Enhancing hadoop performance in homogeneous big data environment assuming configuration of dynamic slots in map-reduce pattern

Article · January 2018

DOI: 10.14419/ijet.v7i4.26962

CITATION
1

READS
204

5 authors, including:

Abolgasem MOHAMED Enfais
University of Zawia
2 PUBLICATIONS **1** CITATION
SEE PROFILE

Adam Amril Jaharadak
Management and Science University
35 PUBLICATIONS **528** CITATIONS
SEE PROFILE

Abdulbasat Alsusaa
University of Benghazi
5 PUBLICATIONS **1** CITATION
SEE PROFILE

International Journal of Engineering &Technology, 7 (4) (2018) 6986-6990

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.26962
Research paper

Enhancing hadoop performance in homogeneous big data environment assuming configuration of dynamic slots in map-reduce pattern

Abolgasem M. Ali Enfais [1] *, Adam Amril Jaharadak [1], Amad Abdelkarim El Marghani [1], Abdulbasat Saleh Alsusaa [2]

[1] School of Graduate Studies, Management and Science University (MSU), Shah Alam, Selangor, Malaysia
[2] Faculty of Arts & science, Benghazi University, Kufra , Libya
**Corresponding author E-mail: gsooma77@gmail.com*

Abstract

Hadoop is a Java-based programming framework that supports the storing and processing of large data sets in a distributed computing environment and it is very much appropriate for high volume of data. It uses HDFS for data storing and uses MapReduce for processing that data. MapReduce is a popular programming model to support data-intensive applications using shared-nothing clusters. The main objective of MapReduce programming model is to parallelize the job execution across multiple nodes for execution. Nowadays, all focus of the researchers and companies toward to Hadoop. Due to this, many scheduling algorithms have been proposed in the past decades. There are three important scheduling issues in MapReduce such as locality, synchronization and fairness. The most common objective of scheduling algorithms is to minimize the completion time of a parallel application and also achieve to these issues. Performance issues are introduced for Hadoop schedulers, and comparative performance analysis between different cases of jobs submission. These jobs are processed in different homogenous data environment and, under fixed or reconfigurable slot between map and reduce tasks for Hadoop MapReduce java programming clustering model. The results showed that when assigning tunable knob between map and reduce tasks under certain scheduler like FIFO algorithm, the performance enhanced 16.66% in inverted index, 55.55% in word count and 11.76% in classification process.

Keywords: *Hadoop; Mapreduce; Parallel Processing; Scheduling Algorithms; Homogeneous Data.*

1. Introduction

Hadoop recently appear as the most interesting open-source software platform utilized for distributed storage and parallel processing of big data by using the MapReduce programming model. Big data can be identified mainly in three parameters which they are complexity, volumes and dynamicity. For complexity parameters, this can make sense when it is understood that the generated data in recent modern scenarios are coming from different sources and different formats, it can be text, audio, videos, symbols or numbers which are different extensions and format. Obtained images through imaging systems are considered as degraded versions of the original view. CT images have different types of degradations such as noise, blur and contrast imperfections [1]. Data loss at the acquisition time and other technical reasons must be considered. The fastest way to deblur an image is by convolving a special kernel to the corrupted image. Laplacian kernels are famous and widely used in this field [2]. Some authors considered the Laplacian sharpening filter and the iterative Richardson – Lucy algorithm, and implemented a mixture of these two techniques to process the CT medical images which reflects the amount of data used to treat it [1]. Additionally, the point spread function (PSF) is one of the essential factors that needed to be calculated, since it will be employed with different types of deblurring algorithms like Richardson – Lucy and its optimized version, Van Cittert and its enhanced version, Landweber, Poisson Map, and Laplacian sharpening filters [8]. All of these models with used algorithms reveal the extent of data that system shall process it per image.

It is clearly that, dealing with such unstructured data will impose huge complexity for purpose of analysis and data mining. There should be efforts as preprocessing to make such data more uniformed in order to make pattern recognition feasible. For the parameter describing the big data regarding its volume, the volume and the size of the data are increased rapidly and dynamically spontaneously in real-time. This can be clear if we imagine that, in modern scenarios of medical data collection to support personnel health records and monitoring health status remotely, the data are collected by different sensors for different health parameters such as heart rate, blood pressure, body temperature and glucose levels, all these data are collected simultaneously and in parallel in one database center for further analysis to send certain patterns recognition or trend analysis as a results for end-user to allow him monitor his health conditions. In this case the data is growing exponentially in size and volume and dynamically in real-time with different format and structure. In the centralized management efforts of the big data analysis, another added complexity factor is represented as the data can come from different distributed sources and many cases from heterogeneous sources.

This paper is focusing in reducing the workload of the Hadoop processing jobs (we refer it also to same meaning as Makespan), by adjusting the ratio between map and reduce for optimal minimum number to achieve the different tasks in homogenous data

clustering, finally to answer different end-users enquiries. In a Hadoop system, the incoming jobs can be either homogeneous or heterogeneous with respect to various features such as number of tasks, data and computation requirements, arrival rates, and execution times. Also, Hadoop resources may differ in capabilities such as data storage and processing units. The assigned priorities and minimum share requirements may differ between users. Moreover, the type and number of jobs assigned by each user can be different. As the mean execution time for a job, reflects the heterogeneity of both workload and cluster factors.

Borase & Banait, (2015) looked in the various clustering algorithms prevailing in order to cluster the data available with the cloud in a homogeneous way. Furthermore, the dispute examined with this research was the scalability of the procedure that was capable of processing information existing in big data. Hence, defined a strategy to resolve the issue prevailing in with an enhanced Density-based spatial clustering of applications with noise (DBSCAN) methodology. In general, the DBSCAN mechanism was capable of identifying the homogeneous data in an effectual way and for now, it was improvised to handle data available in big data along with its scalable nature. Grouping datasets disseminated in a vast spatial manner was accomplished with an efficacy. The boundaries confined in mitigating the entry of irrelevant data into a significantly associated cluster was proficiently devised with this approach. Hence, the clusters were formulated in a highly homogeneous way. The scalability realized with the mechanism devised was also better than other former techniques [7].

Issa, (2015) purported that in recent years, the demand for cloud computing has increased exponentially. This is due to an increasing demand in storing, processing and retrieving a large amount of data in a cloud cluster. The data can be either stored to a cloud network such as scientific data (i.e. Climate modeling, Fusion, Bioinformatics...etc) or use the cloud network for data-intensive tasks such as collecting experimental data, dumping data on parallel storage systems, run large scale simulations...etc. Hadoop was introduced as a solution to handle processing, storing and retrieving Big Data in a cloud environment. It is important for processor architects to understand what processor micro-architecture parameters that affect performance. It is also important for benchmark developers to optimize the benchmark software for a given hardware to achieve maximum performance possible [9].

Faizal, Christopher & Issac, (2018) reported that the Cloud ability to share information, substance and provides certain services to the people connected through the network. The Map Reduce framework in a cloud is an open source implementation, Hadoop have become the defect platform for scalable analysis on large data sets. The current Hadoop only allows static slot configuration, i.e., fixed numbers of map slots and reduce slots throughout the lifetime of a cluster. Found that such a static configuration may lead to low system resource utilizations as well as long completion length. Motivated by this, propose simple yet effective schemes which use slot ratio between map and reduce tasks as a tunable knob for reducing the make span of a given set. By leveraging the workload information of recently completed jobs, our schemes dynamically allocates resources (or slots) to map and reduce tasks [3].

Hadoop is an open source framework with three main components: MapReduce, HBase and Hadoop Distributed File System (HDFS). HDFS is the primary storage for Hadoop; it is highly reliable and uses sockets for communications. The HDFS is horizontally scalable and each cloud cluster consists of a single NameNode and DataNode. The NameNode job acts like a master server that manages the file system namespace which in turn manages access to files by clients. The DataNode manages storage attached to a node that they run on. DataNodes process server reads, write requests, block creation, and replication. HDFS is for batch processing not serving random read or write requests. The HBase is considered the Hadoop database for distributed big data storage, but the two main components of Hadoop are HDFS which is horizontally scalable with a default setting of three copies for storage and MapReduce which is parallelized scalable for computation framework. Hadoop framework consists of several applications developed using MapReduce framework; one of these applications is WordCount. MapReduce is a programmable framework for pulling data in parallel out of a cluster. Parallel processing is when a program executes multiple code paths simultaneously for a massive amount of data. In this paper, we present a detailed performance and power sensitivity analysis for Hadoop WordCount using different processors such as Intel's ATOM D525, Xeon X5690, and AMD's BobCat E-350.

Verma & Sahu, (2018) reported that clustering as a result of the rapid development in cloud computing to investigate the performance of extraordinary Hadoop MapReduce purposes and to realize the performance bottleneck in a cloud cluster that contributes to higher or diminish performance. It is usually primary to research the underlying hardware in cloud cluster servers to permit the optimization of program and hardware to achieve the highest performance feasible. Hadoop is founded on MapReduce, which is among the most popular programming items for huge knowledge analysis in a parallel computing environment [4].

Lin, Chen & Cheng, (2013) presented a check pointing methodology for enhancing the fault tolerance in Hadoop clusters. Slow tasks in the map tasks were detected by means of implementing a technique based on revised condition. Mappers were responsible for generating partial outcomes and those results were also archived by the mechanism devised. The progress rates of varied tasks in an inconstant nature were studied. The projected approach was segregated into three different phases that were given by detection of slow tasks. Longest Approximate Time to End Variable Progress Rate (LATE-VPR) scheduler was designed to become accustomed to the tasks comprising of a diverse rate of progress in real-time. Progress Rate of each and every task was assessed with fulfilling the purpose of picking up the instantaneous progress of the tasks [5].

A standard threshold was predefined to discriminate between the earlier completion and delayed stage of task completion so as to manage them in a differed way. A task that was executed right from the scratch under a speculative execution possibly be delayed in a peculiar circumstance of its late stage. Hence, there was no fruitful reaction was acquired over minimizing the response time of the job concerned at the time proximate to its completion. At this juncture, the comparatively better option was to adapt with slow running tasks than allocating a new task into the scheduler.

2. Methodology

The homogeneous system is defined as the homogeneous characteristics of both workload and cluster. Based on the job sizes, the homogeneous system is classified into two such as homogeneous-small and homogeneous-large. If all the jobs are small, then the system is defined as homogeneous-small. If all the jobs are large, then the system is classified as homogeneous-large. Mostly, the job size parameter has affected the performance of Hadoop Schedulers. The average completion time of all schedulers is almost equal. As the cluster and workload are homogeneous, the COSHH algorithm suggests all resources as the best choice for all job classes. The two conditions are used to develop homogeneous computing environment.

- The same storage representation along with the same results are considered as a correct one in hardware and software module of processor for operations on floating point numbers.
- The correct floating point value is transmitted and guarantee is provided to the communication layer during the process of floating point number between processors.

The module involved in the homogeneous phase is described as a flow diagram in Figure 1. The five representation of data-analyzing Hadoop benchmarks is described in the Heterogeneous phase as a module for makespan and slot configuration prediction. The benchmarks are derived from MapReduce Benchmarks Suite which is described below.

- Inverted Index – Consider text documents as input to generate word as document indexing.
- Histogram Rating – Consider the movie rating data as input to calculate a histogram of input data.
- Word Count – Consider text documents as input to count the occurrence of each word
- Classification – Deliberate the movie rating data as input to classify the movies into predefined clusters
- Grep – take text documents as input and search for a pattern in the files.

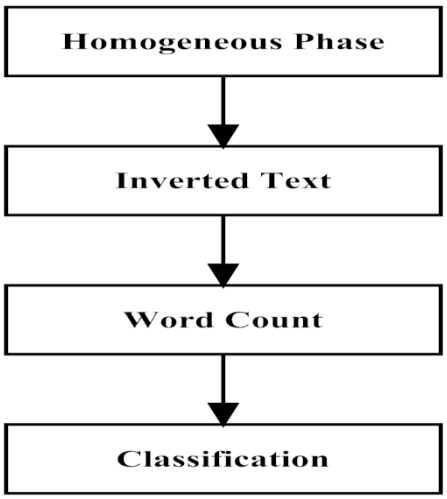

Fig. 1: Flow Diagram of Homogeneous Environment.

2.1. Inverted index

During map task, the system emits <word, docId> tuples with each word emitted once per docId. During reduce tasks, the system joints all tuples on key <word> and produces <word, list (docId)> tuples as an output after removing duplicates. The input format of an inverted index is any web document in text/XML format and the output format is represented as <word> <docId>.

2.2. Histogram rating

Based on the average ratings of movies, bin the movies into 8 bins each with a range of 0.5. The input is of the form <rater_id, rating, date> and the file name is movie_id. During the process of map tasks. The system calculates the average rating for a movie, identifies the bin, and produces <rating,1> tuples. During the process of reduce tasks, the jobs collect all the tuples for a rating and produce a <rating,n> tuple as an output. The input format of histogram rating is denoted as {movie_id:userid1_rating1 userid2_rating2,..} and the output format is represented as <rating> <num_of_user_reviews>.

2.3. Word count

Map-Reduce have become an important platform for a variety of data processing applications. Word Count Mechanisms in Map-Reduce frameworks such as Hadoop, suffer from performance degradations in the presence of faults. Word Count Map-Reduce, proposed in paper presented by Jaiswal & Bhatt (2017), provided an online, on-demand and closed-loop solution to managing these faults [6]. The control loop in word count mitigates performance penalties through early detection of anomalous conditions on slave nodes. Anomaly detection is performed through a novel sparse-coding based method that achieves high true positive and true negative rates and can be trained using only normal class (or anomaly-free) data. The local, decentralized nature of the sparse-coding models ensures minimal computational overhead and enables usage in both homogeneous and heterogeneous Map-Reduce environments. Actually, the managed and processed data were comparable in simulation to Verma & Sahu (2018) model, in which they depended on basic data similar to health data, for instance word count Word count is a typical example where Hadoop map reduce developers start their hands on with [4]. This sample map reduce is intended to count the no of occurrences of each word in the provided input files.
Map emits <word,1>tuples with the same word emitted from the document. Reduce task adds the word count for a given word from all map tasks and outputs the final count. The input format of an inverted index is any web document in text/XML format and the output format is represented as <word><count>.

2.4. Classification

It classifies the input data into k-determined clusters and the centroids value in the cluster are fixed. The classification process uses movie rating data which is of the form <movie_id, list{rater_id,rating}>. The cosine vector similarity is computed in the classification of a given movie with the centroids. Then, the closest movie to the centroid value is determined. The centroid and movie id is emitted in the map rather than emitting the details of movie ratings. Here, further iteration is not included to understand the movie details. The collected movies in a cluster are emitted as <centroid_id, movie_id> which is performed in reduce section. The input format of classification is represented as {movie_id:userid_rating1, userid2_rating2,...} and the output format is <centroid_id, movie_id>.

2.5. Grep

During the mapping process in MapReduce framework, the pattern <regex,1> tuples are obtained as output lines. During the reduce tasks, the counts are added and emit the output as <regex,n>tuples. The input format of grep is a web document in text/XML format.
Simulation Model for Homogenous Environments demonstrated in figure 2, these are the categorized work flow of specified homogenous simulation work environment.

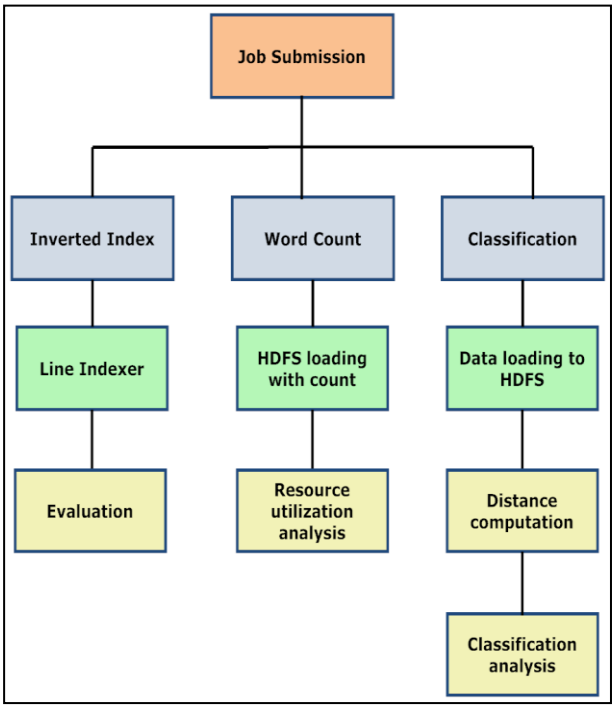

Fig. 2: Homogenous Simulation Environment.

By selecting the corresponding job based on the user favor and convenience, the work environment of homogenous cluster varies.

- In the inverted index job function, there exist two corresponding organized fields dealing with the Line Index followed by the evaluation.

- In the word count job submission, it comprises the HDFS Server loading along with the count receives high priority and resource utilization analysis.
- In the classification job submission, the procedures which are followed is listed as, data loading to HDFS, distance computation and classification analysis.

The following steps explain about the implementation of configuration under homogenous environments.

Self-adjusting slot configurations for homogeneous Hadoop clusters, then homogenous clusters selection, after that, user job submission in Homogenous environment. Inverted Index Job submission .

The user jobs comprises of set of jobs namely

- Inverted Index
- Word count
- Classification.

The choosing Inverted Index option, and selecting Makespan and Resource Utilization.

As shown in figure 3, it is clear that the inverted index takes less makespan time when compared with existing system. The graph is generated based on the count values, which was 15000 of inverted index on previous system and 9000 of existing system.

Enhancement percentage = 9000/100% = 15000/x

X= 16.66 %

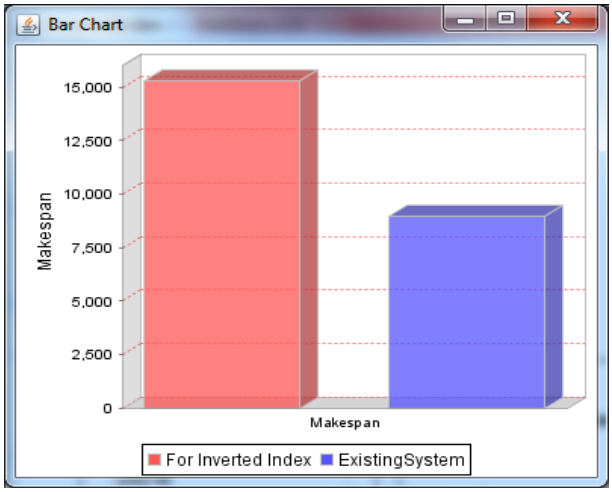

Fig. 3: Makespan Comparison for Inverted Index of Previous And Existing System.

Then applying the mentioned steps to calculate Word count job enhancement and to make comparison of word count over existing system as denoted on figure 4:

$$\text{Percentage of Improvement} = \frac{\text{MS for existing} - \text{MS for WC}}{\text{MS for existing}} \times 100$$

$$= \frac{9000-4000}{9000} \times 100$$

=55.55%

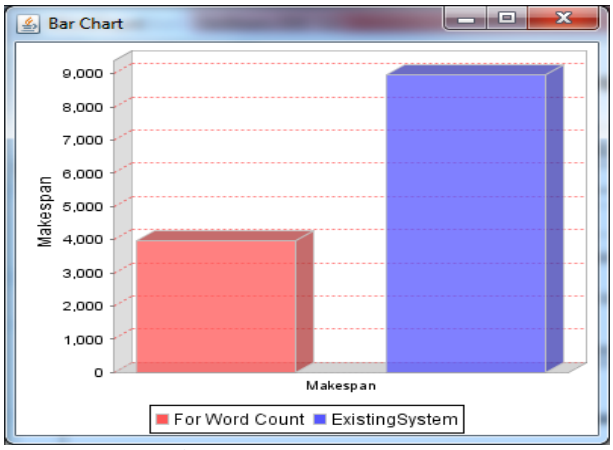

Fig. 4: Comparison of Word Count between Previous and Existing System.

Finally evaluating Classification Job in Homogenous environment. After loading of selected files into HDFS, processing Classification using machine learning technique, then analysis in Classification Job.

To clarify the improvement shown in figure 5 by percentage, the following equation deduct it:

Percentage of Improvement= (MS for existing-MS for classification) / (MS for existing) ×100

% = 8500 – 7500 / 8500 * 100 = 11.76 %

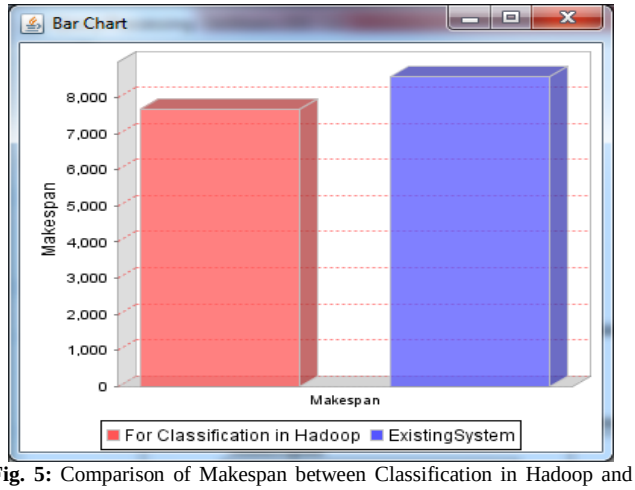

Fig. 5: Comparison of Makespan between Classification in Hadoop and Existing System.

3. Discussion and conclusion

Big and parallel data analysis for mining, knowledge extraction, and pattern recognition has become now in critical demands for modern data processing in many fields such as medical fields, banking fields, social media networks and others. The aim is always to reach to accurate data classification, prediction in optimized performance of decreased time processing and increased resources' utilization.

Hadoop Apache open-source platform is recently considered the most promising architecture of implementation for efficient big data processing loaded from distributed heterogeneous resource then, processing these data by classifying them in homogenous or heterogeneous clusters. The core architecture part of the Hadoop is the MapReduce programming model where a big block of data are divided in smaller ones and assigned for different servers for processing and then collected back again. This job is handled in each master node in typical cluster and tasks assigned to each slave node. The jobs are queued and made ready for processing by different scheduling algorithms. The main concern in performance

evaluation of the Hadoop architecture is the workload as in some cases the jobs can be stuck and resulting in unexpected delay of the time for processing and increased processor and memory consumption. Fixing the slot configuration or the number of map and reduce tasks seems to be not desirable as the previous studies showed that the static slot configuration for fixed number or ratio between and the map and reduce in certain scheduling algorithm tend to increase the makespan or the workload as well as decreases the utilization of the resources.

Therefore, enhancing Hadoop performance in homogeneous big data environment assuming configuration of dynamic slots in map-reduce pattern resulted in improvements of processing time as well as operated processes. Which means by numbers that Hadoop enhanced by 16.66% in case of inverted index, the word count is 55.55% more efficient than the existing system and from the previous results too, Hadoop enhanced by 11.76% in classification option.

References

[1] Al-Ameen, Z., Sulong, G., Gapar, M. D., & Johar, M. D. (2012). Reducing the Gaussian blur artifact from CT medical images by employing a combination of sharpening filters and iterative deblurring algorithms. *Journal of Theoretical and Applied Information Technology*, *46*(1), 31-36.

[2] Al-Ameen, Z., Sulong, G., & Johar, M. G. M. (2012). Fast deblurring method for computed tomography medical images using a novel kernels set. *International Journal of Bio-Science and Bio-Technology*, *4*(3), 9-19.

[3] Faizal, M. M., Christopher, P., & Issac, A. J. (2018). Self-Adjusting Slot Configurations for Hadoop Clusters Using Data Security In Cloud. Self, 5(04).

[4] Verma, K., & Sahu, K. (2018). Implementation of big-data applications using map reduce framework.

[5] Lin, C. Y., Chen, T. H., & Cheng, Y. N. (2013, December). On improving fault tolerance for heterogeneous hadoop mapreduce clusters. In cloud computing and big data (cloudcom-asia), 2013 international conference on (pp. 38-43). IEEE. https://doi.org/10.1109/CLOUDCOM-ASIA.2013.83.

[6] Jaiswal, N., & Bhatt, M. (2017). Big-data application using the mapreduce framework.

[7] Borase, S. D., & Banait, S. S. (2016). Dimensionality reduction using clustering techniques.

[8] Al-Ameen, Z., Sulong, G., Johar, M. G. M., Verma, N., Kumar, R., Dachyar, M., & Singh, S. (2012). A comprehensive study on fast image deblurring techniques. *International Journal of Advanced Science and Technology*, *44*.

[9] Issa, J. A. (2015). Performance evaluation and estimation model using regression method for hadoop WordCount. IEEE Access, 3, 2784-2793. https://doi.org/10.1109/ACCESS.2015.2509598.