AN ENHANCED QUERY OPTIMIZATION TECHNIQUE IN BIG DATA USING ACO ALGORITHM

Deepak Kumar[1], Dr. Vijay Kumar Jha[2]

[1]Research Scholar, Department of Computer Science and Engineering, Birla institute of Technology Mesra, Ranchi
[2]Associate Professor, Department of Computer Science and Engineering, Birla institute of Technology Mesra, Ranchi

---

Abstract - *Now a days large volume of data are being generated which are in unprecedented rate and scale. The conventional data analysis tools are not enough to process and analyze these data. Many types of techniques have been used to optimize the query, Heuristic Greedy [11], Iterative development and Ant Colony algorithms are used to query optimization [15]. Ant Colony Algorithm is employed to locate the optimal answer for a diverse category of difficulties [18]. To surmount the concerns connected with the current techniques, a new multi-objective ant colony-based query optimization method is projected [7]. The consequence of query cost and communication overheads will also be well thought-out [27]. The use of ant colony optimization can uncover optimistic query in order to diminish the query cost [6].*

Key Words: Optimization, Query processing, Bigdata, Ant colony optimization (ACO), Ant colony algorithm etc…

1. INTRODUCTION

Retrieving from large scale of data has become more challenging task since the generation of data is in unprecedented scale and rate. For example, data on web is growing at an enormous rate with new contents, web pages, links, media etc. This our current system becomes unfit to query these data. ACOs are one of the methods used to efficiently collect data or to extract data from larger and larger data sets. [1]. ACO works on the standard of studying artificial structures that come or bears a resemblance to the prototypes of genuine insect provinces [16]. This behaviour is then used to understand, analyse, and solve discrete quantitative problems. There is a lot of data that needs to be assembled and scattered in big data applications. It represents the size we know. Every time the amount of information changes, there is a probability measure that analyses how the data can change due to the change in the size of the data. [12]. For such purposes, we need to define a modified algorithm for problems that involve large amounts of information. All topologies of the same size can be excavated for any known or unknown purpose using the ACO algorithm [17].

Query optimization in large amount of data is playing a significant role in today data retrieval scenario. The queries which perform database operation are generally Insert, Update, Delete, Read etc. These plays a crucial role in data management. Some cloud storage system such as Hadoop [38], Hive [39], Scope [40], Spark SQL [41], Cloudera Impala [15] these are providing efficient solutions for data management in cloud environment. There are a number of techniques to optimize queries performance over big data and still there is a need for improvement.

1.1 QUERY OPTIMIZATION

With recent research using different models to solve the MJQO problem, researchers have implemented various search algorithms to find the right scheme for query execution. However, they could not provide a better solution in terms of query execution time and cost and due to increasing data and large number of tables traditional methods could not solve this optimization problem effectively.

This problem can be tried in many ways. But performance improvements are still needed, improvements in the quality of the solution.

Challenges in Query Optimization

- Large size of data
- High processing cost
- More execution time

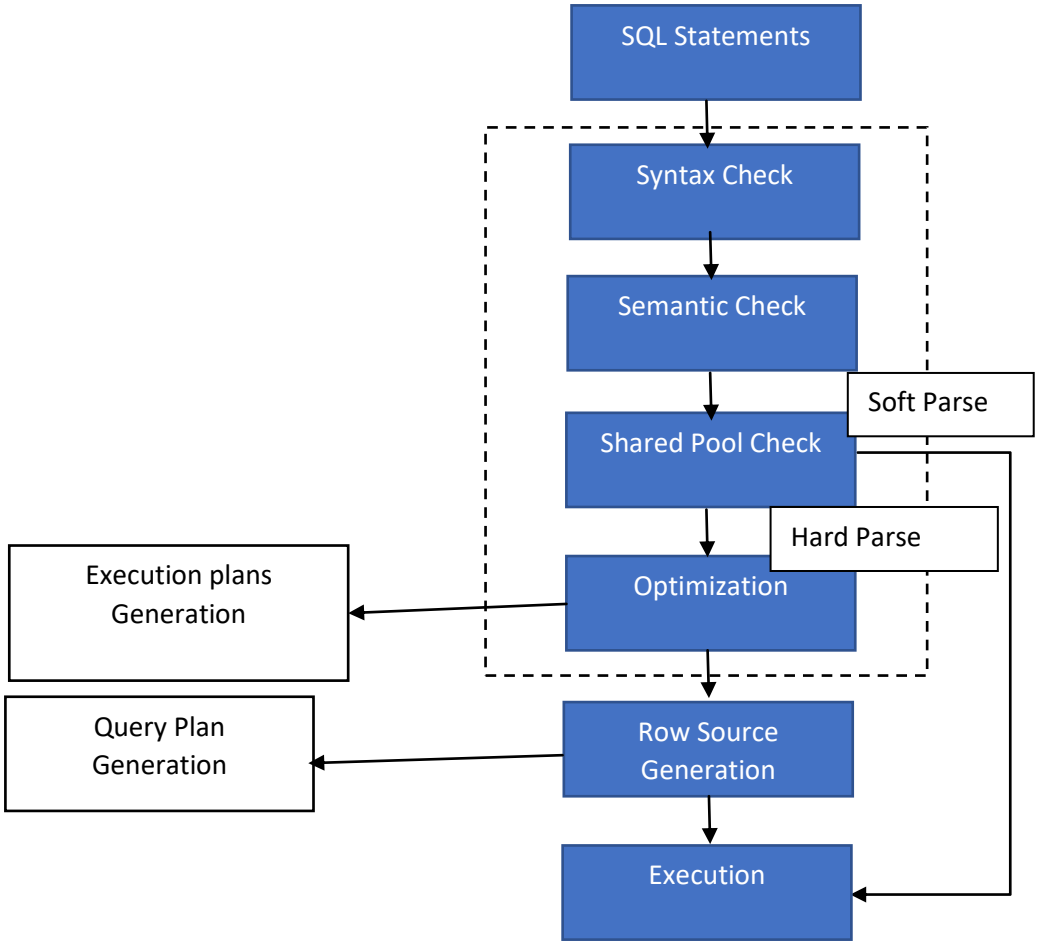

Figure 1: Query Optimization

A typical query processing system consist of some systematic processes which is shown in Figure 1, which translate high-level query language into low-level algebraic expressions. It is the process of parsing and translating, optimizing and executing user submitted questions.

a) Parsing and translation: Check syntax and verify relationships. Translate the question into a similar relational algebraic expression.

b) Optimization: Develop a proper evaluation plan (with minimum cost) for the question plan.

c) Evaluation: The query-execution engine takes a (correct) evaluation plan, executes that plan and answers the question.

d) Semantic check: The semantic test determines whether a statement is meaningful or not.

e) Shared Pool Check: The shared pool check determines the presence of code written in the shared pool and does not take additional steps for database optimization and execution if there is code in the shared pool.

a. Hard Parse: If written code is not existing in the Shared Pool

b. Soft Parse: If written code is existing in the Shared Pool

f) SQL Optimization: Query optimization is a technique where database system compares different query strategies and choose best optimal strategy with least expected cost. During the optimization phase, hard parsing should be done at least once for each specific statement and optimized during parsing.

g) SQL Row Source Generation: Row Source Generator produces a repetitive execution plan that utilizes the execution plan software from the optimizer and the rest of the database. Each iterative execution step provides a set of rows and the last step is to provide rows to the application that issued the SQL statement.

1.2 ANT COLONY OPTIMIZATION:

In recent times, several researchers have paid their attention to a new group of algorithms, called metaheuristics. A metaheuristic is a class of algorithmic notions that can be employed to describe heuristic methods applicable to a large set of diverse problems [2]. The application of metaheuristics has considerably augmented the capability of discovering very high-quality solutions to tough, practically appropriate combinatorial optimization troubles in a realistic time [8]. A predominantly successful metaheuristic is stimulated by the conduct of real ants. Beginning with Ant System [29], a number of algorithmic approaches standing on the very identical thoughts were built up and exercised with substantial success to a multiplicity of combinatorial optimization dilemmas from academic as well as from real-world applications. In this episode, we pioneer ant colony optimization [28], a metaheuristic structure which envelops the algorithmic approach stated above. The ACO met heuristic has been anticipated as an ordinary structure of the current applications and algorithmic options of a range of ant algorithms [3]. Algorithms which fit into the ACO metaheuristic structure will be identified in the subsequent ACO algorithms. Meta heuristic algorithms are algorithms that, in order to avoid native optimization, use some basic heuristic: structural heuristic start of the null solution and the beginning of the complete [26] or local search heuristic full result, including elements. And repeatedly modifying its basics to better comprehend [4]. For a given query statement, the proposed optimization approach delivers the optimal query access plan from a set of valid access plans.

2. RELATED WORK:

Radhya Sahal, Fatma A. Omara (2016), in their study, stated that huge data analysis organisms, like MapReduce, have turned into major areas of concern for several institutions and research groups. Presently, multi-query converted into MapReduce tasks is submitted repetitively with a like tasks [3]. Hence, making use of these related or alike tasks leads to possibilities of repetitions and hence it is essential to evade from repeated calculations of MapReduce tasks. Consequently [27], several researchers have concentrated on the sharing the opportunity for optimum utilization of multi query processing [9]. As a result, the key aim of this job is to learn and contrast expansively two existing techniques in context of sharing opportunity using filtering based on predicate; MRShare and MRShare relaxed [7]. A comparative analysis has been done on TPC-H benchmark and established that the MRShare relaxed method appreciably does better than the MRShare for shared data in context of filters based on a predicate, among multi-query [5].

Yi Shan and Yi Chen (2015), in their study, stated that MapReduce is extensively accredited by both academia as well as industry as an efficient encoding model for big data query processing [4]. He recommended SOSQL, a scalable optimizer for SQL queries by MapReduce. Each vertex is underlined for the table and each edge stands to join the two tables of the question [25]. The goal of optimization is to divide the graphs into a group of sub-graphs and assign each sub-graph to the MRJ so that all the edges of the included graph work in the shortest possible time. [3]. The series of MRJs is called the MRJ Assignment (MRJA). In addition, cost (MRJ) and expense (MRJA) are used to indicate MRJ and MRJA costs, respectively. Experiments on the Google cloud platform have confirmed the scalability and impact of SOSQL on current work [6].

Hai Liu, Pankaj Didwania, Mohamed Y. Eltabakh (2016), in their study, recommended the EXORD system to fill this space using big data query optimization, data correlations [22]. EXORD maintains two types of correlations, clapping all data records [1], and expects a lot of clutter from smooth correlation, but not all data records. To this end, a new three-step approach has been introduced: (1) to review and validate the validity of soft correlation value, and (2) to select soft correlations for operation by maintaining exclusive data records. And manage the question of correlations in optimization and (3) assign and leverage [20]. They proposed a new and original cost-benefit model in line with the highly beneficial soft correlation w.r.t [24]. Question workload mentioned when curing overhead [23]. They demonstrated the resolution of the problem (NP-Hard) [13] and suggested an approximate solution to the problem in multiple times [1]. EXORD can be integrated with various large state of the art data query optimization methods, e.g., indexing, segmentation and more. The EXORD prototype runs on Hadoop as an extension of the Hive Engine [7].

A. Regita Thangam and Dr. S. John Peter (2016), in their study, revealed that the database managing field has branched out to regard settings in which statistics are less accessible, queries are ever more complex, or data are saved tenuously; hence, there has been a confession that the accessible optimization techniques are inadequate. This, in turn, has led to a superfluity of new systems, usually laid under the universal banner of optimizing complex queries that are focused on rewriting the complex queries in a simple manner [21]. Query optimization is a drawback of database application presentation, particularly of those which accumulate history [2], i.e., does data warehousing. SQL is employed as a query language as the majority of data warehouses are based on database system which are relational or are extended relational. In this study [18], authors identified several common areas of concern, themes, and approaches that encompass this work, and the settings in which each part of the task is most suitable [19]. The goal of the paper was to act as a value-added, over the already presented papers on this subject [16], presenting not only a concise outline of each and every technique, but also defining a fundamental structure for comprehending the subject area of general query processing and also reducing the complexity of the queries so as to enhance the query processing and optimization engines [8].

M. Silly (2015), in his study, anticipated the principal setback while scheduling a specified set of episodic tasks, along with intermittent tasks [6], on a uniprocessor machine. Usually, each task is episodic, and worth of its episode relies upon the dynamism of the course of action it controls [14]. Additionally, some other tasks which are time-critical and said to be irregular or intermittent can take place and have to be processed at impulsive times [33]. Author in this paper discussed how this stratagem could be incorporated in a dispersed scheduler by stating that when an irregular task lands at a node, either that task is implemented locally, or it transfers to a different node which offers it within the best response-time [17]. This shows that any group of episodic or irregular tasks which meets the Conditions (C) can be programmed using the scheduling algorithm Earliest Deadline First (EDF) [6]. It also tackles scalability of a group of jobs having specific discharge time. For a group of irregular tasks having specific discharge time, it reveals that the conditions C is essential and adequate for scheduling [4]. This paper also helps in deciding whether a group of episodic jobs with specific discharge time is schedulable or not [9].

Yachen Liu, Hai Liu, Dongqing Xiao, and Mohamed Y.Eltabakh (2018), In their study, found that there was an abundant relationship between data attributes and that the majority were built-in application domains [15]. These associations, if managed in an organized and efficient manner, facilitate different adaptation opportunities [2]. Sadly, state-of-the-art systems are deeply optimized to optimize all aspects, built-in for relational databases, for example, random I / O access, selection selectivity and secondary index, which are generally contemporary large-scale. Data related to infrastructure. , Like, Spark and Hadoop. In this study [13], the EXORD + system was proposed to use the relationship between data in big data query optimization. EXORD + allows two types of correlations - hard (which does not allow exceptions) and smooth (which allows exceptions) [5]. To this end a new three-step approach has been introduced: (1) validation and review of the value of smooth correlations, (2) selecting and establishing smooth correlations for operations, and (3) increasing and taking advantage of question correlation optimization. The most beneficial smooth correlation for the refresh cost-benefit model to adjust is w.r.t [2]. The specified question is the workload. He demonstrated the resolution of the problem (NP-Hard) [32] and suggested an approximation to solve it effectively in multiple times. Additionally, additional algorithms for maintenance and efficiency bring the system up to date under data add and workload changes. The EXORD + prototype is implemented as an extension of the Hive Engine above Hadoop [11]. Experimental evaluation reveals the ability of EXORD + to achieve speeds in excess of 10x at the same time, enabling nominal storage costs [10].

Query processing and optimization in multi-processor environments take place in two parts, the first evaluation query should be reduced to the total work and the minimum workload on the processor should be evenly distributed [30]. Server query rewrite technology for cost-based optimization in platform independent environments. These methods are important in optimizing the use of time and resources [31]. Scalable SQL Query Optimizer based on a comprehensive evaluation of SOSQL using standard TPC-H datasets and the Google cloud platform. It improves SQL query processing efficiency during the optimization and excellence phase [34]. Chelsea Mafria et al. Examines how question processing performance is affected by future memory and consolidation architectures. Using the query workload with AQSIOS and the HMMSim hybrid memory simulator determines how the query processing will be affected [35].

3. QUERY OPTIMIZATION PROCESS:

Query optimization is an effective way to execute SQL queries. The low-cost plan is considered to be the best implementation plan that the optimizer seeks for the SQL statement, to find out the views of the entire candidate [13]. SQL is a non-formal language, so the optimizer is open to rebuild, rearrange and process any array. Problems in Query Optimization Fields [14]

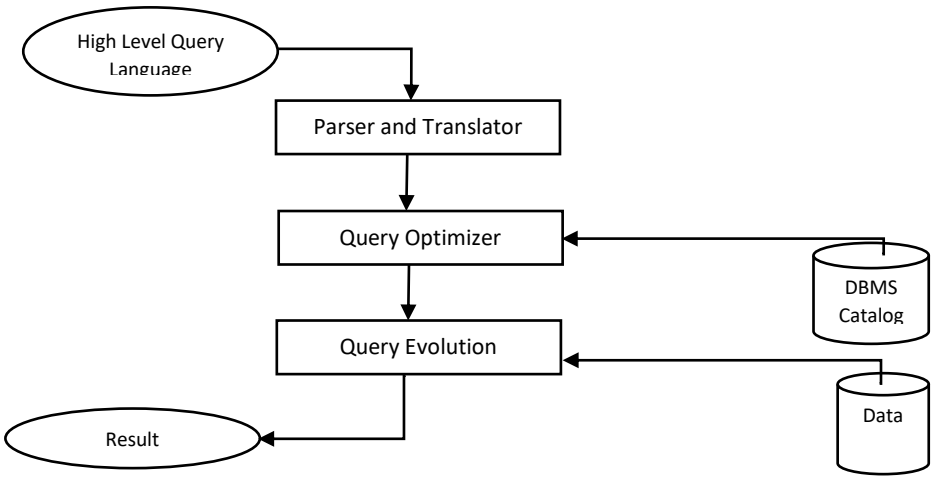

Figure 2: Query Processing

Query Rewrite - Using semantic attributes from one SQL query to another.

Selecting a query implementation plan - join a single query block (i.e., S-P-J block) and a combination [15].

Cost Estimation - In order to differentiate between schemes, we need to estimate their value through the data in the database [24].

3.1 EXISTING SYSTEM:

Multiple Query Optimization (MQO) is an eminent setback in database research. It explains how competently fabricate responses to a group of queries with similar tasks [21]. In a group of queries, each and every query has a set of substitute implementation plans and every plan has a set of jobs. MQO aspires to select a suitable plan for every query and reduces the entire implementation time by carrying out the common [7] tasks only once. The MQO is considered as an NP-Hard optimization problem where their algorithms such as heuristics and genetic algorithms are mostly approximate or produce near optimal solution [22]. Since these resolutions for MQO troubles should uncover the set of plans that produce the answer of each query and included in the minimum cost global execution plan (i.e., a set of plans that answer every and each one of a set of queries) [13]. Whereas, many optimization solutions have been introduced to find the optimal plan based on materialized [6] results (i.e., intermediate results and final answer) which produced from earlier queries. Figure 3 depicts the three basic multi-query optimization techniques listed as follows [12]:

I. Naive technique; each query runs isolated from other queries.

II. Grouping technique; only one instance of the query will be run for a set of identical queries.

III. Materialization technique; when some queries materialized their results or part of their results to other queries [23].

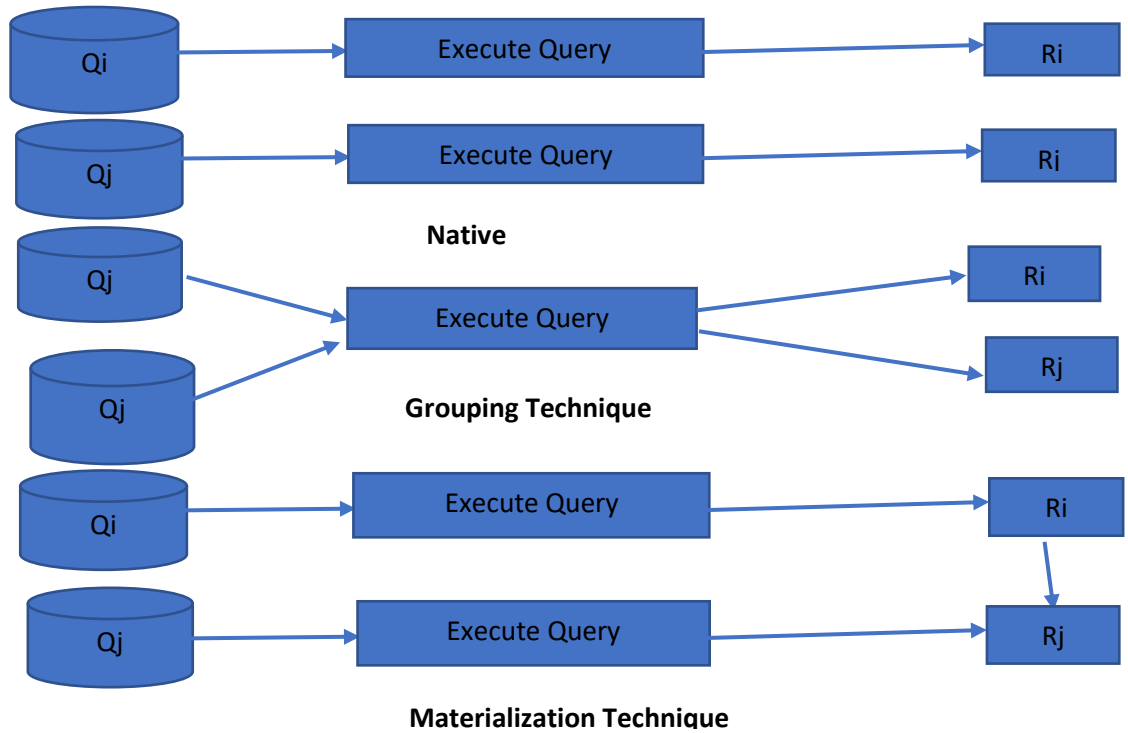

Figure 3. Multi-Query Optimization Techniques

3.2 PROPOSED SYSTEM:

A system was implemented and implemented by a combination of Ant Colony Algorithm and Genetic Algorithm. GA has powerful elasticity, swift global searching capability with superior population scattering facility for widespread amplitude of responses [11]. On the other hand, ACO has low population's spreading capacity, equivalent handing out [20], high convergence rate, and global penetrating skill with a positive feedback machinery. By employing the blend of both algorithms, best and positive features of both can perk up the performance while beating the disadvantages of each [13]. The performance of GA depends on the size of the population and the operator operating on the algorithm. If the population size is low, then it congregates faster [11]. To accomplish a single query, a number of implementation plans may be available. Every implementation plan will be implemented equivalently using an Ant Colony algorithm [10]. And out of these implementation plans, the finest one will be chosen and will be employed in future query implementation using Genetic Algorithm [18]. A flow of this technical procedure is shown in the figure 4 [17].

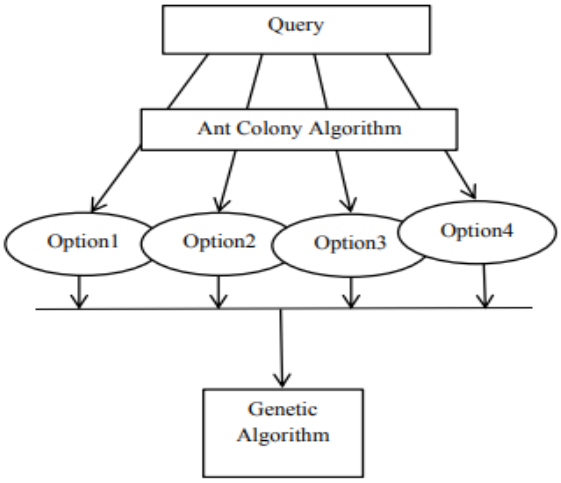

Fig 4: Optimizing solution using a combination of Ant Colony Algorithm & Genetic Algorithm

ANT COLONY BASED OPTIMIZATION(ACO)

ACO is based on real Ant Colonies and artificial System. It is used to solve discrete optimization problem [35][36][37].

ACO algorithms steps:

Step 1. Initiation of parameters. The parameters of the ACO algorithm are enabled.

Step 2. Classification of nodes: Dividing the nodes into center nodes and the geographical coordinates of the nodes and the classification results are used to form the next part of the algorithm.

Step 3. M ants are randomly placed in n nodes and this node is added to the ant tab list.

Step 4. Once the ant selection is complete, the path length is calculated. The relevant tab list is then edited. Repeat step-3 until complete ant seizures occur. The current correct path length is saved and the global iteration path is updated in this iteration.

Step 5. Update the pheromone: The pheromone is updated worldwide in the right way according to the equation in the improved update rules of the pheromone.

Step 6. Repeat control: Go back to the redirected counter

Step 4. Otherwise, the algorithm is terminated and the correct solution is output.

In this manner, the population accessible for genetic algorithm will be cut and ant colony algorithms will congregate quicker [19]. And this way out will be a dynamic one for the dimension of data over the server [17].

3.2 CONCLUSION:

Presently, big data are in boom and handling such a huge volume and variety of data is a big dare. This paper discusses about big data, ant colony optimization technique and query process [9]. Ant colony optimization technique is implemented in Big data which uses tools, softwares and hardwares for the manipulation of its data [8].

The apprehension of hybrid techniques of an ACO Algorithm in the direction of optimization of distributed database queries is yet a new and fresh field. Currently [15], several researches in the context of the conception and execution of ACO hybrids to resolve different kinds of optimization troubles are in evolution [3]. The outcomes have suggested that ACO hybrids are effective and practicable in problems relating to the optimization [14]. In addition, the researchers revealed that this implementation demonstrated practical solutions in relational to the implementation of algorithms, as well as how often the database management system expanded and reached the question with the size of the question. [7]. Several has been done and yet a lot of opportunity remains so as to create optimized resolutions and to filter search tactics using ACO hybrids for the Distributed Database' Queries particularly when the magnitude and intricacy of the associations augments with a number of constraints manipulating the query [6].

REFERENCES

[1] Lingling Shi and Dongzhi He (2019) Multi-join Query in Database Based on Genetic and Ant Colony Algorithm Optimization.
[2] Swati V. Chande and Preeti Tiwari (2019), Join Query Optimization Using Genetic Ant Colony Optimization Algorithm for Distributed Databases.

[3] Z. R. Ren, R. Skjetne, and Z. Gao, "A crane overload protection controller for blade lifting operation based on model predictive control," Energies, vol. 12, no. 1, p. 50, 2019.

[4] T. Yilmaz, R. Ozcan, I. S. Altingovde, and Ö. Ulusoy. (2019). Improving educational web search for question-like queries through subject classification. Information Processing & Management, 56(1), pp.228-246

[5] S. Karimzadeh, and S. Olafsson. (2019). Data clustering using proximity matrices with missing values. Expert Systems with Applications, 126, pp.265-276.

[6] Y. Wu, M. Gong, W. Ma, and S. Wang. (2019). High-order graph matching based on ant colony optimization. Neurocomputing, 328, pp.97-104.

[7] C. Huang, H. Xu, L. Xie, J. Zhu, C. Xu, and Y. Tang. (2018). Large-scale semantic web image retrieval using bimodal deep learning techniques. Information Sciences, 430, pp.331-348.

[8] E. Papenhausen, and K. Mueller. (2018). Coding Ants: Optimization of GPU code using ant colony optimization. Computer Languages, Systems & Structures, 54, pp. 119-138

[9] G. Deepak, and J. S. Priyadarshini, (2018). Personalized and Enhanced Hybridized Semantic Algorithm for web image retrieval incorporating ontology classification, strategic query expansion, and content-based analysis. Computers & Electrical Engineering, 72, pp. 14-25 .

[10] S. Jiang, M. Lian, C. Lu, Q. Gu, S. Ruan, and X. Xie, "Ensemble prediction algorithm of anomaly monitoring based on big data analysis platform of open-pit mine slope," Complexity, vol. 2018, Aug. 2018, Art. no. 1048756.

[11] Q. Zhang, H. L. Chen, J. Luo, Y. T. Xu, C. Wu, and C. Li, "Chaos enhanced bacterial foraging optimization for global optimization," IEEE Access, vol. 6, pp. 64905–64919, 2018.

[12] S. K. Guo, R. Chen, H. Li, J. Gao, and Y. Q. Liu, "Crowdsourced Web application testing under real-time constraints," Int. J. Softw. Eng. Knowl. Eng., vol. 28, no. 6, pp. 751–779, 2018.

[13] Liu, Y.X.; Gao, C.; Zhang, Z.L.; Lu, Y.; Chen, S.; Liang, M.; Tao, L. Solving NP-hard problems with Physarum-based ant colony system. IEEE/ACM Trans. Comput. Biol. Bioinform. 2017, 14, 108–120

[14] Borrotti, M., Minervini, G., De Lucrezia, D. and Poli, I., 2016. Naïve Bayes ant colony optimization for designing high dimensional experiments. Applied Soft Computing, 49, pp.259-268.

[15] Soorya M , Swaraj K.P , "Elliptic Curve Based Data Scrambling with Encryption for Security of Big data ",International Journal of Innovative Research in Science, Engineering and Technology , Volume 5, Special Issue 14, December 2016

[16] Alfredo Cuzzocrea , "Algorithms for Managing, Querying and Processing Big Data in Cloud Environments ",12 January 2016; Accepted: 27 January 2016; Published: 1 February 2016.

[17] Krynicki, K., Houle, M.E. and Jaen, J., 2016. An efficient ant colony optimization strategy for the resolution of multi-class queries. Knowledge-Based Systems, 105, pp.96-106.

[18] Sheshikala, M., Rao, D.R. and Prakash, R.V., 2016. Parallel Approach for Finding Co-location Pattern–A Map Reduce Framework. Procedia Computer Science, 89, pp.341-348.

[19] Asadi, S. and Shahrabi, J., 2016. ACORI: a novel ACO algorithm for rule induction. Knowledge-Based Systems, 97, pp.175-187.

[20] He, Z.N.; Yen, G.G. Visualization and Performance Metric in Many-Objective Optimization. IEEE Trans. Evol. Comput. 2016, 20, 386–402.

[21] Cheng, Y.F.; Shao, W.; Zhang, S.J. An Improved Multi-Objective Genetic Algorithm for Large Planar Array Thinning. IEEE Trans. Magnet. 2016, 52, 1–4.

[22] Zeng, M.F.; Chen, S.Y.; Zhang, W.Q.; Nie, C.H. Generating covering arrays using ant colony optimization: Exploration and mining. Ruan Jian Xue Bao/J. Softw. 2016, 27, 855–878. Available online: http://www.jos.org. cn/1000-9825/4974.htm (accessed on 3 October 2017).

[23] Wang, X.Y.; Choi, T.M.; Liu, H.K.; Yue, X.H. Novel Ant Colony Optimization Methods for Simplifying Solution Construction in Vehicle Routing Problems. IEEE Trans. Intell. Transp. Syst. 2016, 17, 3132–3141.

[24] Wang, Z.Y.; Xing, H.L.; Li, T.R.; Yang, Y.; Qu, R.; Pan, Y. A Modified Ant Colony Optimization Algorithm for Network Coding Resource Minimization. IEEE Trans. Evol. Comput. 2016, 20, 325–342.

[25] Juang, C.-F.; Jeng, T.-L.; Chang, Y.-C. An Interpretable Fuzzy System Learned Through Online Rule Generation and Multiobjective ACO with a Mobile Robot Control Application. IEEE Trans. Cybern. 2016, 46, 2706–2718.

[26] Verdaguer, M.; Clara, N.; Monclús, H.; Poch, M. A step forward in the management of multiple wastewater streams by using an ant colony optimization-based method with bounded pheromone. Process Saf. Environ. Protect. 2016, 102, 799–809.

[27] Kucukkoc, I.; Zhang, D.Z. Mixed-model parallel two-sided assembly line balancing problem: A flexible agent-based ant colony optimization approach. Comput. Ind. Eng. 2016, 97, 58–72.

[28] S. B. Chen, G. Lv, and X. F. Wang, "Offensive strategy in the 2D soccer simulation league using multi-group ant colony optimization," Int. J. Adv. Robot. Syst., vol. 13, no. 1, p. 25, 2016.

[29] M. Marzband, E. Yousefnejad, and A. Sumper, "Real time experimental implementation of optimum energy management system in standalone microgrid by using multi-layer ant colony optimization," Int. J. Elect. Power Energy Syst., vol. 75, pp. 265–274, Feb. 2016.

[30] Sukheja Deepak, Singh Umesh Kumar, "Query Processing and Optimization of Parallel Database System in Multi Processor Environments", Sixth Asia Modelling Symposium

[31] Vamsi Krishna Myalapalli, ASN Chakravarthy, "Revamping SQL Queries for Cost Based Optimization", International Conference on Circuits, Controls, Communications and Computing (I4C)

[32] L. Y. Zuo, L. Shu, S. B. Dong, C. Zhu, and T. Hara, ''A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing,'' IEEE Access, vol. 3, pp. 2687–2699, 2015.

[33] J. Bagherinejad and M. Dehghani, ''A multi-objective robust optimization model for location-allocation decisions in two-stage supply chain network and solving it with non-dominated sorting ant colony optimization,'' Sci. Iranica, vol. 22, no. 6, pp. 2604–2620, 2015.

[34] Yi Shan, Yi Chen, "Scalable Query Optimization for Efficient Data Processing using MapReduce", IEEE International Congress on Big Data

[35] Chelsea Mafrica, John Johnson, "Stream Query Processing on Emerging Memory Architectures", IEEE Non-Volatile Memory System and Applications Symposium (NVMSA)

[36]] J. B. Escario, J. F. Jimenez and J. M. Giron-Sierra, "Ant colony extended: Experiments on the travelling salesman problem", Expert Systems with Applications, vol. 42, no. 1, (2015), pp. 390-410.

[37] F. Li and M. L. Jin, "GACO: a GPU-based high performance parallel multi-ant Colony Optimization algorithm", Journal of Information and Computational Science, vol. 11, no. 6, (2014), pp. 1775-1784.

[38] Z. L. Zhang, C. Gao, Y. X. Liu and T. Qian, "A universal optimization strategy for ant colony optimization algorithms based on the Physarum-inspired mathematical model", Bio inspiration and Bio mimetics, vol. 9, no. 3, (2014), pp. 1-7.

[39] Rong Gu, Xiaoliang Yang, Jinshuang Yan, Yuanhao Sun, Bing Wang, Chunfeng Yuan, and Yihua Huang, "SHadoop: Improving MapReduce performance by optimizing job execution mechanism in Hadoop clusters", Journal of parallel and distributed computing, vol. 74, no. 3, pp. 2166-2179, 2014.

[40] Lalitha Viswanathan, Alekh Jindal, and Konstantinos Karanasos, "Query and resource optimization: Bridging the gap", In IEEE 34th International Conference on Data Engineering (ICDE), IEEE, pp. 1384-1387, 2018.

[41] Yuansheng Lou, and Feng Ye, "Research on data query optimization based on SparkSQL and MongoDB", In 17th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), IEEE, pp. 144-147, 2018.

[42] Zujie Ren, Na Yun, Weisong Shi, Youhuizi Li, Jian Wan, Lihua Yu, and Xinxin Fan, "Characterizing the effectiveness of query optimizer in spark", In 2018 IEEE World Congress on Services (SERVICES), IEEE, pp. 41-42, 2018.

BIOGRAPHIES

Mr. Deepak Kumar is a Research scholar in the department of Computer Science and Engineering, Birla institute of technology Mesra, Ranchi. He has completed his Bachelor Degree from Kuvempu University Shimoga, Karnataka in 2010 and Master Degree from Vinoba Bhave University, Hazaribagh, in 2014. His current research interest includes Big Data Analytic, Data Mining, and Machine Learning.

Dr. Vijay Kumar Jha is working as an Associate Professor in the Department of Information Technology, Birla institute of technology Mesra, Ranchi. He has completed BE (Electronics) from SIT Tumkur in 1996, M.Sc. Engineering in Electronics from MIT Muzaffarpur in 2007 and PhD in Information Technology (Data Mining) from MIT Muzaffarpur in 2011. He has been associated with Birla Institute of Technology, Mesra, Ranchi since 2001. His research interests includes Big data Analysis, Data mining, Network Security, ERP etc.