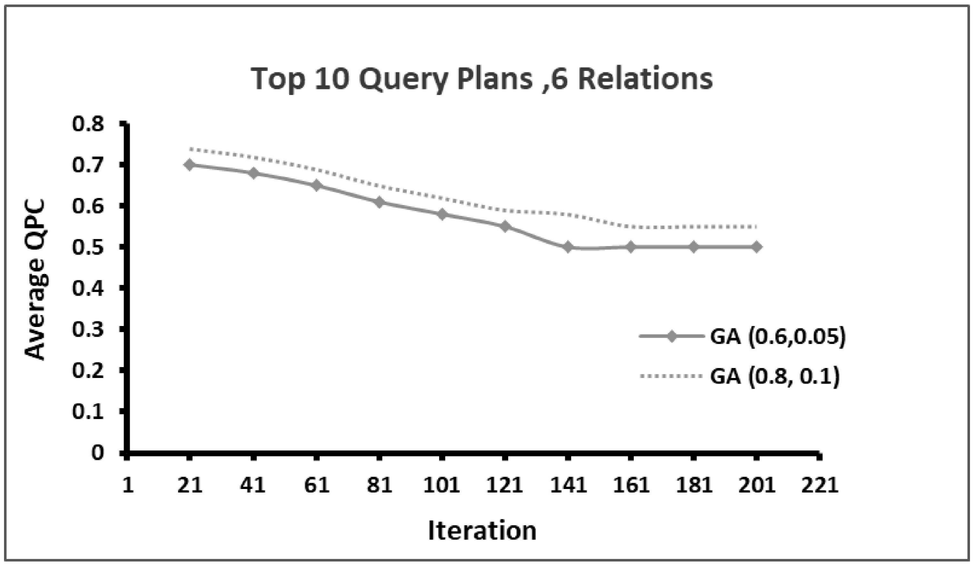


Figure 2. DQPGGA AQPC vs. iterations (6 relations, top-ten query plans)

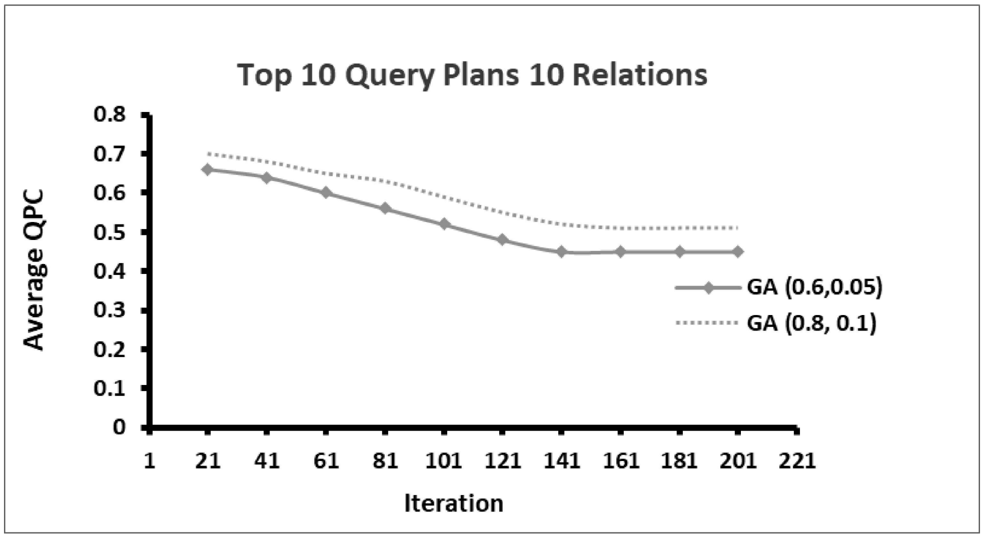


Figure 3. DQPGGA AQPC vs. iterations (10 relations, top-ten query plans)

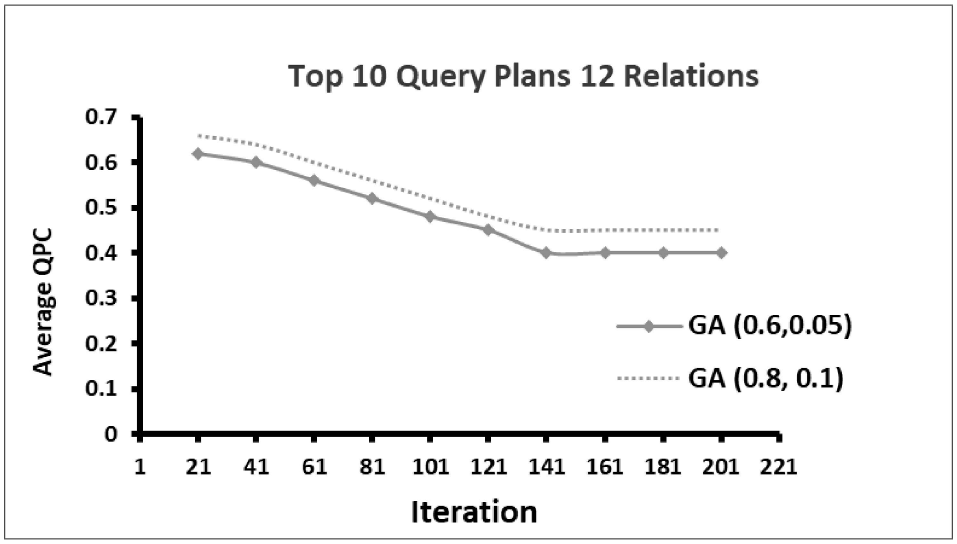


Figure 4. DQPGGA AQPC vs. iterations (12 relations, top-ten query plans).

These graphs were plotted for **GA algorithm** by varying for crossover with Pc ={0.6 , 0.8} and mutation with Pm {0.05,0.1}. The graphs are shown in Figure 8,9,10 convergence to the minimum AQPC for Pc=0.6 and Pm=0.05. shows that Pc=0.6 and Pm=0.05 are best values for generating top-k query plans.

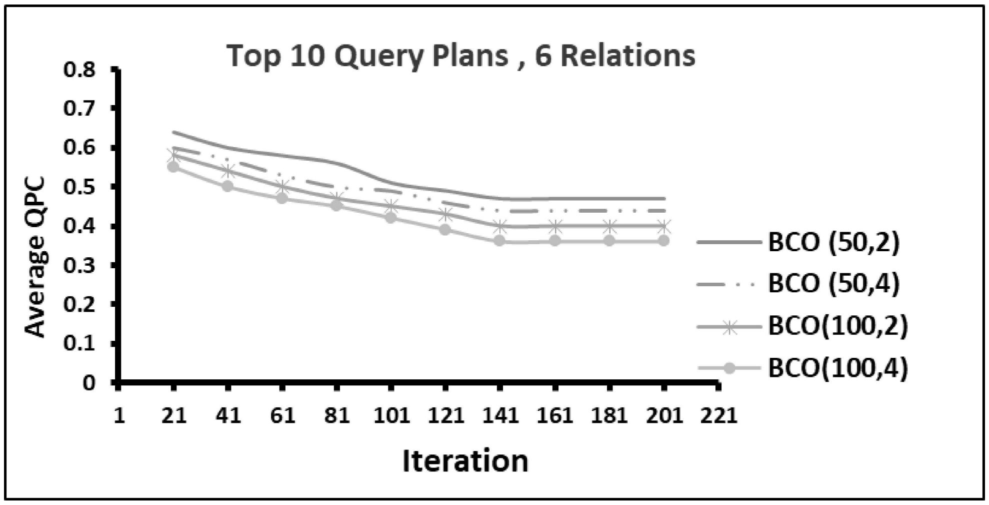


Figure 5. DQPGBCO AQPC vs. iterations (6 relations, top-ten query plans)

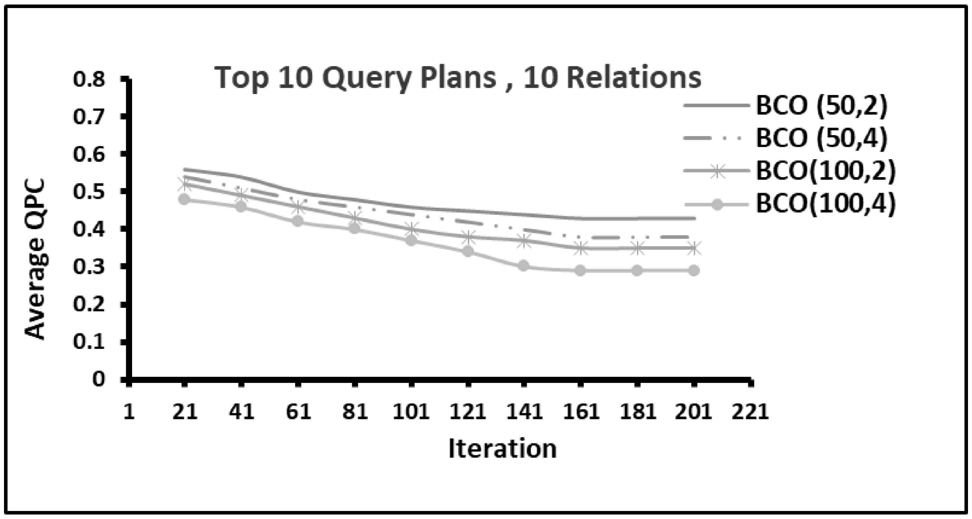


Figure 6. DQPGBCO AQPC vs. iterations (10 relations, top-ten query plans)

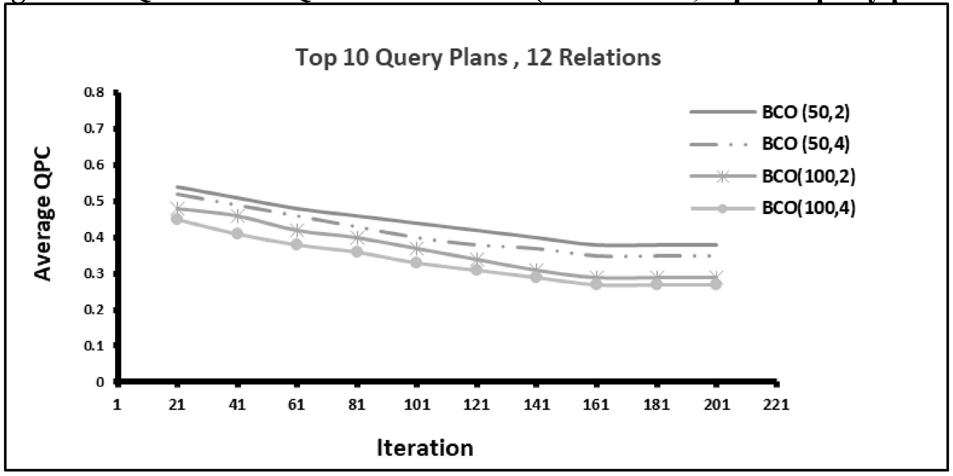


Figure 7. DQPGBCO AQPC vs. iterations (10 relations, top-ten query plans)

These graphs(5,6,7) were plotted for **bee** algorithm by varying the number of bees (NB = 50, 100) and the number of constructive moves (NC = 2, 4), represented as BCO(NB, NC).QPGBCO performs comparatively better for100 bees with NC as 4 i.e., for BCO(100, 4),It can be observed from the graphs that DQPGBCO, for all values of NC, is able to generate the top-K query plans at a lower AQPC, when compared with query plans generated by DQPGGA. This difference in AQPC is significant for BCO(100, 4). It can thus be concluded from the above graphs that DQPGBCO performs better than DQPGGA in terms of the AQPC of the generated top-K query plans.

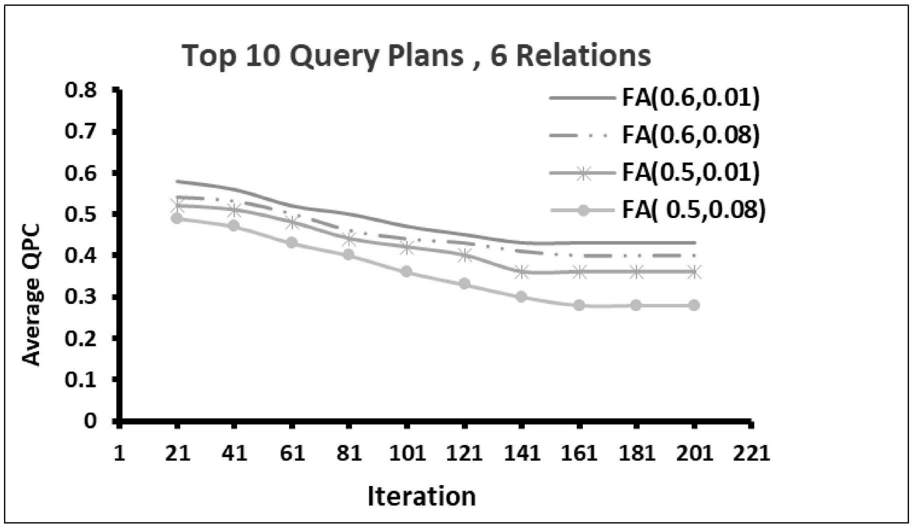


Figure 8. DQPGFA AQPC vs. iterations (6 relations, top-ten query plans)

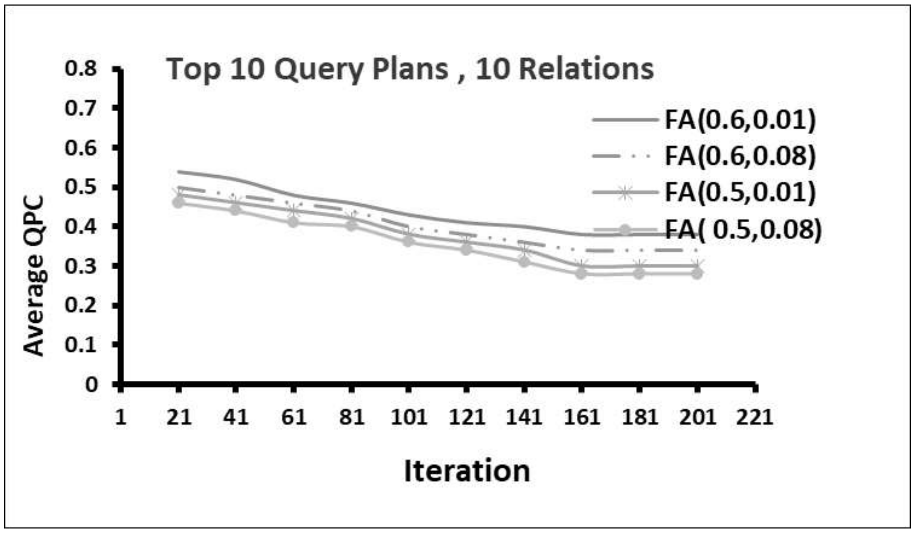


Figure 9. DQPGFA AQPC vs. iterations (10 relations, top-ten query plans)

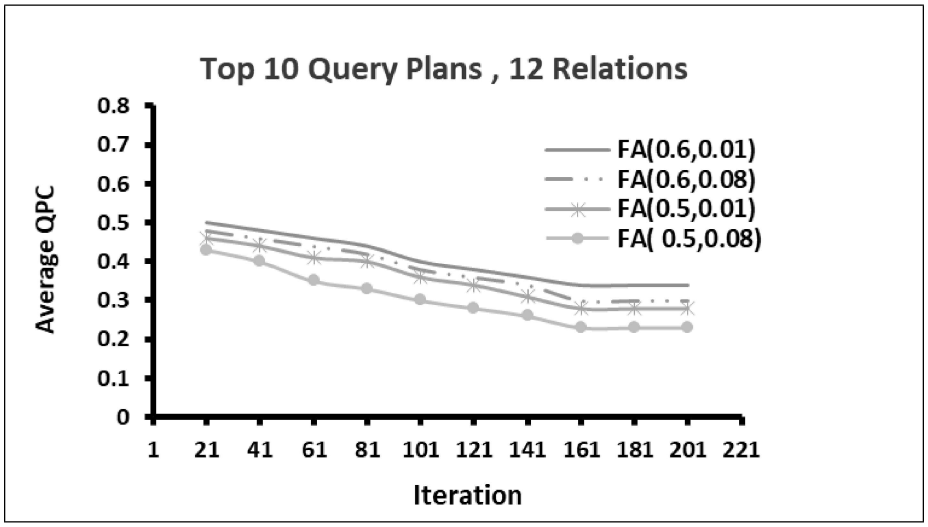


Figure 10. DQPGFA AQPC vs. iterations (12 relations, top-ten query plans)

These graphs were plotted for **FA** algorithm by varying α (α=0.5, 0.6) and γ (γ=0.01, 0.08) and are represented as FA(α, γ). It can be clearly observed from the graphs that DQPGFA is able to generate the Top-10 query plans with the lowest Average QPC (AQPC) for α=0.5 and γ=0.08. This claim is further substantiated from Figure 8,9,10. It can be observed from the graphs that DQPGFA, is able to generate the top-K query plans at a lower AQPC, when compared with query plans generated by DQPGGA and DQPGBCO . graphs that DQPGFA performs better than DQPGGA and DQPGBCO in terms of the AQPC of the generated top-K query plans.

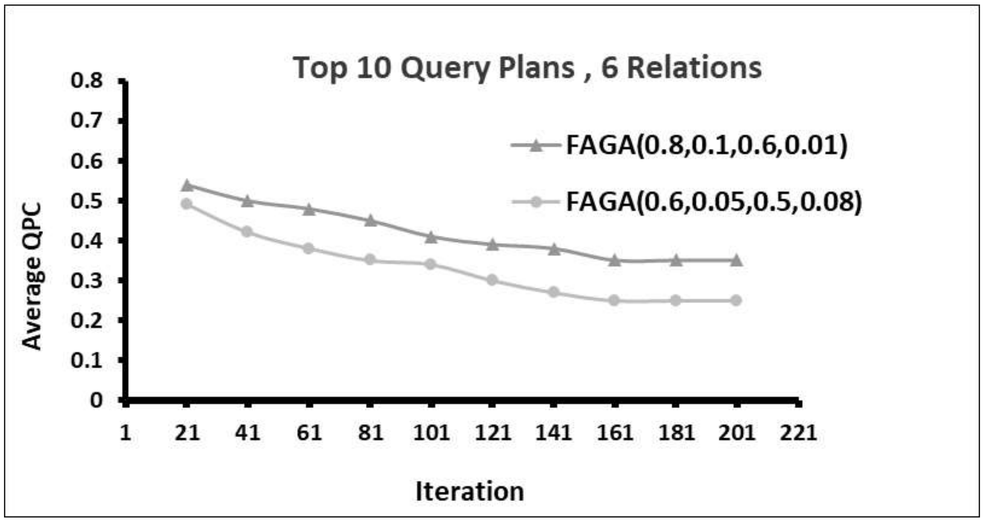


Figure 11. DQPGFAGA AQPC vs. iterations (6 relations, top-ten query plans)

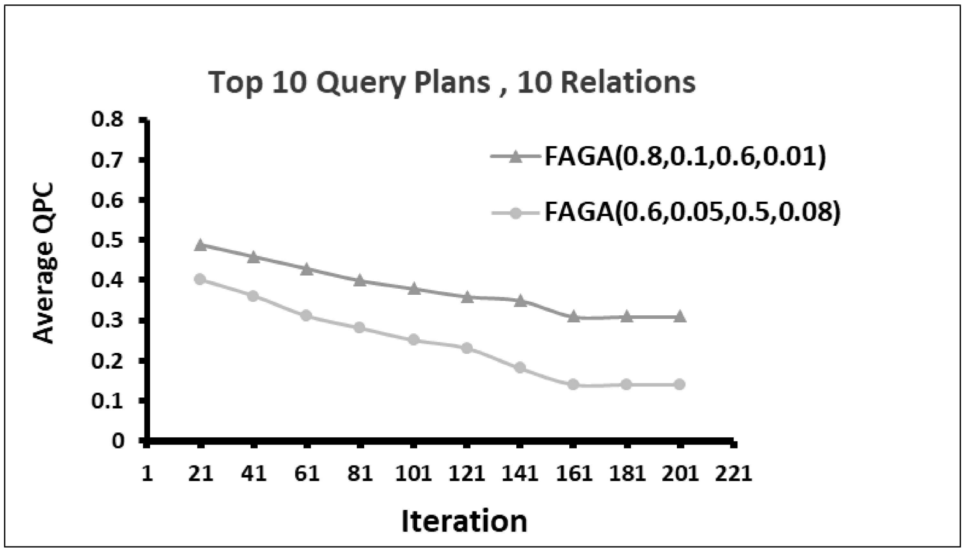


Figure 12. DQPGFAGA AQPC vs. iterations (10 relations, top-ten query plans)

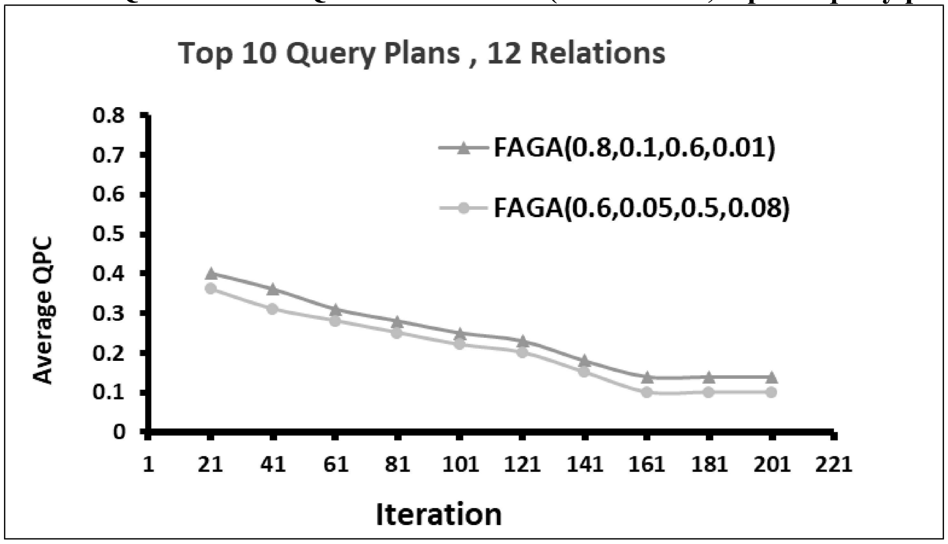


Figure 13. DQPGFAGA AQPC vs. iterations (12 relations, top-ten query plans)

These graphs were plotted for **FAGA** algorithm by varying α (α=0.5, 0.6) and γ (γ=0.01, 0.08),pc(0.6,0.5),pm(0.01,0.08) and are represented as FAGA(α, γ,).

It can be clearly observed from the graphs that DQPGFAGA is able to generate the Top-10 query plans with the lowest Average QPC (AQPC) for α=0.5 and γ=0.08 and pc=0.6 and pm=0.05 This claim is further substantiated from Figure 11,12,13.

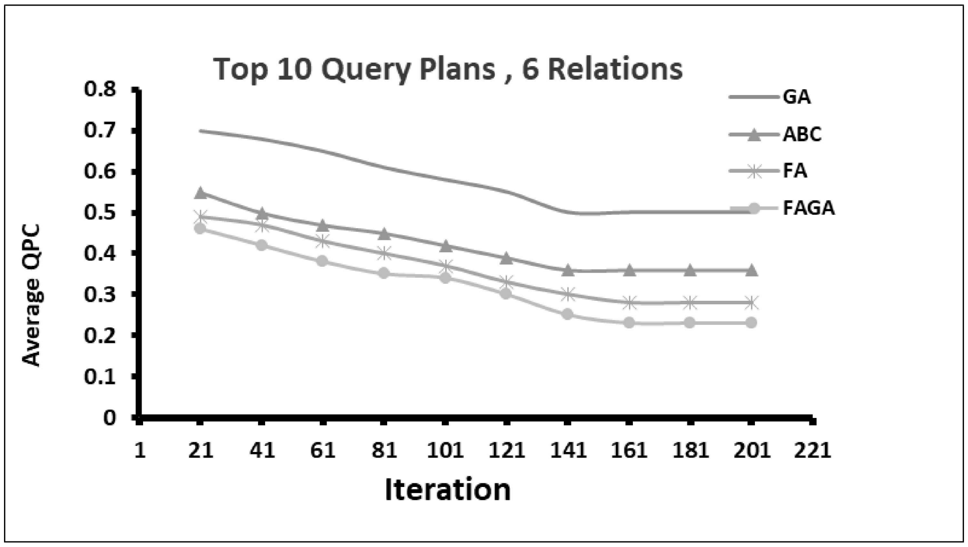


Figure 14. FAGA vs. FA vs. BCO vs. GA AQPC vs. iterations (6 relations, top-ten query plans)

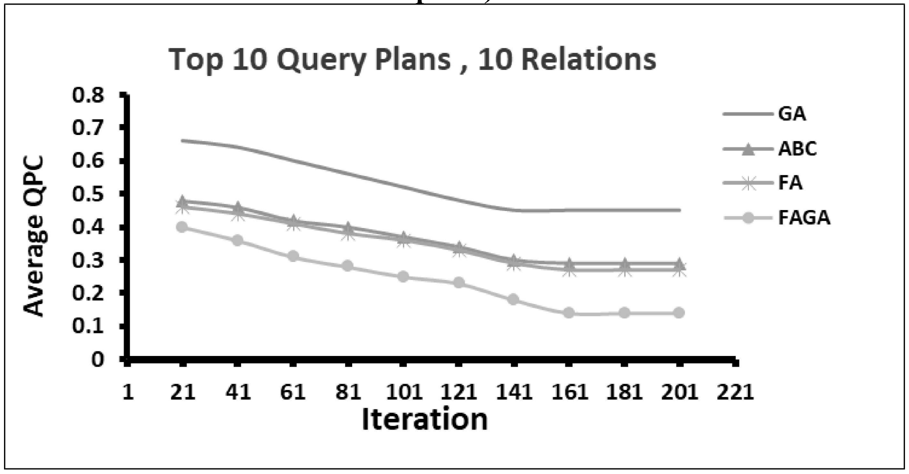


Figure 15. FAGA vs. FA vs. BCO vs. GA AQPC vs. iterations (10 relations, top-ten query plans)

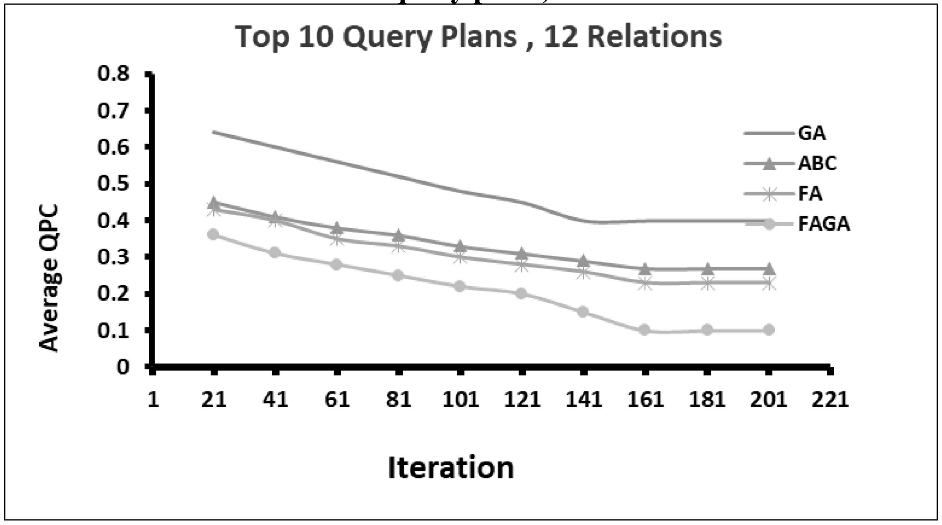


Figure 16. FAGA vs. FA vs. BCO vs. GA AQPC vs. iterations (12 relations, top-ten query plans)

It can be observed from the graphs that DQPGFAGA, is able to generate the top-K query plans at a lower AQPC, when compared with query plans generated by DQPGGA, DQPGBCO and DQPGFA. graphs that DQPGFAGA performs better than DQPGGA , DQPGBCO , DQPGFA in terms of the AQPC of the generated top-K query plans.

5- CONCLUSION:

This paper addresses the DQPG problem with the aim of generating query plans, for a distributed query, that incur lesser total cost of processing a distributed query this regard, use FA algorithm with operators GA that generates 'close' query plans for a distributed query and that involves lesser number of sites and a higher concentration of relations in the participating sites, is proposed. FAGA, which was originally designed for continuous optimization problems, was discretized and adapted to solve the DQPG problem. Experimental based comparisons of FAGA with the GA and BOC and FA algorithm FAGA showed that the former is able to generate query plans that have a comparatively lower QPC. The difference in this QPC increases with increase in the number of relations accessed by the query.

That is, for higher number of relations, FAGA, in comparison to GA,BCO,FA generates query plans that are comparatively more efficient with respect to answering the distributed query. This in turn results in aiding in efficient decision making.

6- Conclusions and recommendations:

- This analytical study facilitates the use of artificial intelligence algorithms to search for data stored in geographically distributed databases, the work of researchers in choosing the most appropriate algorithm, or the combination of two algorithms in order to improve through the features of each algorithm.
- The FAGA algorithm (GA, ABC, FA) is characterized by generating query plans at the lowest cost.
- According to the proposed model, it was found that the FAGA algorithm is better in terms of cost than GA, FA, and ABC.

8- References

[1]Alahmadi, A., Che, D., Khaleel, M., Zhu, M. M., & Ghodous, P. (2015, June). An innovative energy-aware cloud task scheduling framework. In 2015 IEEE 8th International Conference on Cloud Computing (pp. 493-500). IEEE.

[2]Tekkol, T., & Baykara, M. (2023, October). A Comparative Study for the Detection of Attacks on Cloud Computing Systems. In 2023 Innovations in Intelligent Systems and Applications Conference (ASYU) (pp. 1-8). IEEE..

[3]Altaher, R. (2024). Transparency Levels in Distributed Database Management System DDBMS.

[4] Rababaah, H.,2005, "Distributed Databases Fundamentals and Research". Department of Computer and Information Sciences, Indiana University South Bend.

[5] Introduction to Genetic Algorithm.http:// www.rennard .org/ alife/ english/ gavintrgb.html. [Accessed 25 june 2016].

[6]Umbarkar, A. J., & Sheth, P. D. (2015). Crossover operators in genetic algorithms: a review. ICTACT journal on soft computing, 6(1).

[7] Kumar, T. V., Singh, V., & Verma, A. K. (2010, February). Generating distributed query processing plans using genetic algorithm. In 2010 International Conference on Data Storage and Data Engineering (pp. 173-177). IEEE

[8] , A. K. (2011). Distributed query processing plans generation using genetic algorithm. International Journal of Computer Theory and Engineering, 3(1), 38.

[9]Wahid, A., Behera, S. C., & Mohapatra, D. (2015). Artificial Bee Colony and its Application: An Overview. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, *4*(4), 1475-1480

[10] Kumar, T. V., Kumar, L., & Arun, B. (2015). Distributed query plan generation using BCO. *International Journal of Swarm Intelligence*, *1*(4), 358-377.

[11]Yuce, B., Packianather, M. S., Mastrocinque, E., Pham, D. T., & Lambiase, A. (2013). Honey bees inspired optimization method: the bees algorithm. *Insects*, *4*(4), 646-662.

[12] Pal, S. K., Rai, C. S., Singh, A. P.,2012, "Comparative Study of Firefly Algorithm and Particle Swarm Optimization for Noisy Non- Linear Optimization Problems", I.J.Intelligent Systems and Applications, Published Online in MECS (http://www.mecspress.org/), pp:50-57.

[13]Singh, N., Prakash, J., & Kumar, T. V. (2016). Distributed Query Plan Generation Using Firefly Algorithm. *International Journal of Organizational and Collective Intelligence (IJOCI)*, *6*(1), 29-50.

[14]Wahid, F., Ghazali, R., & Ismail, L. H. (2019). Improved firefly algorithm based on genetic algorithm operators for energy efficiency in smart buildings. *Arabian Journal for Science and Engineering*, *44*(4), 4027-4047.

[15]Wahid, F., Alsaedi, A. K. Z., & Ghazali, R. (2019). Using improved firefly algorithm based on genetic algorithm crossover operator for solving optimization problems. *Journal of Intelligent & Fuzzy Systems*, *36*(2), 1547-1562.

[16]Zare, M., Ghasemi, M., Zahedi, A., Golalipour, K., Mohammadi, S. K., Mirjalili, S., & Abualigah, L. (2023). A global best-guided firefly algorithm for engineering problems. Journal of Bionic Engineering, 20(5), 2359-2388