



SCALABLE QUERY PROCESSING IN BIG DATA ENVIRONMENTS WITH HADOOP DISTRIBUTED FILE SYSTEM AND PARTICLE SWARM OPTIMIZATION

S.Tamil Selvan^{*1}, R.Punitha Gowri², V.Vijey Nathan³, C. Saravanan⁴

¹Department of Computer Science and Design, Erode Sengunthar Engineering College, Erode, India.

²Department of Computer Science and Design, Erode Sengunthar Engineering College, Erode, India.

³Department of Computer Science and Engineering, SRM TRP Engineering College, Tiruchirappalli, India.

⁴Department of Information Technology, Velalar College of Engineering and Technology, Erode, India.

¹<http://orcid.org/0009-0004-8835-7342>, ²<http://orcid.org/0009-0007-9490-9387>

³<http://orcid.org/0009-0002-6807-4324>, ⁴<http://orcid.org/0000-0003-0076-4708>

E-mail: *stamilselvan@esec.ac.in, punithagowrircs@esec.ac.in, vijeynathan@gmail.com, saravananvanjii@gmail.com

ARTICLE INFO

Article History

Received: March 20, 2026

Reviewed: April 20, 2026

Accepted: July 01, 2026

Published: July 31, 2026

Keywords:

Hadoop Distributed File System,
Optimization,
Query processing,
Bigdata,
Particle Swarm Optimization.

ABSTRACT

Big data are enormous, intricate sets of information that are too large, too quick, and too difficult to handle using conventional methods. They consist of both structured and unstructured data. Due to the alarmingly high rates of data generation, there is pressure to adopt simple and expensive procedures for data recovery and storage. Conventional database systems cannot handle the processing power of big data. In order to evaluate and query this enormous amount of data, new technologies have recently been created. In this work, a novel hybrid approach for efficient big data query processing through a two-phase approach is introduced. In the first phase, the Hadoop Distributed File System (HDFS), Map Reduce are utilized. Second phase involves optimization using Particle Swarm Optimization (PSO). The proposed algorithm aims to enhance search speed by strategically optimizing various processes. This two-step methodology combines the parallel processing capabilities of HDFS Map-Reduce with the optimization power of PSO, contributing to an overall improvement in query processing efficiency for large-scale datasets. Experimental results demonstrate significant outperformance of the model in expediting query processing.



Copyright ©2026 by authors and Galileo Institute of Technology and Education of the Amazon (ITEGAM). This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

I. INTRODUCTION

The "3 Vs" of Figure 1 Volume, Velocity, and Variety define big data. Traditional query processing methods and database management systems face serious difficulties as a result of these features. Traditional methods find it difficult to deal with the sheer amount of data, speed at which it is produced, wide range of data formats, and requirement to guarantee the accuracy and quality of the data. As such, it is now essential to create new query processing techniques suited to the particular requirements of large data. These days, IT is centralized in all businesses and organizations, where users work with a wide range of computer programs. Numerous data types, such as facts, statistics, numbers, text, and images, as well as information that has been processed and made relevant for appropriate use, are allocated by these applications. A database is an information and data collection of this type. Modern database user interfaces are highly versatile, which makes them important for optimization. Users must deal with a massive amount of data due to the growing use of database management systems. As a result, in order to fulfill a user's request, it is imperative to store this data so that it can be quickly retrieved from the information.

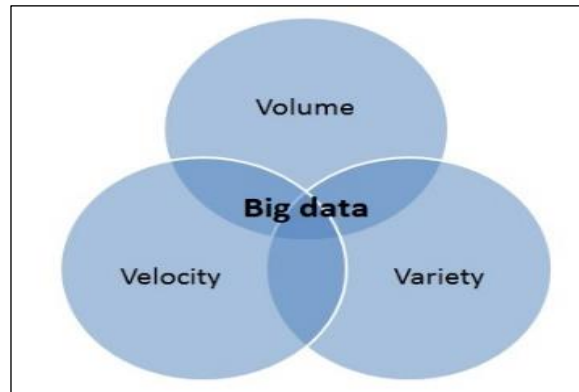


Figure 1: 3 V's of Bigdata.

Source: Authors, (2026).

When it comes to organizing information, databases are really useful. It gives the user the freedom to gather accurate, trustworthy, and timely data for efficient decision-making. As a result, database systems are becoming more and more important. Finding the optimal query optimization strategy becomes more difficult as a result of the daily rise in query complexity. For a very long time, query optimization in databases has been a crucial problem in many different industries. The user must reformulate the complex query with a more efficient and computationally equivalent one in order to lower the execution cost. Main key contributions of proposed work are listed as follows:

- The proposed hybrid approach has two phase of query processing in big data to retrieve the data efficiently.
- In the first phase, Entropy computation is used to extract and manage properties including closed frequent pattern, support, and confidence. Repeated data is removed using the Hadoop Distributed File System and Map Reduce.
- Second phase involves optimization is done using Particle Swarm Optimization.
- Experimental results show that query processing performance of proposed method accuracy was higher and execution time also lower than the traditional methods.

Rest of the paper is organized as follows: Sections 2 discusses related publications. Section 3 shows description of suggested approach Section 4, the suggested hybrid approach is tested and compared to existing approaches. Finally, Section 5 describes all finding and the scope of the study.

II. RELATED WORKS

According to [1] demonstrated the critical role big data plays in the modern society. Big data has many uses in applications, but it also presents difficulties. They have described big data, including veracity, using the four Vs idea. They have addressed a few basic issues that must be resolved in order to turn issues into opportunities. They have discussed developments, big data research, and six key difficulties that arose when processing unstructured data. And also suggested unresolved research issues in the field of big data. Another important problem with heterogeneous Hadoop clusters, as highlighted by [2] is namespace constraints imposed by systems. In a Hadoop context, the issue is sharing name nodes and keeping metadata across name nodes for a large number of files. A single namespace server's capacity to handle and store data is limited [3] expressed apprehension regarding the difficulties presented by supercomputing community. They raised concerns about what happens when data is stored in zettabytes or petabytes. The main topic to concentrate on is the enormous amount of data generated by numerical simulations as a byproduct of supercomputing processes. Three perspectives can be used to analyze the parallelism in large datasets: server applications that require a high bandwidth network to handle all queries, and data applications with a high volume of incoming and outgoing movements. According to [4] there are equity problems with the distinct, profiled critical execution of inquiries. Prediction-based systems are proposed for the current MapReduce-based was developed to scaffold the semantic gap between MapReduce runtimes. Moreover, the query compiler facilitates efficient query planning for rapid and reasonable large-scale data analytics. To get around design flaws and boost large-scale cluster performance, the author conducted tests. [5] have specified the use of comparable input/output approaches for scientific application management.

To maximize I/O throughput, they integrated a two-phase I/O method and data sieving. For large workloads, the suggested effort focuses on providing parallel access to shared datasets. In their clinical studies [6] endorsed the application of naïve Bayes to comprehend the human neuroimaging framework. Although there are various stimulus phases in the research, data collection, classification process offered a very clever look into how data categorization may be controlled if done appropriately. Each response collected as a stimulus offers a comparable testing ground to identify the appropriate heuristic method for data classification of simple activities with complicated surroundings. In [7] conducted a theoretical examination of the PSO algorithm and discussed its history and conception. The specific future research directions this summary suggests are what make it unique [8] suggested a hybrid approach combining PSO and GA to optimize the deployment of an antenna array for the purpose of finding the throughout a substation. Initially, a Direction-of-Arrival (DOA) estimation algorithm was introduced to evaluate its suitability for various array topologies. According to [9] looked at a method for hybridizing PSO and DE, two heuristic algorithms, in order to solve slew-maneuver time-optimal issues with solution route limitations and prevent stagnation. The combined PSODE technique handled the stagnation quite well, reducing computational time compared to merely using DE by 40%. A Dynamic Multi-Swarm Differential Learning Particle Swarm Optimizer was presented [10] By using a local searcher such as the Quasi-Newton method to improve exploitation capability while simultaneously utilizing exploration capability included in the differential mutation, it has an excellent capability of both exploitation and exploration.

By utilizing an Instrumented Indentation Test (IIT), [11] provided a way to extract the mechanical characteristics of its bulk materials utilizing a variety of loading-unloading curves. Finite iterative element simulation in conjunction with Simulated Annealing PSO was utilized to extract the mechanical parameters, such as the strain-hardening exponent (n) and strain-hardening rate (K), from the IIT data.

III PROPOSED WORK

A proficient hybrid structure has been suggested. This system includes data classification and an effective automated preprocessing method for effective data management in large data. Organizations keep many databases to handle and store big data (BD), which comes in a variety of data models and is stored in enormous volumes. For business, querying and BD analysis are essential for gaining insight. This paper improved the BD query optimization process by utilizing both the HDFS map-reduce approach and the PSO algorithm. Figure 3 displays the block diagram for the suggested methodology. There are two stages to the proposed work:

1. Big data arrangement

- Pre-processing
- Feature Extraction

2. Query optimization

Collect and use BD for preprocessing first. The features are extracted from the original data in the second stage, which involves finding the support and confidence value after determining the closed frequent itemset. Finally, apply the entropy method to this value. PSO technique is used in last stage to organize the relevant data. These are explained in brief below.

III.1 BIG DATA ARRANGEMENT

The primary purpose is to convert the unstructured bigdata into an organized manner.

III.1.1 Pre-Processing

In pre-processing phase firstly, it uses the SHA 512 algorithm to calculate hash value (HV) of data during preprocessing. Next, use the HDFS framework to execute the map-reduce using this HV.

HDFS

Large data sets can be reliably and scalable stored and managed using HDFS, a distributed file system. It is made to function with the Hadoop MapReduce (MR) programming model, which is illustrated in figure 2 and is a well-liked method of handling massive data sets. Fast data movement between nodes of HDFS's primary characteristics. When addressing a query, the HDFS has "two" phases. For example, it contains a reduction function Cf and a map function Pf, which are both written as

$$H(Dn) = [Pf, Cf] \quad (1)$$

Map () function

Map () function is the initial function of Map/Reduction tool. Master node (MN) is where this function is dominant. It divides up input data or breaks it up into multiple smaller subprocesses. The worker nodes, which handle these small processes, receive further dispersion of these subprocesses. The MN is then given an acknowledgement.

$$Pf = \text{map}(H(Dn)) \quad (2)$$

where Pf is the result of the map () function and map () denotes the mapping function Reduce () function Reduce () gathers the complete outcomes of each suboperation and combines them to produce an aggregated set of decision-based results that are provided in recognition of the initial high demands. The following formula displays the reduction function.

$$Cf = \text{reduce}(Pf) \quad (3)$$

where reduce () is the function that reduces the components, Pf is the output of the mapping function, and reduced set of data is Cf.

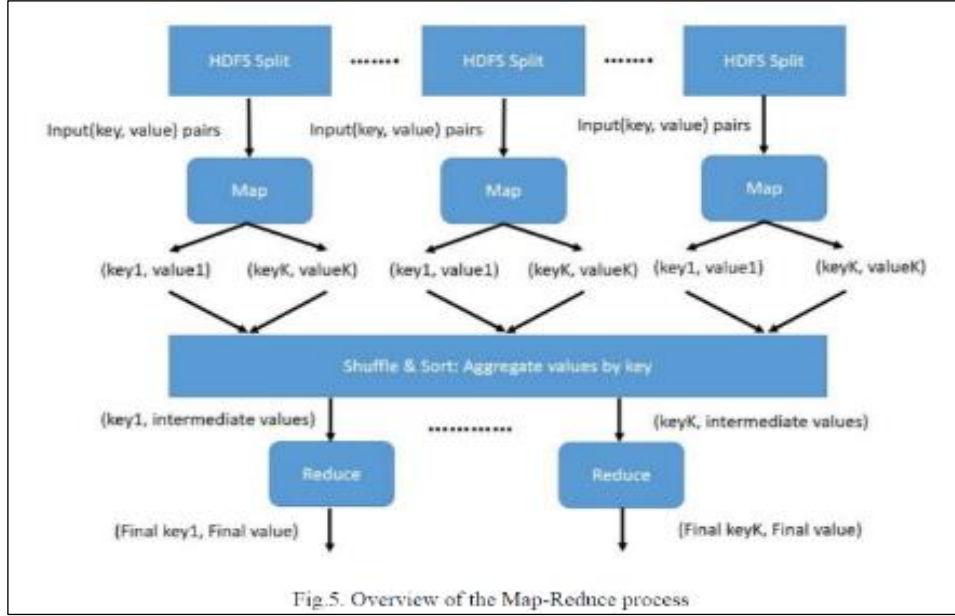


Figure 2: Map-Reduce Process.

Source: Authors, (2026).

III.1.2 Feature extraction

Feature extraction helps in the reduction of the dimensionality of data which is needed to process the data effectively. After eliminating duplicates, crucial features like closed frequent itemsets, support, and confidence are derived from the initial data.

Closed Frequent Itemset ($o(fi)$)

Closed frequent itemsets are frequent itemsets that cannot be extended by adding more items while maintaining the same support. The closed frequent itemset is the maximal frequent itemset, indicating the highest occurrences of a specific item in the dataset.

$$o(f_i) = \{o(f_1), o(f_2), o(f_3), \dots, o(f_n)\} \quad (4)$$

Support(δt)

Once closed frequent itemsets are identified, calculate the support for each itemset. It is usually expressed as a percentage. The support of association rule $U \rightarrow V$ is the percentage of transactions in the database containing both itemsets U and V .

$$\delta_t = Support(U \rightarrow V) = P(U \cup V) \quad (5)$$

Confidence (A_e)

Confidence is a measure of the reliability or strength of an association rule. The confidence of an association rule $U \rightarrow V$ is the percentage of transactions with itemset U that also includes itemset V . It is computed using conditional probability in relation to itemset support, expressed by the equation:

$$A_e = Confidence(U \rightarrow V) = P\left(\frac{V}{U}\right) = \frac{P(U \cup V)}{P(U)} \quad (6)$$

Where $P(U \cup V)$ denotes the total number of transactions that include the itemsets U and V , while $P(U)$ denotes the total number of transactions that include the itemset U .

Entropy calculation

A measure of a set of data's uncertainty or disorder is called entropy. The entropy calculation is used in this subsection to estimate support and confidence. The entropy of a probability distribution, which is connected to its variance, can be calculated for a given dataset using the following formulas.

$$Entropy(A_e) = - \sum_{e=1}^N P(A_e) \log_2 \{P(A_e)\} \quad (7)$$

$$Entropy(\delta_t) = - \sum_{t=1}^N P(\delta_t) \log_2 \{P(\delta_t)\} \quad (8)$$

Here, $P(\delta_t)$ and $P(A_e)$ indicate the likelihood that, at random, the n th support and confidence values will be obtained from the set.

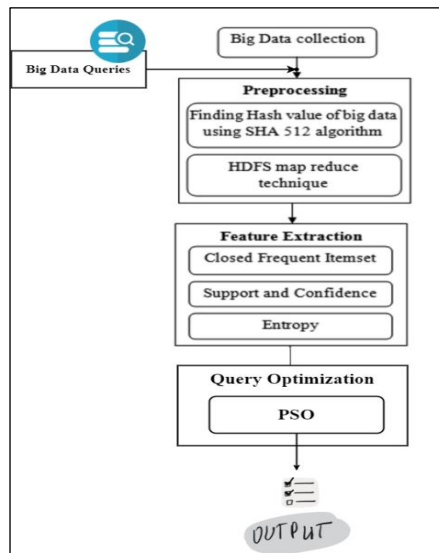


Figure 3: Block diagram for the proposed methodology.
Source: Authors, (2026).

III.2 QUERY OPTIMIZATION

For solving optimization issues, one popular population-based metaheuristic approach is particle swarm optimization (PSO). This algorithm mimics how birds in a flock interact with one another in order to reach their feeding objective. A swarm of birds approaches their food target by combining their social and self-experience. They constantly reposition themselves to establish the finest possible formation based on their own best position as well as the best position of the entire swarm. Iterations are used by the nature-based swarm-intelligence algorithm. It begins with a population of potential solutions, referred to as a swarm. Each particle in this instance stands for a possible fix for the stated issue.

Every iteration updates the velocity and position of every member in the population. Global best value (gbest) and personal best value (pbest) serve as the foundation for these improvements. The particles in gbest model are influenced by the best position identified by any member of the entire population, as opposed to the pbest model where the particles are determined by their own position. Each particle will then converge to this new location as a result.

PSO algorithm

Step 1 The population (particles) has been defined as the $X_1 - X_{n-1}$ weights.

Step 2 For the velocity updating based on weights, the following formula has been applied.

$$v_i(t+1) = wv_i(t) + c_1r_1[xi(t) - xi(t)] + c_2r_2[g(t) - xi(t)]$$

where $v_i(t)$:represents the particle velocity at time t , $xi(t)$:the particle position at time t , $\hat{xi}(t)$:the individual particle best at time t , and $g(t)$ the global best swarm best at time t . c_1 and c_2 : Cognitive coefficients; w : Inertia coefficient

Step 3 The optimal position has then been updated in accordance with needs by comparing the current positions. As needed, the best spots have been updated. Also, the weights have been revised. For w , the range taken into consideration is 0.8–1.2. Here, c_1 and c_2 have a value of 2.

Step 4 The position update or the subsequent equations have been utilized.

$$xi(t+1) = xi(t) + v_i(t+1)$$

Step 5 Classifications have been established and new posts have been allocated.

The PSO algorithm uses the extracted data to find the best answer to the user's inquiry. When compared to other approaches, this one produces better outcomes and provides the best answers for the given situation. As such, the PSO technique is generally appropriate for a class of high-dimensional optimization problems with extremely precise solutions.

IV. RESULTS AND DISCUSSION

IV.1 EXPERIMENTATION AND DATASET

This section explores the NKM methodology's assessment of the suggested query optimization strategy. Performance is examined and contrasted with state-of-the-art methods with different data sizes (10–50 MB). Test data for text classification is taken from publicly available datasets, most notably Reuters-21579. This dataset, which comes from the news agency Reuters, is arranged according to its economic significance. Specifically, the proposed method uses the Reuters-21578 dataset to choose the top 10 classes, with a test set of 1100 articles and a training set of 3875 documents. These records, which came from the Reuters Newswire in 1987, were put together and arranged by Carnegie organizations and Reuters Ltd. David D. Lewis made them available for study in 1990. 21,578 news items total from the dataset, all of which fall into one or more categories. Out of 21,578 documents, only those pertaining to the four most popular topics—"earn (earning)," "acq (corporate acquisitions)," "money-fx (money market)," and "grain" categories. Table 1 shows the Reuters – 8 benchmark datasets.

Table 1: Reuters-8 (R8) benchmark dataset summary.

R8			
Class	# train docs	# test docs	Total # docs
Acq	1596	696	2292
Crude	253	121	374
Earn	2840	1083	3923
Grain	42	10	51
Interest	190	81	271
Money-fix	206	87	293
ship	108	36	144

Source: Authors, (2026).

IV.2 PERFORMANCE ANALYSIS

Here, the suggested hybrid method's and the current method's performances are compared in terms of statistical metrics, Precision (P), Recall (R), Accuracy (A), F-Measure (F), Retrieval Time (RT) and Execution Time (ET).

Table 2: Comparison table for proposed method with existing method based on Precision, Recall, Accuracy and F-measure.

Data size in MB	Proposed Method (HDFS-PSO)				Existing method (NKM)			
	P	R	A	F	P	R	A	F
10	89.2	87.9	88.5	86.4	84.6	83.4	85.4	82.9
20	90.4	89.2	89.7	88.6	85.9	84.8	86.7	84.6
30	91.3	90.4	90.6	89.3	87.5	86.2	87.9	86.5
40	92.1	91.5	91.2	90.8	88.7	87.8	89.2	89.7
50	93.5	93.2	92.9	92.2	90.2	89.5	90.4	90.6

Source: Authors, (2026).

Table 2 compares the precision, recall, accuracy, and F-measure of the proposed method with an existing method. The suggested approach yields 93.5% precision, 93.2% recall, 92.9% accuracy, and 92.2% F-Measure on a 50 MB data set. However, the current approach achieves 90.6% F-Measure, 90.4% Accuracy, 89.5% Recall, and 90.2% Precision. Visually, Figure 4 clarifies the accuracy results shown in Table 2 even more, demonstrating the improved performance of the suggested approach.

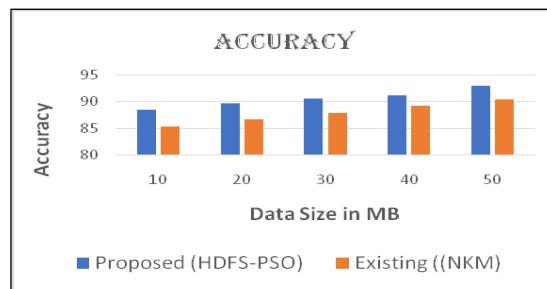


Figure 4: Comparative analysis for the HDFS – PSO with NKM in terms of accuracy.

Source: Authors, (2026).

Table 3 compares the performance of the proposed strategy with an existing method in terms of retrieval time and execution time, both measured in milliseconds (ms), with data sizes ranging from 10 MB to 50 MB.

Table 3: Illustrate the performance of proposed method with existing method based on retrieval time, execution time.

Data size in MB	Proposed Method (HDFS-PSO)		Existing method (NKM)	
	RT	ET	RT	ET
10	345	121	521	135
20	678	189	892	196
30	1487	208	1592	243
40	2571	245	2752	297
50	3489	293	3857	312

Source: Authors, (2026).

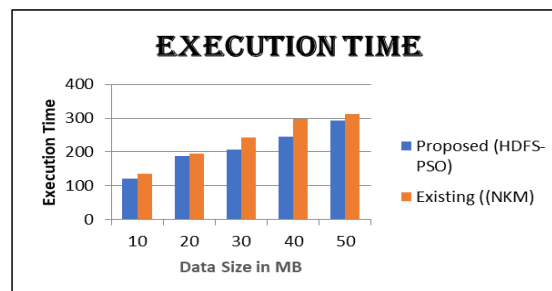


Figure 5: Comparative analysis for HDFS – PSO with NKM on execution time.

Source: Authors, (2026).

The current technique takes 3857 ms for retrieval and 312 ms for execution for a 50 MB data size, whereas the suggested method achieves a retrieval time of 3489 ms and an execution time of 293 ms. For all data sizes, the suggested approach requires less time for retrieval and execution. As a result, the suggested solution outperforms existing methods in terms of performance. Visually, Figure 5 clarifies the execution time results shown in Table 3, demonstrating the enhanced effective

V. CONCLUSIONS

This paper proposes a hybrid approach for query processing in big data, based on Particle Swarm Optimization and the Hadoop Distributed File System. When compared to the current query processing techniques, this hybrid approach is determined to be the most efficient and effective. This hybrid solution uses MapReduce and the Hadoop Distributed File System to remove duplicate data and extract attributes like confidence, support, and closed frequent patterns. Particle Swarm Optimization is used to optimize the query. The results of the proposed work's execution demonstrate and validate that the hybrid technique offers better accuracy and quicker query retrieval. With an average accuracy of 92.9%, the comparison analysis shows that the suggested model fared better than any of the other models in the current study. Additionally, the statistical outcomes of the F-measure (92.2%), recall (93.2%), and precision (93.5%) support the efficacy of the hybrid technique. As a result, this suggested system performs better than the current system. Future improvements to this system's efficiency could include the use of sophisticated optimization techniques and feature selection to speed up retrieval.

VI. AUTHOR'S CONTRIBUTION

Conceptualization: S.Tamil Selvan, R.Punitha Gowri, V.Vijey Nathan C. Saravanan.

Methodology: S.Tamil Selvan, R.Punitha Gowri, V.Vijey Nathan C. Saravanan.

Investigation: S.Tamil Selvan, R.Punitha Gowri, V.Vijey Nathan C. Saravanan.

Discussion of results: S.Tamil Selvan, R.Punitha Gowri, V.Vijey Nathan C. Saravanan.

Writing – Original Draft: S.Tamil Selvan, R.Punitha Gowri, V.Vijey Nathan C. Saravanan.

Writing – Review and Editing: S.Tamil Selvan, R.Punitha Gowri, V.Vijey Nathan C. Saravanan.

Resources: S.Tamil Selvan, R.Punitha Gowri, V.Vijey Nathan C. Saravanan.

Supervision: S.Tamil Selvan, R.Punitha Gowri, V.Vijey Nathan C. Saravanan.

Approval of the final text: S.Tamil Selvan, R.Punitha Gowri, V.Vijey Nathan C. Saravanan.

VII. ACKNOWLEDGMENTS

The author would like to appreciate the effort of the editors and reviewers.

VIII. REFERENCES

- [1] M. Malik, K. Neshatpour, S. Rafatirad, and H. Homayoun, "Hadoop workloads characterization for performance and energy efficiency optimizations on microservers," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 4, no. 3, pp. 355–368, 2018.
- [2] H. Alshammari, J. Lee, and H. Bajwa, "H2Hadoop: Improving Hadoop performance using the metadata of related jobs," *IEEE Trans. Cloud Comput.*, vol. 6, no. 4, pp. 1031–1040, 2018.
- [3] I. Ullah, M. S. Khan, M. Amir, J. Kim, and S. M. Kim, "LSTPD: Least slack time-based preemptive deadline constraint scheduler for Hadoop clusters," *IEEE Access*, vol. 8, pp. 111751–111762, 2020.
- [4] S. Kumar and M. Singh, "A novel clustering technique for efficient clustering of big data in Hadoop ecosystem," *Big Data Min. Anal.*, vol. 2, no. 4, pp. 240–247, 2019.
- [5] M. Soualhia, F. Khomh, and S. Tahar, "A dynamic and failure-aware task scheduling framework for Hadoop," *IEEE Trans. Cloud Comput.*, vol. 8, no. 2, pp. 553–569, 2020.
- [6] H. F. Wang and Y. P. Cao, "An energy efficiency optimization and control model for Hadoop clusters," *IEEE Access*, vol. 7, pp. 40534–40549, 2019.
- [7] D. Wang, D. Tan, and L. Liu, "Particle swarm optimization algorithm: An overview," *Soft Comput.*, vol. 22, no. 2, pp. 387–408, 2018.
- [8] M. Zhu, J. Li, D. Chang, G. Zhang, and J. Chen, "Optimization of antenna array deployment for partial discharge localization in substations by hybrid particle swarm optimization and genetic algorithm method," *Energies*, vol. 11, no. 1, p. 1813, 2018.
- [9] R. G. Melton, "Differential evolution/particle swarm optimizer for constrained slew maneuvers," *Acta Astronaut.*, vol. 148, pp. 246–259, 2018.
- [10] Y. Chen, L. Li, H. Peng, J. Xiao, and Q. Wu, "Dynamic multiswarm differential learning particle swarm optimizer," *Swarm Evol. Comput.*, vol. 39, pp. 209–221, 2018.
- [11] L. Huang, K. Guan, T. Xu, J. Zhang, and Q. Wang, "Investigation of the mechanical properties of steel using instrumented indentation test with simulated annealing particle swarm optimization," *Theor. Appl. Fract. Mech.*, vol. 102, pp. 116–121, 2019.