International Journal of Advances in Computer Science & Its Applications – IJCSIA
Volume 4: Issue 3 [ISSN 2250-3765]
Publication Date : 30 September,2014

Efficient Utilization of Profiles to Reduce Time in Very Large Data Set

Kamini Gupta*, R H Goudar**

***Abstract*- Hadoop is a software framework for analysis of large data sets. Hadoop distributed file system and map reduce paradigm provide an efficient way to deal with terabyte of data being produced every second. MapReduce is known as a popular way to hold data in the cloud environment due to its excellent scalability and good fault tolerance. However, creating profiles for the same job again and again makes it less efficient. This paper proposes an INTERFACE that optimizes time taken to match sampled mapreduce jobs (Js) with already created profiles. It acts as mediator between profile store and worker (nodes).**

***Keywords* - Profile, sampling, tuning, optimization, mapreduce, hadoop distributed file system.**

• I. INTRODUCTION

Big data refers to the information that cannot be processed or analyzed using current tools/system. To get value out of the large volume of data, has become challenge for organization. [7]

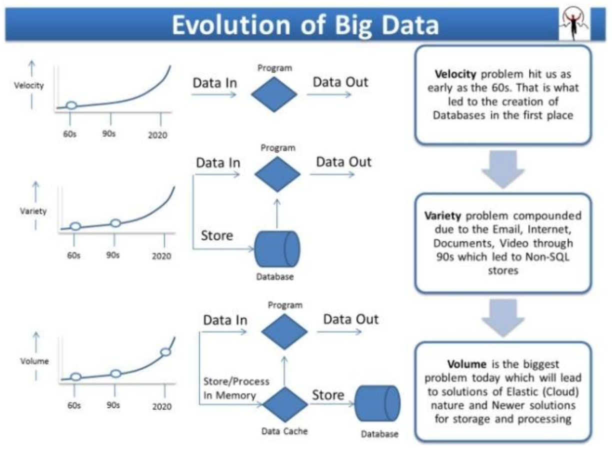
Evolution of Big Data
Velocity
60s
90s
2020
Data In
Program
Data Out
Velocity problem hit us as early as the 60s. That is what led to the creation of Databases in the first place
Variety
Store
Database
Variety problem compounded due to the Email, Internet, Documents, Video through 90s which led to Non-SQL stores
Volume
Store/Process In Memory
Data Cache
Volume is the biggest problem today which will lead to solutions of Elastic (Cloud) nature and Newer solutions for storage and processing

Fig 1a*: Evolution of Big data [10]*

Kamini Gupta,*

Department of CSE, Graphic Era University, Dehradun

*R H Goudar***

Dept. of Computer Network Engineering (CNE),

Visvesvaraya Technological University (VTU), Belgaum- 590018.

Big Data has not evolved today but this process has started with the scientific advancement of technology. Earlier during 60s there raises the demand for database but after that as the streaming data rate increases, the need to analyze data along with it also became a major issue.

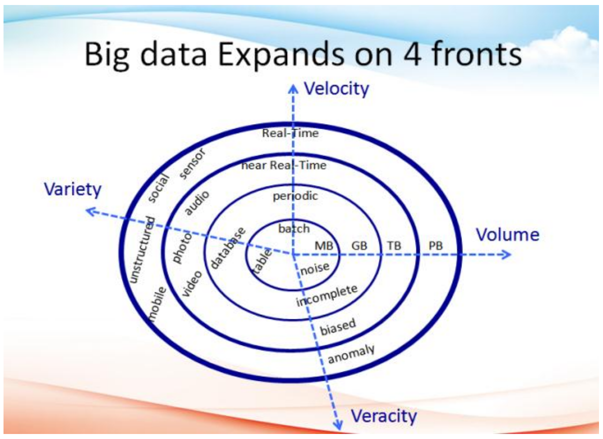

Fig 1b*: Big Data Expands on 4 fronts [11]*

Evolution of big data has marked starting of new era that is working to extract valuable information from data coming every second.
In Current Scenario Big Data can be described with the help of following four characteristics [9][12][13] and their fig1b.
Volume: Amount of data coming every second in database;
Variety: structured/unstructured data.
Velocity: speed at which data is collected and its need to be handled.

Veracity: valuable part of information that is streaming in database.

To handle big data, platform called Hadoop was built with two major components;

1. Hadoop Distributed File System [18].
2. Map reduce programming model [16], [17].

Data in hadoop cluster is broken into smaller pieces called blocks and then distributed over various nodes in cluster .Map and reduce functions are then applied on that block.

Map functions consist of following execution process: [22]

- Read - Refers to reading of map inputs supplied by worker (nodes).
- Map - Refers to dividing job into blocks and then assigning blocks to n mappers .Key, value pairs are formed.
- Collect – Output from map function is collected from various clusters.
- Spill – sorting, combining, compressing, and writing outputs to local disk.
- Merge- Merging of files after spilling process is completed.

Reduce functions consists of following execution process:

- Shuffle – Output of map tasks is assigned as input to reduce tasks.
- Merge – Merging of sorted map outputs.
- Reduce-Reduce function combines the input and then processes them.
- Write – Write the reduced output to distributed file system.

Map function role is to divide that block of data to key value pairs [19][15]. Then they are passed to reduce function which combines them to give final output. Block sent to map function is also called as job. To perform map reduce task many parameters are to be considered as ioformatter, io.sort.mb, io.sort factors [14]. These parameters need to be tuned up properly to process job efficiently. It is very difficult to do it manually for each job as it will fail our objective of time efficiency. So profiler is used which help in tuning up of parameters automatically and define all those tuned up parameters in profile [4][5]. Profiles are store in profile store for future use. Profile store contains all created profiles. Profiler is responsible for collecting job profiles. It consists about execution of every phase in map and reduce processes.

Basic aim of this paper is to reduce the time required to match profiles. Interface proposed here act as a mediator between workers and profile store. INTERFACE contains only recently used profiles, so its easy to search if similar job occurs after some time.
First step is to create sample profile of job that is done by sampler. In second step we will introduce algorithm for profile matching in INTERFACE. Third step deal with job profile storage for further use.

Following are the contributions

Its use is not limited to finding profile in less time but we can also use the INTERFACE to analyze jobs that came recently.

- Other techniques as pattern matching, only considers CPU utilization parameters.
- Less usage of energy and resources which result in good performance rate.
- Jobs do not have to wait longer for their profile as some will have already existing profiles.
- It is cost optimization based technique.
- Time taken to match profile has reduced by nearly 25% as many profiles can be found in INTERFACE.

To demonstrate our approach section II provides related work done in this area. Section III explains our approach in which profile matching scheme is used followed by conclusion in Section IV.

II. LITERATURE REVIEW

Early work done in improving mapreduce performance include work done by Matei Zaharias et.al[17] that focuses on improving performance of hadoop for heterogeneous environment. One research paper related to this study is work done by EamanJahani et.al in paper entitled "Automatic optimization for MapReduceprogram"[2]. In his paper he proposes ideas to successfully detect the opportunities for optimization. In [1] work done by S.Babu et.al also gives us method to determine execution of other jobs using predetermined settings of previous jobs. "In Starfish: A Self tuning System for Big Data Analytics"[3] work done by H Herodotou et.al is related to automatic tuning of parameters. Sasiniveda .Get.alalso did a very good work in this reference in [6] by parallelizing algorithm of data mining, evaluating all the parameters for optimizing of MapReduce paradigm. Approach followed by Nikzad Babaii et.al in [21] is also a better technique in matching profiles based on CPU utilization but problem is that it only considers number of mapper and number of reducers ignoring other parameters as for network resources etc. Issues and challenges discussed by Kyong-Ha Lee et.al [19] gave us good idea about mapreduce and its limitations. Thesis on "PStorM: Profile Storage and Matching for Feedback-Based Tuning of MapReduce Jobs"[8], by MostafaEad gave good ideas for increasing efficiency of Hadoop Distributed File System .The problem with earlier solutions was that they could not provide method for decreasing the time constraint . Solution provided in PStorm were not that much efficient. The solution to this problem has been described in Section III.

OVERVIEW OF INTERFACE

Now we all are aware of the relevance of big data and we are also well aware of methods being opted to tackle this problem. But it is not that easy. Main problem is that we have to do our job in less time. In hadoop we have to first create profile of map reduce job that itself consumes time. So despite of providing solution to tackle the problem we could not achieve our goal to reduce time in handling MapReduce jobs. Then comes the solution to store profiles but it was also not much efficient. Hence the INTERFACE proposed works well in this case.

III PROBLEM DEFINITION

Increase in the efficiency of Profile matching by comparing sampled Map Reduce job with profiles already present in INTERFACE rather than profile Store.

METHODOLOGY

IO FORMATTER is made key because it is a static feature and helps to a large extent to uniquely distinguish between jobs .MapReduce job matching is carried out in three steps:

Sampling – Sampling is only 10% profiling in which only 1 map task execution is done. In profiling applications are run with different sets of configuration parameters to collect utilization or execution profiles.

Profile Matching – To search for a job profile in INTERFACE to distinguish between jobs. After sampling of job, Js, its static feature (IO FORMATTER) is matched with that of key of profiles in the INTERFACE. IO FORMATTER is actually the key of profiles stored in INTERFACE. Fig 2 explains the block structure of matching and storing profiles using INTERFACE. If it is matched then data flow execution of Js and the profiles in INTERFACE is matched. If it matches, then only profile is taken for further execution while profiler is turned off. If profiles not matched in INTERFACE then Profile Store is checked.

Storage – After job profile is created, it is saved in INTERFACE as well as in Profile Store for further use.If INTERFACE limit exceeds then jobs are replaced in terms of least recently used.

3.1 ARCHITECTURE

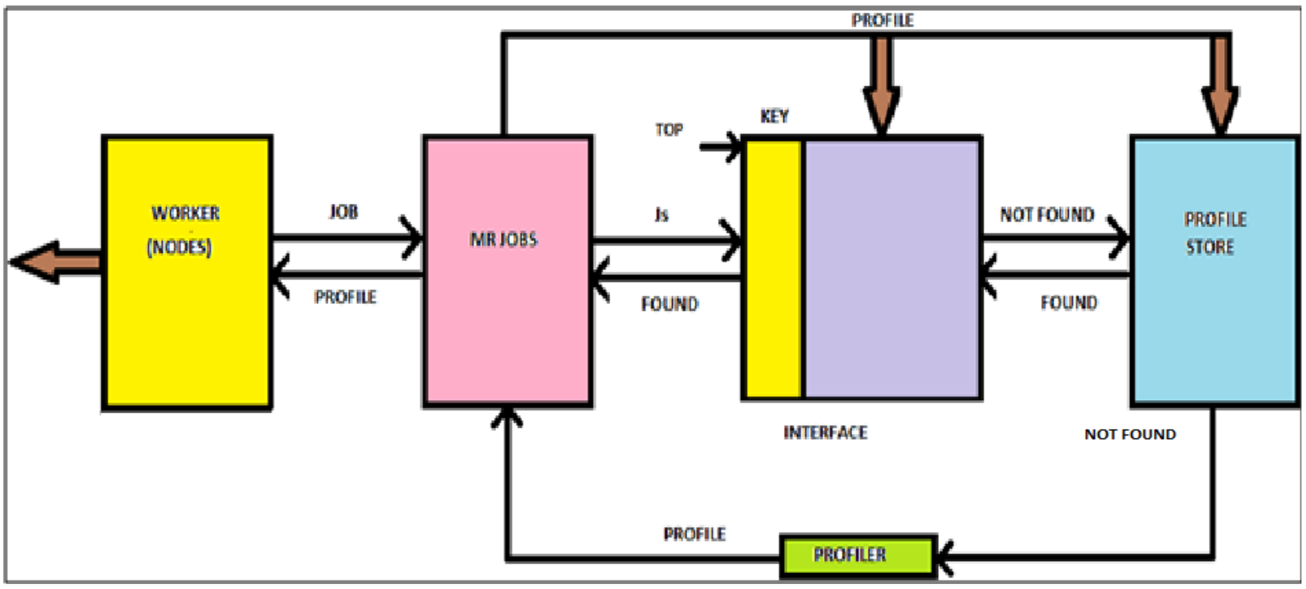

Fig 2 *Block diagram of matching and storing profiles using INTERFACE*

International Journal of Advances in Computer Science & Its Applications – IJCSIA
Volume 4: Issue 3 [ISSN 2250-3765]
Publication Date : 30 September,2014

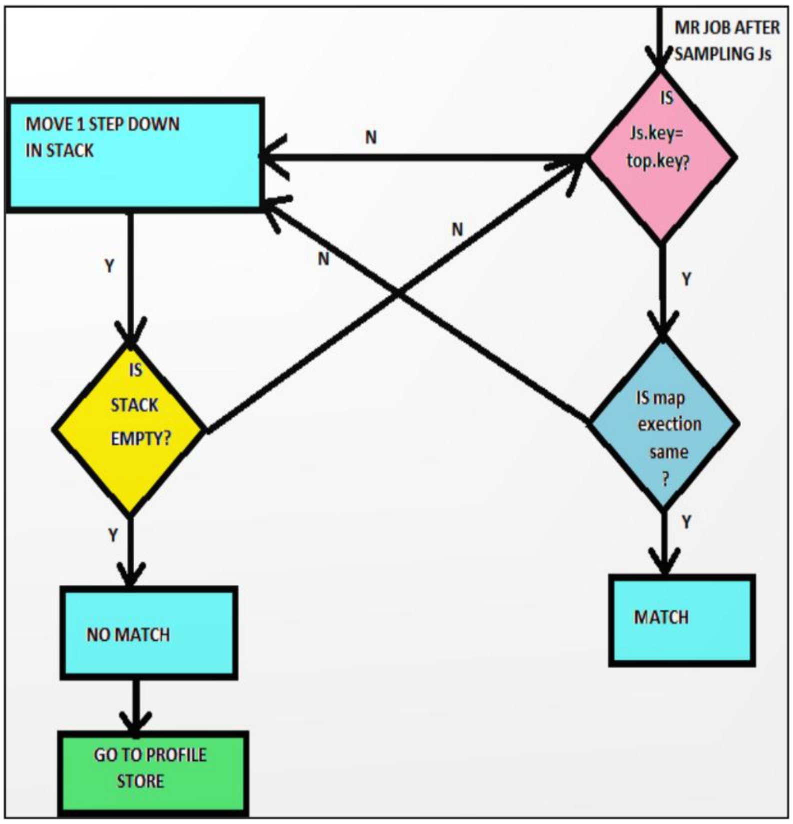

Fig 3*: Flowchart of matching profiles in INTERFACE*

3.2 ALGORITHM:

```
1.. //top is pointer to the topmost element in the INTERFACE
2.//initialize Top←start Address
3.//initialize Flag←1
4.For (; Top! =NULL; top++)
5.{
6.If (Js.IOFORMATTER=top.IOFORMATTER)
7.{
8. If (Js.Data flow execution=top.Data flow execution)
9.{
10. //profile match
11. Flag=1;
12.}}
13.Top++;
14.}
15.If (flag==0)
16.//Go to profile store
```

3.3 ANALYSIS

Hadoop has emerged as an inevitable competitor for current database system in last couples of years. As hadoop is leading existing systems in terms of performance. So reusing of profiles adds to its efficiency.

Using this approach has reduced the time to match and fetch already created profile. Now execution of this job is fast because there is no need of creating profile again for similar or equivalent job.

Good tuning is achieved for parameters using this INTERFACE. It has helped us to achieve efficiency at faster rate than earlier. Achievement is that, time used in matching profiles will be reduced.

IV. CONCLUSION

The proposed profile storing and matching scheme is excellent combination of static and dynamic features of job.

This scheme considers all the parameters that are required to create a profile instead of considering only CPU utilization parameters. Using this scheme, necessity for job profile creation is reduced by 25 % and increases the reusability of profile. It will increase the efficiency of the system with less energy usage.

The present scenario demands work to be done in less time which can be achieved using this proposed scheme.

REFERENCES

[1]. S.Babu, HHerodotou "Cost based optimization of configuration parameters and clustersizing for hadoop" US patent 20,130,254,196, 2013.

[2]. E.Jahani, MJCafarella, Christopher Re. "Automatic optimization for MapReduce programs", proceedings of the VLDB Endowment, 2011.

[3]. H Herodotou, H Lim, G luo, N Borisov, L Dong,"Starfish: A Self tuning System for Big Data Analytics", CIDR, 2011

[4]. F Dong," Extending Starfish to support the growing Hadoop Ecosystem", cs.duke.edu, 2012.

[5]. WC Kim, C Baek, D Lee,"Measuring the optimality of Hadoop optimization ", arxiv.org, 2013.

[6]. Sasiniveda.G, Revathi.N,"Performance tuning and scheduling of large data set analysis in MapReduce paradigm by optimal configuration using Hadoop", Proceedings of AICTE Sponsored National Conference on communication and Informatics [NCCI-2013], SVCE-page 70.

[8]. M Ead " PStorM: Profile Storage and Matching for Feedback-Based Tuning of MapReduce Jobs", test.cs.uwaterloo.caa, 2012.

[9]. Book "Understanding Big Data" by Chris Eaton, DirkDeroos, TomDeutsch, GeorgeLapis, PaulZikopoul

[10]. http://whatis.techtarget.com/definition/3Vs.

[11] .Hadoop mapreduce, http://hadoop.apache.org about hadoop

[12] .Hadoop-0.20.2 http://www.apache.org/dyn/closer.cgi/hadoop/core.

[13] .S. Babu, "Towards automatic optimization of MapReduce programs," presented at the 1st ACM symposium on Cloud computing, Indianapolis, Indiana, USA, 2010.

[14]. D. Jiang et al. Map-join-reduce: Towards scalable and efficient data analysis on large clusters. IEEE Transactions on Knowledge and Data Engineering, 2010

[15]. J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," Commun. ACM, vol. 51, pp. 107-113, 2008.

[16]. Matei Zaharias, Andy Konwinski, et al "Improving Map Reduce Performance in Heterogeneous Environments" IEEE Transactions on Parallel and distributed processing, Vol. 23, No. 19, April 2010.

[17]. Nikzad Babaii Rizvandi1,Albert Y. Zomaya , et.al " On Modeling Dependency between Map Reduce Configuration Parameters and Total Execution Time " IEEE Transactions on Distributed, Parallel, and Cluster Computing , Vol. 23, No. 9, March 2012

[18]. Kyong -Ha Lee, Hyunsik Choi "Parallel Data Processing with MapReduce: A Survey" International Journal of Engineering Research and Applications Vol. 40, No. 4 December 2011

[19]. Mr. Yogesh Pingle, Vaibhav Kohli, Shruti Kamat, Nimesh Poladia Big Data Processing using Apache Hadoop in Cloud System International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622.

[201]. Nikzad Babaii Rizvandi, Javid Taheri, Albert Y.Zomaya" On Using Pattern Matching Algorithms in MapReduce Applications" IEEE transaction on parallel and distributed processing, 2011.