*Quest Journals*
*Journal of Software Engineering and Simulation*
*Volume 10 ~ Issue 12 (2024) pp: 38-50*
*ISSN(Online) :2321-3795 ISSN (Print):2321-3809*
www.questjournals.org

**Research Paper**

# A Review on the Advanced Strategies for Efficient Big Data Storage: REHDFS Implementation and DNA Storage Solutions

Thafzy V M
*Dept. Computer Science and Engineering*

Hasna M
*Dept. Computer Science and Engineering*

***Abstract*****—***Over the past few years, an unprecedented surge in data creation has been observed, resulting in colossal amounts of information generated on a daily basis. Projections indicate that by 2020, the volume of data to be stored could reach a staggering 44 trillion gigabytes. However, while data volumes continue to escalate rapidly, our conventional databases face limitations in effectively managing and processing such vast and intricate datasets. Compounded by the absence of a durable physical storage medium capable of enduring various environ- mental conditions, the challenge becomes more pronounced. Simultaneously, the concept of Big Data has evolved into a pivotal new resource, yet current analysis frameworks confront daunting hurdles related to scalability, swift data ingestion, performance optimization, and efficient storage management. To address the mounting and exponentially expanding volumes of diverse data being generated at an accelerated pace, numerous researchers have focused on two potential solutions. One avenue involves enhancing existing technologies with intelligent enhancements, as seen in the development of REHDFS. Alternatively, innovative breakthroughs in chemistry, such as leveraging DNA, have emerged as promising solutions.*

***Index Terms*****—***Big Data management, REHDFS, DNA digital storage, Automated machine learning, NoSql database, K-means clustering, SEEDDUP, PSA/PCA.*

***R****eceived 13 Dec., 2024;* ***R****evised 22 Dec., 2024;* ***A****ccepted 25 Dec., 2024 © The author(s) 2024.*
***P****ublished with open access at* ***www.questjournas.org***

## I. INTRODUCTION

Big Data transcends sheer volume; it encompasses data of diverse types and swift generation rates. Essentially, it represents an immense stream of varied data forms that pose challenges in management and processing using current tools and architectures. To grasp the nature of data feeding into Big Data systems, consider sources such as bank transactions, web server logs, social media interactions from platforms like Twitter or Facebook, multimedia content like MP3 songs and video blogs, web page content, personal documents, medical reports, and an extensive array of other data formats.

The global digital content landscape is experiencing expo- nential growth and is projected to soar to 35 zettabytes in the coming decade. This surge is predominantly fueled by the proliferation of social networking platforms like Face- book, Twitter, WhatsApp, and the escalating use of sensors across various domains. Within this vast pool of data lies a wealth of emerging market and environmental trends. Access to such real-time insights empowers organizations to make well-informed decisions, propelling them towards competitive edges.

Traditionally reliant on data mining tools for structured historical data analysis, organizations are redirecting invest- ments towards developing robust tools capable of decipher- ing unstructured, rapidly expanding data. This shift aims to comprehend prevalent market trends and predict customer behavior accurately. Leveraging these analytical tools not only enhances decision-making processes but also facilitates necessary adjustments in strategies and policies for heightened competitiveness in the market.

Traditional data storage systems, particularly relational databases, face significant difficulties in keeping up with the growing demands for storage and analysis. As a result, there is a critical need to develop advanced tools for big data storage and analytics to meet these escalating challenges effectively.

*A. Data Structures of Big Data*

Big Data manifests in various formats, spanning from Struc- tured and Semi-Structured to Quasi-Structured and Unstruc- tured data types. Fig. 1. illustrates these diverse data structure types alongside their respective growth patterns.

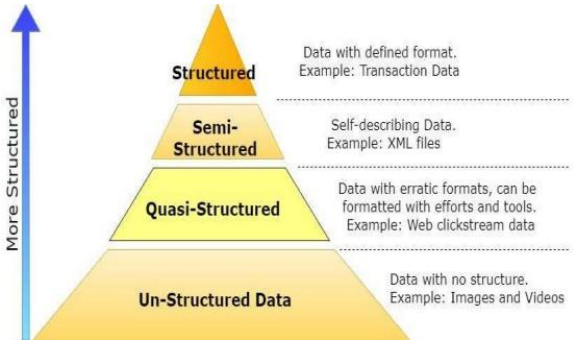


Fig. 1. The pyramid of data structures of Big data [1]

*1) Structured Data:* Structured data embodies clear organi- zation, description, presentation, and explicit length or format. This data type can be effortlessly rearranged or processed using various data mining tools. Imagine it as a meticulously organized library where books are systematically arranged, labeled, and readily accessible. Consequently, organizations find this data type highly conducive for effective utilization.

*2) Semi-Structured Data:* Semi-structured data exhibits a loosely defined or irregular structure, allowing a schema-less description format. Unlike traditional database constraints, this data model enables the amalgamation of information from diverse sources with related yet differing properties. It is adaptable for database integration and information sharing on the Web. This data type may be irregular, incomplete, and prone to rapid or unpredictable changes. It lacks a fixed schema, being both schema-less and self-describing, carrying its own schema information.

*3) Quasi-Structured Data:* Quasi-structured data lies be- tween semi-structured and unstructured data types. It is un- structured yet exists in an erratic format that necessitates specialized tools for handling. Essentially, quasi-structured data lacks definitive structure but can be managed with specific processing tools.

*4) Unstructured Data:* Unstructured data, often termed messy or disorganized raw data, primarily comprises textual information and various multimedia files. This data lacks a formalized structure, making its organization unidentifiable. Examples include word processing documents, books, jour- nals, health records, emails, multimedia content like videos, presentations, web pages, photos, audio files, and a myriad of other business documents.

*B. Big Data Challenges*

In existing literature, numerous studies have delved into the multifaceted challenges presented by big data. Addressing these challenges and refining the performance of machine learning algorithms necessitates attention to several key as- pects:

- Streamlining data processing and analysis to minimize computation time.
- Innovating new methodologies capable of scaling for big data, accommodating diverse and loosely structured data, while handling high computational complexities.
- Ensuring efficient synchronization between disparate sys- tems and mitigating bottlenecks at system integration points.
- Devising robust security measures to fortify data privacy and shield individual and corporate information from vulnerabilities proliferating alongside data expansion.
- Enhancing algorithmic efficiency to manage the vast storage demands imposed by large datasets.
- Enabling real-time data analysis from diverse sources to derive immediate value from knowledge.

The above stated various types of Big data can be ef- ficiently managed using various storage management and optimization techniques. This include distributed file systems, NoSQL databases, data compression, data deduplication, tiered storage, data archiving, data replication and data encryption.

This research examines two unique methods designed to improve Big Data storage and management. The first method explores an enhanced version of the Hadoop Distributed File System (REHDFS), emphasizing various block placement strategies to provide a scalable framework with a load-aware

block access mechanism. It also incorporates two models—one pessimistic and the other optimistic—to enable efficient ran- dom read and write operations.

DFS, an abbreviation for distributed file system, represents the approach of storing files across multiple nodes in a distributed fashion. Essentially, DFS offers an abstraction of a unified, expansive system where the collective storage across nodes within a cluster sums up to create a larger storage entity. The Hadoop Distributed File System (HDFS) functions as the primary storage system in a Hadoop cluster. Primarily crafted for use on cost-effective commodity hardware, it is built on a distributed file system architecture. HDFS is engi- neered with a focus on storing data in sizable blocks rather than smaller fragments. Within the Hadoop ecosystem, HDFS ensures fault tolerance and high availability, enhancing the resilience of the storage layer and the devices interconnected within the Hadoop cluster.

The Hadoop Distributed File System (HDFS) is designed with a master-slave architecture, as shown in Fig. 2. The NameNode serves as the master node, managing metadata and overseeing the file system namespace. It maintains details about file locations, permissions, and the directory and file structure. DataNodes function as the slave nodes, handling the actual storage of data. They manage the storage resources of the nodes and execute read and write operations under the guidance of the NameNode.

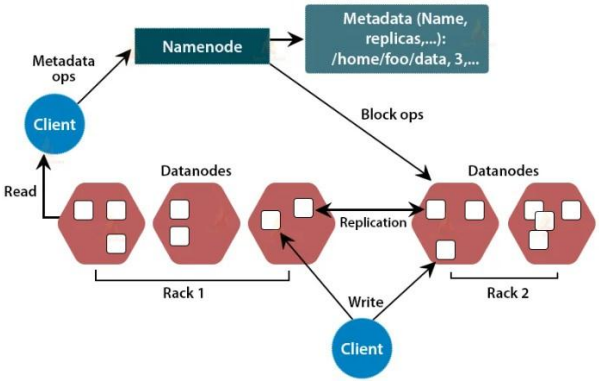


Fig. 2. HDFS Architecture [2]

The key elements of HDFS architecture include fault tol- erance through data replication across multiple DataNodes, scalability by allowing easy addition of more nodes, and the emphasis on data locality processing data where it's stored to reduce network traffic. Files are broken down into blocks (typically 128 MB or 256 MB in size) and distributed across the DataNodes, with replication to ensure fault tolerance. The NameNode keeps track of block locations and manages access to files, while the DataNodes store and retrieve these blocks as per the NameNode's instructions.

The second solution revolves around DNA strands, showcas- ing an extraordinary capacity to securely store extensive data, potentially providing a breakthrough solution for maximizing data storage in minimal space. Advancements in synthetic biology have notably enabled the laboratory-based creation (in vitro) of artificial DNA strands. These synthetic DNA strands possess the same remarkable characteristics as natural DNA but are created without relying on specific DNA templates. This freedom allows the synthesis of virtually any sequence of nucleotides—adenine (A), thymine (T), cytosine (C), and gua- nine (G)—in a laboratory setting. Unlike natural DNA, these artificially produced strands might not necessarily contain genes responsible for life processes; instead, any sequence of nucleotides can be assembled, opening the door for encoding digital information into a quaternary representation—a process known as DNA coding.

Once encoded into DNA form, specialized machines called sequencers facilitate the retrieval of the encoded sequence. DNA sequencing, a biological process, involves reading and decoding any DNA strand to reveal its quaternary content. These foundational biological processes of DNA synthesis (writing) and sequencing (reading) operate similarly to a noisy channel, establishing an encoding workflow fundamental to digital storage within DNA structures.Workflow of DNA storage is shown in the Fig. 3.

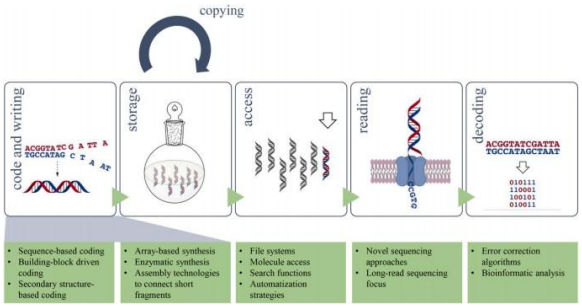


Fig. 3. DNA storage work flow

## II. LITERATURE SURVEY

### *A. AutoML with Bayesian Optimizations*

Automated Machine Learning (AutoML) [3] presents nu- merous applications within Big Data processing, management, and systems. Firstly, it can autonomously enhance machine learning model's performance on extensive datasets by dy- namically choosing suitable algorithms, fine-tuning hyperpa- rameters, and identifying optimal features.

Secondly, AutoML enhances its functionality by simplifying the creation and deployment of machine learning models within a big data environment. For instance, it can facilitate automatic scaling and distribution of models across machine clusters, as well as select the most efficient storage and processing methods tailored to specific datasets.

Furthermore, AutoML plays a crucial role in automating feature engineering processes for big data, significantly reduc- ing the time and resources needed to prepare extensive datasets for machine learning analysis. Automated Machine Learn- ing (AutoML) covers the entire machine learning pipeline,

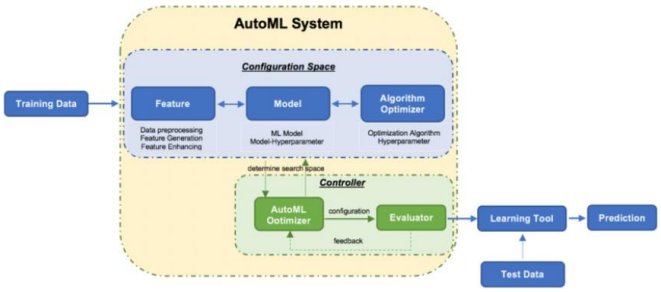


Fig. 4. An illustrative workflow of an AutoML system [4]

automating processes such as data preprocessing, model se- lection, training, and deployment. AutoML employs various techniques to achieve this level of automation, some of which include:

- Hyperparameter tuning in AutoML automates the pro- cess of finding the optimal set of hyperparameters for a specific machine learning model. This is done using techniques like grid search, random search, or Bayesian optimization to explore the best combinations.
- Feature selection and engineering in AutoML automati- cally identify the most relevant features in a dataset. It also carries out essential feature engineering tasks, such as scaling, normalization, and dimensionality reduction, to improve model performance.
- Model selection is a crucial function in AutoML that automatically determines the best-suited machine learn- ing model for a dataset. This is achieved by evaluating the performance of different models or through advanced techniques like ensembling or stacking, which combine the predictions from multiple models.
- Neural Architecture Search (NAS), a specialized area of AutoML, automates the design of neural network archi- tectures. Its main goal is to identify the most effective architecture for a given task and dataset.
- Automated deployment, another critical aspect of Au- toML, simplifies the process of moving machine learning models into production. This includes tasks like model versioning, continuous monitoring, and ensuring scalabil- ity for smooth integration into operational environments.

*1) Methodology:* The workflow of an AutoML system involves two primary components as shown in Fig. 4. Firstly, a configuration space outlines what aspects are automated in the process. This includes data

preprocessing (such as feature engineering), model selection, and optimization algo- rithm selection, covering the fundamental areas of machine learning applications. Feature engineering, an integral part of this, enhances predictive features through methods like dimension reduction, feature generation, and encoding. The configuration search within feature engineering spans two spaces: hyperparameters for dimension-reduction techniques and choosing and creating features.

Next, the phase that deal with selection of model focuses on automatically identifying a set of classifiers and fine- tuning their hyperparameters to achieve optimal learning per- formance. This involves a hierarchical exploration, selecting the classifier first and then determining the corresponding hyperparameter values. Similarly, the algorithm optimization step involves exploring optimization algorithm types and their respective hyperparameter values within a hierarchical search space.
Secondly, the controller dictates the approach to solving the AutoML problem. It comprises two main components: an AutoML optimizer and an evaluator. The optimizer techniques align with the defined search spaces in the first part and vary in efficiency in generating configurations until an optimal one is discovered. Various optimization techniques like simple search, greedy search, and derivative-free optimization have been studied, each with its specific costs and advantages.

*B. NoSQL Databases*

Health data is widely acknowledged for its intricate, sensi- tive, and substantial nature, characterized by complex struc- tures, ownership concerns, and diverse data formats. Over the past two decades, concerted efforts have been directed towards efficiently storing health data in electronic formats, aiming to provide access to those who require it. The spectrum of health data spans structured, semi-structured, and unstructured for- mats, encompassing textual information, data originating from diverse medical devices, visual representations like charts, graphs, images, as well as audio, video files, and sensor- generated data.
Modern health data often includes extensive textual com- ments used to evaluate an individual's mental well- being, adding to its complex and varied nature. As a result, the storage, management, and processing of these different data formats require distinct approaches. The challenge is to effi- ciently store diverse types of health data, each with varying volumes, within a single data repository, while also meeting the performance requirements of such multifaceted informa- tion.
Numerous studies have highlighted the proficiency of NoSQL databases in managing complex representations of Big Data. The utilization of NoSQL databases notably en- hances query performance and facilitates horizontal scalability. However, crafting a NoSQL database demands a non-trivial approach. The conventional method employed for designing relational databases, starting with a conceptual model, pro- gressing to a logical model, and culminating in implementing a physical model, doesn't consistently align with NoSQL databases.
While design of the relational database primarily centers on optimizing storage, the design strategy for NoSQL databases shifts focus toward enhancing query performance. As a result, it adopts a query-driven approach. The crux of effective NoSQL database design hinges upon comprehending intricate application query patterns, marking the initial and critical stride toward query-driven database design.

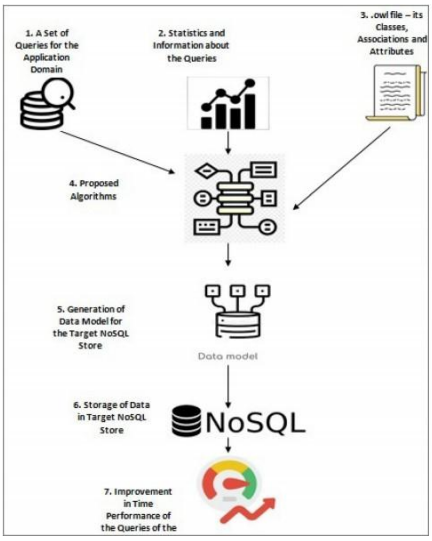


Fig. 5. The complete process of developing a data model for a NoSQL database. [5]

*1) Methodology:* The process illustrated in Fig. 5, pre- sented as an algorithm, demonstrates its effectiveness by comparing the query response times of the resulting database to those of a relational design within the same domain, using identical queries. Surprisingly, this topic hasn't received signif- icant attention within the research domain. Despite meticulous data model design by software industry developers, consid- ering application domain, data characteristics, and storage options, this aspect hasn't been emphasized.

### *C. K-Means Clustering Algorithm*

K-means Clustering serves the purpose of grouping data into distinct clusters, each comprised of similar data points. Positioned within the domain of data mining, this algorithm is instrumental in partitioning and analyzing data, particularly in handling large volumes.

The K-means algorithm operates through five fundamental steps. Initially, the number of clusters, denoted as 'k', is chosen. Subsequently, k points are randomly selected along the data axis, termed as centroids, even if they don't precisely correspond to existing data points. These centroids act as pivotal markers. The algorithm proceeds by calculating the distance from each data point to all k centroids, gauging their dissimilarity based, for instance, on income disparity.

Following this, clusters are formed by associating each data point with the nearest centroids. The subsequent phase involves recalculating the centroids by computing the aver- age value from the data points within each cluster. These recalculated averages become the new centroids, replacing the previous values. This iterative process continues through steps two to four, iteratively updating centroids until their positions stabilize and cease to change.

### *D. Signature Based Compression Technique*

SIBACO [6]is a pioneering signature-based compression method designed to augment existing practices while seam- lessly integrating with column and hybrid storage systems. SIBACO aims to further curtail storage costs effectively.

Central to SIBACO is the hypothesis that employing mul- tiple data compression schemes proves more efficacious for intricate big data scenarios. This approach enables incre- mental compression and selective decompression, tailoring compression schemes based on diverse column types and data attributes. SIBACO systematically divides data into clusters of columns or rows, applying the most suitable compression scheme to individual or grouped columns/rows.

The method relies on a knowledge base to select com- pression schemes, utilizing a mapping system that correlates specific data signatures with appropriate compression schemes. Leveraging insights from the database catalog and historical data, SIBACO effectively determines and applies the optimal compression scheme to enhance storage efficiency. SIBACO attains superior compression ratios through distinct combina- tions and compression methodologies tailored to the data type and distribution within columns. Its compression methodology comprises three sequential stages, illustrated in Fig. 6. (i) identification of compatible columns, (ii) segmentation and clustering of these compatible columns, and (iii) subsequent selection of compression techniques based on data types. Upon retrieval, SIBACO utilizes an ordered index system, facilitating the extraction of compressed files by employing the relevant compression scheme.

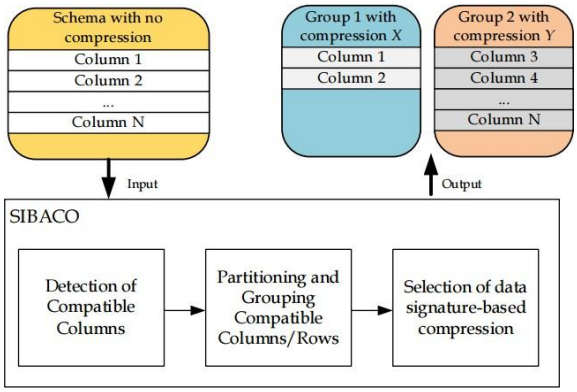


Fig. 6. The three stages of SIBACO [6]

*E. Hidden Markov Model and Constraint Satisfaction Prob- lem*

Hidden Markov models (HMMs) stand as robust machine learning algorithms extensively employed in various machine learning scenarios. HMMs have demonstrated successful ap- plications in diverse domains such as speech recognition, face detection, bioinformatics, financial analysis, among others. The surge in the utilization of HMMs for extensive datasets, characterized by a significant number of states and observa- tions, has sparked researchers' interest in enhancing HMMs for big data applications. This growing trend emphasizes the research focus on developing novel methodologies to adapt HMMs to the context of big data, aiming to augment their performance. A novel method for optimizing big data revolves around leveraging the constraint satisfaction problem, a funda- mental concept within metaheuristics. This approach adopts a CSP graph-based strategy, specifically targeting the reduction of the state space inherent in hidden Markov models. The objective is to enhance learning and prediction tasks associated with HMMs.

*1) Methodology:* In this methodology, HMMs are treated akin to a CSP, albeit not restricted to a formalism. This accounts for the utilization of CSP solver algorithms aimed at resolving such challenges. Notably, this concept extends beyond solely addressing states or observations; it simultane- ously tackles both aspects—an aspect distinguishing it from other methodologies. While primarily tailored for big data applications, this approach also demonstrates adaptability to conventional smaller datasets.

*F. SEEDDUP: A Three-Tier SEcurE Data DedUPlication Architecture-Based Storage Over Cloud*

The content-aware chunking algorithm [7] adapts the chunk size according to file content, adeptly handling minor informa- tion updates and data partitioning while maintaining minimal variance in chunk size. It's discerned that this method of content-aware chunking proves more adept for data dedupli- cation .

Subsequently, hash values, also termed as fingerprints, are generated for each chunk utilizing hashing algorithms like MD-5, SHA-1, and SHA-256. These unique hash values serve to distinctly identify each chunk. These hashes are then transmitted to a centralized cloud infrastructure for the purpose of deduplication. Fig. 7.depicts the SEEDDUP system for cloud storage. The SEEDUP comprised of four entities: Key Generation Center (KGC), Cloud Server (CS), Data Owners (DOs) and Data Users (DUs).

Presently, prevailing data deduplication systems rely on a singular centralized server for the deduplication process. However, these systems encounter limitations due to restricted processing capabilities, proving insufficient for efficiently han- dling the vast scale of petabytes/terabytes characteristic of big data deduplication.

*G. Distributed Principal Subspace Analysis*

Principal Subspace Analysis (PSA), along with its counter- part Principal Component Analysis (PCA), stands as a preva- lent method for reducing dimensionality in signal processing and machine learning. However, the conventional centralized approaches of PSA/PCA are becoming outdated in the era

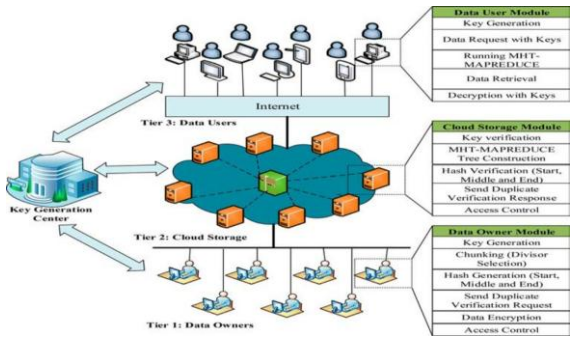


Fig. 7. SEEDDUP System Architecture
[7]

of big data, where the volume of samples or the sample dimensionality often surpasses the storage and computational capacities of individual machines. Consequently, the focus has shifted to distributed solutions for PSA/PCA, where data gets distributed across multiple machines, and collaboration among these machines yields an estimation of the principal subspace. This research delves into the realm of distributed PSA/PCA within a network of interconnected machines with- out a central server. The contributions of this study are threefold: introducing two algorithms tailored for distributed PSA/PCA: one for sample partitioned data and another for partitioned (raw) features; analyzing the convergence of the proposed algorithms towards the true subspace in the case of sample wise partitioned data; and conducting extensive experiments, encompassing synthetic and real world data, to validate the efficacy of these algorithms. Particularly, in the scenario of

sample-wise partitioned data, an MPI-based distributed implementation is explored to understand the in- terplay between network topology, communication costs, and the impact of straggler machines on the effectiveness of the proposed algorithms.

*1) Methodology:* The methodology encompasses several innovative algorithms addressing distributed Principal Sub- space Analysis (PSA). Firstly, there's F-DOT, a novel algo- rithm focusing on feature-wise distributed PSA. Secondly, S- DOT, a sample-wise distributed PSA algorithm, is introduced, accompanied by a variant SA-DOT that dynamically ad- justs consensus iterations for orthogonal iterations. Theoretical guarantees for convergence are provided for S-DOT and SA- DOT. Additionally, experiments utilizing Message Passing Interface (MPI) aim to assess communication costs in real- world scenarios. Extensive numerical experiments compare the efficiency of these proposed algorithms against existing distributed and baseline methods.

The core objective of this paper revolves around solving PSA concerns when data is partitioned across interconnected nodes either by features or samples, within an arbitrary net- work. To tackle dimension reduction in distributed settings for these data partitions, the proposed algorithms aim to identify the principal eigenspace of the data covariance matrix without relying on a central entity to collate or coordinate the data. The foundational algorithm for all proposed solutions is Orthogonal Iteration (OI), known for eigenspace estimation in centralized settings. Adapting OI to distributed setups requires meticulous handling to maintain orthonormality (in the case of F-DOT) and network consensus (for S-DOT and SA-DOT).

Extensive experimental results support these claims, even though theoretical guarantees for the F-DOT algorithm are not provided. Experimental simulations highlight its efficiency. The study delves into real-world distributed networks using MPI and examines various algorithmic parameters like net- work connectivity and data dimension. Furthermore, these distributed PSA developments, rooted in OI, generalize to the distributed PCA problem specifically when distinct eigenval- ues of the covariance matrix exist. However, the focus remains on the distributed PSA problem without emphasizing distinct eigenvalues, limiting the scope of the study.

### *H. REHDFS System*

The Hadoop Distributed File System (HDFS) stands as a remarkably scalable and fault-tolerant data storage platform renowned for its cost-effectiveness and reliable storage capac- ity. It accommodates vast quantities of data, allowing storage of files exceeding 1 TB, and adeptly manages both structured and unstructured data. Data within the HDFS is segregated into blocks, strategically replicated across numerous data nodes. This replication not only bolsters throughput and access speed but also fortifies the system against faults.

However, a limitation of the conventional HDFS is its restriction to sequential reads and writes, lacking support for random access. This section introduces an enhanced version of HDFS termed REHDFS

This section serves to illustrate the enhanced features of the HDFS system compared to its traditional counterpart, outlining the distinctive optimistic and pessimistic models instrumental in enabling random write operations.

Features of REHDFS System: Within HDFS, files undergo segmentation into blocks, distributed across multiple data nodes to ensure resilience against faults. By default, HDFS employs a predetermined block size, yet users retain the flexibility to modify this parameter. The implementation of HDFS architecture allows for experimentation with various block placement strategies. However, conventional HDFS sys- tems lack support for random read and write operations. The enhanced REHDFS system introduces several novel function- alities, encompassing:

- Diverse block placement strategies
- Customizable block size selection
- Strategies for block caching and retrieval
- Advanced methodologies for Data Node selection
- Incorporation of capabilities for random read and write operations

*1) Methodology:* The methodology implemented in en- hanced HDFS are-one, Random Write: File Level Lock with a Pessimistic Model (FLPM). As depicted in Fig.8., within a pessimistic model, a file can exist in three distinct states: closed, open, and locked.

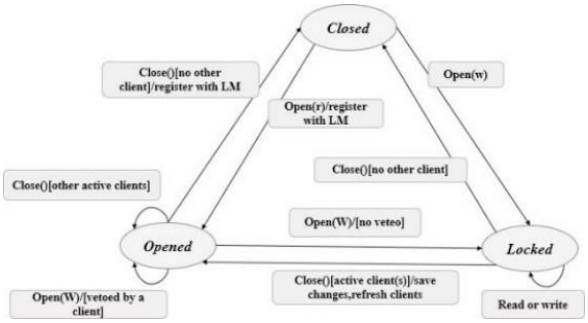


Fig. 8. The pessimistic model state diagram of a file
[8]

In this model, each file starts in a closed state and transitions to an open (locked) state when a client initiates access. This process involves the client being registered via the lock man- ager (LM). The accompanying figure illustrates the sequential changes in file states driven by client actions and subsequent requests. In the Optimistic Model with File Level Consistency (OMFC), depicted in Fig. 9, a client goes through four dis- tinct states: inactive, registration with the validation manager, update (modifying read blocks) and record (attempting to save the altered blocks). The diagram below outlines the sequential states of a file in the optimistic model.

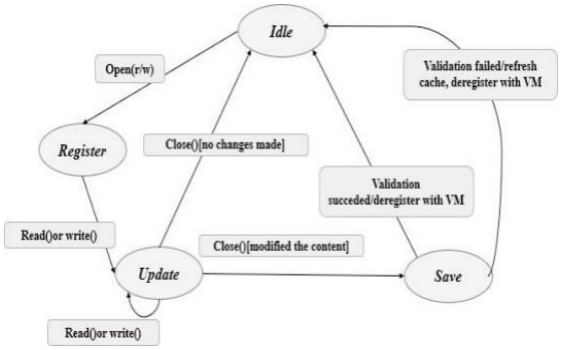


Fig. 9. The state diagram of a customer in the optimistic model
[8]

*I. DNA genetic storage media*

The paper [8] encompasses that storing computer data poses challenges not just in terms of volume but also in preserving data over time. Existing storage technologies have finite lifespans and are susceptible to fragility. Consequently, there's a growing need for computer media capable of securely retaining immense data over extended periods. DNA emerges as a compelling solution owing to its durability and remarkable information density. As a result, DNA storage is increasingly considered a fitting choice for the forthcoming generation to address these persistent challenges in data storage. Fig.10. shows the overview of DNA data storage system.

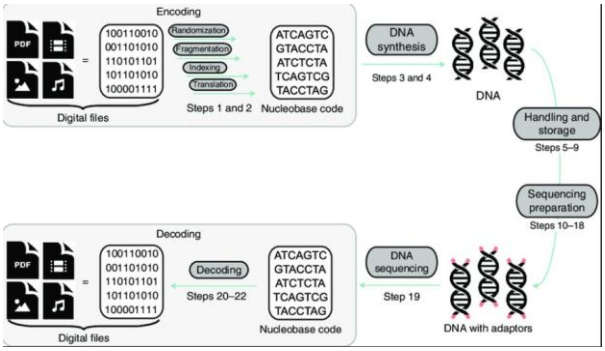


Fig. 10. DNA Digital Data Storage
[8]

*1) Methodology:* The process of storing information within a DNA molecule involves two main stages: the Processing stage and the Encoding/Decoding algorithm.

**Encoding Algorithm**:

- Examine data stream: A
- Check the size of the data: [r, c, n] = size(A), where r is the number of rows, c is the number of columns, and n is the number of matrices
- Calculate the size of the DNA sequences for the image
- Generate a zero matrix of the DNA sequences' size
- While the DNA sequences' size equals the maximum:
  - Convert even the smallest data piece into binary format
  - Embed the binary DNA code of an individual data cell into the DNA sequence
  - Proceed until the maximum size of the DNA se- quence is reached

**Decoding Algorithm**:

- Retrieve the DNA sequence
- Calculate the size: length and size of each cell
- Translate DNA into actual data for decoding:
  - Transform a single cell into data
  - Embed the transformed value into the template data matrix
  - Cease the conversion process upon completing the entire DNA sequence

## I. DISCUSSION

### *A. Comparison of DNA digital storage with traditional sys- tems*

The properties of DNA promises it as an exceedingly promising avenue for digital data storage. As per a "Nature" journal article, scientists propose a rough theoretical estimation suggesting that 1kg of DNA could conceivably house the entirety of the world's digital information. Fig. 9. presents a table juxtaposing DNA molecules with prevalent storage devices like hard disks and flash memories.

DNA digital storage holds significant promise due to its exceptional data density and enduring longevity. By encoding vast quantities of data within DNA molecules, it achieves an unprecedented storage capacity in a remarkably compact space, surpassing conventional storage methods by leaps and bounds. Additionally, DNA's remarkable stability ensures data preservation for millennia when stored under optimal condi- tions, making it an appealing choice for long-term archiving needs. The current efficacy of DNA storage faces challenges, including the high costs linked to synthesis and sequencing, slower read and write speeds, and the intricacies involved in encoding and decoding data. Overcoming these obstacles through technological advancements and cost-efficient strate- gies will play a pivotal role in maximizing the potential of DNA digital storage, potentially reshaping the landscape of data storage and archival practices in the future.

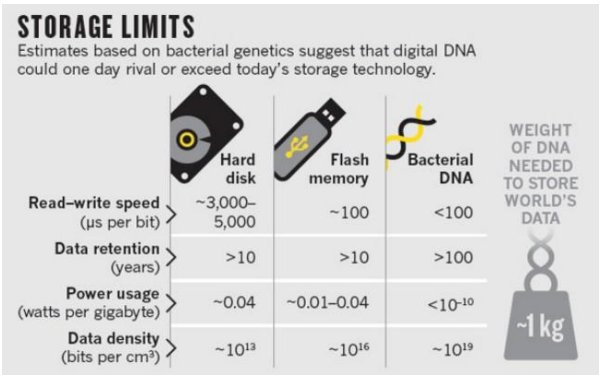


Fig. 11. A Comparative Analysis of DNA and Conventional Digital Data Storage [8]

### *B. Evaluating Learning Algorithm Performance in the Context of Big Data*

Utilizing big data and AI technology unlocks numerous opportunities to improve business performance and efficiency, including:

- Forecasting and leveraging new industry and market trends.
- Analyze customer behavior to streamline and automate customer segmentation.
- Tailoring and improving digital marketing campaigns to improve efficiency and impact.
- Developing intelligent decision support systems driven by big data, AI and predictive analytics.

- Big data analytics is fundamental to several AI disciplines such as machine learning, natural language processing (NLP), computer vision, deep learning, and recommen- dation systems. These areas utilize vast datasets to extract valuable information.

*C. Evaluating Power Consumption Using Big Data Analytics*

REHDFS marks a major leap forward in storage technology, notably recognized for its substantial reduction in power

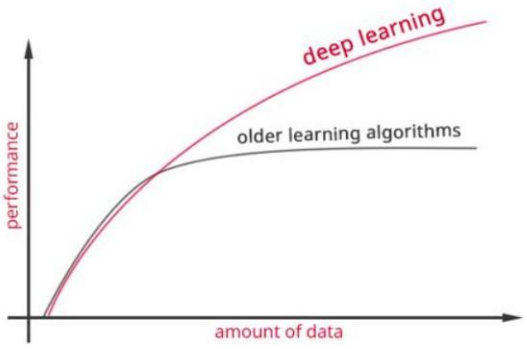


Fig. 12. Benefits of Big Data for AI or Deep Learning Algorithmic Performance[3]

consumption when compared to conventional storage systems. This breakthrough is made possible through advanced data management strategies, sophisticated algorithms, and opti- mized hardware configurations that focus on energy efficiency while maintaining data accessibility and performance. En- hanced HDFS excels in reducing power usage during data storage, retrieval, and processing, providing both economic benefits and promoting a greener, more sustainable approach to large-scale data storage solutions. Fig. 11. illustrates how this reduction in power consumption positions REHDFS as a leading solution in the field of energy efficient storage systems, supporting modern sustainability goals, and addressing the increasing demand for environmentally friendly technology solutions.

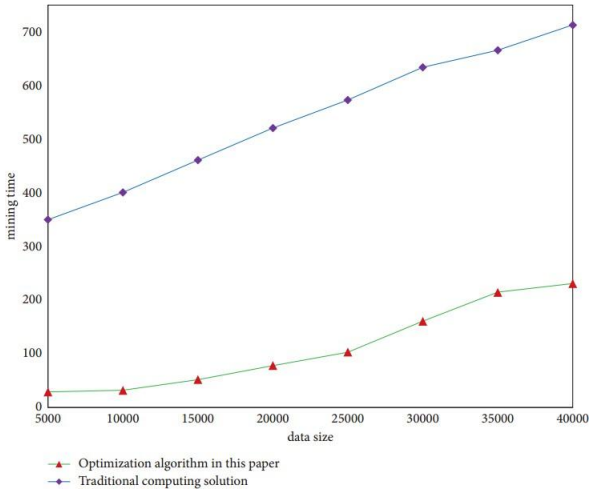


Fig. 13. Evaluating Power Consumption Using Big Data Analytics [2]

*D. Performance analysis of mining time with Big Data*

REHDFS offers a significant breakthrough with its notably reduced data mining time when compared to traditional storage systems. This improvement is achieved through cutting-edge data retrieval techniques, efficient algorithms, and optimized hardware configurations that accelerate the process of locating and accessing stored information. REHDFS excels in minimiz- ing the time required for data mining tasks, enabling faster access to critical information within vast datasets, as depicted in Fig. 13. This efficiency not only boosts operational speed but also significantly lowers the time and resources needed for data analysis and decision-making. By speeding up mining operations, REHDFS stands out as an innovative solution, transforming the

efficiency and responsiveness of data-driven tasks across various domains, from research and analytics to large-scale industrial applications.

their modifications. When multiple clients operate on the same file, the OMFC model surpasses the FLPM model. Future enhancements to REHDFS aim for comprehensive random write capabilities alongside heightened security.

The second method involves harnessing DNA memory strands, enabling the storage of vast data volumes within a minuscule space for long-term preservation. DNA, when conserved under specific conditions (dry, dark, and cold), retains encoded information for thousands of years. The process involves encoding and decoding data into binary form, transferring it to and from abbreviated DNA strands within an extensive virtual DNA molecule. Storing data in DNA offers simplicity, convenience, cost-efficiency and long- term data fidelity. However, this technology currently remains prohibitively expensive, demanding substantial resources and high-end hardware.

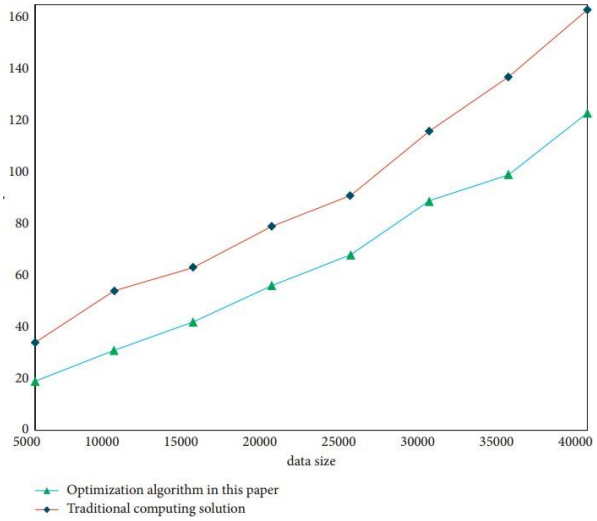


Fig. 14. Performance analysis of mining time with Big Data [2]

## II. CONCLUSION

The exponential surge in data production has spurred a pressing need for enhanced storage devices, urging the exploration of sustainable data retention options that are faster, more energy-efficient, and sustainable. Consequently, researchers have pivoted towards adapting, augmenting existing technologies, or inventing novel, more efficient methodologies to meet this demand. This paper provides a concise overview of available options, their functionalities, and, notably, their added value.

The primary approach focuses on enhancing the HDFS storage system, resulting in a more robust REHDFS. Improved algorithms empower the system to handle a greater volume of jobs, with a primary emphasis on random writing employing optimistic and pessimistic models. The divergence between these models lies in their approach to random writing. In the pessimistic model (FLPM), a client must secure a file lock before editing, potentially leading to deadlocks within client groups attempting consecutive file alterations. Conversely, the optimistic model (OMFC) allows all clients to edit cached blocks, with the earliest timestamp client able to preserve

## REFERENCES

[1] R. M. J. A. Manar Sais, Najat Rafalia, "Distributed storage optimization using multi-agent systems in hadoop," E3S Web of Conferences 412, 01091, 2023.

[2] A. Sultana, "Unraveling the data structures of big data, the hdfs architecture and importance of data replication in hdfs," International Research Journal of Engineering and Technology (IRJET), 2018.

[3] N. S. M. A. S. S. Aristeidis Karras, Christos Karras, "Automl with bayesian optimizations for big data management," Information jounal, 2023.

[4] S. Y. K. G. Krishna, Vaibhav, "Integrating advanced machine learning in information systems research: What can automated machine learning and transfer learning offer,"

[5] N. M. Poly Sil Sen, "An ontology-based approach to designing a nosql database for semistructured and unstructured health data," Springer Science+Business Media, LLC, part of Springer Nature, 2023.

[6] E. S. N. N. Constantinos Costa, Marios Costa, "Towards a signature based compression technique for big data storage," Cyprus' Research Promotion Foundation RESTART programme, 2020.

[7] B. R. B. . P. Chitra, "Seeddup: A three-tier secure data deduplication architecture-based storage and retrieval for cross-domains over cloud," IETE Journal of Research, 2021.

[8] J. A. Manar Saisa, Najat Rafaliaa, "Intelligent approaches to optimizing big data storage and management: Rehdfs system and

dna storage," The 3rd International Workshop on Statistical Methods and Artificial Intelligence (IWSMAI), 2022.
[9] N. S. M. A. S. S. Aristeidis Karras, Christos Karras, "Sdn helps big-data to optimize access to data," arXiv:2012.15295v1 [cs.DC], December,2020.
[10] S. A. Imad Sassi and A. Bekkhoucha, "A graph-based big data optimiza- tion approach using hidden markov model and constraint satisfaction problem," springer Open, Big Data Journal, 2021.
[11] B. X. Arpita Gang and W. U. Bajwa, "Distributed principal subspace analysis for partitioned big data: Algorithms, analysis, and implementa- tion," arXiv:2103.06406v3 [cs.LG], 2021.
[12] C. W. J. G. Y. L. U. F. M. P.- L. D. K. R. A. Zongheng Yang, Badrish Chandramouli, "Qd-tree: Learning data layouts for big data analytics," arXiv:2004.10898v1 [cs.DB], 2020.