The Scientific Temper (2023) Vol. 14 (4): 1398-1403
E-ISSN: 2231-6396, ISSN: 0976-8653
Doi: 10.58414/SCIENTIFICTEMPER.2023.14.4.50
https://scientifictemper.com/

RESEARCH ARTICLE

A general stochastic model to handle deduplication challenges using hidden Markov model in big data analytics

Sahaya Jenitha A.*, Sinthu J. Prakash

Abstract

Background: Since increased interest of consumers, cloud computing is needed to store and access the information about their data in their convenient way. In recent days, cloud computing offers many services stipulated by the internet. Data duplication is one of the main challenges in big data analytics that leads to increased data storage and processing time. Therefore, there is a need to develop a data deduplication process. It eliminates excessive copies of data as well as decreases the storage space. In order to preserve the accurate data information without any duplication, joint probability distribution computes the likelihood of two events occurring together at the same time and thus it leads to removing the redundant data before data is sent to the cloud server.

Methods: This paper presents a general stochastic model (GSM) algorithm that uses hidden markov model, likelihood estimation, markov chain transition, and poisson distribution model.

Findings: Joint probability distribution computes the likelihood of two events occurring together at the same time and thus it leads to removing the redundant data before data is sent to the cloud server.

Novelty and applications: This paper proposes the GSM to handle redundant data by a multi-level process using hidden markov model (HMM), likelihood estimation, transition probability and poisson distribution model (PDM).

Keywords: Hidden markov model, Markov chain transition, Likelihood estimation, Poisson distribution.

Introduction

Data duplication offers a complete availability by saving the duplicate data to all the cloud servers. Duplication may occur in the server, storage devices, network sites and host. The server is utilized to copy the files from one network to another is the duplication host. Duplication in the array is the data replication in the arrays. There are two ways in data duplication. One is synchronous duplication and the other is asynchronous duplication.

The duplication that occurs in the real time is called as the data deduplication. In real time the lost data cannot be recovered. It also includes latency which leads to increase in cost. It is done by means of the array duplication and network duplication. Duplication that needs low bandwidth for transmission over distances is termed as the asynchronous duplication. As the data are copied and transmitted over distances there may be delay in copying data and some data might be lost due to these issues. This is supported by array, network and host based replication.

Material and Methods

The process of data deduplication is used to eliminate the duplicate data to reduce the storage memory in the database. Not only does it reduce the memory, it is also used to save time and to search the data in the database. This paper proposes the general stochastic model (GSM) to handle redundant data by a multi-level process using hidden markov model (HMM), likelihood estimation, transition probability and Poisson distribution model (PDM). The GSM considers stock market data as an input. The GSM is described in the flow diagram in Figure 1.

Developing Temporary Database

A temporary database is needed to store the data. A Matrix is one of the structures to collect the data. The $D = m \times n$

Department of Computer Science, Cauvery College for Women, Bharathidasan University, Tiruchirappalli, Tamil Nadu, India.

***Corresponding Author:** Sahaya Jenitha A., Department of Computer Science, Cauvery College for Women, Bharathidasan University, Tiruchirappalli, Tamil Nadu, India, E-Mail: jenitaguhan@gmail.com

How to cite this article: Jenitha, S.A., Prakash, S.J. (2023). A general stochastic model to handle deduplication challenges using hidden Markov model in big data analytics. The Scientific Temper, **14**(4):1398-1403.

Doi: 10.58414/SCIENTIFICTEMPER.2023.14.4.50

Source of support: Nil

Conflict of interest: None.

Received: 12/10/2023 **Accepted:** 15/11/2023 **Published:**31/12/2023

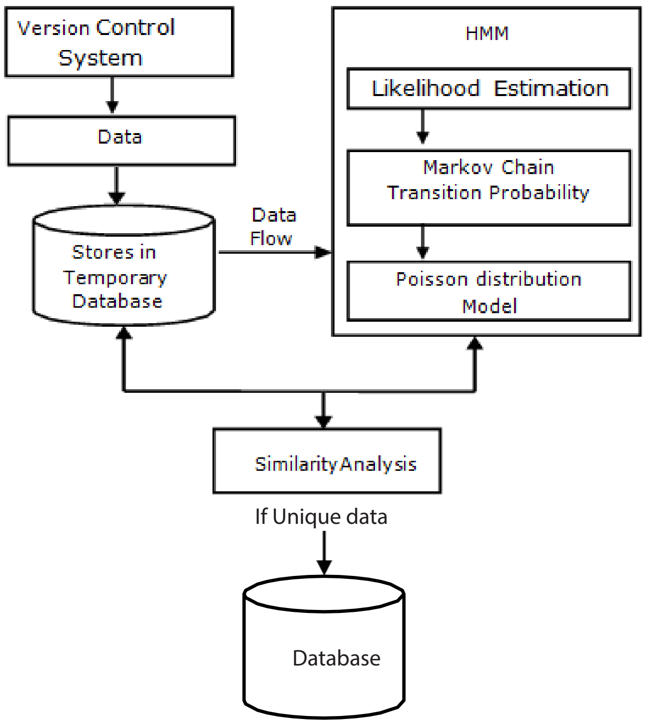

Figure 1: The work flow of GSM

matrix is initialized. Each data is represented in the row of the matrix as *m* as an instance. The attributes of the data are stored in the column of the matrix as *n*. It is necessary to check whether the same information is stored multiple times. Also, it will be taking more time to search the relevant information in the database. The output of the matrix data structure can be represented in eqn. (1).

$$D = \begin{pmatrix} d_{11} & d_{12} & d_{13} & \ldots\ldots & d_{1m} \\ d_{21} & d_{22} & d_{23} & \ldots\ldots & d_{2m} \\ \ldots & \ldots & \ldots & \ldots\ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots\ldots & \ldots \\ d_{n1} & d_{n2} & d_{n3} & \ldots\ldots & d_{nm} \end{pmatrix} \quad (1)$$

Finally, the processed data will be stored in the cloud server. It is used for answer the query from clients**.**

Hidden Markov Model

The HMM is used to analyze the sequence and temporal data. Also, it computes distribution of events that can be observed []. An event is a set of all possible outcomes**.** The *N* is a number of data that is stored in *D*. Let *G* be the group of data that is denoted by the eqn. (2). Every group of data which consists of existing instance d_{mn} data needs to be used for making a HMM to analyze whether the current data information is redundant or not.

$$G = \left\{\left\{d_{mn}\right\}_{n=1}^{n+1}\right\}_{n=1}^{N} \quad (2)$$

A HMM is built by a number of finite states where the states are called hidden states. The hidden states are associated with transition probabilities. It is computed by the Markov chain transition probability. The system goes into the states using transition probability at each time.

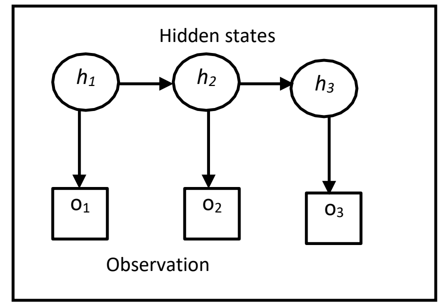

Figure 2: The general structure of HMM

An output of transition probability is made of observable symbols. It is computed by observation probabilities that are derived from the current state. A HMM is usually defined by following parameters (*H, O, P_{ij},*)

Let H_s be the hidden states as *{h1, h2, h3)* where *s = 1* to 3. Let h_1 is be the LOW (L), h_2 be the MODERATE (M), and $h_3$be the HIGH(S).

Let P_{ij} be the transition probability and the $p_{ij}$is the set of the probabilities that the system enters from state h_i to state h_j.

Let O_e be the observed symbols as *{o1, o2, o3}* where *e = 1* to *3*. The *O* is a function of *H*. The *H* and *O* will have a finite of possible states. The statistics values of observed symbols are calculated using the PD. This system considers o_1 is be the INCREASES (I), o_2 be the DECREASES (D), and o_3 be the NO_CHANGES (N) changes in for each d_{mn} instance.

Let π be the initial probabilities where {π_i} the probability of system starts in h_i. The general structure of HMM is explained in the Figure 2.

Likelihood Estimation

Likelihood estimation is a joint probability that calculates the likelihood of two states *(H* and *O*) occurring together. Joint probability is the probability of event *H* occurring at the same time that event *O* occurs. It is computed by the eqn. (3).

$$P(I, D,N, L, M, S) = P(I|L)\ P(D|M)\ P(N|S) \times P(L)\ P(M|L)\ P(S|M) \quad (3)$$

In eqn. (3), all the information of this parts *P(L) P(M|L) P(S|M)* belongs to transition probability. Transition probability can be computed by using Markov chain transition probability (MCTP) matrix. Also *P(I|L) P(D|M) P(N|S)* parts can be derived using PDM.

Markov Chain Transition Probabilities

The MCTP is used to find the transition probabilities of hidden states. Markov chain is a process that the system moves from one state to immediate adjacent state []. The h_{uv} is the hidden states probabilities that the state enters from $h_u$to state h_v. It is represented by the following eqn. (4).

$$P_{ij} = \begin{array}{c c} & \begin{array}{ccc} L & M & N \end{array} \\ \begin{array}{c} L \\ M \\ S \end{array} & \begin{pmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{pmatrix} \end{array} \quad (4)$$

A transition probability matrix *P* is defined to be a stochastic matrix with*0 pij 1*elements. The summation of its row or column should be one. The *P* is derived from a set of *H*and the elements are computed using the eqn. (5).

$$P_{kl} = \frac{p_{uj}}{p_{u1} + p_{u2} + p_{u3}} \quad (5)$$

Where p_{ij} is the current hidden state and $p_{u1} + p_{u2} + p_{u3}$ is the row of current hidden state. The processing of states transition probabilities is explained in Algorithm 1.

Algorithm 1 State Transition Probabilities

Input: Hidden states H (L, M, and *S*)

Output: Probabilities of all combination hidden states.

Step 1: In every pair of hidden states hij, the inter relationship between two hidden states has to be computed evaluating where *L=L, L=M, L=S, M=L, M=M, M=S, S=L, S=M, S=S,*

Step 2: If any transition is occurred between two hidden states, the respective P*ij = 1.*

Step 3: The step 2 iterate until reach all row hidden state transition in *P.*

Step 4: If greater than 1 event is occurred in *P*, the ratio is to be found between total number of possible events and number of transition occurred for a states.

Step 5: If there is no state transition occurred between *hij* and *hij+1,* the respective P*ij =0.*

Step 6: Generating all neighborhood state transition and providing probabilities for each state.

Poisson Distribution Model

The poisson distribution model (PDM) is a discrete probability distribution that expresses the probability of given a number of events occurring in a fixed interval of time. It can be produced a sequence of hidden *H* states with observation *O*. The numbers of possible combinations for H and *O* are computed by using the eqn. (6).

$$M = H^O = 3^3 = 27, \quad (6)$$

where *H* is the total number of hidden states, *O* is the total number of observations. The combination possible events *Cs* and it contains *{c1, c2, c3, , c27)*. The total combinations are to be applied to find the probabilities of their respective probabilities. The previous combination states probability is to be added with the current combination probability. The total possible events probability can be computed using the eqn. (7).

$$\begin{aligned} C_S(P_0) &= C_S\, e^{-\mu} \\ C_S(P_1) &= C_S(P_0) \times \frac{\mu}{1} \\ C_S(P_2) &= C_S(P_1) \times \frac{\mu}{2} \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ C_S(P_n) &= C_S(P_{n-1}) \times \frac{\mu}{n} \end{aligned} \quad (7)$$

where *s = 1* to *27, e* is *2. 71828* that is base of natural *log*, is a mean that is ratio between number of times each hidden states and observation occurred.

The various probability values of MCTP and PDM are evaluated using emission probability matrix.

Table 1: Data analysis to find the similar data

Sl. No.	*Volume*	*Difference*	*Observation symbols*
1	5789		
2	5800	11	I
3	5800	0	N
4	5801	1	I
5	5798	2	I
6	5795	-3	D
7	5798	3	I
8	5800	2	I
9	5801	1	I
10	5801	0	N

$$Z = \begin{array}{c} \\ L \\ M \\ S \end{array}\!\! \begin{array}{c} \begin{array}{ccc} I & I & I \end{array} \\ \begin{pmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{pmatrix} \end{array}, \begin{array}{c} \\ L \\ M \\ S \end{array}\!\! \begin{array}{c} \begin{array}{ccc} D & D & D \end{array} \\ \begin{pmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{pmatrix} \end{array}$$

$$\begin{array}{c} \\ L \\ M \\ S \end{array}\!\! \begin{array}{c} \begin{array}{ccc} D & D & D \end{array} \\ \begin{pmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{pmatrix} \end{array}, \begin{array}{c} \\ L \\ M \\ S \end{array}\!\! \begin{array}{c} \begin{array}{ccc} I & N & D \end{array} \\ \begin{pmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{pmatrix} \end{array} \quad (8)$$

From Z, the highest probability is to be found from all possible combination.

Similarity and Analysis

For grouping data redundancy, the system finds similarity in between data values in attribute basis. It is explained in the Figure 3. Before sending processed data to the cloud server, the destination node needs to be determined to store the data. The example data with sequence are described using in Table 1.

From the Table 1, there were *I* symbols indicated if the attribute values were increased, *D* was indicated if values of attributes were decreased, and *N* symbols were indicated in case of there were no changes in respective attribute values. When the n^{th} day's value subtracted from $(n-1)^{th}$ day's value if gets > 0, symbol is I. When n^{th} day's value subtracted from $(n-1)^{th}$ day's value if gets < 0, then the symbol is *D*. The n^{th} day's value subtracted from $(n-1)^{th}$ day's value if gets 0, then the symbol.

The Table 1 and Figure 3 shows the sequence will be how produced using difference of values. From the sequence, the proposed model can predict the future values of data.

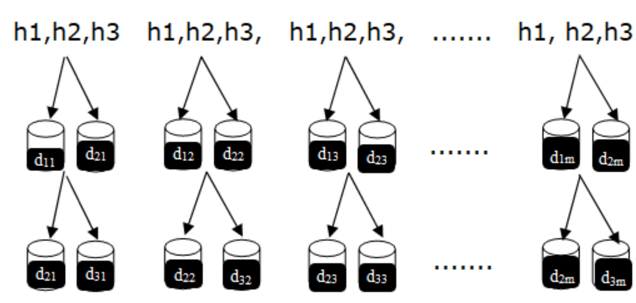

Figure 3: Similarity analysis comparing between attributes

Table 2: Minimum and maximum chunk size with best fit parameters used by TTTD-P

Big datasets (Gigabytes)	*Minimum chunk size (bytes)*	*Maximum chunk size(bytes)*	*Optimal parameter (Divisor, threshold)*
Dataset1 (streaming data)	461	2800	(280,1900)
Dataset2(git hub data)	420	2800	(270,1800)
Dataset3(stored data in drives)	191	2800	(270,1800)

Experimental Results

Introduction

The experiments in this paper analyze and compare the results on different parameters of data deduplication for big data storage system. The scope of the experiments focuses on three core phases of efficient data deduplication for big data storage system. The overall objective of all experiments is to find a good combination of efficient data deduplication for big data storage with the following goals:

- Deduplication ratio is maximal (Table 3)
- CPU usage is minimal

Table 3: Experimental results comparison for data deduplication of big datasets

Before deduplication Input data size(GB) datasets	*Data dedupe algorithm*	*Redundant data size (GB)*	*After deduplication output data size(GB)*	*Deduplication ratio (Input/ output Size)*	*Dedup. time (m.se c)*	*Thruput KB/s*	*Data reduction In %*
174.096609 570	RabinCDC	24.746572643	149.35003692	1.16569512 24	2645196 3	339	14.21
	TTTD	39.900983962	134.19562560	1.29733446 07	2200666 0	443	22.91
	AE	41.001354672	133.09525489	1.30806022 88	1150085 4	1023	23.55
	FASTCDC	39.901023712	134.19558585	1.29733484 50	2641785	3324	22.92
	DEBucket	49.901027316	124.19558225	1.40179389 97	1635083	5313	28.66
	Proposed Method	50.987698956	120.86795784	1.58698508 50	1109390	6453	30.45
226.992897 156	RabinCDC	83.64454814	143.34834901	1.58350548 66	3823697 3	437	36.85
	TTTD	100.46684505	126.52605210	1.79404077 96	3712480 4	543	44.25
	AE	103.64761135	123.34528580	1.84030460 25	1274565 8	1029	45.66
	FASTCDC	100.46711672	126.52578043	1.79404463 17	3821978	3337	44.27
	DEBucket	118.46725039	108.52564676	2.09160602 98	2252592	5343	52.19
	Proposed Method	123.47847847	103.37874878	3.74873847 84	2090908	6345	60.90
156.027040 026	RabinCDC	26.238750875	129.78828915	1.20216578 12	2487498 4	331	16.82
	TTTD	38.823370578	117.20366944	1.33124705 71	2406772 4	423	24.88
	AE	41.333531642	114.69350838	1.36038248 57	1081521 1	1001	26.49
	FASTCDC	38.825812606	117.20122742	1.33127479 51	2488509	3318	24.88
	DEBucket	58.826066794	97.200973232	1.60520038 88	1537053	5311	37.70
	Proposed Method	67.489849894	89.38738378	1.87938989 97	1290390	6321	45.67

- Storage space saving is maximal
- Data reduction is maximal
- Disk reader write I/O's is minimal

Experimental Results and Analysis

The experiments are performed to evaluate the performance of Rabin CDC, TTTD, AE, FAST CDC,TTTD-P , DE-based bucket indexed data deduplication and the proposed deduplication method. Three big datasets are taken one from streaming data, the other is from github code repository data and finally the stored data in drives.

Chunking Efficiency

In order to reduce the overall amount of data, the efficiency of the chunking algorithm is the most important factor. The aim of these experiments is to find optimal parameters that detect maximum redundancy with very low CPU utilization. Deduplication algorithm's efficiency strongly depends on the number of calculations performed on the underlying data as well as on how much data is processed in each step.

Deduplication ratio (DR)

The overall deduplication ratio is defined as input data size before deduplication divided.

DR = Input data size before deduplication/Output data size after deduplication (9)

Deduplication time

Deduplication time is the time required by deduplication technique to give output response.

Chunking throughput

Throughput is a measure of how many units of information a system can process in a given amount of time. In data deduplication, chunking throughput is number of formation of chunks in a given period and typically measure in bits per second (bps), megabits per second (Mbps) or gigabits per seconds (Gbps).

Throughput =Total Data Size/Deduplication Time (10)

Table 2 shows the big datasets used in the experiments. These are real-world big datasets, whose sources are the web servers and mail servers. Then data from git hub repository and already stored data's are also used.

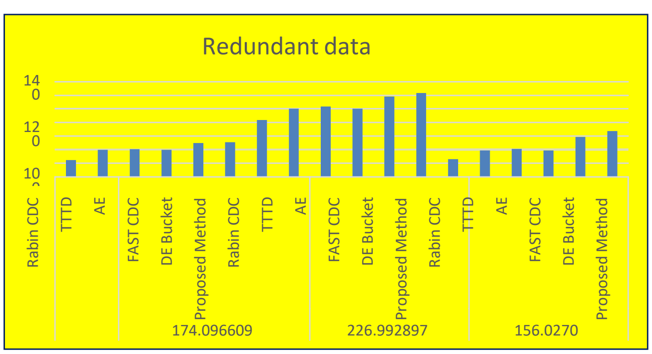

Figure 4: Redundant data detected in existing and proposed algorithms

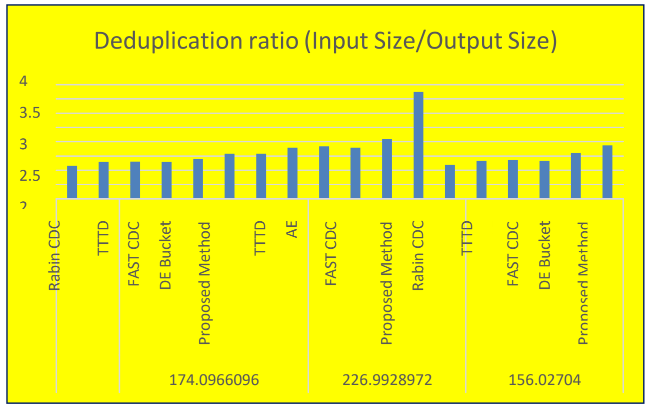

Figure 5: Deduplication ratio in existing and proposed algorithms

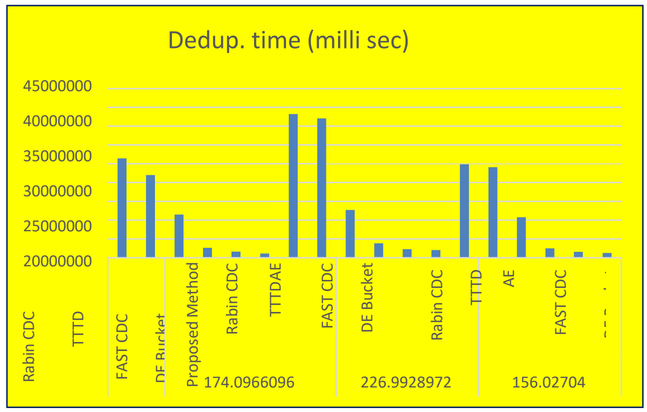

Figure 6: Deduplication time in existing and proposed algorithms

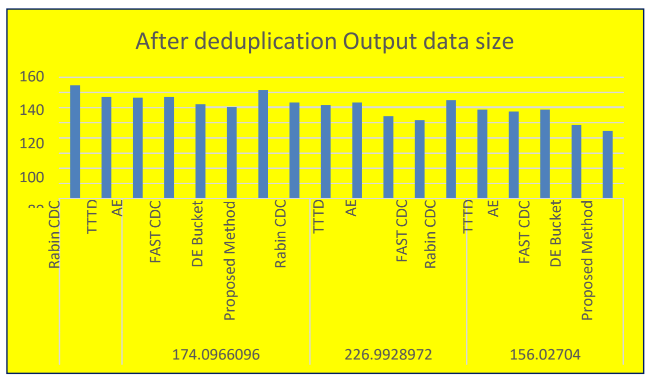

Figure 7: After deduplication output data size in existing and proposed algorithm

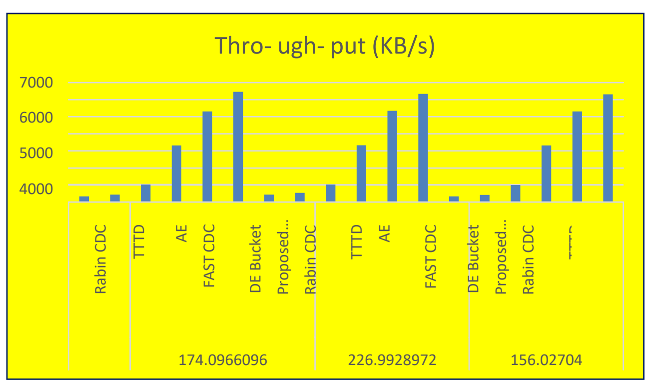

Figure 8: Throughput in existing and proposed algorithms

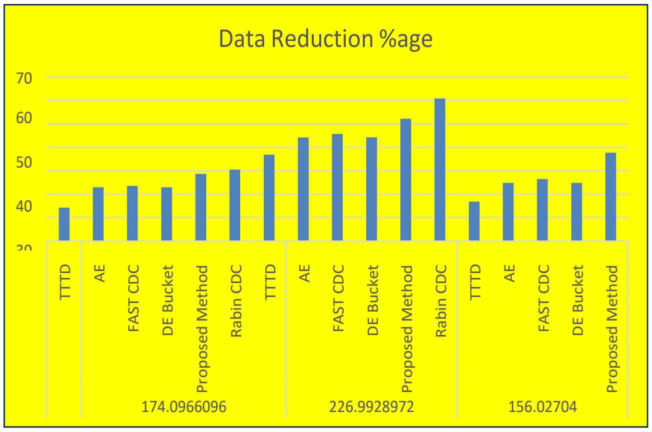

Figure 9: Data reduction percentage existing and proposed algorithms

Figures 4-9 shows the redundant data size, the output data Size in terms of giga bytes after deduplication, the deduplication ratio that is the ratio of input size and output size, deduplication time which is measured in terms of milliseconds, throughput and finally the data reduction percentage. From the given table and figure it is clear that the proposed method outperforms in all terms of metrics than the other existing methods.

Conclusion

The main goal of this research is to reduce duplicate data. This reduces the storage space and provides a better indexing for storing the big data in an optimized way. The original data size after deduplication, the corresponding duplicate data size, deduplication ratio, throughput are clearly explained. This paper addresses the problem of duplicate data reduction at chunk-level deduplication , a fast indexing for the input and output operations in disk read, maximum elimination of duplicate data, higher deduplication ratio, higher chunking throughput and also reduces the computation overheads. The proposed method is proven that it is efficient than that of the other existing methods.

Acknowledgement

We sincerely thank the College Administration, Principal and Head, Department of Computer Science for providing facilities to complete this work.

References

A.SahayaJenitha and Sinthu Janita Prakash**, "**A Stochastic Based Model For Version Control Based System to Handle Duplication Specific Scenario", ASL, Vol.26, Number: 05, pp.408-413, May 2020.

Hu, J.-Y & Zhang, T.-Y & Zhang, C.-M. (2004). "Texture classification using fractional Brownian motion and probabilistic neural network", 26. 389-393.

Jinyan Hu and Taiyi Zhang "Texture analysis and classification through extended fractional Brownian motion", Proc. SPIE 4554, Object Detection, Classification, and Tracking Technologies, (24 September 2001)

Liu, S. C., & Chang, S. (1997). "Dimension estimation of discrete-time fractional Brownian motion with applications to image texture classification", *IEEE Transactions on Image Processing*, *6*(8), 1176-1184.

Qi, D., Yu, L., & Feng, X. (2008, September). "A detection method on wood defects of CT image using multifractal spectrum based on fractal Brownian motion", In *2008 IEEE International Conference on Automation and Logistics* ,pp. 1539-1544, IEEE.

R. Kirubakaran, C. M. Prathibhan and C. Karthika, "A cloud based model for deduplication of large data", *2015 IEEE International Conference on Engineering and Technology (ICETECH)*, Coimbatore, 2015, pp.1-4, DOI: 10.1109/ICETECH.2015.7275007.

Sant'Ana, R., Coelho, R., & Alcaim, A. (2006). "Text-independent speaker recognition based on the Hurst parameter and the multidimensional fractional Brownian motion model", *IEEE Transactions on Audio, Speech, and Language Processing*, *14*(3), pp.931-940.

X. Jin, L. Wei, M. Yu, N. Yu and J. Sun, "Anonymous deduplication of encrypted data with proof of ownership in cloud storage",2013 IEEE/CIC International Conference on Communications in China (ICCC), Xi'an, 2013, pp. 224-229, DOI: 10.1109/ICCChina.2013.6671119.

X. Xu and Q. Tu, "Data Deduplication Mechanism for Cloud Storage Systems," *2015 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Xi'an, 2015, pp. 286-294, DOI: 10.1109/CyberC.2015.71

Xie, W., & Xie, W. (1997). "Image object detection based on fractional Brownian motion". *Journal of Electronics (China)*, *14*(4), 289-294.