PARALLEL PROCESSING OF HEALTHCARE DATASETS USING MAP REDUCE

K.Venkateshkumar [1], D.Chandrakala [2]

[1]PG Scholar, [2]Professor, Department of Computer Science and Engineering,

Kumaraguru College of Technology, Coimbatore, Tamil Nadu, (India)

ABSTRACT

A workflow application for efficient parallel processing of data downloaded from an Internet portal. The partitions input files into subdirectories which are further split for parallel processing by services installed on distinct computer nodes. The goal is to assess achievable speed-ups and determine which factors influence scalability and to what degree. Data processing services are implemented for assessment of context (positive or negative) in which the given keyword appears in a document. The resultant execution times as well as speed-ups are presented for data sets of various sizes along with discussion on how factors such as load imbalance and memory/disk bottlenecks limit performance. The input datasets downloaded from the internet is stored as a collection of files in a directory structure. Depending on the sizes and characteristics of the real data sets from a new technology portal, execution times and speedup are available by using Hyper threading technology. The existing programming model makes it to parallize and distribute computations and to make such computation fault-tolerant. The network bandwidth is a scarce resource. Redundant execution can be used to reduce the impact of slow machines and to handle machine failure and data loss.

Keywords: Mobile Ad Hoc Networks, Clustering, Gateway, Cluster Head Election, Node Degree.

I. INTRODUCTION

Big data is an all-encompassing term for any collection of data sets so large or complex that it becomes difficult to process them using traditional data processing applications. Big data refers to data sets whose size is beyond the ability of typical database software tools to capture, store, manage and analyze. Key enablers for the growth of Big Data are Increase of storage capacities Increase of processing power Availability of data. Effectively used Big Data can transform data into insights and intelligence, delivered where they're needed to make and implement better strategic and operational decisions.

Big data requires exceptional technologies to efficiently process large quantities of data within tolerable elapsed times. A suitable technology includes A/B testing, data fusion and integration, signal processing, simulation, genetic algorithms, natural language processing, time series analysis and visualization. Big data is a term that refers to data sets or combinations of data sets whose size (volume), complexity (variability) and rate of growth (velocity) make them difficult to be capture, manage, process or analyze by conventional technologies and tools, such as relational databases and statistics or visualization packages within the time to make them useful. The size used to determine whether a particular data set is considered big data is not firmly defined and continues to

change over time, analysts and practitioners currently refer to data sets from 30-50 terabytes to multiple petabytes as big data.

The nature of big data is primarily driven by the unstructured nature of much of the data that is generated by modern technology such as that from web logs, radio frequency ID (RFID), sensors embedded in devices, Internet searches, social networks such as smart phones, GPS devices, and call center records, face book, portable computers. It must be combined with structured data (typically from a relational database) from a more conventional business application, such as Enterprise Resource Planning (ERP) or Customer Relationship Management (CRM).

Similar to the complexity, or variability, aspect of big data, its rate of growth, or velocity is largely due to the ubiquitous nature of on-line, real-time data capture devices, systems, and networks. It is expected that the rate of growth of big data will continue to increase for the foreseeable future Specific new big data technologies and tools have been and continue to be developed. Much of the new big data technology relies heavily on massively parallel processing (MPP) databases, which can concurrently distribute the processing of very large sets of data across many servers.

1.1 The Characteristics of Big Data are

Big Data has been described by its attributes are volume, velocity, variety and veracity.

Volume & Velocity- Volume and velocity refer to the sheer quantity of Big Data available – Often hundreds of terabytes or even peta bytes of data – and the speed at which data must be stored and/or analyzed, which could reach tens of thousands of transactions per second in some cases.

Fig.1. Big Data

Varity- Variety refers to the huge variation in the types and sources of Big Data are highly structured files to unstructured video and audio information.

Veracity- Veracity refers to the level of quality and trustworthiness that can be ascribed to a data set.

Complexity- Difficulties dealing with data increase with the expanding universe of data sources and are compounded by the link, match and transform data across business entities and systems.

The emergence of big data and the potential to complex analysis of very large data sets is a consequence of recent advances in the technology. If big data analytics are adopted by agencies a large amount of stress may be placed upon current ICT systems and solutions which presently carry the burden of processing, analyzing and archiving data. Government agencies will need to manage these new requirements efficiently in order to deliver

net benefits through the adoption of new technologies. In particular technology includes low cost storage arrays, in memory processing, cloud based storage and processing together with a range of new software. The emergence of Cloud Computing over the last few years represents the single most important contributor with cloud storage. Cloud Computing offers to store, and perform computational analysis on increasingly large data sets.

1.2 Hadoop

Hadoop is open source for distributed processing of large data sets across clusters of servers. Hadoop is designed to scale up from a single server to thousands of machines with a very degree of fault tolerance. The Apache Hadoop framework is composed of the following modules,

Hadoop Distributed File System (HDFS) a distributed file-system that stores data on the commodity machines, providing very high aggregate bandwidth across the cluster.

Hadoop YARN a resource-management platform for managing compute resources in clusters and using them for scheduling of users applications Hadoop MapReduce a programming model for large scale data processing.

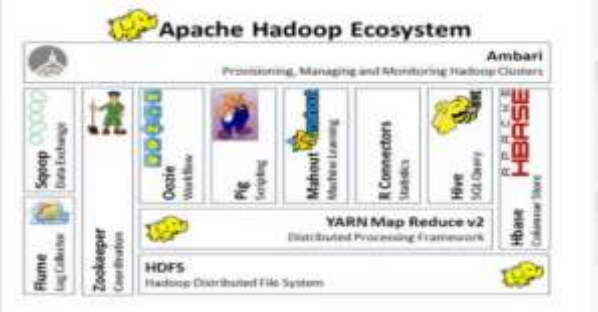

Fig.2.Components of Distributed File System

1.3 HDFS

2. HDFS is a distributed file system to distribute data.

1.4 Map/Reduce

3. It is an offline computing engine. Handles distributed Applications.

II. RELATED WORK

Several heuristics have been proposed to choose cluster heads in an adhoc network.

1. Lowest-ID Clustering Algorithm (LIC)

The node with the minimum ID is chosen to be a cluster head. Major drawbacks of this algorithm are its bias towards nodes with smaller ids which may lead to the battery drainage of certain nodes, and it does not attempt to balance the load uniformly across all the nodes.

***2.* Highest Connectivity Clustering Algorithm (HCC)**

This algorithm is also known as connectivity-based clustering algorithm. Each and every node will broadcast its ID to the neighbor nodes within its transmission range. The degree for each node is calculated and the node that contains the maximum number of neighbors is selected as the cluster head. Disadvantages are there will be lower throughputs when the degree of the node increases.

2. Weighted Clustering Algorithm (WCA)

The weighted clustering algorithm (WCA) is based on the use of a combined weight metric. i.e., the number of neighbors, distance with all neighbors, mobility and cumulative time for which the node acts as the cluster head. The weight values are broadcast by each node and so each node knows the weight values of all other nodes and other cluster heads in the system.

3. An On-Demand Weighted Clustering Algorithm (WCA) for Ad hoc Network

In this work, a weighted clustering algorithm (WCA) is presented which takes into consideration the number of nodes a cluster head can handle ideally (without any severe degradation of the system performance), transmission power, mobility and battery power of the nodes. Most of the existing clustering algorithms are invoked periodically but this algorithm is not periodic. Its invocation is adaptive based on the mobility of the nodes. More precisely, the election procedure is delayed as long as possible to reduce the computation cost. Frequent updates result in high information exchange among the nodes resulting in high communication overhead. The algorithm is executed only when there is a need, i.e., when a node is no longer able to attach itself to any of the existing cluster heads. This algorithm performs significantly better than both of the Highest-Degree and the Lowest-ID heuristics.

4. Distributed Clustering for Ad Hoc Networks

Distributed Clustering Algorithm (DCA) is presented that generalizes the previous approaches by allowing the choice of the cluster heads based on a generic weight associated to each node: The bigger the weight of a node, the better that node for the role of cluster head.

III. PROPOSED SYSTEM

A workflow application for efficient parallel processing of data downloaded from an Internet portal. The partitions input files into subdirectories which are further split for parallel processing by services installed on distinct computer nodes.

A. Parallel Data Processing Frameworks

The Map Reduce scheme is efficient for large data processing on a cluster of machines. They demonstrated how several applications could be implemented in the framework including: distributed grep for text matching, count of URL visits from server logs, web link graph, searching for most important words in documents, inverted index and distributed sort. The solution is able to cope with machine failures. The most relevant to this work is the distributed grep in which the input is split into 15000 pieces each 64 MB in size and the output reduced to 1 file. The entire computation takes approximately 150 seconds with overhead for program propagation and

delays caused by the GFS system. Recently, parallel data processing in modern distributed environments has gained attention.

B. Parallel Text Processing Applications

The work is processing of data that can be gathered from the Internet, in the context of information retrieval. The latter is the process of identification and obtaining relevant documents based on a query. There are approaches for parallel handling of queries on a multiprocessor system such as where the speed-up of 11.3 is obtained for 16 processors. There are several methods available for parallel text search. A text search mechanism on a low cost cluster with a performance model and verification of the latter against real experiments. The factor that influences performance is load balancing. Work shows performance evaluation of parallel information retrieval on a multiprocessor system with consideration of the number of CPUs, threads, disks etc. However, the number of CPUs and threads are limited to 4 and 32 respectively.

C. Workflow Modeling

In order to achieve efficient parallel processing of big data, the author proposes a workflow application depicted in Figure 3. 1. The healthcare datasets are downloaded from the portal of Inertia at http://nt.interia.pl. The downloaded datasets are stored in a designated location. The workflow considers the following steps, assuming the data is already in a designated location/ space, available as a collection of files. Service parallel process directory splits the initial directory into reasonably large subdirectories, partitions input files in a successive subdirectory and initializes parallel computations by services assess_context_of_keyword.

Service assess_context_of_keyword assesses the context of the given keyword in a list of files in parallel. The application, written in C using the Threads' library, reads file names assigned to the application and partitions into arrays to be assigned to particular threads. Each thread processes files to detect existence of the given keyword and then positive and negative descriptive words. If the keyword is present in the file (which can be an article in a portal, for example), the context of keyword within file (document, article) *a* is evaluated as fc(keyword, a) = ep(a)−en(a) where ep(a) is the number of positive descriptive words in *a* while *en*(*a*) is the number of negative descriptive words in *a*.

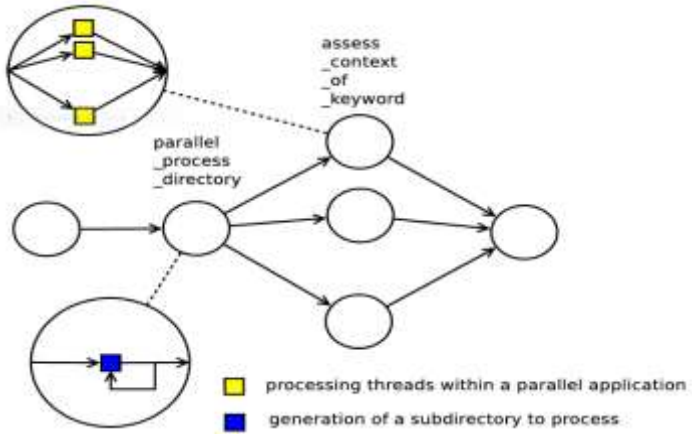

Fig.3. A Workflow for parallel processing of data

IV ACKNOWLEDGEMENT

Thanks to guide who supported me in all aspects of my project work. We are thankful to the team of the institution for providing us an opportunity to present our project in a conference.

IV. RESULTS

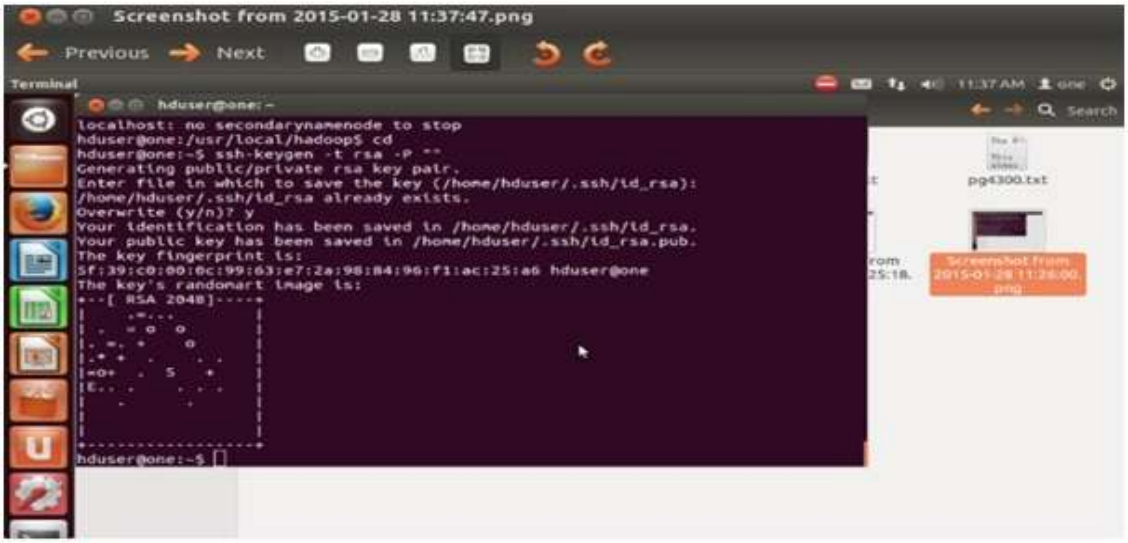

Fig 5.1 RSA Key generation

- RSA key generation for ssh configuring by accessing the system as password less.RSA key generate private and public key for secure transaction.

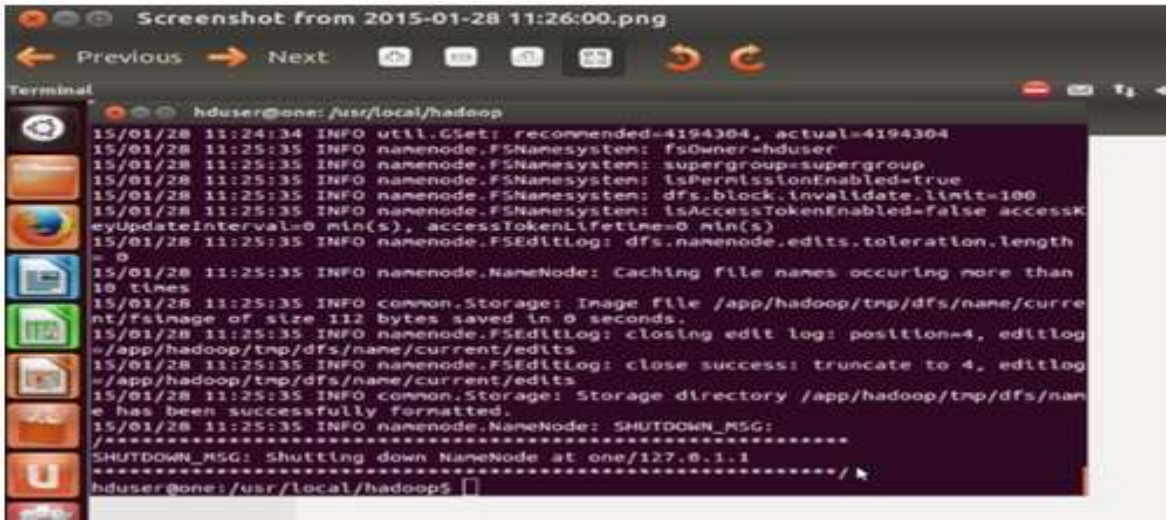

Fig 5.2 Name Node Formation

- The first step to starting up your Hadoop installation is formatting the Hadoop file system which is implemented on top of the local file system of your “cluster”.

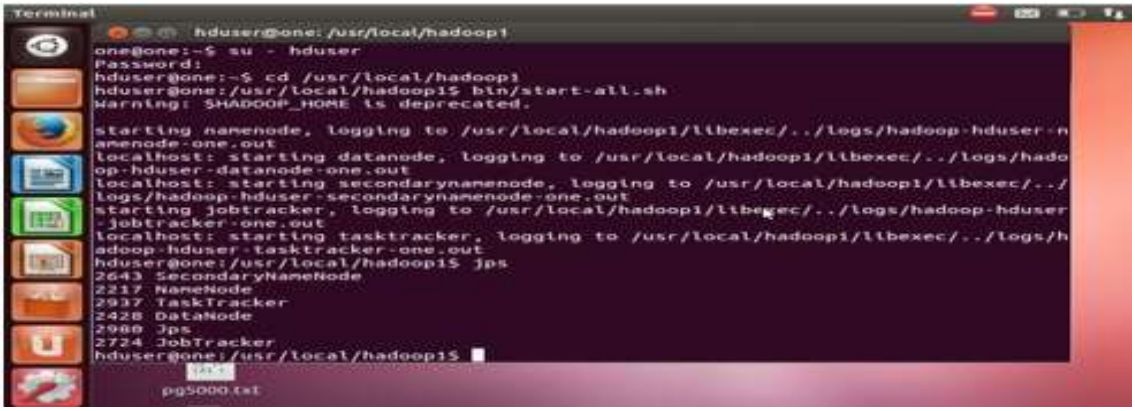

Fig 5.3 Starting Single Node

- Starting the single node cluster through name node. It starts name node, secondary name node, task tracker, data node, jps, job tracker.

Fig 5.4 Ending Single Node

- Stopping the single node cluster. It stops name node, secondary name node, task tracker, data node, jps, job tracker.

V. CONCLUSION

The workflow application for parallel processing of data sets presumably downloaded from the Internet. Parallel processing is performed at two levels: several computers in parallel and on multiple processors/cores within each node. Optimization through partitioning of data allowing fast startup of processing was presented. Depending on the sizes and characteristics of the real data sets from a new technology portal, execution times and corresponding speed-ups in the range between 26.3 and 36.5 were obtained on 64 cores, 32 of which were available thanks to the Hyper Threading technology. Factors limiting the speed-up were discussed with impact on performance.

In the future, the author plans extending the solution to other systems including processing on GPU cards as well as Intel Xeon Phi technologies.

REFERENCES

[1]. David P. Anderson. ,"Boinc: A system for public-resource computing and storage", In Proceedings of 5th IEEE/ACM International Workshop on Grid Computing, Pittsburgh, USA, November 2004.

[2]. J. Balicki, H. Krawczyk, and E. Nawarecki, "Grid and Volunteer Computing" Gdansk University of Technology, Faculty of Electronics, Telecommunication and Informatics Press, Gdansk, 2012. ISBN: 978-83- 60779-17-0.

[3]. Rajkumar Buyya, "High Performance Cluster Computing, Programming and Applications", Prentice Hall, 1999.

[4]. Stefan Büttcher, Charles L. A. Clarke, and Gordon V. Cormack. "Information Retrieval", MIT Press, 2010.

[5]. Sang-Hwa Chung, Soo-Cheol Oh, Kwang Ryel Ryu, and Soo-Hee Park "Parallel information retrieval on a distributed memory multiprocessor system. In Algorithms and Architectures for Parallel Processing", 1997. ICAPP 97, 1997 3rd International Conference on, pages 163–176, 1997.

[6]. Pawel Czarnul, "Integration of compute-intensive tasks into scientific workflows in bees clusters". In Proceedings of ICCS 2006 Conference, University of Reading, UK, May 2006. Springer Verlag. Lecture Notes in Computer Science, LNCS 3993.

[7]. Pawel Czarnul "A model, design, and implementation of an efficient multithreaded workflow execution engine with data streaming, caching, and storage constraints", The Journal of Supercomputing, 63(3):919–945, 2013.

[8]. Pawel Czarnul "Modeling, run-time optimization and execution of distributed workflow applications in the Jee- based BeesyCluster environment", The Journal of Supercomputing, 63(1):46–71, 2013.

[9]. Pawel Czarnul and Krzyszt of Grzeda, "Parallelization of electrophysiological phenomena in myocardium on large 32 & 64-bit Linux clusters", In Springer-Verlag, editor, Proceedings of Euro PVM/MPI 2004, volume LNCS 3241, pages 234–241 Budapest, Hungary, Sept. 2004.

[10]. Jeffrey Dean and Sanjay Ghemawat. Maps reduce: simplified data processing on large clusters. In Proceedings of the 6th conference on Symposium on Operating Systems Design & Implementation - Volume 6, OSDI'04, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.