SQL Optimization in Oracle using Hybrid Genetic and Ant Colony Algorithm

Rajan Kusi[1], Kabir Kumar Sinkemana[2], Sanjeeb Prasad Panday[1*] and Shashidhar Ram Joshi[1]

[1]Department of Electronics and Computer, Institute of Engineering Pulchowk Campus, Lalitpur
[2]Nepal College of Information Technology, Balkumari, Lalitpur

***CORRESPONDING AUTHOR:**

Sanjeeb Prasad Panday
Email: sanjeeb@ioe.edu.np

ISSN : 2382-5359(Online), 1994-1412(Print)

DOI:
https://doi.org/10.3126/njst.v21i2.62353

Date of Submission: 27/10/2021
Date of Acceptance: 21/12/2022

ABSTRACT

In this paper, the input user Structured Query Language (SQL) query is converted into an optimized SQL query using a hybrid algorithm. The main aim is to reduce query execution time using PHP language and oracle database. These performance has been evaluated using different performance metrics: Cost of individuals, Query execution time. The hybrid algorithm method combines the evolutionary effect of the Genetic Algorithm (GA) and the cooperative effect of Ant Colony Optimization (ACO). A GA with a great global converging rate aims to produce an initial optimum for allocating initial pheromones of ACO. An ACO with great parallelism and effective feedback is then served to obtain the optimal solution. A fused algorithm of a GA and ACO to solve SQL optimization problems is an innovative solution that presents a clear methodological contribution to the optimization algorithm. In the simulation result, we found the algorithm of a GA and ACO to solve SQL optimization problems in Oracle. It is an innovative solution that presents a clear methodological contribution to the optimization algorithm.

Keywords: Ant Colony, Genetic, SQL, Query Optimization, Query Execution plan

1. INTRODUCTION

Structured Query Language (SQL) Statements are the specialized language used to retrieve data from the database. We can get the same results by writing different SQL queries. However, using the best query is important when performance is considered. Making a query run faster reduces the number of calculations that the software (and, therefore, hardware) must perform. We need some understanding of how SQL makes calculations. First, let us address high-level things that will affect the number of calculations needed to make and, therefore, query run time: table size, joins, and aggregations.

Genetic algorithms (GAs) are adaptive metaheuristic search algorithms classified as evolutionary computing algorithms. It is a technique inspired by natural evolution that Charles Darwin postulated as a theory of natural evolution (Yan 2021; Holland 1975). This algorithm reflects the process of natural selection, where the fittest individuals are selected for reproduction to produce offspring of the next generation. The phases considered in a genetic algorithm are Initial population, Fitness function, Selection, Crossover, and Mutation.

Ant colony algorithms are becoming popular approaches for solving combinatorial optimization problems in database SQL systems. The basic idea of ant heuristics is the natural behavior of ants that succeed in finding the shortest paths from their nest to food sources. They communicate via a collective memory consisting of pheromone trails (Wagh & Nemade 2017; Chande & Sinha 2010; Pan & Wang 2009). As ants have weak global insight into their environment, they move randomly when no pheromone is available. However, it tends to follow a path with a high pheromone level when many ants move in a common area, leading to an autocatalytic process. An ant does not choose its direction based on the level of pheromone exclusively but also considers the neighborhood of the nest and the food place, respectively, into account. It allows the discovery of new and potentially shorter paths.

2. MATERIALS AND METHODS

2.1 Hybridization Algorithm of a GA and ACO

A system has been implemented combining the Ant Colony algorithm and Genetic algorithm. GA has strong flexibilityand quick global searching ability with a higher population scattering ability for the extensive amplitude of answers. On the other hand, ACO has a low population scattering ability, parallel processing, high convergence speed, and global searching ability with a positive feedback mechanism. By combining both algorithms, strong features of both can improve the performance while overcoming the drawbacks of each. The performance of GA depends on the population and operator used in the algorithm. If the population size is small, then it converges fast.

2.2 Basic Principle of Adynamic Combination of GA and ACO

“Fig 1” shows the speed-time curve of the genetic algorithm and ant colony algorithm. For the genetic algorithm, there is a higher convergence speed to the optimal solution during the preliminary stage (t0~ta), but it will significantly reduce after ta. However, during the preliminary stage (t0~ta) of the ant colony algorithm, the searching speed is very slow for lacking pheromones. Then after pheromones reach a certain degree (after ta), the convergence speed to optimal solution improves quickly. The basic principle of a dynamic combination between a genetic algorithm and an ant colony algorithm is that we can utilize a genetic algorithm to get initial pheromones. After that, obtain the optimal solution by ant colony algorithm (Zhao *et al.* 2016).

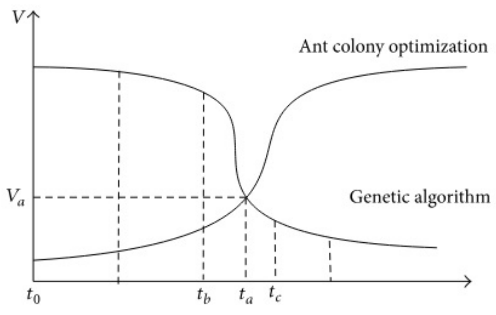


Fig. 1. Difference in the speed-time curve of GA and ACO

2.3 Hybridization of GA and ACO

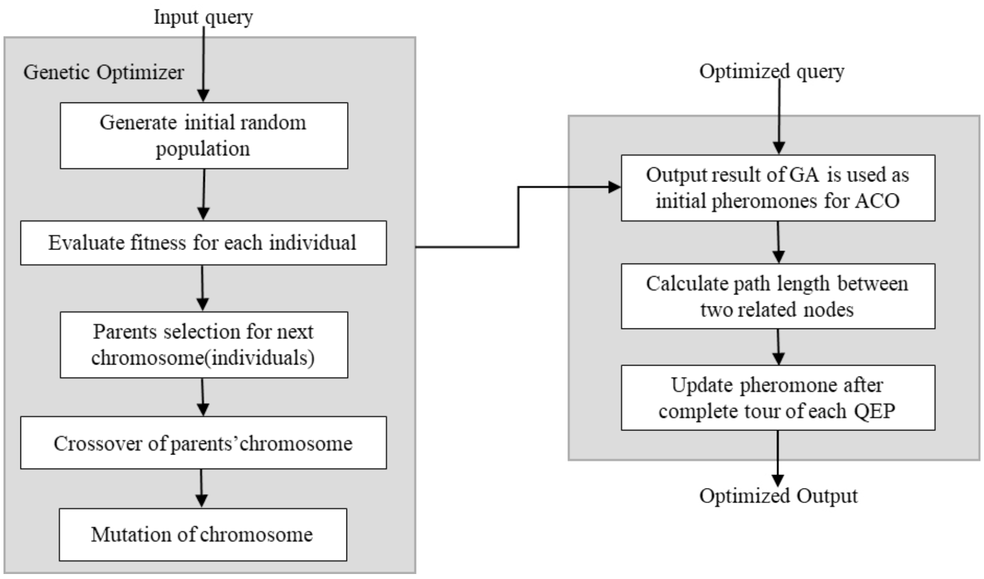

Input query
Genetic Optimizer
Generate initial random population
Evaluate fitness for each individual
Parents selection for next chromosome(individuals)
Crossover of parents' chromosome
Mutation of chromosome
Optimized query
Output result of GA is used as initial pheromones for ACO
Calculate path length between two related nodes
Update pheromone after complete tour of each QEP
Optimized Output

Fig. 2. Hybrid GA and ACO

Fig. 2 shows the block diagram of the hybrid GA and ACO process that consists of various steps. Query translator converts the high-level input language (SQL) into relational algebra expression. Then it is converted into query tree structures. Different query trees can be generated from a single input query. These query trees are treated as query execution plans (QEPs). The solution of a query is the set of all QEPs for the query.

(Butey *et al*. 2012; Horng *et al*. 1994; Petković 2010) In GA, the initial population is randomly selected and will contain certain chromosomes. Eachchromosomeis again validated using the fitness function, and each individual's fitness is calculated. Evaluation of fitness is applied to eliminate the worst-fitted members in the population. Once the initial population is created, two fit chromosomes are selected from a population thatcan mate and create their offspring. The fit chromosomes are selected, allowing them to produce offspring in such a way that would lead to chromosomes closer to the optimal solution and, therefore, less diversity. The primary transformation is performed by crossover and mutation. One point crossover is used, a random crossover point is selected, and the tails of both chromosomes are swapped to produce new offspring. The crossover probability and mutation probability are calculated for this operation. The remaining members in the population are given to the ant colony optimizer. Here, the node of the query tree represents a state, and the tree's edge represents apath length. The initial pheromones for the routes of ACO are settled according to the results of the GA.When artificial ants move from one state to another, the path length between these two states is calculated. If a single ant travels all possible states of the query, then the tour is completed. After completing a tour of the artificial ant, a pheromone is updated for each ant. Once the stop criteria are met, the output result will be the best solution for the input SQL query.

Query translator converts the high-level input language (SQL) into relational algebra expression (Ceri & Gottlob 1985). Then it is converted into query tree structures. Different query trees can be generated from a single input query. These query trees are treated as query execution plans (QEPs). The solution of a query is the set of all QEPs for the query. Each execution plan can be considered a solution program for the problem of finding a good access path to retrieve the required data.

1. Estimated cost model for fitness function

The simple cost model is based on two assumptions (Dong & Liang 2007)

a) The uniform distribution of attribute values and

b) The sum of the size of the intermediate relations determines the cost of a join tree.

The cost of the join tree is

$$\text{Cost} = \textstyle\sum_{i=0}^{n-1} \text{n}\left(\text{t}_\text{i}\right) \text{ ....(1)}$$

The Number of tuples is given by

$$\text{n}\left(\text{t}\right) = \frac{n\left(r\right) \times n\left(s\right)}{\pi C_j \in C \max\left(V\left(C_j, r\right), V\left(C_j, s\right)\right)} \text{ .... (2)}$$

And the Number of distinct values that appear in the relation r for attribute A

$$V\left(A,t\right) = \begin{cases} V\left(A,r\right) A \in r - s \\ V\left(A,s\right) A \in s - r \\ \min\left(V\left(A,r\right), V\left(A,s\right)\right) A \in r \; A \in s) \end{cases} \text{ .... (3)}$$

2. Crossover and Mutation Probability.

The self-adaptive crossover and mutation probability functions are as follows:

$$P_c = \begin{cases} P_{c0} & ,f \le \bar{f} \\ P_{c1}\left(\frac{P_{c0}}{P_{c1}}\right)\left(\frac{f_{max} - f}{f_{max} - \bar{f}}\right) & ,f > \bar{f} \end{cases} \text{ .... (4)}$$

$$P_c = \begin{cases} P_{m0} & ,f' \le \bar{f} \\ P_{m1}\left(\frac{P_{m0}}{P_{m1}}\right)\left(\frac{f_{max} - f'}{f_{max} - \bar{f}}\right) & ,f' > \bar{f} \end{cases} \text{ ....(5)}$$

2.4 Ant Colony Algorithm Technique Used in Query Optimization

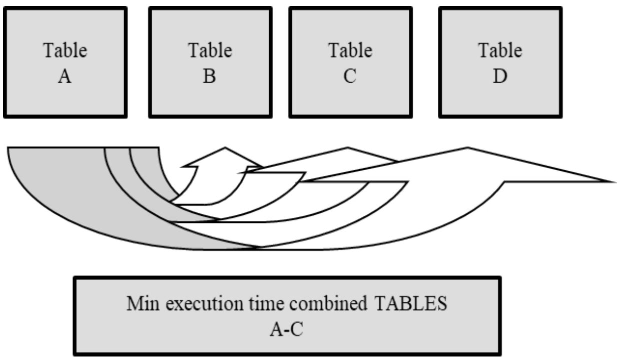


Fig. 3. First step of thetour of theant colony algorithm

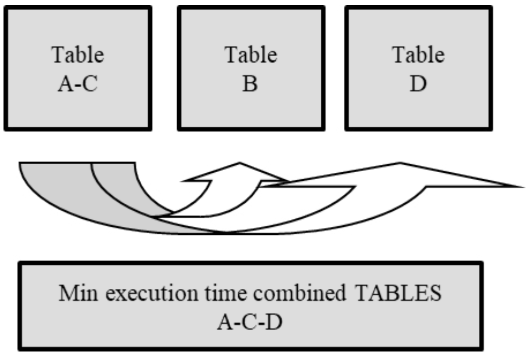


Fig. 4. Second step of thetour of the ant colony algorithm

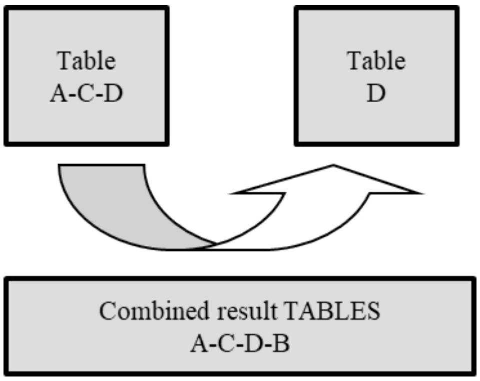
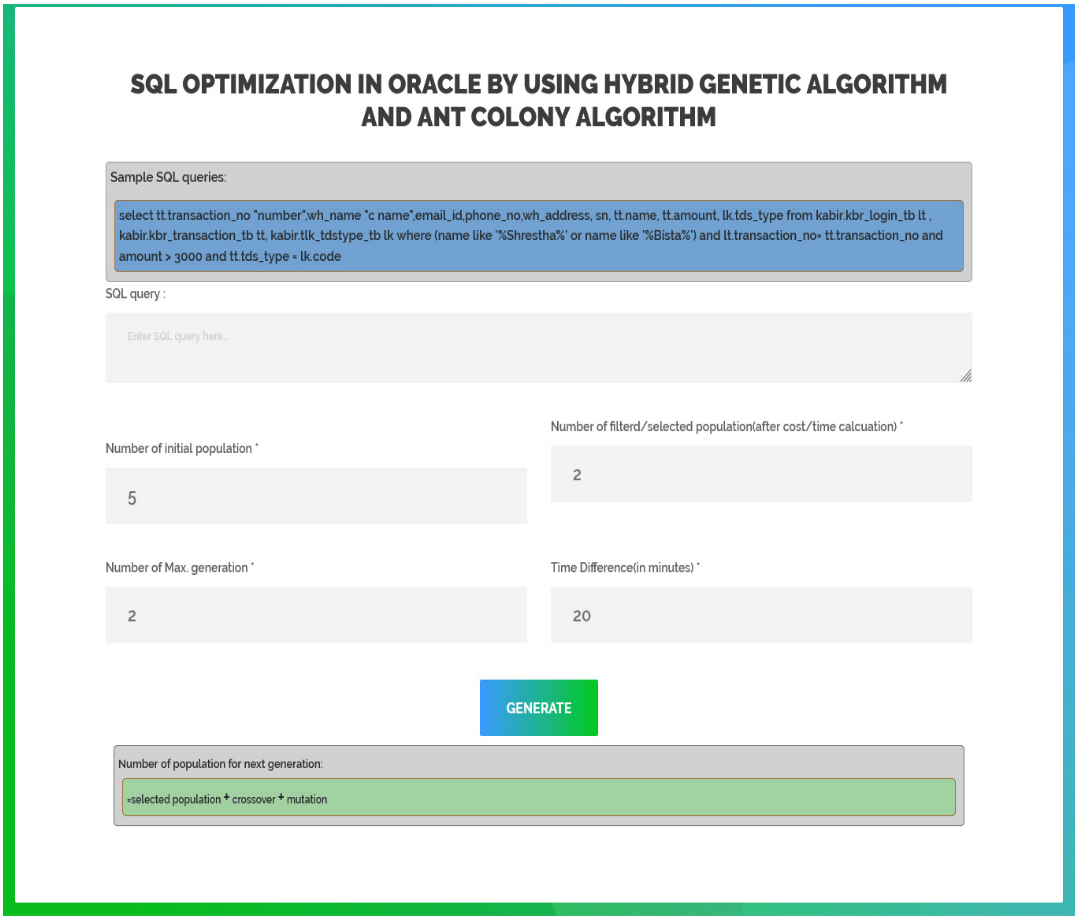


Fig. 5. Third step of thetour of theant colony algorithm

Fig. 3, 4, and 5 show the various steps of the tour of the ant colony algorithm. The four join tables, i.e., A, B, C, and D, denote state, and their relationship as the path, are considered. Artificial ants start their tour from table A. After reaching the next state, the estimated cost and execution time are calculated. Here joining tables A-C has a minimum estimated execution time than joining tables A-B and A-D. Then different ants continue the tour from combined table A-C to the remaining state i.e., table B and table D, and calculate the estimated cost and execution time. The join tables A-C-D has minimum execution time compared to join tables A-C-B. Now only one node i.e., table B, remains. The tour is completed when the ant reaches node B. Finally, the estimated cost and execution time is calculated for the query with these four join tables. This way, an optimized query is obtained after artificial ants complete all possible tours.

3. SIMULATION AND RESULT

The input parameters considered in simulation and experiments are the input SQL query, the Number of the initial population, the initial population, the number of selected populations, the Number of maximum generations, and the cut-off time.


SQL OPTIMIZATION IN ORACLE BY USING HYBRID GENETIC ALGORITHM AND ANT COLONY ALGORITHM

Sample SQL queries:

```
select tt.transaction_no "number",wh_name "c name",email_id,phone_no,wh_address, sn, tt.name, tt.amount, lk.tds_type from kabir.kbr_login_tb lt , kabir.kbr_transaction_tb tt, kabir.tlk_tdstype_tb lk where (name like '%Shrestha%' or name like '%Bista%') and lt.transaction_no= tt.transaction_no and amount > 3000 and tt.tds_type = lk.code
```

SQL query :

Enter SQL query here...

Number of initial population *: 5

Number of filterd/selected population(after cost/time calcuation) *: 2

Number of Max. generation *: 2

Time Difference(in minutes) *: 20

Number of population for next generation:

=selected population + crossover + mutation

Fig. 6. Query optimization processing form

Fig. 6 is a form designed for the query optimization process. For the query optimization process, five input parameters are considered. These are; input SQL query, number of the initial population, number of selected population, number of maximum generation, and cut-off time.

The result of a query with four join tables for original input query, ACA, GA, and Hybrid ACA and GA for conditioning experiment on 5 million tuples records are obtained as below:

No. of join tables	Least execution time			
	Original(Input)	ACA	GA	Hybrid GA and ACA
4	56 sec	53 sec	22 sec	19 sec
5	57 sec	54 sec	22 sec	19 sec
6	57 sec	54 sec	22 sec	19 sec
7	59 sec	56 sec	24 sec	19 sec
8	61 sec	59 sec	25 sec	20 sec
9	66 sec	63 sec	35 sec	21 sec
10	68 sec	64 sec	35 sec	21 sec
11	70 sec	66 sec	35 sec	24 sec
12	70 sec	67 sec	35 sec	24 sec
13	80 sec	67 sec	35 sec	24 sec
14	86 sec	67 sec	35 sec	25 sec
15	90 sec	76 sec	43 sec	25 sec
16	90 sec	76 sec	43 sec	28 sec
17	92 sec	78 sec	43 sec	28 sec
18	98 sec	81 sec	43 sec	28 sec
19	99 sec	82 sec	45 sec	32 sec
20	110 sec	90 sec	45 sec	32 sec

Table 1. Comparison of the execution time of GA, ACA, and Hybrid algorithm

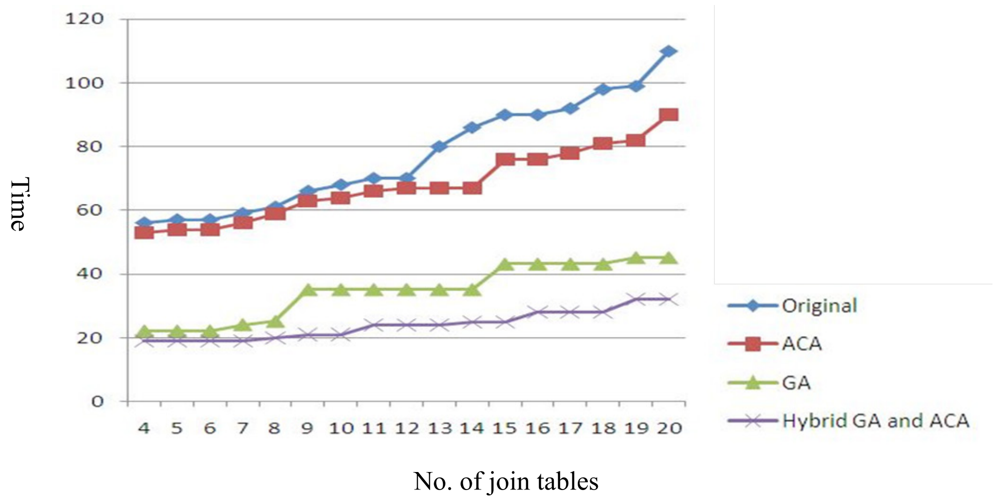


Fig 7. Graphical representation of acomparison of execution time

Fig. 7 shows the graphical representation of execution time vs. number of join tables for original input query, ACA, GA, and Hybrid algorithm.

4. CONCLUSION

This paper mainly focuses on the generation of population, then selecting the best population with the least execution time for parent query and performing crossover of the best population in Oracle. It has been found that the execution time for an optimized query using a hybrid genetic algorithm and ant colony algorithm in Oracle is less than the original query. Also, the next performance parameter, i.e., the CPU cycle for an optimized query using hybrid GA and ACA, is less than that of the original query. The SQL query optimization in the oracle database using a hybrid genetic algorithm and ant colony algorithm has been performed.

ACKNOWLEDGMENT

The authors are highly thankful to the Nepal College of Information Technology (NCIT) and the Institute of Engineering, Tribhuvan University, for providing us with this opportunity and guidance. The authors also thank the referees for their careful reading and valuable suggestions.

REFERENCES

Butey, P. K., S., Meshram, and R. L. Sonolikar, 2012. Query Optimization by Genetic Algorithm (Volume: 3, Number: 1). Journal of Information Technology and Engineering, International Science Press. https://www.ispjournals.com

Chande, S. V., and M. Sinha, 2010. Query Optimization Using Genetic Algorithms (No. Volume: 2, Issue: 3). Research Journal of Information Technology. https://doi.org/10.3923/rjit.2010.139.144

Ceri, S., and G. Gottlob, 1985. Translating SQL into Relational Algebra: Optimization, Semantics, and Equivalence of SQL Queries. IEEE Transactions on Software Engineering, SE-11(4), 324–345.

Dong, H., and Y. Liang, 2007. Genetic algorithms for large join query optimization. Association for Computing Machinery, New York, NY, United States. https://doi.org/10.1145/1276958.1277193

Holland, J. H., 1975. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press, 55 Hayward St., Cambridge, MA, USA.

Horng, J.-T., C.-Y., Kao, and B.-J. Liu, 1994. A Genetic Algorithm for Database Query Optimization. IEEE World Congress on Computational Intelligence - Orlando, FL, USA. https://doi.org/10.1109/icec.1994.349926

Pan, W., and L. Wang, 2009. An Ant Colony Optimization Algorithm Based on the Experience Model (Volume: 6). Fifth International Conference on Natural Computation, ICNC 2009, Tianjian, China. https://doi.org/10.1109/ICNC.2009.690

Petković, D. š. 2010. Comparison of Different Solutions for Solving the Optimization Problem of Large Join Queries. IEEE. https://doi.org/10.1109/DBKDA.2010.1

Wagh, A., and V. Nemade, 2017. Query Optimization using Multiple Techniques (Volume: 163, Number: 3). International Journal of Computer Applications, Foundation of Computer Science, New York, NY 10001, USA.

Wagh, A., and V. Nemade, 2017. Query Optimization using Modified Ant Colony Algorithm (No. Volume: 167, Number: 2). International Journal of Computer Applications, Foundation of Computer Science (FCS), NY, USA. https://doi.org/10.5120/ijca2017914185

Yan, X.-S. 2021. Nature-Inspired Optimization Algorithms (Second Edition). Science Direct

Zhao, F. T., Z., Yao, J., Luan, and X. Song, 2016. Article ID 2167413, A Novel Fused Optimization Algorithm of Genetic Algorithmand Ant Colony Optimization (Article ID: 2167413). Hindawi Publishing Corporation, Mathematical Problems in Engineering. https://doi.org/10.1155/2016/2167413