Algorithms for Intelligent Systems Series Editors: Jagdish Chand Bansal · Kusum Deep · Atulya K. Nagar

Dinesh Goyal · Valentina Emilia Bălaş · Abhishek Mukherjee · Victor Hugo C. de Albuquerque · Amit Kumar Gupta *Editors*

Information Management and Machine Intelligence Proceedings of ICIMMI 2019



Algorithms for Intelligent Systems

Series Editors

Jagdish Chand Bansal, Department of Mathematics, South Asian University, New Delhi, Delhi, India

Kusum Deep, Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee, Uttarakhand, India

Atulya K. Nagar, School of Mathematics, Computer Science and Engineering, Liverpool Hope University, Liverpool, UK

This book series publishes research on the analysis and development of algorithms for intelligent systems with their applications to various real world problems. It covers research related to autonomous agents, multi-agent systems, behavioral modeling, reinforcement learning, game theory, mechanism design, machine learning, meta-heuristic search, optimization, planning and scheduling, artificial neural networks, evolutionary computation, swarm intelligence and other algorithms for intelligent systems.

The book series includes recent advancements, modification and applications of the artificial neural networks, evolutionary computation, swarm intelligence, artificial immune systems, fuzzy system, autonomous and multi agent systems, machine learning and other intelligent systems related areas. The material will be beneficial for the graduate students, post-graduate students as well as the researchers who want a broader view of advances in algorithms for intelligent systems. The contents will also be useful to the researchers from other fields who have no knowledge of the power of intelligent systems, e.g. the researchers in the field of bioinformatics, biochemists, mechanical and chemical engineers, economists, musicians and medical practitioners.

The series publishes monographs, edited volumes, advanced textbooks and selected proceedings.

More information about this series at http://www.springer.com/series/16171

Dinesh Goyal · Valentina Emilia Bălaş · Abhishek Mukherjee · Victor Hugo C. de Albuquerque · Amit Kumar Gupta Editors

Information Management and Machine Intelligence

Proceedings of ICIMMI 2019



Editors Dinesh Goyal Poornima Institute of Engineering and Technology Jaipur, Rajasthan, India

Abhishek Mukherjee CISCO Technologies Milpitas, CA, USA

Amit Kumar Gupta Poornima Institute of Engineering and Technology Jaipur, Rajasthan, India Valentina Emilia Bălaș Department of Automatics and Applied Informatics Aurel Vlaicu University of Arad Arad, Romania

Victor Hugo C. de Albuquerque Universidade de Fortaleza Fortaleza, Brazil

ISSN 2524-7565 ISSN 2524-7573 (electronic) Algorithms for Intelligent Systems ISBN 978-981-15-4935-9 ISBN 978-981-15-4936-6 (eBook) https://doi.org/10.1007/978-981-15-4936-6

© Springer Nature Singapore Pte Ltd. 2021

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd. The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Contents

A Study of Data Hiding Using Cryptography and Steganography Priya Mathur and Amit Kumar Gupta	1
A Review on Offline Signature Verification Using Deep Convolution Neural Network Deepak Moud, Sandeep Tuli, and Rattan Pal Rana	15
Designing of SAW-Based Resonator Under Variable Mass Loadfor Resonance ShiftYateesh Chander and Manish Singhal	23
Ablation of Hepatic Tumor Tissues with Active Elements and Cylindrical Phased Array Transducer Sarita Zutshi Bhan, S. V. A. V. Prasad, and Dinesh Javalkar	31
A Brief Analysis and Comparison of DCT- and DWT-Based Image Compression Techniques Anuj Kumar Singh, Shashi Bhushan, and Sonakshi Vij	45
Topic Modeling on Twitter Data and Identifying Health-Related Issues	57
Effort Estimation Using Hybridized Machine Learning Techniques for Evaluating Student's Academic Performance	65
Frequency Sweep and Width Optimization of Memos-Based Digital Logic Gates	77
Performance Improvement of Heterogeneous Cluster of Big Data Using Query Optimization and MapReduce Pankaj Dadheech, Dinesh Goyal, Sumit Srivastava, Ankit Kumar, and Manish Bhardwaj	85

Signaling Load Reduction Using Data Analytics in Future Heterogeneous Networks Naveen Kumar Srinivasa Naidu, Sumit Maheshwari, R. K. Srinivasa, C. Bharathi, and A. R. Hemanth Kumar	. 101
Modelling and Simulation of Smart Safety and Alerting Systemfor Coal Mine Using Wireless TechnologyOm Prakash and Amrita Rai	. 111
Green Algorithmic Impact of Computing on Indian Financial Market Krishna Kumar Singh and Sachin Rohatgi	. 119
Coarse-Grained Architecture Pursuance Investigation with Bidirectional NoC Router	. 127
Home Automation and Fault Detection	. 135
Performance Evaluation of Simulated Annealing-Based Task Scheduling Algorithms Abhishek Mishra, Kamal Sheel Mishra, and Pramod Kumar Mishra	. 145
Epileptic Seizure Onset Prediction Using EEG with MachineLearning AlgorithmsShruti Bijawat and Abhishek Dadhich	. 153
A Review of Crop Diseases Identification Using Convolutional Neural Network Pooja Sharma, Ayush Sogani, and Ashu Sharma	. 163
Efficiency of Different SVM Kernels in Predicting Rainfall in India M. Kiran Kumar, J. Divya Udayan, and A. Ghananand	. 169
Smart Trash Barrel: An IoT-Based System for Smart Cities Ruchi Goel, Sahil Aggarwal, A. Sharmila, and Azim Uddin Ansari	. 177
Iterative Parameterized Consensus Approach for Clustering and Visualization of Crime Analysis K. Lavanya, V. Srividya, B. Sneha, and Anmol Dudani	. 183
Assistive Technology for Low or No Vision	. 199
A Socio Responding Implementation Using Big Data Analytics S. GopalaKrishnan, R. Renuga Devi, and A. Prema	. 203

х

Contents

An Optimised Robust Model for Big Data Security on the Cloud Environment: The Numerous User-Level Data Compaction Jay Dave	209
An Adapted Ad Hoc on Demand Routing Protocol for Better Link Stability and Routing Act in MANETs Yatendra Mohan Sharma, Neelam Sharma, and Pramendra Kumar	217
An Efficient Anonymous Authentication with Privacy and Enhanced Access Control for Medical Data in WBAN K. Mohana Bhindu, R. Aarthi, and P. Yogesh	227
A Study on Big Data Analytics and Its Challenges and Tool K. Kalaiselvi	237
"Real-Time Monitoring with Data Acquisition of Energy Meter Using G3-PLC Technology" Deepak Sharma and Megha Sharma	245
A Profound Analysis of Parallel Processing Algorithms for Big Image Applications K. Vigneshwari and K. Kalaiselvi	255
Breast Cancer Detection Using Supervised Machine Learning: A Comparative Analysis Akansha Kamboj, Prashmit Tanay, Akash Sinha, and Prabhat Kumar	263
An Analytical Study on Importance of SLA for VM Migration Algorithm and Start-Ups in Cloud T. Lavanya Suja and B. Booba	271
In-Database Analysis of Road Safety and Prediction of Accident Severity Sejal Chandra, Parmeet Kaur, Himanshi Sharma, Vaishnavi Varshney, and Medhavani Sharma	277
Identifying Expert Users on Question Answering Sites Pradeep Kumar Roy, Ayushi Jain, Zishan Ahmad, and Jyoti Prakash Singh	285
SentEmojis: Sentiment Classification Using Emojis Sangeeta Lal, Niyati Aggrawal, Anshul Jain, Ali Khan, Vatsal Tiwari, and Amnpreet Kaur	293
Protection of Six-Phase Transmission Line Using Bior-6.8 Wavelet Transform	301
Protection of Nine-Phase Transmission Line Using Demeyer Wavelet Transform	315

A Comparative Analysis of Benign and Malicious HTTPs Traffic 32 Abhay Pratap Singh and Mahendra Singh	27
Comparative Study of the Seasonal Variation of SO2 Gas in PollutedAir by Using IOT with the Help of Air Sensor33Vandana Saxena, Anand Prakash Singh, and Kaushal33	37
Fake News Detection: Tools, Techniques, and Methodologies34Deependra Bhushan, Chetan Agrawal, and Himanshu Yadav	47
Synthesis and Analysis of Optimal Order Butterworth Filterfor Denoising ECG Signal on FPGASeema Nayak and Amrita Rai	59
Prognosis Model of Hepatitis B Reactivation Using Decision Tree 37 Syed Atef, Vishal Anand, Shruthi Venkatesh, Tejaswini Katey, and Kusuma Mohanchandra	71
Novel Approach for Gridding of Microarray Images	77
Neighbours on Line (NoL): An Approach to Balance Skewed Datasets 38 Shivani Tyagi, Sangeeta Mittal, and Niyati Aggrawal	87
Competent of Feature Selection Methods to Classify Big Data Using Social Internet of Things (SIoT)	93
Big Data Analytics: A Review and Tools Comparison	9 9
Secured Cloud for Health Care System)7
Minimising Acquisition Maximising Inference—A Demonstration on Print Error Detection 41 Suyash Shandilya 41	13
Data Management Techniques in Hadoop Framework for HandlingSmall Files: A Survey42Vijay Shankar Sharma and N. C. Barwar	25
Maintaining Accuracy and Efficiency in Electronic Health Records Using Deep Learning 43 A. Suresh and R. Udendhran 43	39
An Extensive Study on the Optimization Techniques and Its Impacton Non-linear Quadruple Tank ProcessT. J. Harini Akshaya, V. Suresh, and M. Carmel Sobia	45

xii

Contents	xiii
A Unique Approach of Optimization in the Genetic Algorithm Using Matlab T. D. Srividya and V. Arulmozhi	451
Deep Learning Architectures, Methods, and Frameworks:A ReviewAnjali Bohra and Nemi Chand Barwar	465
Protection of Wind Farm Integrated Double Circuit Transmission Line Using Symlet-2 Wavelet Transform	477
Predicting the Time Left to Earthquake Using DeepLearning ModelsVasu Eranki, Vishal Chudasama, and Kishor Upla	489
Fully Informed Grey Wolf Optimizer AlgorithmPriyanka Meiwal, Harish Sharma, and Nirmala Sharma	497
A Study to Convert Big Data from Dedicated Server to Virtual Server G. R. Srikrishnan, S. Gopalakrishnan, G. M. Sridhar, and A. Prema	513
Fuzzy Logic Controller Based Solar Powered Induction MotorDrives for Water Pumping ApplicationAkshay Singhal and Vikas Kumar Sharma	519
Identity Recognition Using Same Face in Different Context	527
Using Hybrid Segmentation Method to Diagnosis and Predict Brain Malfunction K. Dhinakaran and R. A. Karthika	535
A Blockchain-Based Access Control System for Cloud Storage R. A. Karthika and P. Sriramya	545
Cost-Effective Solution for Visually Impaired	555
When Sociology Meets Next Generation Mobile Networks	567
Image Enhancement Performance of Fuzzy Filter and Wiener Filterfor Statistical DistortionPawan Kumar Patidar and Mukesh Kataria	583
A Review of Face Recognition Using Feature Optimization and Classification Techniques Apurwa Raikwar and Jitendra Agrawal	595

xiv Con	ntents
Rapid Eye Movement Monitoring System Using Artificial IntelligenceTechniquesM. Vergin Raja Sarobin, Sherly Alphonse, Mahima Gupta,and Tushar Joshi	605
Analysis of Process Mining in Audit Trails of Organization Swati Srivastava, Gaurav Srivastava, and Roheet Bhatnagar	611
Modern Approach for the Significance Role of Decision Support System in Solid Waste Management System (SWMS) Narendra Sharma, Ratnesh Litoriya, Harsh Pratap Singh, and Deepika Sharma	619
Integration of Basic Descriptors for Image Retrieval Vaishali Puranik and A. Sharmila	629
Bitcoin Exchange Rate Price Prediction Using Machine LearningTechniques: A ReviewAnirudhi Thanvi, Raghav Sharma, Bhanvi Menghani, Manish Kumar,and Sunil Kumar Jangir	635
A Critical Review on Security Issues in Cloud Computing Priyanka Trikha	643
Smart Traveler—For Visually Impaired People	653
Comparative Study of Stability Based AOMDV and AOMDV Routing Protocol for MANETs Polina Krukovich, Sunil Pathak, and Narendra Singh Yadav	663
IoT-Based Automatic Irrigation System Using Robotic Vehicle Sakshi Gupta, Sharmila, and Hari Mohan Rai	669

Check for updates

Performance Improvement of Heterogeneous Cluster of Big Data Using Query Optimization and MapReduce

Pankaj Dadheech, Dinesh Goyal, Sumit Srivastava, Ankit Kumar, and Manish Bhardwaj

1 Introduction

Hadoop is scalable and capable of managing substantial data volumes with same or homogeneous kind of clusters since the data movement and processing capacities of servers remain same all around the network. In case of heterogeneous clusters, each node comprises of a host with different speeds of storage and processing capabilities. The task of processing tasks then is changed from slow-end nodes into high-end nodes. This strategy works good once the number of information to be processed is less or job load is low, but it fails to improve the rate of job processing on Hadoop heterogeneous clusters if processing and data involves huge volumes of data sets. The improvement of the hardware product is another issue where processors can be improved but it reveals quite expensive. Another solution for this matter is to improve performance of those heterogeneous clusters using different analytical techniques to analyze structured and unstructured information [1]. The MapReduce algorithm

A. Kumar e-mail: iiita.ankit@gmail.com

D. Goyal · M. Bhardwaj Poornima Institute of Engineering & Technology, Jaipur, Rajasthan, India e-mail: dinesh8dg@gmail.com

M. Bhardwaj e-mail: manishbhardwaj.it@gmail.com

S. Srivastava Manipal University Jaipur, Jaipur, Rajasthan, India e-mail: sumit.310879@gmail.com

© Springer Nature Singapore Pte Ltd. 2021 D. Goyal et al. (eds.), *Information Management and Machine Intelligence*, Algorithms for Intelligent Systems, https://doi.org/10.1007/978-981-15-4936-6_9

P. Dadheech (🖂) · A. Kumar

Department of Computer Science & Engineering, Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur, Rajasthan, India e-mail: pankajdadheech777@gmail.com

divided the data classification activities and assigned those jobs to different computers on the system. When every task was computed and finished processing, it was gathered and introduced as the final outcome. The MapReduce algorithm was shown to be very effective as now, large data sets could be readily processed and that also in a reduced time period. The group of Mike Caferella and Doug Cutting with additional members in 2005 employed the MapReduce algorithm and laid the base of Hadoop. It is able to deal with large amount of data sets as in big data and provide insights and meaning of those data statistically. The data generated today can be in any kind of format, be it a simple text message, an audio clipping, video or even a file with information of customers. Broadly speaking data may be user generated, machine generated, structured form where data is organized to offer some significance, or unstructured in which information is in raw format. An analytical survey shows that 33% of information generated digitally can be useful for different situations. But today on 0.5% of information is used to seriously analyze the situation. The limited analytical capabilities of the existing algorithmic approaches a very major chunk of information insights and meanings have been missing. The usage of MapReduce in data warehousing systems implies that analytic queries aren't sufficient for the data outburst experienced now. A quick look on information generation and classification reveals that the rate of ability of data movement on/off hard drive is slow compared to the online data development. Though the capacities of hard drives are also advancing day by day yet, processor speeds or functionality have comparable growth rates as of hard drives. The major bottle neck of this data throughput mismatch occurs due to the motion of information on and off the hard disk drive [2]. Hadoop is also an open-source framework which is used to store in addition to process information using different approaches in a distributed environment. It can manage data flow in a distributed manner across a various bunch of computers in such a way that data access gets fast, reliable, and secure. It entails use of computing and programming models for handling data storage or distribution over large and widespread geographical areas. Going with the conventional databases was quite a good alternative with predefined software written to utilize them. It might be used to analyze information in consumer in addition to business standpoint but it worked well only till the data was confined to some less quantity. With larger volumes of information, processing large number of data became an impossible task. The processing and managing of information within a single or bunch of computers has been also a tedious task as there were risks of node failures, data reduction, and slow processing times. The conventional database servers proved to be a massive risk to store and process data when the amount of data doubled exponentially [3].

2 Literature Review

Another such implementation of good clustering algorithm is conducted by the authors of the paper, [4] where they have used K-means and TF-IDF on MapReduce for clustering of distributed documents. The authors have proposed that as data is accumulating over Internet rapidly, the amount of documents or data to be processed has starting causing overheads for Web servers. The authors have proposed a new type algorithm to cluster the data they are taking in form of documents. They have initiated a preprocessing phase where they calculate the respective TF-IDF weights in the documents for further processing. The experiments conducted reveal that the precision of clustering has improved by incorporating K means and TF-IDF algorithm [5]. It can help in processing thousands of documents processed concurrently at high pace and in stipulated time period. As mentioned in earlier works of distinct authors it is evident that big data requires a lot of mining along big data sets to find the right keyword, but it comes with a cost that lots of organizations have to pay as to finding a difficult pattern in large sets is already a tedious task. This problem is raised as a concern in the paper by Rong [6]. The algorithm implemented here is introduced in MapReduce framework to map the memory as and when required the data items from big data. The authors have accessed and analyzed few of the Web logs spanning over simple databases and reached to the conclusion that traditional database do not offer a good solution in managing and accessing because data cannot reside on a single machine and data movement over long distances is not feasible enough. The cost of computing also increases manifold when the data items in a cluster are in peta or zeta bytes. Many developers and researchers have shifted their focus on developing algorithms in MapReduce framework as it supports a wide variety of options and can contend with real case scenarios which was lacking in earlier implementations [7, 8].

As per the authors [9] deep neural networks (DNN) are a good way to represent the complexity of speech recognition systems. The authors have used singular value decomposition (SVD) in order to maintain and restructure the data model for learning effects of speech recognition in complex systems. The weights of each sub-point are collected and applied SVD upon. The paper gives a general idea on how the complexity of speech recognition systems could be managed by fine tuning, modeling the sparseness and training the recovery systems in manners so as they could be recognized and worked upon faster calculations.

The expenses of big data are relevantly written by [10] in his paper. He explains the scenario of testing tools, hardware, and software counterparts of the Hadoop and the big data that can help organizations to discover new ways to use the day and provide good monetization. He also advocates in finding the performance bottlenecks and what could be the solution to the idealistic features to improve them for further proceedings.

Another such analysis conducted by the author [11] in his paper suggests that clustering can really help large organizations process data that are not labeled and help them manifest all the resources effectively. They have used the method of filtering,

uploading and then applying *K*-means to find effective results. The document clustering can be improved by using methods that distinguishably divided/filters or prunes documents and then held them in the Hadoop account. The subsequent steps of managing them into a cluster can help the document to be utilized at its fullest and reduce the data management jobs. The author has incorporated Davis–Bouldin index that measures the quality of cluster that is generated post Hadoop implementation step.

An application is designed around the architecture that once a document is fed into a system, it is cleansed from less important words by using Porter Stemmer, then for calculating the weighted frequency, TF-IDF is applied [12, 13]. The clustering algorithm used here, *K*-means, require inputs in numerical forms; therefore, the weight of important words are given as an input to the *K*-means algorithm after the document intermediary is uploaded on Hadoop ecosystem. Then, the subsequent steps are utilized to check the quality of clusters where closeness in clusters depicts similarity in documents and distance in clusters depicts the dissimilarity of clusters [14, 15]. One big issue that arises here is that the larger the data set the more effective the implementation begins. It is applicable for MapReduce frameworks which work well in highly complex environments.

The cluster quality is measured in Davis–Bouldin metrics which provides a statistical analysis of the document clustering done so far. The more the value of a cluster is nearer to 1, the bad the quality of cluster is. In this way, it is easier to manage cluster quality by large organizations and the impact of clustering in distributed documentation [16, 17]. The preprocessing steps help in determining the needful from whole lot of documents. The document clustering is one of the main headaches for companies that involve lot of go through in data sets. While big data continues to thrive on Internet, it is becoming more complex and expensive to maintain sanity among data processing. The need of such preprocessing clusters is highly requited so that it can be solve and managed efficiently without introducing the costs to storage, computation, and other means. The intent of clustering document has attracted many eminent researchers. One such research group have studied and designed algorithm for text clustering [18].

3 Proposed Work

3.1 Methodology

In this work, we enhance the performance of heterogeneous clusters on Hadoop by minding a set of policies that increase the input/output signal queries, enhances the routing of the algorithm into heterogeneous clusters, so improving the operation of processing of inquiries in such a fashion that they are readily attached with the ideal portion of implementation at minimal period because we could without raising the values at calculating degree [19]. This work includes that scenario in which

we processed the complex information through clustering. We handle the clustering processing through query improvisation.

3.1.1 Solving Problems Through Iteration and Phasing

The information employed here for doing experiments have been all log files of sites that will be in the form of unstructured information. It is the very good method through which we could establish the efficacy of processes and it is also closely associated with the real-world environment where information is recorded in such condition. The information thus additionally assists in imagining the situation of true use of procedures which are closely associated with the real-world situation where one needs to take care of data that may arrive in any shape and shape [16, 20]. The information since log files have assisted in obtaining the question entry procedure for an e-commerce site that must take care of countless search key words, necessitates clustering of information items to purchase and supplying precise results in rather short periods of life. Within this paper breaking up the question processing system for MapReduce in successive pragmatic stages such as filtering, clustering, and enhancing the question for quicker execution procedures. Initially, the stage involves iterative actions to clean out the query and discover dependencies one of the question so that after a query is implemented, task tracker does not need to count on calculating query dependencies again after it had been calculated at the first actions. To lessen the query execution time, enhance the efficacy of clusters and such as the pragmatic processing in which the questions are enhanced through clustering them to comparable measures, employing TF-IDF, calculating weights, and handling the similarity matrix and ultimately submitting the question for implementation [8].

The procedure used here for calculating minimum rate and supplying output in comparatively smaller timeframe is represented Fig. 1.:

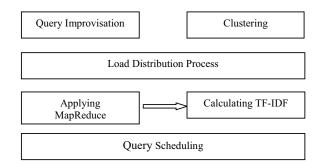


Fig. 1 Phasing methodology for heterogeneous clusters

3.1.2 Query Improvisation

Query improvisations are actually just a procedure through which people strengthen the performance of both clustering [6, 21]. Clustering can be employed when queries are brought into the parser and semantic analyzer they always compute the dependencies among these queries. But the moment this parsed question has been sent into the Hadoop's MapReduce to do the exact dependencies which have been calculated for your Hive query is dropped between your alterations. Once the dependencies are calculated they are used for semantics extractions in Hive QL chip. In the second phase, we are able to use these dependencies like logical predicates and input tables to method that the dependencies among distinct questions to become closely attached with each other during transition [9, 22]. When the semantic gap between both Hive and also Hadoop is bridged from these intermediary measures, they are sometimes readily utilized for clustering equivalent inquiries at query level, and thus by minute steps improving the query job performance (Fig. 2).

Steps of Query Improvisation Process which use the following property:

1. **Selection of Query**: A select with conjunctive conditions on the attribute list is equivalent to a cascade of selects upon selects:

$$QA1^{\wedge}A2^{\wedge}\dots An(R) \equiv QA1(QA2(\dots(QAn(R))\dots))$$

2. Calculating the Commutative property of Query: We use select operation by commutative

$$A1(QA2(R)) \equiv QA2(QA1(R))$$

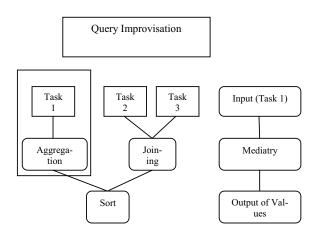


Fig. 2 Query improvisation process

90

Performance Improvement of Heterogeneous Cluster of Big Data ...

3. **Calculating the value of cascade of p**: The value of cascade is the of project operations is equivalent to the last project operation of the calculated value:

 $pAList1 (pAList2(...(pAListn(R))...)) \equiv pAList1(R)$

4. Calculating the Commuting Query with the value of p: Given a P's and Q's attribute list of A1, A2...An. The P and Q selection operations can be commuted:

Commutative of Q: A1, A2...An(Qc(R)) $\equiv Qc(P, A1, A2...An(R))$

5. **Calculate the value of** *X*: We merge the join and Cartesian product operations which are commutative:

 $RXQ \equiv QXR$ and $RXS \equiv QXR$

- 6. **Commuting Query with or** *X*: Select can be commuted with join (or Cartesian product) as follows:
 - a. If all of the attributes in the select's condition are in relation *R*, then Query, $c (R \text{ Query}) \equiv (\text{Query } c (R)) \text{ Query}$
 - b. Given select the condition c composed of conditions c1 and c2, and c1 contains only attributes from *R*, and c2 contains only attributes from Query, then Query $c(R \text{ Query}) \equiv (\text{Query } c1 (R)) (\text{Query } c2 (\text{Query}))$
- 7. Commutative of set operations $(U, \cap, -)$: Union and intersection operations are commutative; but the difference operation is not:

$$RUQ \text{ Selection} \equiv QUR, R \cap Q$$
$$\equiv S \cap Q, R - Q \neq Q - R$$

8. Associativity of *X*, *U* and \cap : All four of these operations are individually associative. Let θ be any one of these operators, then:

 $(R\theta Q)\theta T \equiv R\theta (Q\theta T)$

9. Commuting s with set operations $(U, \cap, -)$: Let θ be any one of the three set operators, then:

$$Qc(R\theta Q) \equiv (Qc(R))\theta(Qc(S))$$

10. Commuting P with U: Project and union operations can be commuted:

 $PAList(RUQ) \equiv (PAList(R))U(PAList(Q))$

3.1.3 Applying Clustering in Initial Phases

A lot of applications and systems use clustering as a part of dividing similar and dissimilar data items together. Clustering of data sets here can help achieve focusing on the items that are useful and discarding items which are generally not useful [23]. Looking from the view of a log file and applying clustering on it, we get clusters of user prone data that categorize the behavior of the item buying selections. The interactions of a user with an e-commerce Web site can identify good buying patterns easily. When this single end user data is collected from the multi-million users of the Web site and analyzed, the results tell a whole lot of insight on the consumer behavior, selection, buying options, and fields which need improvisation.

The log files used here symbolizes the unstructured data category and recording the data from different fronts, a presentation of heterogeneous Hadoop clusters [24, 25]. In this iterative clustering model when the data is captured from user end, it is optimized to remove static and dynamic parts of it in the clustering process itself where static parts are the decisions where dynamic parts are variants of time stamping, invoice numbers, delimiters, etc. It could be done through iteration process but applying a standard clustering algorithm with changes improves the efficiency as well.

3.1.4 Proposed Approach Algorithm

Step 1: We will read the log transition.

Step 2: We store the transition value in Q_i , Where $i = 1, 2, 3 \dots n$. Where n is equal to the log transition value.

Step 3: We store the query which is request from client and store in array by get Query () method.

IQ = I get query. Where *I* is number of query request. We store the query in array to create cluster. By method table put (null, array, parameter1, parameter 2 *n*); To convert the array in object they are $Q_i = \text{Table}$ —get query ().

Step 4: Merge the pair of most similar queries (q_i, q_j) that does not convert the same queries. If $(Q_i$ is not in C_i) then store the frequency of the item and the increase the value.

Step 5: Compute the simulating Matrix if Q_i in C_i .

Step 6: If Q_i is not C_i then compute new cluster $IQ = New (C_i)$.

Step 7: Go to Step 3.

Step 8: Step go to Step 2.

Performance Improvement of Heterogeneous Cluster of Big Data ...

Nodes	Specifications	Specifications			
	RAM (in GB)	Processors	Memory (in GB)		
NameNode	2	1	30		
DataNode 1	1	1	30		
DataNode 2	1	1	30		
JobTracker	2	1	30		

Table 1	System	specification
---------	--------	---------------

4 Experimental Setup

4.1 System Specifications

The system deployed here to test the proposed work contains data nodes, name nodes, job trackers, and task trackers on virtual machines [26, 27]. The machines have different capacity and hardware configurations. The nodes are created in such a manner that each node has its own specialty and try to resemble to real-world environment (Table 1).

4.2 Load Distribution to High- and Slow-End Clusters

Once the data is sorted initially, it can be forwarded to the clusters for processing one by one. Here, comes the tricky part because ideally Hadoop does not contain same type of hardware architecture underneath [28]. There can be nodes that operate as high-end clusters with good processing speeds but there also can be slow-end clusters with lower speeds than their high-end counterparts. When a job or couple of jobs is fed into Hadoop ecosystem, they are sent to both types of processors depending on the availability. When a high-end cluster finishes job processing, the data waiting to be processed is sent to the high-end systems. This data movement introduces cost to the processing, data storage, and security as well.

4.3 Applying MapReduce

As demonstrated in previous sections, when initially the system requirements, job processing on clusters, and queries to implement are decided, it becomes whole lot easier to deal with the unstructured data formats [29, 30]. The cleansed, clustered, and diversified data items can now be sent to MapReduce to extract the key value pairs for further processing. In MapReduce, the applications can be used to classify unstructured documents into a set of meaningful terms which can provide value to

the clusters. The MapReduce algorithm can help in retrieving important information from a pool of data which cannot be achieved manually.

Implementation of MapReduce:

Map:

Step 1:	Prepare the Map() input
Step 2:	Input unstructured data file: (log_file.txt) Run the Map() code–Map() is run exactly once for
	each <i>K</i> 1 key value, generating output organized by key

values K2.

Function that divides data sets:

(log_file_intermediate)

Splitthefile(log_filetranstation.txt):(lft) + (lft2) + (lft3)....+

(lftn)

Step 3: Collect Output: (log_file_output; lfIdt, 1)

Reduce:

C is the Count of the individual terms.

Step 1: Input file which was output from map function:

(log_file_output; lfIdt, [c])

Step 2: Run the Reduce() code Function to summarize the

intermediate terms and give a final value to all the

detected valid terms in the module: S = Sum(c)

Step 3: Provide output to sum up the file values.

(log_part, countId, S)

5 Results and Discussion

5.1 Results

The subsequent results have been established by directing the proposed methodology in the system. The heterogeneous Hadoop clusters were used to test the working of big data sets which emulated the *IO* throughput, average rate of execution, standard deviations and finally the test execution time.

94

Performance Improvement of Heterogeneous Cluster of Big Data ...

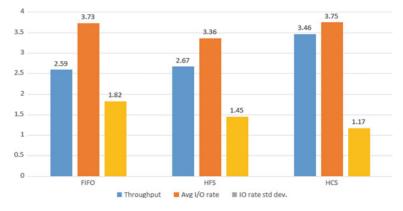


Fig. 3 Comparison of different Hadoop schedulers

5.1.1 Comparison of Productivity Among Hadoop Schedulers

Comparison among the most efficient schedulers used in Hadoop, i.e., FIFO, HFS, and HCS. The following graphs establish the point that schedulers can overtake other schedulers which are used in HDFS and internal mechanisms to share a file or job over the several default systems to execute and store them reliably (Fig. 3).

The figure shows a trade-off among various schedules which may lead to optimized performance of systems in HDFS for clustering big data. The response time caused by the data loads, which were actually the logs of an e-commerce Web site and the response time took by the schedulers in mapping and reducing them to store them on the DFS, manually extract as well as manage them. It also establishes the fact that surmounting amount of data as is general in big data sets; Hadoop can optimize the processing instead of data loading and unloading frequently to get better insights about data. This also proves the fact that once data is accepted by a node, the average time to consume and process it may vary depending upon the type of scheduler being used.

5.1.2 Comparative Study of FIFO/HCS/HFS

See Table 2; Figs. 4, 5 and 6.

5.2 Performance Evaluation

The clusters that were created for this work are working and derived results as formulated. The average I/O rate of data movement, standard deviations, and time

Scheduling algorithm	Data size	Previous methodology	Proposed methodology	
		Execution time	Execution time	
FIFO	500 MB	4020	3958	
	1 GB	7700	7156.356	
	2 GB	9900	9165	
	5 GB	14,500	11,130	
	10 GB	Not consider	12,780	
HFS	500 MB	3950	3925	
	1 GB	7200	7065	
	2 GB	9200	9085	
	5 GB	13,500	10,980	
	10 GB		12,400	
HCS	500 MB	3900	3875	
	1 GB	7000	6998	
	2 GB	9000	8912	
	5 GB	12,000	10,260	
	10 GB		12,610	

 Table 2
 Efficient job scheduling for MapReduce clusters [16]

PREVIOUS V/S PROPOSED METHODOLOGY

FIFO_PREVIOUS FIFO_PROPOSED HFS_PREVIOUS FFS_PROPOSED FCS_PREVIOUS FCS_PROPOSED



Fig. 4 Previous versus proposed methodology

Performance Improvement of Heterogeneous Cluster of Big Data ...

🗛 Applications Races System 👹 🌸 🗹		🗉 rastilizalhest-Desissa 🗖 Aplicatos Rass System 👹 🕸 💆			
1	rootj/localhest;issrishare;hadoop	Fie Edt ylew Search Jeminal Heb		1	rwtjikalheit-D
Fie fait yiew Search Jernical Belp		15/98/25 07:34:15 INFO mapred.3xbClient:	Data-lacal map tasks-20	File Est yes Search Jerninal Help	
(98/23 (8:55:12 DV9 mapret.)obCloent:	Deta-local map tasks-20	16/88/25 67:34:15 INFO mared.3xbClient:	SLIPTS KOLLIS REDUCES-478047	16/88/28 88:47:25 DVD supred_lobClient:	Data-local mp tasko-15
(86/23 89:55:32 399 mapred.3xb(loent:	9,075 K0.125 KERUES-45758	16/88/25 07:34:15 IMPD macred_JobClient:	File Input Format Counters	16/88/28 98:47:25 DVD supred_lob/light:	SLOTS HELLES REDUCES=2009462
(8/23 8:55:12 DPD mored.3ddClient:	File Diput Fornat Counters	16/88/25 07:34:15 IMF0 macred.lobClient:	Bytes Read-1128	16/88/28 80:47:25 DEG sapred_lobClient:	File Input Format Counters
(8/23 8:55:32 DFD mored.3dxClient:	Bytes Redrill28	16/88/25 87:34-35 39F0 macred .bbClient:	File dutaut Format Counters	16/08/20 90:47:25 DFG moved_lobClient:	Butes Read=1228
(48/23 49:55:32 399 supred.lobCloent:	File Output Format Counters	16/08/25 07:34:15 DVFD mored_3xbClient:	Bites Witter-75	16/01/20 00:47:25 DWD supred JobClient:	File Output Format Counters
(46/23 89:55:12 30F0 maprel.3cbClient:	Bytes Written-74	16/88/25 07:34:15 DVFD marred_bbClient:	FileSystemCounters	16/05/20 00:47:25 DFD supred.lcbClient:	Butes Written-74
(46/23 49:55:32 399 mpref.3dxTient:	FileSystemCounters	16/86/25 07:34:15 30/9 nared.3dxClent:	FILE BITES READ-820	16/08/20 00:47/25 DMO mapred ltdk[limt;	FileSystemCounters
(46/23 49:55:32 399 mpref.3xbCloent:	FILE BITES READ-RED	16/8/25 07:34:25 30F0 mored.3ddClient:	KIPS BYTES READ-2418	16/85/28 ME:47:25 DV0 supred_lob/lient:	FILE BITES READ-EDA
6/88/23 (0:55:32 DVP0 mapred.lobCloent:	KOFS BITES READ-DAUB			16/05/20 00:47:25 DVD supred_lob(lient:	HOPS BITTES READ-3418
108/23 08:55:32 20F0 mapred.3dbClient:	FELE BYTES WROTTEN-GAUNDA	16/88/25 07:34:15 10F0 mapred.3xbClient:	FILE BITES WUTTEH-604425	16/98/28 08:47:25 DF0 suprof.JobClint:	FILE BYTES WITTER-60-501
(48/23 89:55:32 DR9 mpred.3xXCluent:	KOPS BITES WILTTEN-LEADSON	16/88/25 07:34:15 INFO mapred.JobClient:	KOFS_BITES_WOTTEN-01405035	16/98/28 98:47:25 DF0 spred_lobClient:	KOPS BYTES WETTER-OHABSEH
6/81/23 89:55:32 39/9 mpred.3xbClient:	Np-Retuce Fratework	16/88/25 47:34:35 DMP0 mapred.3xbClient:	Rip-Reduce Fratework	16/81/28 ME47:25 DVD supred_lobClient:	Rep-Reduce Framework
(81/23 89:55:32 20F0 supred.3ubClient:	Nup output materialized bytes-473	16/08/25 47:14:15 INFD mapred.3xbClient:	Rap output materialized bytes=#14	16/85/21 ML47.25 2070 seprel.JobClient:	New output meterialized bytes-565
100/23 09:55:32 D90 mapred.3dxClient:	Re last records/08	16/08/25 07:34:15 INFO magned.3xbClient:	Nap input records=18	16/68/29 88:47:25 DF0 supred.3cbClient:	Rip (rout records=18
(46/23 89:55:32 TMP9 supref.)xbXTient:	Reduce shuffle bytes=813	16/86/25 07:34:15 INFO mapred.3xbClient:	Reduce shuffle bytes=874	16/81/21 ME47:25 DVD supred.3cbClient:	Refuce shuffle bytes+668
(81/23 89:55:32 39/0 sepret.3dXlient:	Spilled Records+188	15/95/25 07:34:15 39F0 migred.3dbCljent:	Sollied Records=100	16/06/20 00:47:25 DWD supred_lob(lient:	Saciled Records+100
198/23 81:55:32 DHD maxed.lobClient:	No output bytes=713	16/88/25 07:34:15 10F0 mored.3xbClient:	Rip output bytes=714	16/98/28 30:47:25 DMO supred JobClient:	Rip pitput bytes=788
6/81/23 (8:55:32 DPD mored.3xXTient:	Tes1=001663008	15/98/25 07:34:15 DMFD mapred.3cbClient:	Total committed heap usage (bytes)=1639683988	16/98/28 00:47:25 DV9 mapred.JobClient:	Total comitted have occase (bytec)+15156110
6/98/23 09:55:32 39F0 mipred.3dbCloent:	CPU time spent (ms)+12780	16/88/25 07:34:15 INFO mapred JobClient:	(P) the spect (m)=1240	16/88/28 NE:47:25 30F0 mapred.JobClient:	(PU time spent ins)=(2638
6/88/23 89:55:32 20F0 mapred.3xbClient:	the local data with the lo	16/06/25 07:34:35 DVFD mapred_bbbClient:	The state with the same	16/86/28 30:47:25 DWD mapred JobClient:	Mit your stational
(46/23 89:55:32 DP0 mared.3ddCisent:	SPLIT NW BITES-1291	16/98/25 47:34:25 20F0 marred.300Client:	GPLIT RAN BYTEG-1200	16/98/29 98:47:25 DFD supred JobClient:	SPLIT RAN BITES-1298
6/88/23 89:55:32 39F0 mpred.3xbClient:	Contine input records+0	16/98/25 07:34:15 INFO macred.JobClient:	Conduine input records-0	16/82/28 98:47:25 DR0 supret_lobClient:	Costine input records+8
6/81/23 49:55:32 309 maprel.luXlient:	Reduce input records=58	16/80/25 07:34:25 30F0 mored.3dbClient:	Reduce input records/58	16/88/28 ME47:25 DWD supred.lcbClient:	Reduce input records/58
448/23 00-55-32 DMF8 mapred.3dbClient:	Reduce input groups=5			16/98/28 98:47:25 DPD supred.JobClient:	Refuce input groups-5
P86/23 89:55:32 30F0 mipred.3xbCluent:	Conduine output records=0	16/98/25 07:34:15 INFO mapred.3xbClient:	Reduce input groups=5	16/08/20 00:47:25 DMO mapred.lcbClient:	Costine output recontine
(46/23 49:55:32 30F0 supred.lob(lisent:	Physical nenory (bytes) snapshot-UEN29468	16/88/25 07:34:15 INFO mapred.3xbClient:	Combine output records+0	16/06/20 30:47:25 DWD sapred lobClient:	Physical memory (butes) statishet-04250252
(01/23 00:55:32 20F0 supred.3ddClient:	Reduce putput records-5	16/88/25 07:34:15 JNFD mapred.3xbClient:	Physical newry (bytes) snapshot=0858847232	16/05/20 10:47:25 DWD supred.JobClient:	Reduce output records-5
(8/23 89:35:32 309 mpref.3dXlbert:	Kirtual setury (bytes) snapshot=8682967936	16/88/25 87:34:35 10F0 mapred.3cbClient:	Reduce output records=5	16/85/21 00:47:25 DVD sapred_lob/liert:	Virtual memory (bytes) snapshot+00110080344
(81/23 89:55:32 39/9 mpret.3ddCient:	Rep patput records-68	16/88/25 47:34:15 IMF0 mapred.lobClient:	Virtual memory (bytes) snapshot-8813785216	16/06/28 00:47:25 DWO supred_lobClient:	Rep subput records+58
P86/23 89.55.32 39/9 (s.Text0/538,	TextSF310 : write	16/98/25 07.34:15 2070 mapred.3xbClient:	Rep output records-50	16/98/28 Mt:47:25 DW9 fs.Text0P530:	
(8)/23 89:55:12 2049 fs.Tex104520;	Date & time: Tue Aug 23 09:55:32 P0T 2006	16/88/25 67:34:15 INF0 fs.Test0F520:	- Test0F510 : vrite	16/88/01 86:47:25 DRI fs.Text0F500:	Date & time: Son Aug 28 88:47:25 PDT 283
PRE/23 89:55:32 DRI fs.Tect0F301:	Number of files: 18	16/8U/25 47:34:15 IMF0 fs.Test0FSIR:	Date & time: Thu Aut 25 87:34:15 PDT 2016	16/8/28 90:47:25 DRI 15.76/0F538:	Number of files: 10
(48/23 69:55:32 3940 fs.Text0F530: Total	Mbytes processed: 18	16/88/25 67:34:15 39F0 fs.Test0F520:	Rober of files: 18	16/80/20 ME-47:25 DVP0 fs.Test0F530: Tota	(Waytes processed: 18
36/06/23 09:55:32 DRI Fs.Test0FSD: Throughput.ub/sec: 2.5000039000259		16/88/05 87:34:15 DIFD fs. Test0F500: Total Mextes processed: 18		16/88/08 88:47:25 099 fs.Text0F530: Throughout millisec: 1.4636836781363433	
06/08/23 48:55:32 3049 fs.Tect04530: Avenue 30 rate ob/sec: 3.7300303550492		16/16/25 (0:14:15 30/26 fs.Test0/550: Throughout ab/sec: 3.89635617582846		14 con cits and cit. No. Both die Territorithe Averages Tol state of lines. 3 Westmanning and the	
		16/8/25 47:34:25 30F0 fs. fest0F50: Average 10 rate sb/sec: 4.3442344804245		16/00/20 MI-47-25 DNR fs.Test0F510: 10 rate std deviation: 1/37029390201005 VV I	
				MARCH MARCH DATA IN Faller Faller Test exer time sec 图 极心 Settings t	
		16/66/25 67:34:35 1040 fs.Test04520: 10 rate std devlation: 0.996996260838672 16/66/25 67:34:35 1040 fs.Test04520: Test exec time sec: 586.521		MALCH NEW 23 THE FLIMPSTE	
rest(localitest hadoos)#		16/86/25 07:34:15 04/9 15.76509500; 16/86/25 07:34:15 04/9 fs.76509500;	Test exec time sec: 508.521	[root8localhest Desktop1#	

Fig. 5 Performance analysis of FIFO/HCS/HFS with query improvisation

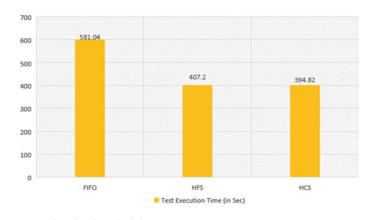


Fig. 6 Test execution time by schedulers

taken to execute the loaded files prove that the formulated method is working as expected.

5.3 Schedulers at Work

5.3.1 First in First Out (FIFO)

The scheduler used here is FIFO which is used to schedule the resources of big data sets in the heterogeneous clusters and concurrently running TestDFSIO as benchmarking.

5.3.2 Hadoop Fair Scheduler

The test results are the resultant of the working of fair scheduler algorithm on the big data sets for heterogeneous clusters in Hadoop environment.

6 Conclusion

The conducted experimentation also establishes the fact that if a practice is described to handle different usage case scenarios, one could complete lessen the amount being spent on computing and may gain on counting upon dispersed programs for rapidly executions.

The major points of thought that were resulted as conduction of this work are as following:

Calculating question dependencies: This really may be lowered by keeping an index or question addiction desk which cannot only cut back time but affect that the total calculating in an excellent phrase. The processor could be utilized to method an increasing number of data within the dispersed nodes of Hadoop. The heterogeneous clusters used in the Hadoop can reap by simply in carrying the relevant and shedding the insignificant jobs to method.

Job scheduling mechanics: The default scheduler of Hadoop and also MapReduce could be your FIFO scheduler (First in First Out) which seems important as so when Hadoop project arrives. However, at a dynamic setting of big data if the payable size of data is to be processed, using in everything that comes could become unfruitful and squander of reference, because while in the end to make money from these kinds of big chunks of data, then it must be sorted at the past. Therefore, scheduling the tasks that process this info therefore the clusters using high-performance can manage big numbers sets and clusters with slow-ends can assist in additional processing systems. The study experiments and work conducted under this work have emulated quite surprising results, many of them function as selection of schedulers to program tasks, placement of data in the similarity matrix, clustering before scheduling inquiries and moreover, iterative, mapping along with reducing and binding the inner dependencies collectively to steer clear of query stalling and execution times.

98

7 Limitation

The big data set has many probing worries for computing and data storage. Since the major part of big data analytics is related to dynamic nature of data, data is deleted, manipulated, and retrieved frequently. This ad hoc-based processing of data includes streaming data in and out of the storage systems as the requirement but this also introduces a large chunk of processing.

References

- 1. Liu, Z. (2015). Efficient storage design and query scheduling for improving big data retrieval and analytics, Dissertation, Auburn University, Alabama.
- Zongben, X., & Shi, Y. (2015). Exploring big data analysis: Fundamental scientific problems. Springer Annals of Data Science, 2(4), 363–372.
- Tinetti, F. G., Real, I., Jaramillo, R., & Barry, D. (2015). Hadoop scalability and performance testing in heterogeneous clusters. In *The proceedings of the 2015 international conference* on parallel and distributed processing techniques and applications (PDPTA-2015), Part of WORLDCOMP'15 (pp. 441–446).
- 4. Wan, J., Yu, W., & Xu, X. (2009). Design and implement of distributed document clustering based on MapReduce, ISBN 978-952-5726-07-7, 2009.
- 5. Kamtekar, K., & Jain, R. (2015). *Performance modeling of big data* (pp. 1–9). Washington University in St. Louis.
- Das, T. K., & Mohan Kumar, P. (2013). Big data analytics: A framework for unstructured data analysis. *International Journal of Engineering and Technology (IJET)*, 5(1), 153–156. ISSN: 0975-4024.
- Liu, F. H., Liou, Y. R., Lo, H. F., Chang, K. C., & Lee, W. T. (2014). The comprehensive performance rating for hadoop clusters on cloud computing platform. *International Journal of Information and Electronics Engineering*, 4(6), 480–484.
- Rong, Z., & De Knijf, J. (2013). Direct out-of-memory distributed parallel frequent pattern mining, ACM, BigMine'13. In *Proceedings of the 2nd international workshop on big data, streams and heterogeneous source mining: Algorithms, systems, programming models and applications* (pp. 55–62). ISBN: 978-1-4503-2324-6, https://doi.org/10.1145/2501221. 2501229.
- Li, B., & Guoyong, Y. (2012). Improvement of TF-IDF algorithm based on Hadoop framework. In *The 2nd international conference on computer application and system modeling* (pp. 0391– 0393), Paris, France: Atlantis Press.
- 10. Kamtekar, K. (2015). Performance modeling of big data, May 2015.
- Jagtap, A. (2015). Categorization of the documents using K-Means and MapReduce. International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753, 2015.
- Das, T. K., & Kumar, P. M. (2013). BIG data analytics: A framework for unstructured data analysis. *International Journal of Engineering and Technology (IJET)*, 5(1), 153–156. ISSN: 0975-4024.
- Novacescu, F. (2013). Big data in high performance scientific computing. In *International Journal of Analele Universității "Eftimie Murgu* (vol. 1, pp. 207–216). "Eftimie Murgu" University of Resita, ANUL XX, NR.
- Rao, B. T., Sridevi, N. V., Reddy, V. K., & Reddy, L. S. S. (2011). Performance issues of heterogeneous Hadoop clusters in cloud computing. *Global Journal of Computer Science and Technology, XI*(VIII).

- Xie, J., Yin, S., Ruan, X., Ding, Z., Tian, Y., Majors, J., Manzanares, A., & Qin, X. (2010). Improving MapReduce performance through data placement in heterogeneous Hadoop clusters. In *Proceedings of the 19th international heterogeneity in computing workshop* (pp. 1–9), Atlanta, Georgia.
- 16. Liu, J., et al. (2015). An efficient job scheduling for MapReduce clusters. *International Journal of Future Generation Communication and Networking*, 8(2), 391–398.
- Fahad, A., Alshatri, N., Tari, Z., Alamri, A., Khalil, I., Zomaya, A. Y., Foufou, S., & Bouras, A. (2014). A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE Transactions on Emerging Topics in Computing*, 2(3), 267–279. Digital Object Identifier https://doi.org/10.1109/tetc.2014.2330519.
- 18. MonaP. EMC Corporation (2014). Virtualizing Hadoop in large-scale infrastructures.
- Aggarwal, C., & Han, J. (2014). An introduction to frequent pattern mining. In *Frequent pattern mining*, Springer. ISBN 978-3-319-07820-5 (Print) 978-3-319-07821-2 (Online), https://doi.org/10.1007/978-3-319-07821-2.
- Victor, G. S., Antonia, P., & Spyros, S. (2014). CSMR: A scalable algorithm for text clustering with cosine similarity and MapReduce. In *IFIP international conference on artificial intelligence applications and innovations*, *AIAI 2014: Artificial intelligence applications and innovations*, *AIAI 2014: Artificial intelligence applications and innovations* (pp. 211–220), AICT 437.
- Novacescu, F. (2013). Big data in high performance scientific computing. EFTIMIE MURGU RESITA, ANUL XX, NR. 1, (pp 207–216). ISSN 1453–7397.
- 22. Xue, J., Li, J., & Gong, Y. (2013). *Restructuring of deep neural network acoustic models with singular value decomposition* (pp. 2365–2369), ISCA, INTERSPEECH.
- Akdere, M., Cetintemel, U., Riondato, M., Upfal, E., & Zdonik, S. B. (2012). Learning based query performance modeling and prediction. In *IEEE 28th international conference on data engineering* (pp. 390–401).
- Thirumala Rao, B., Sridevi, N. V., Krishna Reddy, V., & Reddy, L. S. S. (2011). Performance issues of heterogeneous Hadoop clusters in cloud computing. *Global Journal of Computer Science and Technology*, XI(VIII).
- Kumar, A., Goyal, D., Dadheech, P. (2018). A novel framework for performance optimization of routing protocol in VANET network. *Journal of Advanced Research in Dynamical & Control Systems*, 10(02), 2110–2121. ISSN: 1943-023X.
- Dadheech, P., Goyal, D., Srivastava, S., & Kumar, A. (2018). A scalable data processing using Hadoop & MapReduce for big data. *Journal of Advanced Research in Dynamical & Control Systems*, 10, (02), 2099–2109. ISSN: 1943-023X.
- Dadheech, P., Goyal, D., Srivastava, S., & Choudhary, C. M. (2018). An efficient approach for big data processing using spatial boolean queries. *Journal of Statistics and Management Systems (JSMS)*, 21(4), 583–591.
- Dadheech, P., Kumar, A., Choudhary, C., Beniwal, M. K., Dogiwal, S. R., & Agarwal, B. (2019). An enhanced 4-way technique using cookies for robust authentication process in wireless network. *Journal of Statistics and Management Systems*, 22(4), 773–782. https://doi.org/10. 1080/09720510.2019.1609557.
- Kumar, A., Dadheech, P., Singh, V., Raja, L., & Poonia, R. C. (2019). An enhanced quantum key distribution protocol for security authentication. *Journal of Discrete Mathematical Sciences* and Cryptography, 22(4), 499–507. https://doi.org/10.1080/09720529.2019.1637154.
- Kumar, A., Dadheech, P., Singh, V., Poonia, R. C., & Raja, L. (2019). An improved quantum key distribution protocol for verification. *Journal of Discrete Mathematical Sciences and Cryptography*, 22(4), 491–498. https://doi.org/10.1080/09720529.2019.1637153.

100