Map Reduce: A Survey Paper on Recent Expansion

Shafali Agarwal
JSS Academy of Technical Education, Noida, 201301, India

Zeba Khanam
JSS Academy of Technical Education, Noida, 201301, India

***Abstract*—A rapid growth of data in recent time, Industries and academia required an intelligent data analysis tool that would be helpful to satisfy the need to analysis a huge amount of data. MapReduce framework is basically designed to compute data intensive applications to support effective decision making. Since its introduction, remarkable research efforts have been put to make it more familiar to the users subsequently utilized to support the execution of massive data intensive applications.**

Our survey paper emphasizes the state of the art in improving the performance of various applications using recent MapReduce models and how it is useful to process large scale dataset. A comparative study of given models corresponds to Apache Hadoop and Phoenix will be discussed primarily based on execution time and fault tolerance. At the end, a high-level discussion will be done about the enhancement of the MapReduce computation in specific problem area such as Iterative computation, continuous query processing, hybrid database etc.

Keywords—Map Reduce; Hadoop; Iterative Computation; Phoenix; Databases

I. INTRODUCTION

In the present days, a voluminous data handling is a prime concern topic for researchers. Many applications like data mining, Image processing, data analytic etc are required processing of huge amount of data. In 2004, Google [1] had invented a MapReduce framework suitable for parallel data processing in distributed computing environment. MapReduce is a processing paradigm of executing data with partitioning and aggregation of intermediate results. It works to process data in parallel in which splitting of data, distribution, synchronization and fault tolerance are handled automatically by the framework. Map reduce framework is famous for large scale data processing and analysis of voluminous datasets in clusters of machines.

A MapReduce framework can be categorized into mainly two steps such as [2]:

Map Phase:

- Initially split the data into key value pair and fed into mapper which in turn process each key value pair and generate intermediate output.

Reduce Phase:

- The Intermediate key value pair first collected, sorted and grouped by key and generate values associated with each key.
- The receiver produces final output based on some calculation and stores it in an output file.

Despite being featured such as scalability in clusters, ensuring availability, handling failures Google's MapReduce has been unusable for certain kind of applications requires iterative computation, execution of high-level language such as SQL and work on an Internet desktop grid. Since the MapReduce introduced, numerous MapRduce frameworks have been developed by several companies including Google's MapReduce [1], Apache's Hadoop MapReduce [3], AMPLab's spark [4], SASReduce [5], Disco [6] etc. A lot of research has been done to address the issues highlighted above and some recent MapReduce implementation helps to overcome the limitations of the prior framework. While we consider databases, an author described salient features of MapReduce implementation and its performance comparison with the parallel database. According to report, MapReduce works well in different storage systems and provide a good framework to fault tolerance for large jobs [7].

Initially the paper describes the MapReduce classification as well as an introductory explanation of its applications such as distributed pattern based searching, geospatial query processing, web link graph traversal, distributed sort, machine learning applications etc. The primary focus of this survey paper is to highlight some MapReduce implementation worked well to accomplish a specific purpose and compared with previously available frameworks. A remarkable performance improvement over the existing system seems after comparison. Later we discussed the recent enhancements which help to solve the issues related to iterative computation, efficient continuous queries execution and hybrid database.

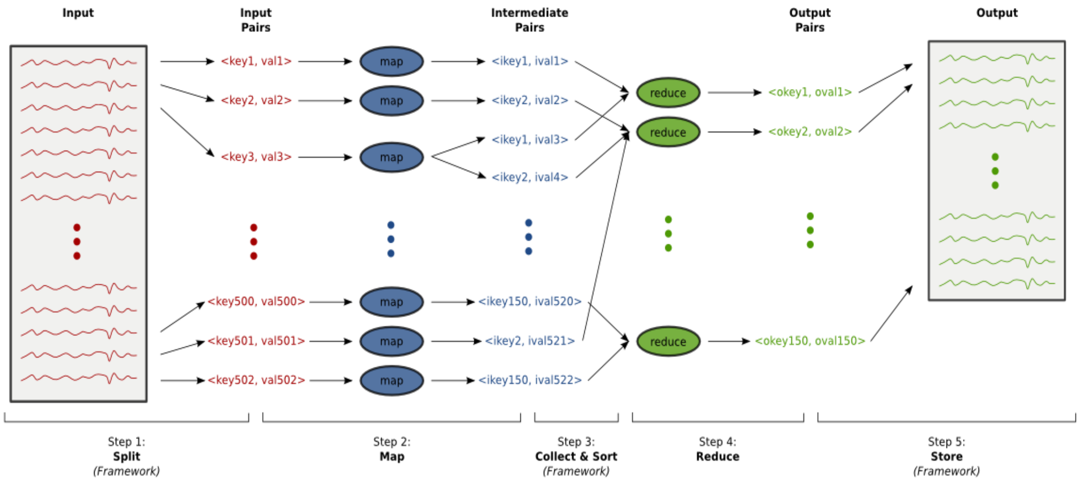

Fig. 1. Map Reduce framework

II. Map Reduce Classification

Map Reduce data analytic applications are categorized on the basis of their functions [8]:

A. *Clustering based algorithm*

These algorithms are memory sensitive as cluster-based algorithm required a large amount of storage. To measure the parameter values of multiple clusters, a massive computation makes it compute intensive method. for eg. K-means, Fuzzy K-means, canopy clustering etc.

B. *Classification Algorithm*

This algorithm works on a training set and query set to compute k nearest values which required a sufficient memory space to store the data. It is also compute intensive method because a vector product is carried out to calculate the similarity between two vectors. For eg. K-nearest neighbor etc

Author [9] analyzed the different mechanism to improve the memory utilization on the multi-core machine for MapReduce. Author had also explored three given applications with respect to efficient memory utilization.

1) Hash Join- It is a variant of broadcast join by Blanas et al [10]. In the join operation, only Map function is used to join two tables i.e. data table (S) and reference table (R). A hash join is not compute intensive application and its time complexity is O(|S|).

2) KMeans- K-means application is used to partition a set of n sample objects into K clusters for input parameter K. This algorithm is memory intensive and compute-intensive which in turn limiting the number of clusters K-means can generate. The time complexity is O(|n||k|).*

3) K-nearest neighbors- K-nearest neighbors is a classification algorithm that uses a large in-memory data set. KNN method uses two data sets, a query set Q and a training set T. It chooses K closet elements in T based on a computed distance between data points in both sets. The time complexity of the method is O(|Q||T|) because it calculates the distance between every point in Q and in T. So the KNN is compute intensive as well as memory intensive application.*

III. Map Reduce Applications

Map Reduce implementation is used in various data intensive computation because of the functionality of parallel processing of massive data. A short introduction of related applications is given below:

A. *Distributed pattern based searching*

Distributed grep command is used to search a pattern in the given text distributed over a network. Here map function searches for the pattern and produces the output so no intermediate result writes. Hence reduce function is just copied the intermediate result to output in distributed pattern searching [1].

Example: A big data of medical health record is analyzed using parallelization and pattern searching property of MapReduce taking into consideration [11]:

1) Public dataset- It consists of various reports of patients from US Food and Drug administration.

2) Biometric Datasets- It is having human characteristics like images [12].

3) Bioinformatics Signal datasets- This dataset represents the recording of vital signs of a patient. e.g. Electrocardiography ECG

4) Biomedical Image datasets- A dataset having a collection of scanning of medical images such as ultrasound images.

B. *Geospatial Query Processing*

With the technological advancement in location based service, MapReduce helps to find out the shortest path in Google map for a given location. Here Map function searches all connecting paths from source to destination with distance value. After sorting the keys, the Reduce function emits the path which is of shortest distance.

An algorithm LoNARS [13] has implemented to improve Reduce task scheduling by considering data locality and network traffic. Even author achieved 15% gain in data shuffling time and up to 3-4% improvement in job completion time.

C. Distributed Sort

Distributed sort is used to arrange the data in sorted manner split across multiple sites. In Map Reduce implementation, initially input data is given to map function to convert it into intermediate data which is stored in a local disk buffer. In next step, data is transmitted to the appropriate reducer function over the network. A number of reduce functions sort the data according to given key value and writes the output [14].

Author represents massive data sorting using Apache Hadoop open source software framework with the help of three map reduce functions [15]:

- Teragen: used to generate input data to be sort.
- Terasort: Sample the input data and used them with Map Reduce to sort the data.
- Teravalidate: At last sorted output data is validated.

This method is I/O intensive as it works on data input/output.

D. Web Link Graph Traversal

A large-scale graph is also known as web graph. For eg. According to a survey Facebook is having more than 1 billions of users (vertices) and more than 140 billions of relationships (edges) among them in 2012 [16].

Basically Map Reduce model is not suitable for iterative data analysis application that’s why it is assumed to be inadequate for graph traversal. In order to accomplish large scale graph processing, Surfer and GBASE are used as an extension of Map Reduce that are proposed to make it suitable for graph processing.

Surfer- Surfer is an engine used in graph processing. It works with two components i.e. Map Reduce and propagation. Map Reduce processes data parallel in terms of key/value pair whereas propagation is an iterative computational pattern that propagate data from a vertex to its neighbors in the graph.

GBASE- GBASE [17] executes block compression to store homogeneous region of the graph. When a graph traversal query is fired, GBASE selects the grid having a block that is relevant to query. Therefore only relevant required data is fed into Hadoop jobs.

E. Term Vector per Host

This term refers to summarize the important words of a document or multiple documents. A map function finds out the term vector for a particular host name as (host name, term vector) pair and pass this data to reduce function for a given host. Now reduce function add these term vectors and produces a final output in terms of (host name, term vector) [18].

F. Machine Learning Applications

Machine learning is a branch of artificial Intelligence which deals with the building of systems that learn from data without need of explicit programming for all the possible conditions.

Author [19] discussed the case of Netflix prize data which is an online DVD rental company. Netflix wants to predict the user preferences of movies based on their rating. In order to get the data map function is used to generate a table which contains information regarding users and their movie preferences. After completing this process, reduce function derive a contingency table for each group of intermediate results depicts user preferences about movies.

G. Data Clustering

Data clustering is a fascinating field for researchers involved in Image processing, data mining and document retrieval area. Data clustering is used to solve the computational complexity arises due to the voluminous data used in processing by dividing complete data set into small data subsets based on certain criteria.

Author [20] used parallel K-means clustering using map reduce to minimize the efforts make to handle a large data sets. A key feature of this algorithm is the use of combiner used to partially combine the intermediate values of map function with the same key.

H. Inverted Index

It is an index data structure storing mapping from contents such as words or numbers to its locations in the database file or in a document. Inverted Index is used in data retrieval in a large database management system. This process receives a list of document as input and produces word to document indexing. Alternatively it is used to track the position of words in a given document.

A map function parsed each document and retrieved its document Id with the word. Later reduce function accepts all pair of given words and emits corresponding word with its list of relevant documents. Hence complete output pairs represent an Inverted Index of the database.

IV. Map Reduce Models and their Comparison

A. Hadoop vs. Phoenics++

Many map reduce implementations have been discussed in the previous section. From which Hadoop is given by Apache and support distributed memory clusters. Similarly phoenix++ works on shared memory multicore systems [21]. Author [22] compared the performances for word count problem running on Amazon elastic compute cloud (Amazon EC2) of both systems and concluded that phoenix++ is superior to Hadoop in terms of execution time. According to him phoenix++ is faster than Hadoop by 28:5 on four virtual CPUs for 7.4 seconds versus 211 seconds.

B. Phoenix vs. Phoenix2

Phoenix was introduced as a Map Reduce model which can work on shared memory machine and symmetric multiprocessors with scalability [23][24]. This model was not appropriate for many types of workloads because of certain functionalities. Author described the revised version of phoenix2 as phoenix++ with the introduction of containers which eventually reduces the memory requirement. It hides task scheduling details and represents a basic map reduce model. Containers are used to store emitted key-value pair by key and storing them in combiners which stored all emitted values with the same key. This increases the necessity of writing high-performance code which eventually improves scalability over phoenix2.

C. Hadoop vs. BitDew Map Reduce

Google invent a new map reduce programming for Internet Desktop Grids using BitDew middleware. The main feature of this implementation highlights a firewall friendly protocol, fault tolerance, result certification, two level schedulers and more. The Author presented new optimizations to BitDew MapReduce in terms of aggressive task backup, intermediate result backup, task re-execution, mitigation and network failure hiding.

A new framework is proposed by the authors [25] which emulated key aspects of Internet Desktop Grid and as well as compared it with apache Hadoop framework. According to their report BitDew Map Reduce framework is able to handle all stress tests whereas Hadoop is not suitable with wide area network topology which includes PC hidden behind firewall and NAT. Additionally BitDew Map Reduce is more successful in terms of fairness, resilience to node failures and network disconnections.

D. Map Reduce Parallel Computation vs. PRAM

The author compared Map Reduce parallel computation model to PRAM (Parallel Random Access Machine) model and analyzed that parallelization of computation on the relatively small number of machines makes Map Reduce model more efficient than PRAM model. However, complete running time for mapper & reducer reaches polynomial time rather than linear. In the paper, authors explained the idea to compute a minimum spanning tree of a dense graph in only two rounds whereas PRAM model requires $\Omega(\log n)$ rounds [26].

V. Map Reduce Enhancement

A. Peacock: An improved version of Phoenix

Phoenix++ is a Map Reduce implementation best worked with shared memory multicore platform. An application distributed sort is efficiently carried out with the introduction of built-in containers. Initially, Map Reduce starts with partitioning the complete data set into equal size portion, each of which is processed by map workers. Further in next step, containers invoked to group the emitted values with the same key and stored them in combiners.

Combiner object passes the data to reduce phase after a run on all cross-thread emitted values. At last reduce phase parse the data and produce the final result stored in result buffer array. With the help of container phoenix++ implementation reduces the overhead occurred in intermediate data storage.

Later author had described a refined version of phoenix++ known as peacock. Peacock is a MapReduce system with workflow customization execution flow which reduced the overhead of intermediate data which is having only one emitted value per key [27].

B. HaLoop and Spark for Iterative Computation

An extended version of MapReduce known as HaLoop used for data-intensive applications also work well for Iterative computation. Author devices Iterative task with three iterations that have two features-

1) Data source of each iteration is having two parts, one is variant and another is invariant.

2) Convergence of iterative procedure to a fixed point might need a progress check at the end of each iteration.

In the iterative computation, additional functions Add Map and Add Reduce of HaLoop works for efficient processing of data. Here different units of HaLoop functions work constantly for variant part of data and stored the Intermediate value of invariant data locally. Hence reduces unnecessary scanning of invariant data.

The reducer just compared the data that has been catched from the previous iteration with the newly generated results to check whether a fixed point is achieved. This strategy helps in time saving with the advent of local storage of invariant data.

Spark is another implementation of Map Reduce, useful for performing iterative computation. A storage abstraction called resilient distributed dataset (RDD), which is a collection of tuples across a set of machines inputted to the map function. A usual processing of map function takes place with the tuples of each partition of RDD and further reduce function is used for aggregation of the resulted tuples. A key feature in spark implementation is the use of intermediate data of RDD stored locally in memory and reused it in subsequent iteration computation. Hence a faster processing of iterative function is carried out [28].

C. Hadoop Online Prototype (HOP)

A new improved version of Hadoop MapReduce framework was proposed by the authors [29] which supports intermediate data to be pipelined between operators and named it Hadoop online prototype (HOP). HOP helped to widen the range of the domain of the problems like a continuous queries execution. According to his study, MapReduce framework can be used for event monitoring and stream processing.

D. Reduced Input size to solve graphs

We know that MapReduce is known for parallel processing of peta byte scale data. An idea of the author is to apply some filtering technique so that the input size can be reduced in distributed manner, resulting to a much smaller problem instance can be solved on a single machine.

Author [30] mainly emphasized on the related graph problems such as for minimum spanning tree, maximal matching, approximate weighted matching, approximate vertex and edge covers and minimum cuts. The given algorithm represents the trade-off between available memories on the machine and numbers of map reduce rounds. Later to proven his idea, the author depicted the implementation of the maximal matching algorithm and represents that how to compute a maximal matching in three map reduce rounds in the model of [31]. Finally author concluded that if the machine have memory O(n) then this algorithm required O(log n) rounds.

E. HOG: Hadoop on Grid

The author proposed a Hadoop Map Reduce framework executed on open science grid which covers all institutions span in USA. The framework is different in terms of data availability and detection and resolution of the zombie datanode problem from those which are dedicated to a cluster or cloud. It creates multi institutions failure domain and also provides wide area data analysis as well as map data centers across U.S. This proposed system has experimented with 1100 nodes on grid and provided comparable performance than cluster [32].

F. Cloud Data Management System

Map Reduce is a programming model which implements applications over cloud data storage system. Various service providers provided data management systems over cloud such as Google's Bigtable [33], Yahoo's PNUTS/Sherpa, Amazon's Dynamo, Microsoft's Dryad ets.

G. Summarizing Large Text Based On Map Reduce Framework

Author proposed a technique to summarize large collection of text using semantic similarity based clustering and topic modeling using Latent Dirichlet Allocation (LDA) over Map Reduce framework [34]. The proposed method is evaluated in terms of scalability, compression ratio, retention ratio, ROUGE and pyramid score. Experiment results have shown the better scalability and reduced time complexity of summarization of large text data over Map Reduce framework. Author also proposed a multilingual text summarization over Map Reduce framework as his future work.

VI. Map Reduce and Data Processing Tools

A. HadoopDB

The author suggested a hybrid system of parallel database and Map Reduce based system named HadoopDB to utilize performance and efficiency of parallel database as well as scalability, flexibility and fault tolerance of Hadoop. The ability of HadoopDB makes extensible support for performing data analysis at the large scale of workloads. [35]

B. Hive

Hive- an open source data warehousing system used by various companies like Yahoo, facebook etc to store and process huge data sets on commodity hardware [36]. Hive works on a SQL like declarative language- HiveQL to execute queries. Hive contains a system catalog - Metastore – which includes schemas and statistics, useful in data exploration, query optimization and query compilation. Authors are aiming to develop methods for multi-query optimization techniques and generic n-way joins process in a single map-reduce job.

C. Apache Pig

Apache Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs with a salient property that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets [37]. There is a compiler in Pig's infrastructure layer that produces sequence of map reduce programs. Pig is used a declarative language i.e. Pig Latin which has the properties 1). Ease of programming; 2). Optimization opportunities; 3). Extensibility.

D. SCOPE (Structured Computations Optimized for Parallel Execution)

Scope is a declarative and highly extensible language for web scale data analysis on large clusters. This is much like to SQL so users don't require training to use it. Users can easily develop their own functions to solve problems by implementing their own functions and versions of operators: extractors (parsing and constructing rows from a file), processors (row-wise processing), reducers (group-wise processing), and combiners (combining rows from two inputs). SCOPE compiler generates a parallel execution plans which is further optimize by its optimizer [38].

VII. Related Work

The most related work associated with the introduction of various MapReduce models and its relation with the database processing [39]. This tutorial provides the insight about how to improve the performance by increasing availability of the system in case of failure, reduced network communication overhead, process scheduling etc. [40] Authors performed a detailed study about its open source implementation-Hadoop and few factors such as 1) I/O mode, the way of a reader retrieving data from the storage system, 2) data parsing, the scheme of a reader parsing the format of records, and 3) indexing, which is used to speeding up data processing. According to the given study [41] in case of complex analytical task, Hadoop is slower by a factor of 3,1 to 6.5 as compare to parallel data base systems. Later it has been notified that by tuning the above given factors, performance of Hadoop system is improved by a factor of 2.5 to 3.5 for the same benchmark. A critical comparison is carried out between parallel database and MapReduce that criticize the performance of MapReduce [42] for large data bases. According to survey parallel DBMS is more suitable for large scale data processing whereas MR excels in complex analytics and ETL. Basically an interface is required between parallel DBMS and MapReduce to gain the performance excellence of both systems. A large scale data management arise the interest about cloud environment. Author described the concept of cloud computing, related research and its implementation based on VCL (Virtual Computing Laboratory) [43].

VIII. Conclusion

MapReduce provides a distributed parallel computing across multiple nodes and return result on a particular node. MapReduce plays a vital role in parallel data processing because of its salient features such as scalability, flexibility and fault tolerance. Previous Research showed that Map Reduce framework is not sufficient to handle some specific kind of applications. It raised a question regarding improvement and enhancement of the Map Reduce architecture to address those issues and challenges. In this survey paper, our focus was on the extended Map Reduce framework with additional functionalities to support some specific kind of tasks. Initially, we reviewed Google invented Map Reduce architecture and its various applications. Many organizations have invented various Map Reduce frameworks with additional features after Google's invention. We had compared the design and functionalities of frameworks with Apache Hadoop and Phoenix.

A lot of research work has been done on the extension of Map Reduce carried out with new functionalities and mechanism to optimizing it for a new set of problems. We reviewed the extended version of Mapreduce for more data intensive applications such as HaLoop and Spark Map Reduce work well for Iterative computation. Another improved version of Hadoop is known as hadoop online prototype (HOP) designed to support continuous query execution & event handling concluding with the introductory description of HadoopDB which helps to improve the performance of the system with combined features of parallel database and Hadoop database. At last a brief introduction of different data processing tool such as HadoopDB, Hive, Apache Pig and SCOPE used with Map Reduce has been discussed.

IX. Future Research Directions

Map Reduce was initiated by Google to handle big data analysis which is unstructured data such as web document. We have discussed a number of Map Reduce models still researchers can develop a more efficient Map Reduce with improved functionalities. Similarly a new user friendly data processing language can be introduced to make data handling easier.

References

[1] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters" In ACM OSDI, 2004.

[2] A. Elsayed, O. Ismail, and M. E. El-Sharkawi, MapReduce: State-of-the-Art and Research Directions, International Journal of Computer and Electrical Engineering, Vol. 6, No. 1, February 2014.

[3] "Hadoop," available at http://hadoop.apache.org/.

[4] M. Zaharia, M. Chowdhury, Michael J. Franklin, Scott Shenker and Ion Stoica, Spark: Cluster Computing with Working Sets University of California, Berkeley, May, 2010.

[5] D. Moors, Whitehound Limited, UK, "SASReduce An implementation of MapReduce in BASE/SAS", Paper 1507-2014

[6] E. Bugnion, S. Devine, K. Govil, and M. Rosenblum, Disco: Running Commodity Operating Systems on Scalable Multiprocessors (1997).

[7] J. Dean and S. Ghemawat, MapReduce: A flexible data processing tool, Communications of the ACM, Vol. 53 No. 1, Pages 72-77 10.1145/1629175.1629198

[8] K. Ericson and S. Pallickara, On the Performance of High Dimensional Data Clustering and Classification Algorithms (2013).

[9] Y. Zhang, "Optimized Runtime Systems for MapReduce applications in Multi-core clusters", A thesis in Houston Texas, May 2014.

[10] S. Blanas, J. M. Patel, V. Ercegovac, J. Rao, E. J. Shekita, and Y. Tian, "A Comparison of Join Algorithms for Log Processing in MapReduce," in Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data, SIGMOD '10, (New York, NY, USA), pp. 975–986, ACM, 2010.

[11] E. A Mohammed, B. H Far and C. Naugler, Application of the MapReduce programming framework to clinical big data analysis:Current landscape and future trends, BioData Mining. Vol. 7, 22. Oct 29, 2014.

[12] M Jonas, S Solangasenathirajan and D Hett. Annual Update in Intensive Care and Emergency Medicine 2014. New York – USA: Springer. Patient Identification, A Review of the Use of Biometrics in the ICU; pp. 679–688. (2014).

[13] E. Arslan, M. Shekhar & T. Kosar, "Locality and Network-aware reduce task scheduling for data intensive applications", published in Proceedings DataCloud'14 Proceedings of the 5th International workshop on Data Intensive Computing in the Clouds, page 17-24, ISBN:978-1-4799-7034-6

[14] D. Gillick, A. Faria and J. DeNero, "Map Reduce: Distributed Computing and Machine Learning", Dec-2006

[15] Owen O'Malley, "TeraByte Sort on Apache Hadoop", Yahoo! owen@yahoo-inc.com May 2008.

[16] S. Sakr, Processing Large Scale Graph Data: A Guide to Current Technology, National ICT Australia, June 2013.

[17] U Kang et al., GBASE: A Scalable and General Graph Management System, San Diego, California, U.S.A. ACM978-1-4503-0813-7/11/08. Aug-2011.

[18] J. Dean, "Experience with MapReduce, An Abstraction for Large Scale Computation", Google, Inc., proceedings of the 15th international conference on parallel architecture and compilation techniques, ACM New york, US, ISBN:1-59593-264-X doi>10.1145/1152154.1152155

[19] S. Chen and S. W. Schlosser,"Map reduce meets wider varieties of applications", IPR-TR-08-05, Research at Intel (2008).

[20] W. Zhao, H. Ma & Q. He, "Parallel K-means clustering based on mapreduce", cloudcom, LNCS 5931, pp 674-679 © springer-verlag Berlin Heidelberg 2009.

[21] J. Talbot, R. M. Yoo, and C. Kozyrakis, "Phoenix++: Modular mapreduce for shared-memory systems," In Proc. of the second international workshop on MapReduce and its applications, pp. 9–16, 2011.

[22] C. Cao, F. Song, D. G. Waddington, "Implementing a high performance recommendation system using Phoenix++", In Proc. of Internet Technology and Secured Transactions, 8th International Conference for, DOI 10.1109/ICITST.2013.6750200, pages 252-257, dec 2013.

[23] C. Ranger, R. Raghuraman, A. Penmetsa, G. Bradski, and C. Kozyrakis. Evaluating MapReduce for multi-core and multiprocessor systems. In Proc. of the 13th Int'l Symposium on High Performance Computer Architecture, pages 13–24, 2007

[24] R. M. Yoo, A. Romano, and C. Kozyrakis. Phoenix rebirth: Scalable MapReduce on a large-scale shared-memory system. In Proc. of the 2009 IEEE Int'l Symposium on Workload Characterization, pages 198–207, 2009.

[25] L. Lu, H. Jim, X. Shi, Fedak G.,"Assessing MapReduce for Internet Computing: A Comparison of Hadoop and BitDew-MapReduce", In Proc. of the Grid Computing (GRID), ACM/IEEE 13th International Conference on Grid Computing, DOI:10.1109/Grid.2012.31, ISBN: 978-0-7695-4815-9, pp. 76-84, Sept 2012.

[26] H. Karloff, S. Suri and S. Vassilvitskii, A Model of Computation for MapReduce, at AT & T Labs and Yahoo! Research (2010).

[27] S. Wu, Y. Peng, H. Jin, J. Zhang,"Peacock: a customizable Map Reduce for Multicore plateform, In Proc. of the Journal of Supercomputing, DOI 10.1007/s11227-014-1238-2, Springer Science + Business Media New York pages:1496-1513, June 2014

[28] Li, F., Ooi, B-C., Özsu, M. T., Wu, S., Distributed Data Management Using MapReduce. ACM Comput. Surv. 0, 0, Article A (0), 41 pages. DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000 (2013).

[29] T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein,"MapReduce online" in UC Berkeley, 2010.

[30] G. D. F. Morales, A. Gionis and M. Sozio. Social Content Matching in MapReduce. 37th International conference on very large databases, Seattle Washington, Proceedings of the VLDB Endowment, Vol. 4, No. 7 Copyright 2011 VLDB Endowment 2150-8097/11/04.

[31] A. V. Goldberg and S. Rao. "Beyond the flow decomposition barrier", JACM, 45(5):783–797, 1998.

[32] C. He, D. Weitzel, D. Swanson, Y. Lu. HOG: Distributed Hadoop MapReduce on the Grid Published by SC Companion: High Performance Computing, Networking Storage and Analysis (2012).

[33] Sakr S, Liu A, Batista DM, Alomari M, A survey of large scale data management approaches in cloud environments. IEEE Commun Survey Tutorials 13(3):311-336 , 2011.

[34] N K Nagwani, Summarizing large text collection using topic modeling and clustering based on MapReduce framework, Journal of Big Data 2:6 DOI 10.1186/s40537-015-0020-5, 2015.

[35] A. Abouzeid, K. B. Pawlikowski, D. Abadi, A. Silberschatz and A. Rasin, HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads, VLDB Endowment '09' August Lyon France, 24-28, 2009

[36] A. Thusoo, J.S. Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhang, S. Antony, H. Liu, and R. Murthy, Hive – A Petabyte Scale Data Warehouse Using Hadoop. Proc. of ICDE, 2010.

[37] Hadoop Pig. Available at http://hadoop.apache.org/pig

[38] R. Chaiken, et. al. Scope: Easy and Efficient Parallel Processing of Massive Data Sets. In Proc. of VLDB, 2008.

[39] J. Zhao, J. Pjesivac-Grbovic, MapReduce: The Programming Model And Practice, 2009.

[40] D. Jiang, B. C. Ooi, L. Shi and S. Wu, The performance of mapreduce: An in-depth study. Proc. VLDB Endow., 3 pp. 472–483 (Sept 2010),

[41] A. Pavlo, E. Paulson, A. Rasin, D.J. Abadi, D.J. DeWitt, S. Madden, and M. Stonebraker. A comparison of approaches to large-scale data analysis. In Proceedings of the 35th SIGMOD international conference on Management of data, SIGMOD '09, pages 165–178. ACM, 2009.

[42] M. Stonebraker, D. J. Abadi, D. J. DeWitt, S. Madden, E. Paulson, A. Pavlo, and A. Rasin. MapReduce and parallel DBMSs: friends or foes? Communictions of the ACM, 53(1):64{71, 2010.

[43] M. A. Vouk, Cloud Computing-Issues, Research and Implementations, Journal of Computing and Information Technology, CIT 16, 4, 235-246, doi-10.2498/cit.1001391, (2008).

Author Profiles

Shafali Agarwal is associated as an assistant professor with JSSATE, Noida, formerly she worked with NIET, Greater Noida. She has received her Ph.D. from Singhania University, Rajasthan. Her research areas are MapReduce Implementation and fractal analysis which is a part of Image processing. She has published papers in national conference, International conference and International journal which are indexed by ACM, Springer, Citeseer, ProQuest, Index Copernicus, EBSCO, Scribd and many more. She has done graduation in 2001, master in computer applications in 2004 and after that MPhil in 2007. She got published a book titled "Data Structure using C" for engineering students. She is an active member of IEEE Computer society.

Zeba Khanam is presently working at the department of Computer Science, JSSATE,Noida .She has received her PhD degree from Jamia Millia Islamia (Central University),New Delhi. Her research focuses on evolving the legacy systems using refactoring techniques with aspect oriented programming.

Her research interests are Map Reduce Framework, Software Re engineering and Reverse Engineering. Her recent publications include articles in IEEE Xplore, Wseas Transactions, Procedia Engineering,WORLDComp'11 proceeding