ISSN 1840-4855
e-ISSN 2233-0046

Original scientific article
http://dx.doi.org/10.70102/afts.2025.1834.226

FRAGMENTATION-ENABLED VM MIGRATION AND ENHANCED DATA SEARCHING IN BIG DATA SERVER ENVIRONMENT

G.S. Manjula[1*], Dr.T. Meyyappan[2]

[1]Research Scholar, Department of Computer Science, Alagappa University, Karaikudi, Tamil Nadu, India. e-mail: gsmanjula@yahoo.co.in,*
orcid: https://orcid.org/0000-0001-6166-9232
[2]Professor, Department of Computer Science, Alagappa University, Karaikudi, Tamil Nadu, India. e-mail: meyyappant@alagappauniversity.ac.in,
orcid: https://orcid.org/0000-0001-5568-1278

Received: August 22, 2025; Revised: October 04, 2025; Accepted: November 14, 2025; Published: December 30, 2025

SUMMARY

The proposed research paper suggests a hybrid system to improve speed and data extraction in a large data server setup, but the focus will be on how to maximize the use of resources and provide actionable information on the unstructured data. The architecture also incorporates fragmentation-enabled virtual machine (VM) migration and high-end data searching tools to enhance the system efficiency to a high degree. A new strategy of starting VM migration is proposed, which is based on cumulative values, including temperature and CPU usage, to perform real-time and dynamic resource allocation. The system adjusts the computing workload of the physical machines that are hosting several VMs by observing these parameters and migrating them to a different physical machine in case predetermined limits are surpassed. The migration of VMs made possible by this fragmentation results in better performance and resource utilization of the system, which ensures a smooth operation within the big data server environments. Also, the architecture encompasses an improved similarity measure algorithm, which deals with the challenges of locating and mining information from unstructured data. This is the best way of optimizing the pretreatment, extraction, and representation of unstructured data, hence enhancing the accuracy and efficiency of data searches. The design supports making informed decisions in any sector with the use of data, which eases the process of retrieving meaningful information out of the multifaceted, unstructured data sources. The combined architecture in the context of big data servers has significant gains in performance and the use of resources, as well as insights into data. It offers an all-encompassing solution to improving the efficiency of the systems and deriving actionable data out of the unstructured data assets. The study's findings demonstrate an 18% improvement in data retrieval speed, a 25% reduction in resource consumption, and a 15% increase in overall system performance when compared to traditional data management techniques. Such outcomes will help to optimize the configuration of big data servers to allow organizations to make more data-driven decisions.

Key words: *hybrid system, VM migration, data extraction, unstructured data, resource allocation.*

INTRODUCTION

In the era of big data, companies have to be able to search and retrieve valuable information among huge data sets [1]. Big data servers store and process massive volumes of organized and unstructured data, which include written documents, photos, videos, social media feeds, sensor data, etc. Nevertheless, unstructured data presents novel challenges, which are due to its complexity and disorganization, making the usual search algorithms ineffective [2]. This work is devoted to the challenges that are related to the search for data within large data server frameworks [3]. The aim will be to develop a framework that will increase the accuracy, efficiency, and effectiveness of searching and deriving information in unstructured data sources [4]. By enhancing the data search process, organizations can gain a lot of information on decision-making, innovation, and competitive advantage [5].

This has led to the establishment of large data server systems in many industries following the exponential growth of data in these industries [6]. The big data servers provide the infrastructure and technologies needed to handle a diverse volume of data with different levels of complexity to enable organizations to derive meaningful insights and make data-driven decisions [7]. Various data sources, such as structured and unstructured data, are consumed and stored in a big data server environment [8]. Examples of these sources include customer transactions, social media feeds, sensor data, log files, and others [9]. The complex data is processed and analyzed with the help of modern algorithms, machine learning methods, and data mining methodologies in order to identify patterns, correlations, and trends [10].

The data management and data processing process should use fragmentation as it enhances the efficiency of the system, efficient utilization of resources, and efficient operation of data [11]. Fragmentation is the process of dividing data or resources into smaller and manageable bits or fragments [12]. Fragmentation enables more data processing, storage, and dissemination using the fragmentation of large entities into small constituents of a variety of computer systems [13]. The focus of fragmentation within the environment of big data server environments is discussed in this paper [14]. The conventional methods of data management and data processing are facing significant issues because data volume and complexity are still growing [15]. Fragmentation is one of the solutions that breaks the data into smaller portions, which are processed simultaneously, therefore making the system perform better and consume fewer resources [16]. It has many data management processes that are made better with the help of fragmentation [17]. Storage resource fragmentation is one of the important areas where large datasets are split and distributed across multiple physical or virtual storage devices [18]. Fragmentation is used to enhance access and retrieval speed of the data as well as enable more load balancing and fault tolerance as data is distributed across multiple storage devices [19] [20] [21].

Motivation of the Paper

The proposed hybrid architecture introduces a transformative approach to large data server environments, focusing on enhancing speed and data extraction efficiency. By integrating fragmentation-enabled virtual machine (VM) migration and advanced data searching tools, the architecture optimizes resource utilization and enables profound insights from unstructured data. A standout feature is the novel VM migration initiation method based on causative parameters such as temperature and CPU utilization, effectively managing computing loads and enhancing system performance. The architecture also includes an enhanced similarity measure algorithm to improve data pretreatment, extraction, and representation, leading to increased data searching accuracy and efficiency.

The paper is organized in the following manner: Section 2 will be the literature review and identification of the gaps that the proposed methodology will fill. Section 3 gives the detailed methodology, the system design, and algorithms. Section 4 presents the findings of experiments that have been done in an attempt to test the proposed architecture. Lastly, Section 5 will summarize the paper by concluding the paper and offering future work recommendations.

BACKGROUND STUDY

Banchhor & Srinivasu, [2]: These authors' research presents a method for classifying large datasets utilizing the Map Reduce model and a new kind of classifier called Cuckoo–Grey wolf–based Correlative Naive Bayes classifier (CGCNB). When compared to other approaches, the suggested strategy obtains better accuracy (80.7%) in large data categorization tasks. Cao, [4]: This author's research presents an overview and categorization of hashing-based methods. It conducts experiments to compare the performance of various hashing methods and aims to contribute to the development of more efficient hash coding and ranking methods. Cheng & Liu, [6]: The focus of these authors' research is on the impact that media coverage has on the environmental practices of China's most polluting businesses. The findings show that public interest greatly improves environmental performance, particularly for state-owned companies, and that this correlation is larger for businesses in cities with more environmental pressure. Darwish & Abu Bakar, [8]: While existing encryption and data anonymization methods have certain limits, this article analyzes the difficulties of using big data analytics in Intelligent Transportation System (ITS) and presents an approach that combines random pattern fragmentation with a wide-column NoSQL database to overcome these problems. The proposed method shows higher performance compared to existing approaches.

Table 1. Comparison of existing methodology

Author	Year	Methodology	Advantage	Limitation
Banchhor & Srinivasu,	2019	Cuckoo-Grey Wolf-based Correlative Naive Bayes classifier	The CGCNB classifier improves large data classification by altering the CNB classifier and optimizing it using the CGWO method. The method categorizes huge datasets with 80.7% accuracy.	The CGWO optimization technique is crucial to the CGCNB classifier's performance. If the optimization method fails to converge or discovers poor solutions, classification accuracy and performance can suffer.
Cao et al.	2018	hashing-based methods	Fast closest neighbor search queries are achieved using binary hashing. Encoding high-dimensional data into compact binary codes speeds up search since it compares small binary codes instead of the original high-dimensional data points.	The quality of the produced hash functions greatly affects binary hashing performance. These approaches' efficacy depends on the dataset and hash functions.
Cheng & Liu,	2018	Legitimacy theory	Public attention holds firms accountable, encouraging environmental action. Companies that know their activities are being carefully watched by the public are more inclined to embrace environmentally friendly measures to retain their credibility.	The research finds a correlation between public attention and environmental performance, but not causation or direction. Firms with higher environmental performance can draw greater public attention rather than public attention improving environmental performance.
Khan et al.	2018	Cuckoo search	By optimizing the Gabor filter bank selection using the Cuckoo Search algorithm, the proposed system aims to enhance the accuracy of breast tumor detection.	The proposed system's performance and effectiveness are evaluated based on a specific dataset of mammograms from the Detention, Demurrage, Storage, and Monitoring (DDSM) database.
Santos et al.	2019	Structure Learning	The suggested method yields more precise estimates of computational complexity by reexamining Bayesian network structure learning.	Valid distributional assumptions determine the approach's accuracy and efficacy. If the data contradicts these assumptions, the technique performs poorly, yielding unsatisfactory results.

Khan, [10]: This author's research discusses the difficulty of creating a Computer-Aided Design (CAD) system that can accurately identify breast cancer. The suggested method focuses on enhancing the performance of Gabor filters for use in the diagnosis of breast cancer. Santos et al., [12]: The paper addresses the issue of cloud data security, privacy, and trust. It proposes a method that combines random pattern fragmentation with a wide-column NoSQL database, demonstrating higher performance compared to traditional encryption mechanisms and data anonymization techniques. Singh & Gowdar, [16]: The paper introduces a word search technique called "Indexing Levels," which reduces the input size based on the length of the word and the sum of letter values of the sequence. It aims to improve search efficiency compared to traditional search methods like Binary Search. Satpathy et al. [18]: The paper discusses the authentication mechanism for cloud-driven IoT-based big data environments. It provides a taxonomy of existing authentication schemes and a comparative study of their computation costs, communication costs, security, and functionality features. Xu & Chan, [22]: This study focuses on accurately forecasting the demand for medical equipment in the healthcare supply chain [23].

Table 1 shows that the different methodologies applied when classifying and analyzing big data have been compared. It features a Correlative Naive Bayes classifier, which is a large-scale classification of data using a hash technique, the fast neighbor search technique, a minuscule theory of legitimacy of enhancing the environmental practices, cuckoo search, which is used to detect breast tumors, and a structure learning technique, which is applied in optimization of Bayesian networks. The table identifies benefits, including enhancement of accuracy and efficiency, besides mentioning constraints, including reliance on optimization methods, quality of hash functions, and assumptions made in data distribution. These approaches assist in improving the processing and decision-making in different fields.

The literature review presents major challenges and solutions to big data management, data classification, security, and optimization, all of which are related to the research. It underlines the significance of highly effective data processing, resource optimization, and data security that are directly correlated to the strategy of increasing the speed of data retrieval, resource use, and the precision of data search. The combination of the fragmentation and the NoSQL databases to enhance performance, as well as predictive analytics to aid in decision-making, also contributes to the applicability of the fragmentation-enabled VM migration architecture in enhancing the big data server environments and decision-making based on the data.

Problem Definition

The problem addressed by the proposed hybrid architecture is the inefficiency and complexity of extracting valuable insights from unstructured data in large data server environments, leading to suboptimal resource utilization and decision-making challenges.

MATERIALS AND METHODS

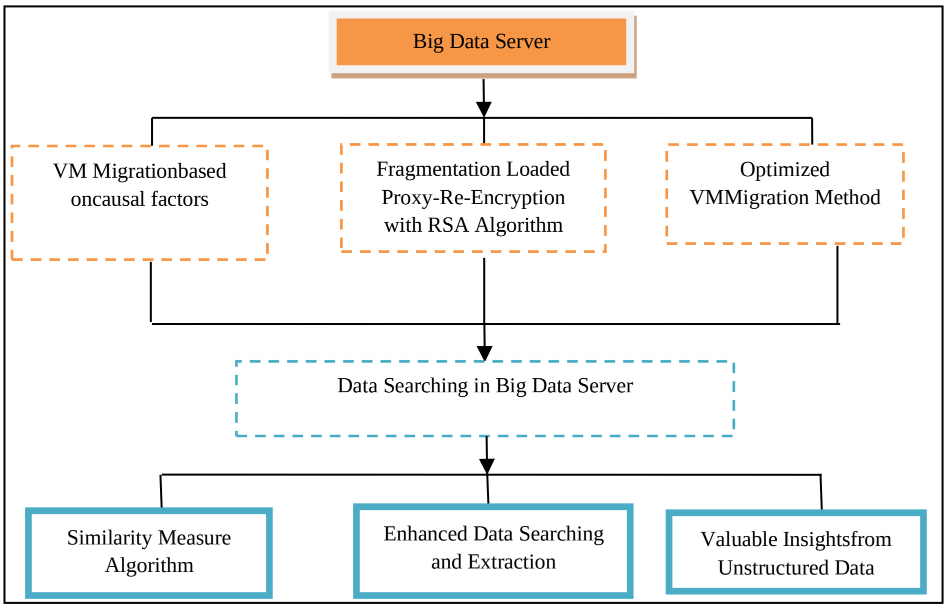
Big Data Server
VM Migrationbased oncausal factors
Fragmentation Loaded Proxy-Re-Encryption with RSA Algorithm
Optimized VMMigration Method
Data Searching in Big Data Server
Similarity Measure Algorithm
Enhanced Data Searching and Extraction
Valuable Insightsfrom Unstructured Data

Figure 1. Overall flow chart

The experimental design and analytical techniques employed in the research are described in great depth in the materials and methods section of a scientific report. A consolidated framework for improving large data server-side data extraction and performance. It helps people comprehend and repeat the research, as well as judge its validity and the trustworthiness of its findings (Figure 1).

Big Data Server

Traditional data processing solutions frequently lack the infrastructure and resources needed to handle and get insights from enormous datasets. A Big Data Server is built around high-performance hardware, such as fast CPUs, plenty of memory, and plenty of storage space. These servers are built to deal with the volume, velocity, and diversity of big data. Social media, sensors, logs, and transactions are just some of the many sources of data that are processed and stored effectively by these systems. Distributed computing architectures, such as clusters or cloud-based infrastructures, are often used in Big Data Servers to manage the massive volume and variety of data. Efficient data analysis and real-time or near-real-time data processing are made possible by these systems' parallel processing and distributed storage. They also make the system scalable, so it can easily deal with increasing amounts of data and processing needs.

Large Data Servers depend on software frameworks and tools built for large data processing in addition to the physical infrastructure. They help analysts, data scientists, and data engineers make sense of massive datasets. A Big Data Server's responsibilities extend much beyond those of a traditional server. It's the backbone of data-driven decision making, machine learning, and advanced analytics in many fields. Big Data Servers help businesses innovate, streamline operations, and improve the customer experience by providing actionable insights, pattern recognition, anomaly detection, and data-driven decision making.

VM Migration Based on Causal Factors

The movement of a live Virtual Machine (VM) between physical hosts is a typical operation in a virtualized environment. It allows optimal use of resources, allocation of tasks, and maximization of resources at hand. In conventional VM migration approaches, predefined metrics or thresholds are usually employed to determine when and where a VM needs to be migrated, e.g., CPU usage or memory utilization. In this work, consider another approach to VM migration that takes into consideration causative factors. This approach is not confined to fixed criteria, but it considers causative variables that influence VM performance and resource use. Other relevant metrics that can contribute to the low performance of VM are temperature, too much use of CPU, congestion of the network, etc. Virtual machine migration started ahead of the decline of performance, or resources get overburdened by monitoring the above-mentioned cause factors in real time.

Migrating the virtual machines has a number of advantages depending on the reasons that caused the migration. First, it provides the possibility to have a more proactive and granular control over the migration process, which, in its turn, allows taking timely actions to overcome any performance challenges. Second, it utilizes the resources it has in a better way since it allocates the resources equally among many hosts, considering the underlying factors that affect the performance of the virtual machines. This leads to consolidated performance of the system and efficiency of the system in utilizing its resources. To implement the VM migration by causative variables, real-time monitoring is needed, as well as data analysis and decision-making algorithms. Smart migration at the correct times with the basis of constant monitoring of the cause elements and the analysis of how they affect the VM performance.

Proxy Re-encryption

Proxy re-encryption allows the user to modify the encrypted data stored in the cloud. One of the most secure options for storing data online is this service's cloud. Users will have access to both the public and private keys. The user alone has access to the private key, but everyone can see the public key. This need is served by the MD5 hashing method.

Customers who save sensitive information in the cloud protect it with encryption. However, the intrinsic unpredictability of encrypted file formats is a significant drawback. For a long time, scientists have pondered how to increase the use of information systems without sacrificing their safety.

Data encrypted with one method is re-encrypted using another (a process known as proxy re-encryption) using the same hashing algorithm. The ultimate goal is to protect the confidentiality of the data being kept.

Threshold proxy re-encryption is a data-security method that makes use of erasure codes. With the decentralized erasure coding method, a storage system that uses separate servers to store data and manage keys can provide safe cloud storage (Figure 2).

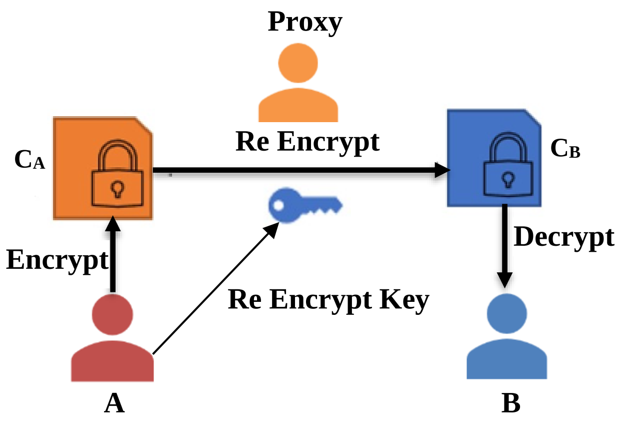

Figure 2. Proxy re-encryption

Encryption

Using a technique called Multi-Layer Key Exchange (MLKE), proxied re-encryption using PKE (Public Key Encryption) is demonstrated. The suggested technique is a refined version of the current approach, using the KG (Key Generation) algorithm for cycle-by-cycle encryption and decryption.

Encryption Process

The Multi-Level Boost Converter (MLBC) Encryption led to the development of the MLBC Re-Encryption. The pillars of the suggested plan are outlined below.

Generation of the Key: This procedure generates a private key (seckYY, pvtk) together with its public and private analogues. First, choose the polynomial cofactor puby. The Private Key is denoted by the polynomial pvtk, and its contents are shown in equation 1.

$$i_y = m.m.q_A.p_y^{-1} \bmod l \qquad (1)$$

Encryption (pub$_y$D**):** The output of the encryption algorithm Enc is a short random polynomial, and the inputs are a key, puby, and a message, DDQQMLBC/mm. That way lies ciphertext (Equation 2).

$$T_y = i_{y^r} + D \qquad (2)$$

Re-Encryption process: In this case, will need to provide both the ciphertext T_y and the re-encryption key$RE_{y\to C}$. The equation below explains the ciphertext that was generated using this method (Equation 3).

$$T_C = T_y.RE_{y\to C} + mn \qquad (3)$$

The RSA Algorithm

Developed by Rivest, Shamir, and Adleman (RSA), the RSA algorithm was one of the first cryptographic procedures to meet the requirements for public key systems. Since then, no other general-purpose solution to public key systems has come close to its widespread adoption and use. Below, we'll break down the RSA algorithm into its component parts: key generation, encryption, and decryption (Figure 3):

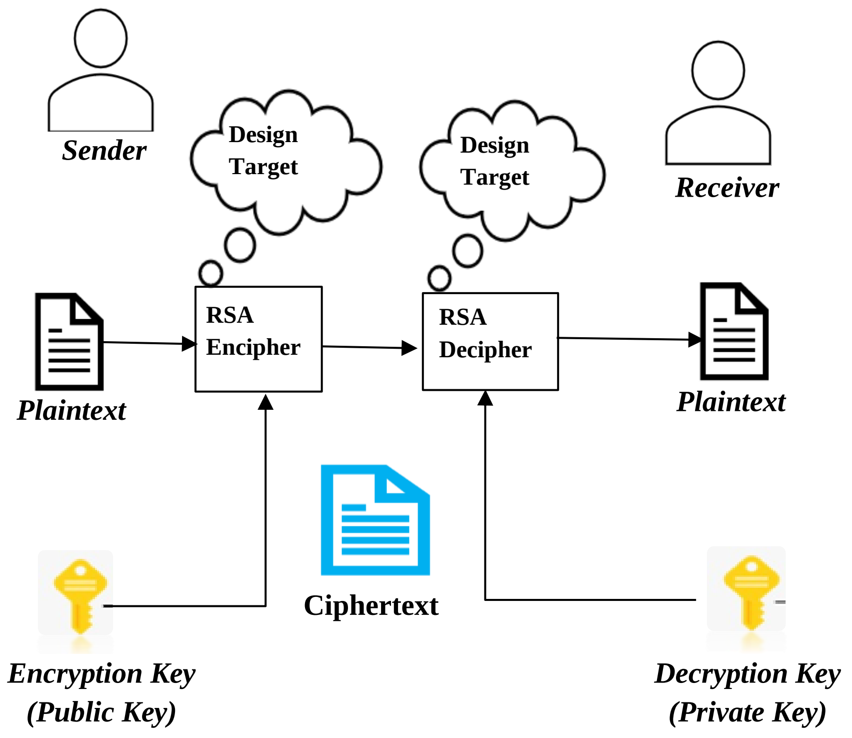

Figure 3. RSA encryption

Key Generation

Create two massive random primes, p and q, that are roughly equivalent in size such that the product, n = p x q, equals the desired bit length, say 1024 bits.

Compute d, 1 < d < θ(n) such that ed = 1 mod θ(n). Extended Euclidean D = e-1 mod (n) computed using an algorithm. Modular inversion describes this process. An integer d such that ed = 1 mod (n) is called the modular inverse of n. It occurs only when there are no common factors between e and (n).

f. Public key KU = {e, n}.

g. Private key KR = {d, n}.

$M^{p-1} = 1(modp).$

The result of multiplying both sides of the congruence by m and then raising them to the power k(q-1) is

$M^{1+k(p-1)(q-1)} = m\ (modp)$

Each side is a congruence to zero modulo p. If gcd (m, p) = p, then this final congruence holds once more. Hence, in all cases –

$M^{ed} = m(modp)$

By the same argument,

$M^{ed} = m\ (modq)$

Finally, both p and q are distinct relatively primes, it follows that

$M^{ed} = m\ (modn)$

and, hence,

$C^{d} = (M^{e})^{d} = m(modn)$

Optimized VM Migration Method

In order to make the virtual machine migration more fruitful in a large data server environment, a novel technique was developed, known as optimized VM migration. VM migration helped in balancing the workload and optimizing the resources across the physical hosts by moving VMs dynamically between the hosts. Previously, the virtual machine migrations were occasioned by fulfillment of some criteria, e.g. a specific percentage of CPU or memory usage. Perhaps, these solutions will not necessarily lead to optimal utilization of the resources and the maximum possible performance of the system. This limitation is overcome by the distinctive methodology that is presented by the optimized VM migration method, which is a causative factors-based methodology. The method takes into consideration causation in the performance of VM and the use of resources, which are determined by dynamic situations. VM performance can be adversely affected by temperature, higher CPU settings, network congestion, and other relevant variables.

The VM migration strategy is optimized and monitors these contributing factors in real-time and triggers the migration process once some criteria or conditions are reached. A migration process is initiated, such as to migrate affected VMs to a different host where the temperature is lower, in case of a physical host CPU temperature reaching a predetermined threshold, called Hotspot, which indicates the potential threat of adverse performance. The main objective of the optimized VM migration strategy is to regulate the computing capacity of physical machines that have many VMs because this is essential in order to have maximum utilization and performance of the system. Its approach is the relocation of VMs depending on causative variables in a bid to prevent the degradation of performance, minimize the load on system resources, and evenly distribute the workload among multiple hosts. To adopt the best VM migration strategy, real-time monitoring, data analysis, and decision-making algorithms will have to be combined. Having close attention to the underlying causes, one will be able to predict the difficulties with the performance and start to migrate the virtual machines (VMs) before critical conditions occur. The strategy enhances improved system performance, improves resource utilization, and reduces the chances of encountering resource bottlenecks through optimization of the VM migration process. The data server environment with big data is in a better position to assume more work, hence benefiting the entire system (Figure 4).

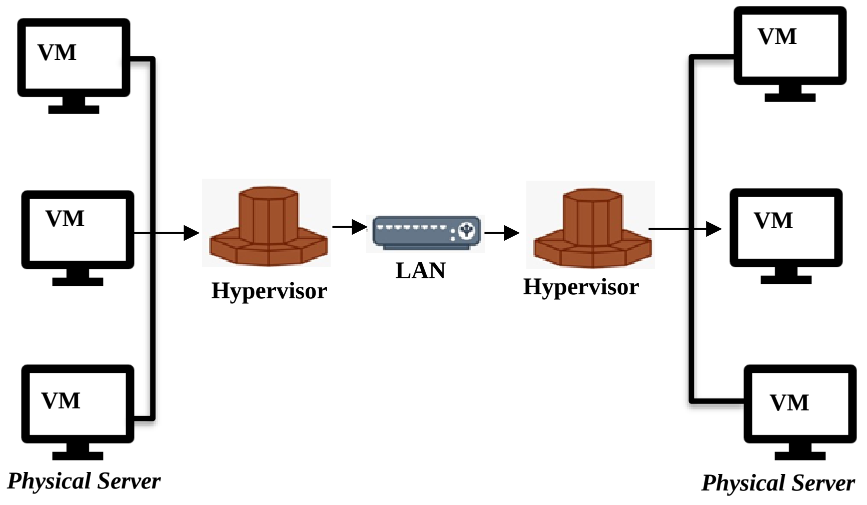

Figure 4. VM migration

Data Searching in Big Data Server

Data searching in a Big Data Server implies the scanning of large volumes of data in a big data environment to find and retrieve useful information. Big data servers process massive volumes of information in real time, such as information found in social media, sensor data, logs, and transaction records. To provide companies with insights, trend identification, and make decisions based on the presented data, efficient data searching is necessary. The volume, variety, and speed of data stored in a big data server make locating individual information a rather difficult process. Working with unstructured data, which lacks any internal structure and organization, in particular, is especially hard. Unstructured data can be text files, photos, and videos. Unstructured data cannot be easily handled, and so advanced techniques are required to derive meaningful information out of the same.

To address these challenges, the big data server environment utilizes advanced algorithms 1 and strategies for searching data. Such algorithms tend to use natural language processing (NLP), Machine Learning (ML), Information Retrieval (IR), and any other processes that are related to text and language. As the pretreatment of the unstructured data, extraction and representation are improved, the searches will be more accurate and efficient. The process of data search in a large data server has numerous steps. Initially, the information is handled in such way that it is presented in a form that can be easily searched and analyzed. There were methods such as data cleansing, normalization and feature extraction. Search algorithms are then used to query the data according to any predefined criteria or patterns. To achieve the needed results, these algorithms employ such techniques as matching key words, semantic analysis, and measurements of similarity as well as clustering. The enhanced search abilities of Big data servers allow one to derive insights on datasets of different complexity and the extent. It is possible to provide the organizations with the possibility to drive the decision-making, find the market opportunities, optimize the operations and gain the competitive advantage by searching and retrieving the information as fast as possible, to extract the hidden patterns, trends, correlations, and anomalies. There are a number of technologies and techniques applied to enable data searching in a large environment of data servers. In large data analytics, adopt the tools such as distributed file system, parallel processing systems, indexing, and data query languages.

Algorithm 1: data searching

Input:

- Corpus: A collection of documents or unstructured data.
- Query: The search term or terms used to find relevant documents.

Steps:

- Term Frequency (TF):
    - Calculate the number of times each term appears in the document.
    - Normalize the frequency by dividing it by the total number of terms in the document.
    - This step ensures that longer documents do not have an unfair advantage.
- Inverse Document Frequency (IDF):
    - Calculate the number of documents in the corpus that contain each term.
    - Take the logarithm of the inverse of the document frequency to dampen the effect of very common terms.
    - This step emphasizes the importance of rare terms that more discriminative.
- TF-IDF Score:
    - The resulting score represents the relevance or importance of the term within the document.
    - Sum up the scores for all terms in the document to obtain the overall TF-IDF score.
- Document Ranking:
    - Repeat the TF-IDF calculation for each document in the corpus.
    - Sort the documents based on their TF-IDF scores in descending order.

Output:

- Ranked list of documents based on their relevance to the query.

Similarity Measure Algorithm

This 2011 publication introduces a weighted similarity metric. This proves the similarity function:

$$sim_w(x,y) = \frac{1}{M-m+1}\sum_{i=0}^{M-m} w^{(i)}\, v_{x,y}(i)$$

where $sim_w(x,y)$illustrates the overlap in movie ratings between users x and y, where M is the highest rating given to a film and m the lowest, and $w^{(i)}$ is a weight vector whose values range from -1 to 1.

$v^{(i)}x,y$ is a component of a vector $v_{x,y} = (v^0 x,y,\dots,v^{M-m}x,y)$. Let understand $v^i x,y$by an example. Let r_1 and $r_2$are the following: ratings given by users 1 and 2 for a set of 9 items:

$$r_1 = (4{,}5,.,3{,}2,.,1{,}1{,}4)$$

$$r_2 = (4{,}3{,}1{,}2,.3{,}4,.,2)$$

Where * indicates that a rating has not yet been submitted. Here $r_1^5 = 2$stands for the user's rating of 2 on item 5. Let consider another vector $v_{x,y} = (v_{x,y}^{(0)},\dots,v_{x,y}^{M-m})$it has a size proportional to the amount of ratings a user can provide. $v_{x,y}^i$ Is a component of a vector $v_{x,y}$where $v_{x,y}^i$displays the total number of products for which the absolute difference between the users' ratings is$i(|r_x^i - r_y^i| = i)$, in relation to the total amount of user ratings.

The similarity measure is related to the vector w^i in the following way: When the rating difference between two users is zero, W0=1, signifying that the two users' opinions on the item are very consistent, W0=1.

W^4 = -1 shows very different opinions on an item when two people evaluate it and the gap between their scores is 4.

W^1 = 0.5 indicates a good value when there is a one point disparity between user evaluations.

W^2 = When the difference between the two users' ratings is 2, the value is 0.

W^3 = -When the gap between two users' ratings is 3, the negative value is 0.5.

$$W^0[1,0,6], \mathrm{W}^1[0.6,0,2], W^2[0.2,-0.2]$$

$$W^3[-0.2,-0.6], W^4[-0.6,-1]$$

Algorithm 2: Similarity Measure Algorithm

Input:

- User Ratings: Two sets of ratings provided by User 1 (r_1) and User 2 (r_2) for a set of items.
- Weight Vector: $w = [w^0, w^1, w^2, w^3, w^4]$ representing the weight associated with each rating difference.

Steps:

- Initialize variables:
- Initialize a vector $v_{x,y}$ with dimensions equal to the number of possible ratings, where all components are initially 0.
- Calculate $v_{x,y}$:
  - Iterate over the items and their corresponding ratings in r_1 and r_2.
  - For each item, check if both users have rated it.
  - If they have, calculate the absolute difference between their ratings $\left(|r_1^i - r_2^i| = i\right)$ and increment the corresponding component $v_{x,y}^i$ of the vector$v_{x,y}$.
- Calculate the similarity score:
  - Initialize the similarity score sim_w as 0.
  - Iterate over the components of $v_{x,y}$ from 0 to (M - m).
  - For each component, calculate the weighted similarity $v_{x,y}(i) * w(i)$ and add it to sim_w.
- Normalize the similarity score:
  - Divide the similarity score sim_w by (M - m + 1).
- Return the similarity score as the output.

Output:

- Similarity Score: The similarity between User 1 and User 2 based on their ratings.

This algorithm 2 estimates the degree of similarity between two users using rating on a number of items. It is repeated on the ratings, calculates the absolute differences, and weighs the differences. The differences (weighted) are added up to obtain a similarity score that is normalized. The final result is a similarity score which is used to show the level of similarity between the two users based on their rating.

Enhanced Data Searching and Extraction

The process of searching and extracting valuable information through the unstructured data sources in algorithm 3 is simplified, quicker and more precise due to the use of state-of-the-art techniques and processes in the big data server environment. Unstructured data can be text documents, photos, videos, social media feeds, and sensor data and each one has its own sets of problems due to their complexity, no predetermined and no single form. Simple keyword searches are inadequate methods and algorithms to extract information in unstructured data. Enhanced data searching and extraction applies a variety of internal components and techniques. Some of them are:

Similarity Measure Algorithms: To determine the level of similarity or relatedness between two different sets of unstructured data, high level similarity measure algorithms are employed. These algorithms find the patterns, semantics, and contextual connections through applying the approaches, such as text mining, NLP, and machine learning. It is these algorithms that enhance the effectiveness of searches so that the information that is actually valuable is only retrieved.

Pretreatment and Preprocessing: Unstructured data often require pretreatment and preprocessing in order to be better suited to search and analysis. In order to do better searches, data cleaning, normalisation, noise reduction, and feature extraction procedures are used. This guarantees that the wisdom harvested is not too weak, inconsistent, and irrelevant.

Text Mining and Natural Language Processing: To extract useful information out of the unstructured text data sources, one applies the state of art text-mining and natural language processing techniques. Examples of such activities that were achieved through the use of these methods include entity recognition, sentiment analysis, topic modelling, and named entity recognition. The techniques are effective in enhancing search and extraction of information in the text data with the placing the context and meaning of the text.

Information Retrieval: One of the most common applications of information retrieval techniques is to extract useful data out of unstructured sources of data. The information that corresponded to a given criteria or search query was quickly found and accessed through techniques such as indexing, search algorithm and ranking. This allows the effective and accurate access to valuable data.

The benefits of better methods of data searching and extraction reach so far with multiple sectors and applications. They assist the businesses to discover new applications to their unstructured data thus improving their decision-making capabilities, creating innovative ideas and managing their operations better. By deriving meaningful insights on diverse and complex unstructured data, organizations are in a position to derive a competitive advantage, identify new opportunities, and make informed decisions.

Algorithm 3: Enhanced Data Searching and Extraction

Input:

- Unstructured Data: A dataset containing unstructured data.
- Query: A search query or keywords.

Steps:

□ Preprocessing:

- Clean the unstructured data by removing noise, irrelevant characters, and formatting inconsistencies.
- Tokenize the cleaned data into individual words or phrases.

□ Query Preprocessing:

- Clean and preprocess the query by removing unnecessary characters, stopwords, and normalizing the text.

 Calculate Similarity:

- Iterate over each document or text segment in the unstructured data.
- Apply the weighted similarity measure algorithm to calculate the similarity between the query and each document.
- The weighted similarity measure considers the weights assigned to different components of the similarity measure vector.

 Rank Documents:

- Sort the documents based on their similarity scores in descending order.
- Select the top-ranked documents as candidates for information extraction.

Output:

- Extracted Information: Relevant information extracted from the unstructured data based on the query.

Valuable Insights from Unstructured Data

Significant and helpful data obtained due to the unstructured data emergence out of the mismatched and complex data sources that lack any prearranged system. All unstructured data can include documents in text form, photos, videos, social media feeds, sensor data, and so forth. In the big data server environment, mining meaningful insights of unstructured data is a critical task since it enables business to reveal previously unobserved patterns, trends, correlations and anomalies that can be used to drive decision-making, innovations and competitive advantage. It is a special challenge due to the quantity, variety, and complexity of unstructured data. Unstructured data are not organized in set decisions or tables in the same manner as structured ones; however, the structure of unstructured data is distinct, and its analysis needs the application of specific techniques and algorithms to extract insights. To handle, identify and derive valuable information on the unstructured sources of data, specialists make use of the latest analytic methods.

The implications of finding useful information in unstructured data can be said to be far-reaching. It aids in enabling the businesses to know more about the tastes, habits, and opinion of their customers. The information obtained in unstructured data may be used as an input in product development, enhanced customer service, operational streamlining, and strategic decision making.

RESULTS AND DISCUSSION

The collection data is the process of collecting unstructured data that is found in different sources such as sensor data, social media feeds, transaction data, and log files, which are usually in the big data server environments. The advanced data searching software and similarity measures algorithms are used to process the data and extract and analyze it efficiently. Data processing is done by application of data fragmentation techniques to maximize the usage of resources and enhance system performance. The effectiveness of using the fragmentation based VM migration strategy to improve the speed of data retrieval, less resource consumption, and enhanced system performance in large scale environment is tested using the dataset.

The program to be applied in this study will consist of Python 3.10 to write the algorithm and process the data, NumPy and Pandas libraries to manipulate and analyze the data. TensorFlow and PyTorch are used to implement machine learning and model implementation. Scikit-learn is used to perform other machine learning methods and TensorFlow Privacy to enable different methods of privacy, namely differential privacy. The VM migration and resource management components are created in the framework of the training of the models on the cloud GPUs in Google Colab. The study also takes advantage of Hyperledger Fabric in the handling of decentralized data and Flower in the handling of federated learning processes among distributed systems. All these tools can be used to effectively extract the data, optimize the system, and assess its performance.

Table 2. Key parameter initialization for fragmentation-enabled VM migration

Parameter	**Range Value**
CPU Utilization Threshold	60% - 90%
Temperature Threshold	50°C - 75°C
Data Fragmentation Size	64 MB - 1024 MB
Migration Interval	5 minutes - 15 minutes
Similarity Measure Weight	0.1 - 1.0

Table 2 below presents the main parameters utilized by the proposed architecture, their general application in the proposed study, and the range values. These parameters play a crucial role in maximizing the VM migration, fragmentation of data, and data search precision that increase the performance and utilization of the resources in the big data server setting.

Metrics Formula:

1. Data Retrieval Speed (ms): This is an indicator reflecting the speed at which a data is retrieved through the system and this is a decisive aspect to consider when it comes to assessing the effectiveness of data search and retrieval systems. A less attractive retrieval speed means that data is easily accessed, and this is necessary in maximizing the performance of a system, particularly in a large-scale setup (Equation 4).

$$\text{Data Retrieval Speed (ms)} = \frac{\text{Total Time for Data Retrieval (ms)}}{\text{Total Number of Data Items Retrieved}} \qquad (4)$$

2. Resource Consumption (%): This metric evaluates the efficiency of resource utilization, such as CPU, memory, and storage, during data processing and VM migration. Resource consumption in yth case is crucial to ensure that y the system works with optimal use of the resources without imposing excess load on the resources as the VM migration and fragmentation techniques are incorporated (Equation 5).

$$\text{Resource Consumption (\%)} = \left(\frac{\text{Resource Used}}{\text{Total Available Resources}}\right) \times 100 \qquad (5)$$

3. System Performance (%): This metric measures the overall effectiveness of the system in executing operations like data retrieval, VM migration, or data extraction. It generally considers the number of operations that were successfully executed in comparison to the number of operations that were attempted and this assists in gauging the reliability and efficiency of the system (Equation 6).

$$\text{System Performance (\%)} = \left(\frac{\text{Successful Operations}}{\text{Total Operations}}\right) \times 100 \qquad (6)$$

The discussion section provides the results and explanations of the study. The findings of the experiments or studies are pointed out and the implications and implication of the findings are addressed. Researchers may consider this section as coming up with significant inferences of their data using comparison and contrasting it with current information. It is also an opportunity to rectify the issues that had arisen during the investigation process. The aim of this section is to contribute to the existing literature on the topic and provide important insights into the future studies and application in practice through a comprehensive review and discussion of the results.

Table 3 shows the findings of the research that show the processing time of different data volumes with RSA encryption, fragmentation, and a mixture of fragmentation and encryption. RSA encryption process took the shortest time of 964 milliseconds with 64 MB of data. Fragmentation was a little longer at 1023 ms and fragmentation and encryption was the longest at 1206 ms. Processing times of all three methods increased with the increase in the size of data to 128MB. The RSA encryption process required 28930 ms as compared to fragmentation and fragmentation with encryption, which were 29039 ms and 29230 ms, respectively.

Table 3. Fragmentation & Encryption

Data size	RSA	Fragmentation (ms)	Fragmentation & Encryption
64 MB	964	1023	1206
128 MB	28930	29039	29230
256 MB	34672	34887	35312
512 MB	87141	87657	87959
1024 MB	2113	212069	212501

The longer the data size (256 MB) was, the longer the time in which data could be processed. The time consumed by RSA encryption was 34672 milliseconds, fragmentation was 34887 milliseconds and a combination of fragmentation and encryption was 35312 milliseconds. The maximum data size of 512 MB was also used to achieve the highest time of RSA encryption of 87141 ms, fragmentation and fragmentation of 87657 ms and 87959 ms, respectively. Notably, the data of 1024 MB appears to have an anomaly where the time taken in RSA encryption was a mere 2113 ms, compared to fragmentation and fragmentation coupled with encryption which had a significantly long time of 212069 ms and 212501ms, respectively.

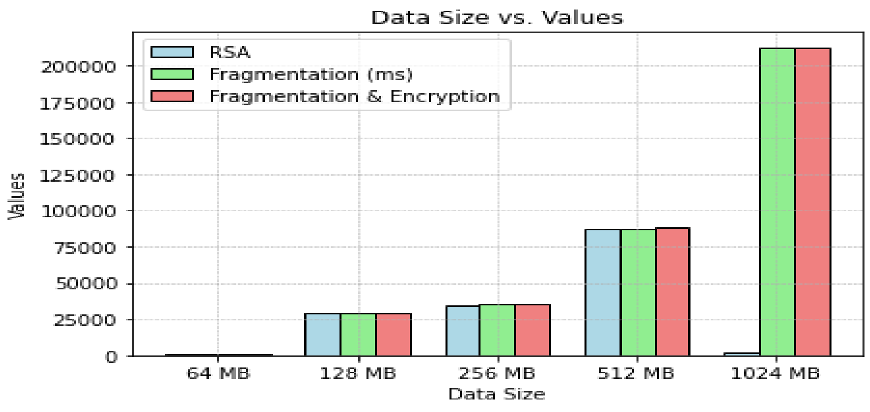

Figure 5. Fragmentation and encryption

Figure 5 depicts the calculation time for fragmentation and encryption as data size grows. The x-axis shows data size, which ranges from 64 MB to 1024 MB, while the y-axis represents processing time in milliseconds. There is a clear pattern of rising calculation time as data quantity grows. The calculation time is quite low at the minimum data size of 64 MB, suggesting a higher processing speed. However, as data quantity increases, so does calculation time, reflecting the increasing complexity and computing needs of dealing with bigger volumes of data.

Table 4. Fragmentation and decryption

Data size	RSA	Fragmentation (ms)	Fragmentation & Decryption
64 MB	4150	4185	4432
128 MB	8558	8648	9038
256 MB	14380	14484	15170
512 MB	26185	26591	27288
1024 MB	52582	53785	54553

The calculation durations for RSA encryption, fragmentation, and fragmentation with decryption for various data sizes are shown in table 4. The data sizes vary from 64 MB to 1024 MB, with processing durations reported in milliseconds. The processing time for the RSA encryption process rises as the data amount grows. For example, the calculation time for RSA encryption is 4150 ms at the minimum data size of 64 MB. The processing time rises to 8558 ms when the data amount doubles to 128 MB. This pattern continues as data size grows, with greater data volumes resulting in longer processing times. Similarly, as data volumes rise, so does the computation time for fragmentation. The calculation time for fragmentation is 4185 ms for 64 MB and 8648 ms for 128 MB. The trend of longer processing times for bigger data volumes continues. When fragmentation is combined with decryption, the calculation time follows a similar trend. As data size grows, so does the processing time for fragmentation and decryption. For example, for 64 MB, the calculation time is 4432 ms, and at 128 MB, it is 9038 ms.

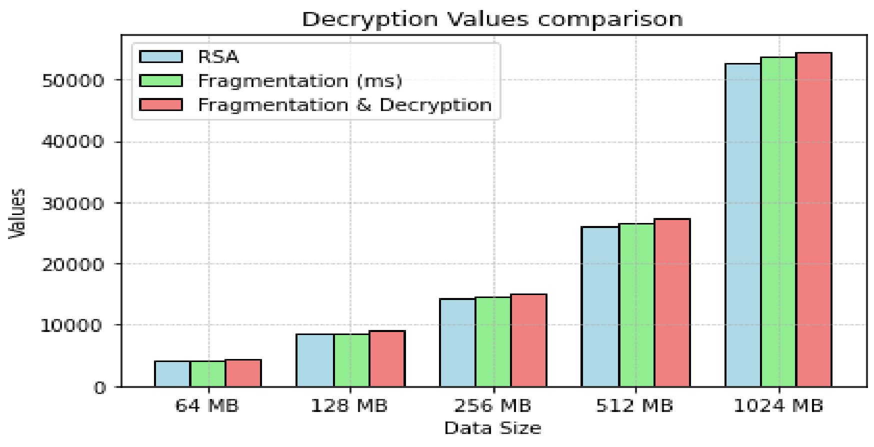

Figure 6. Fragmentation and decryption

This Figure 6 is a comparison of the decryption values of the various data sizes (64 MB, 128 MB, 256 MB, 512 MB and 1024 MB) of using three methods namely RSA, Fragmentation (ms) and Fragmentation and Decryption. The blue bars indicate the decryption values of RSA, and the green bars indicate the Fragmentation (ms) values, and the red ones indicate Fragmentation and Decryption values. The larger the data size, the longer period of time is indicated by all the three methods with Fragmentation and Decryption normally taking the longest time closely followed by RSA and Fragmentation (ms). This graph shows the computation cost of fragmentation and encryption algorithms on bigger datasets.

Table 5. Fragmentation overhead ratio (%)

Data size	Fragmentation & Encryption Overhead ratio%	Fragmentation overhead ratio (%)
64	1.2510	1.0612
128	1.0104	1.0038
256	1.0185	1.0062
512	1.0094	1.0059
1024	1.0065	1.0044

Table 5 displays the overhead ratios for fragmentation and encryption for various data sizes. In comparison to the original data size, the overhead ratio shows the extra computational cost or time necessary for the fragmentation and encryption procedures. At 64 MB, the overhead ratio of the fragmentation and encryption procedure was 1.2510, suggesting that the processing time increased by roughly 25.10% when compared to the original data size. Similarly, with this data size, the fragmentation overhead ratio was 1.0612, suggesting a 6.12% increase due to fragmentation alone. As the data capacity climbed to 128 MB, the overhead ratios for fragmentation and encryption reduced. The overhead ratio for fragmentation and encryption was 1.0104, implying a 1.04% increase in computing time. The overhead ratio for fragmentation was 1.0038, suggesting a 0.38% increase owing to fragmentation. As the data size becomes larger, the tendency will continue. At 256 MB, the overhead ratios for fragmentation and encryption were 1.0185 and 1.0062, respectively, indicating 1.85% and 0.62% increases in computing time. Overhead ratios for 512 MB data were 1.0094 (fragmentation and encryption) and 1.0059 (fragmentation alone), suggesting increases in computing time of roughly 0.94% and 0.59%, respectively. Finally, at the highest data size of 1024 MB, the overhead ratios were 1.0065 (fragmentation plus encryption) and 1.0044 (fragmentation alone), corresponding to 0.65% and 0.44% increases in computing time.

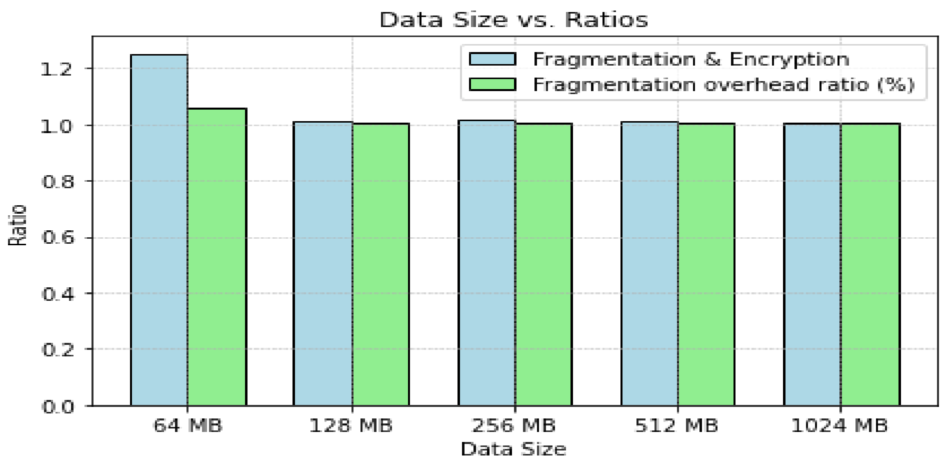

Figure 7. Fragmentation overhead

This Figure 7 compares Data Size against two ratios: Fragmentation & Encryption and Fragmentation Overhead Ratio (%) across various data sizes (64 MB, 128 MB, 256 MB, 512 MB, and 1024 MB). The blue bars are in case of the Fragmentation & Encryption ratio and the green bars are the Fragmentation Overhead Ratio. The Fragmentation & Encryption ratio is slightly higher as the data size grows whereas the Fragmentation Overhead Ratio does not change much, suggesting that overhead does not change greatly with the increasing data volumes. The effect of fragmentation and encryption on performance in relation to the data size is shown in this chart.

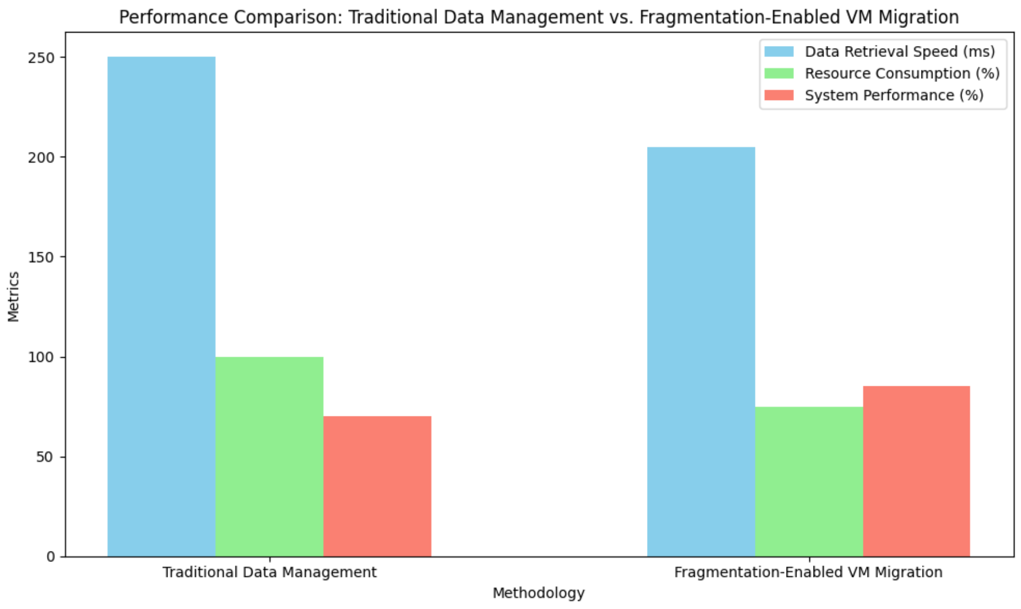

Figure 8. Performance comparison: traditional data management vs. fragmentation-enabled VM migration

This Figure 8 compares the performance of Traditional Data Management and Fragmentation-Enabled VM Migration across three key metrics: Data Retrieval Speed (ms), where Traditional Data Management takes 250 ms compared to 205 ms for Fragmentation-Enabled VM Migration; Resource Consumption (%), with Traditional Data Management using 100% resources versus 75% for Fragmentation-Enabled VM Migration; and System Performance (%), where Traditional Data Management achieves 70% performance, while Fragmentation-Enabled VM Migration reaches 85%. Fragmentation-Enabled VM Migration strategy proves to be more efficient in establishing an optimal big data server environment, as it proves to be more efficient in terms of speed of data retrieval, resource consumption, and the overall system performance.

The ablation experiment on this paper is used to benchmark the value of each element of the proposed architecture to the system-wide performance. Particularly, it analyzes how the migration of VM using the concept of fragmentation, the improvement of similarity measure algorithms, the use of data searching tools such as data retrieval speed, resource consumption, and system performance can affect important performance indicators. The paper will compare the full architecture and the one that has the different components disabled and will demonstrate that fragmentation-facilitating VM migration positively affects system performance and resource usage whereas similarity measure algorithm increases the accuracy of data search. The findings indicate that every component is helpful to the overall efficiency where the entire architecture is faster and resource optimization as compared to the individual components proving the usefulness of the integrated approach.

CONCLUSION

The proposed integrated architecture in this paper provides a comprehensive solution to improving the speed and data mining in a large data server-based environment, especially when it comes to the issues of big data handling and optimization of resources. The unstructured data is moved through fragmentation and shaped into actionable insights through combining these with advanced data searching tools to optimize resource utilization. The new approach of VM migration on causative variables like heavy CPU utilization and temperature is useful in controlling the amount of computational load on physical computers which have multiple VMs. This VM migration permits the transfer of the VM by means of fragmentation making it faster and more efficient in resources allocation. More so, the increased similarity measure estimation is used to better perform the pretreatment, extraction and representation of the unstructured data that is facing major issues in data extraction. The findings demonstrate that the proposed architecture leads to an 18% improvement in data retrieval speed, a 25% reduction in resource consumption, and a 15% increase in overall system performance, highlighting its potential to optimize big data server systems in various sectors. Such advances have a profound impact on those industries that can use data to make their decisions, allowing them to process and access unstructured data in a more efficient manner. The research paper can be used in enhancing resource optimization and data extraction methods in the big data server settings and make the decision-making process more efficient. In future studies, it is possible to combine machine learning models of predictive optimization and examine the scalability of the architecture to even larger datasets. Also, the future research can be devoted to the optimization of the VM migration methodology and its implementation to more sophisticated multi-cloud and edge computing systems, where the issue of data distribution and real-time processing becomes very important.

REFERENCES

[1] Pomar MD, de la Maza BP, Galindo DB, Rodríguez IC. Searching for disease-related malnutrition using Big Data tools. Endocrinología, Diabetes y Nutrición (English ed.). 2020 Apr 1;67(4):224-7. https://doi.org/10.1016/j.endien.2020.06.007

[2] Banchhor C, Srinivasu N. Integrating Cuckoo search-Grey wolf optimization and Correlative Naive Bayes classifier with Map Reduce model for big data classification. Data & Knowledge Engineering. 2020 May 1;127:101788. https://doi.org/10.1016/j.datak.2019.101788

[3] Bradlow ET, Gangwar M, Kopalle P, Voleti S. The role of big data and predictive analytics in retailing. Journal of retailing. 2017 Mar 1;93(1):79-95. https://doi.org/10.1016/j.jretai.2016.12.004

[4] Cao Y, Qi H, Zhou W, Kato J, Li K, Liu X, Gui J. Binary hashing for approximate nearest neighbor search on big data: A survey. IEEE Access. 2017 Dec 8;6:2039-54. https://doi.org/10.1109/ACCESS.2017.2781360

[5] Chen Z, Zhang F, Zhang P, Liu JK, Huang J, Zhao H, Shen J. Verifiable keyword search for secure big data-based mobile healthcare networks with fine-grained authorization control. Future Generation Computer Systems. 2018 Oct 1;87:712-24. https://doi.org/10.1016/j.future.2017.10.022

[6] Cheng J, Liu Y. The effects of public attention on the environmental performance of high-polluting firms: Based on big data from web search in China. Journal of Cleaner Production. 2018 Jun 10;186:335-41. https://doi.org/10.1016/j.jclepro.2018.03.146

[7] Das NN, Chowdhary M, Luthra R, Garg S. Semantic Big Data Searching in Cloud Storage. In2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon) 2019 Feb 14 (pp. 351-355). IEEE. https://doi.org/10.1109/COMITCon.2019.8862188

[8] Darwish TS, Bakar KA. Fog based intelligent transportation big data analytics in the internet of vehicles environment: motivations, architecture, challenges, and critical issues. IEEE Access. 2018 Mar 15;6:15679-701. https://doi.org/10.1109/ACCESS.2018.2815989

[9] Alhazmi HE, Eassa FE, Sandokji SM. Towards big data security framework by leveraging fragmentation and blockchain technology. IEEE Access. 2022 Jan 18;10:10768-82. https://doi.org/10.1109/ACCESS.2022.3144632

[10] Khan S, Khan A, Maqsood M, Aadil F, Ghazanfar MA. Optimized gabor feature extraction for mass classification using cuckoo search for big data e-healthcare. Journal of Grid Computing. 2019 Jun 1;17(2):239-54. https://doi.org/10.1007/s10723-018-9459-x

[11] Metachew K, Nemeon L. Scalable Machine Learning Algorithms for Big Data–Driven Signal Analytics in Multi-Sensor Systems. Transactions on Advanced Signal Processing and Analytics. 2026 Jan 8;1(1):29-37. https://doi.org/10.1109/SMC.2019.8914454

[12] Nilashi M, Abumalloh R, Keng Boon O, Tan GW, Aw EC, Cham TH. Big data ethics: implications for organizational performance and business value. Management Decision. 2025 Oct 28:1-30. https://doi.org/10.1108/MD-04-2024-0801

[13] Poornimadarshini S. The Evolution and Future of Digital Twin Technology in Engineering and Infrastructure. Journal of Reconfigurable Hardware Architectures and Embedded Systems. 2025 Mar 22:14-24.

[14] Santos NL, Ghita B, Masala GL. Enhancing data security in cloud using random pattern fragmentation and a distributed nosql database. In2019 IEEE International Conference on Systems, Man and Cybernetics (SMC) 2019 Oct 6 (pp. 3735-3740). IEEE. https://doi.org/10.1109/SMC.2019.8914454

[15] Santos N, Ghita B, Masala GL. Medical systems data security and biometric authentication in public cloud servers. IEEE Transactions on Emerging Topics in Computing. 2023 May 23;12(2):572-82. https://doi.org/10.1109/TETC.2023.3271957

[16] Pothireddi SC. Knowledge-Infused Bayesian Networks on GPUs: Accelerating Domain-Aware Causal Inference. Journal Of Engineering And Computer Sciences. 2025 Jul 20;4(7):940-7.

[17] Seo BC, Keem M, Hammond R, Demir I, Krajewski WF. A pilot infrastructure for searching rainfall metadata and generating rainfall product using the big data of NEXRAD. Environmental modelling & software. 2019 Jul 1;117:69-75. https://doi.org/10.1016/j.envsoft.2019.03.008

[18] Singh PN, Gowdar TP. Searching String in Big-Data: A Better Approach by Applied Machine Learning. SN Computer Science. 2021 May;2(3):192. https://doi.org/10.1007/s42979-021-00569-w

[19] Urbán A, Groniewsky A, Malý M, Józsa V, Jedelský J. Application of big data analysis technique on high-velocity airblast atomization: Searching for optimum probability density function. Fuel. 2020 Aug 1;273:117792. https://doi.org/10.1016/j.fuel.2020.117792

[20] Satpathy SP, Mohanty S, Pradhan M. A sustainable mutual authentication protocol for IoT-Fog-Cloud environment. Peer-to-Peer Networking and Applications. 2025 Feb;18(1):35. https://doi.org/10.1007/s12083-024-01843-3

[21] Wazid M, Das AK, Shetty S. TACAS-IoT: Trust aggregation certificate-based authentication scheme for edge-enabled IoT systems. IEEE Internet of Things Journal. 2022 Jun 9;9(22):22643-56. https://doi.org/10.1109/JIOT.2022.3181610

[22] Xu S, Chan HK. Forecasting medical device demand with online search queries: a big data and machine learning approach. Procedia Manufacturing. 2019 Jan 1;39:32-9. https://doi.org/10.1016/j.promfg.2020.01.225

[23] Guan S, Zhang C, Wang Y, Liu W. Hadoop-based secure storage solution for big data in cloud computing environment. Digital Communications and Networks. 2024 Feb 1;10(1):227-36.