PROCESSING LARGE / BIG DATA SET THROUGH MapR AND PIG

Arvind Kumar-Senior ERP Solution Architect / Manager, Derex Technologies, Inc.

Abstract : We live in the data age. It's not easy to measure the total volume of data stored electronically, but an IDC estimate put the size of the "digital universe" at 0.18 zettabytes in 2006, and is forecasting a tenfold growth by 2011 to 1.8 zettabytes.* A zettabyte is

1021 bytes, or equivalently one thousand Exabyte's, one million petabytes, or one billion terabytes. That's roughly the same order of magnitude as one disk drive for every person in the world.

MapReduce is a programming model for data processing. The model is simple, yet nottoo simple to express useful programs in. Hadoop can run MapReduce programs writtenin various languages, MapReduce programs are inherentlyparallel, thus putting very large-scale data analysis into the hands of anyone withenough machines at their disposal. MapReduce comes into its own for large datasets. MapReduce is a framework for performing distributed data processing using the MapReduce programming paradigm. In the MapReduce paradigm, each job has a user-defined map phase, which is a parallel, share-nothing processing of input; followed by a user-defined reduce phase where the output of the map phase is aggregated).

Pig raises the level of abstraction for processing large datasets. With MapReduce, thereis a map function and there is a reduce function, and working out how to fit your data processing into this pattern, which often requires multiple MapReduce stages, can be a challenge. With Pig the data structures are much richer, typically being multivaluedand nested; and the set of transformations you can apply to the data are much more powerful—they include joins, for example, which are not for the faint of heart in MapReduce.

Keywords: MapR, Big Data, PIG, HDFS. Hadoop

1. INTRODUCTION

Hadoop Map/Reduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner. A Map/Reduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the reduce tasks. Typically both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.Typically the compute nodes and the storage nodes are the same, that is, the Map/Reduce framework and the Distributed FileSystem are running on the same set of nodes. Thisconfiguration allows the framework to effectively schedule tasks on the nodes where data is already present, resulting in very high aggregate bandwidth across the cluster.The Map/Reduce framework consists of a single master Job Tracker and one slave Task Tracker per cluster-node. The master is responsible for scheduling the jobs'component tasks on the slaves, monitoring them.

Pig provides an engine for executing data flows in parallel on Hadoop. It includes alanguage, Pig Latin, for expressing these data flows. Pig Latin includes operators formany of the traditional data operations (join, sort, filter, etc.), as well as the ability forusers to develop their own functions for reading, processing, and writing data.Pig is an Apache open source project. This means users are free to download it as sourceor binary, use it for themselves, contribute to it, and—under the terms of the Apache License—use it in their products and change it as they see fit.Pig runs on Hadoop. It makes use of both the Hadoop Distributed File System,HDFS, and Hadoop's processing system, MapReduce.HDFS is a distributed filesystem that stores files across all of the nodes in a Hadoopcluster. It handles breaking the files into large blocks and distributing them acrossdifferent machines, including making multiple copies of each block so that if any onemachine fails no data is lost. It presents a POSIX-like interface to users. By default, Pigreads input files from HDFS, uses HDFS to store intermediate data between MapReducejobs, and writes its output to HDFS

In this paper, we illustrate the use of a methodology that we feel would facilitate and support the combined use of MAPR and PIG. We do a case study and illustrate how both methodologies can be used in industries.

The remainder of this article is organized as follows. Section 2 talks about our research objective. Section 3 gives a brief background of MapR and the PIG. Section 4 talks about MapR and PIG framework, components and Hadoop. Section 4 discusses our proposed methodology and Section 5 talks about how to install MapR and Pig in systems describes a case study where we applied our methodology.

Section 6 talks about our methodology and difference between PIG and MapR. Section 7 talks about conclusion section...

2. RESEARCH OBJECTIVE

Researchers have tried find the best usability of both methodologies i.e. MapReduce and PIG in big data analysis. There are several methodologies available, but so far there is no single one methodology available, which satisfy all requirements of complete big data handling. The primary goal of big data analytics is to help companies make better business decisions by enabling data scientists and other users to analyze huge volumes of transaction data as well as other data sources that may be left untapped by conventional business intelligence (BI) programs. These other data sources may include Web server logs and Internet clickstream data, social media activity reports, mobile-phone call detail records and information captured by sensors. Some people exclusively associate big data and big data analytics with unstructured data of that sort, but some consulting firms also consider transactions and other structured data to be valid forms of big data. Researchers tried to find scenarios, whereMapReduce methodologies can be used, and in which scenarios PIG is suitable. Furthermore, our aim is to create single tool which can satisfy both requirements of MapReduce and PIG

3. BACKGROUND

3.1 MapR

MapReduce is a framework for performing distributed data processing using the MapReduce programming paradigm. In the MapReduce paradigm, each job has a user-defined map phase, which is a parallel, share-nothing processing of input; followed by a user-defined reduce phase where the output of the map phase is aggregated). Typically, HDFS is the storage system for both input and output of the Hadoop MapReduce is a programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster. Seen from a high level, MapReduce is really two things:

1. A strategy or model for writing programs that can easily be made to process data in parallel.
2. A framework that runs these programs in parallel, automatically handling the details of division of labor, distribution, synchronization, and fault-tolerance.

In the map-reduce programming model, work is divided into two phases: a map phase and a reduce phase. Both of these phases work on key-value pairs.

3.2 PIG

Pig started out as a research project in Yahoo! Research, where Yahoo! scientists designedit and produced an initial implementation. Researchers felt that the MapReduce paradigm presented byHadoop “is too low-level and rigid, and leads to a great deal of custom user code thatis hard to maintain and reuse.” At the same time they observed that many MapReduceusers were not comfortable with declarative languages such as SQL. Thus they set out to produce “a new language called Pig Latin that has been designed to fit in a sweet spotbetween the declarative style of SQL, and the low-level, procedural style of MapReduce.” Yahoo! Hadoop users started to adopt Pig. So, a team of development engineers wasassembled to take the research prototype and build it into a production-quality product. About this same time, in fall 2007, Pig was open sourced via the Apache Incubator. The first Pig release came a year later in September 2008. Later that same year, Piggraduated from the Incubator and became a subproject of Apache Hadoop. Early in 2009 other companies started to use Pig for their data processing. Amazon alsoadded Pig as part of its Elastic MapReduce service. By the end of 2009 about half ofHadoop jobs at Yahoo! were Pig jobs. In 2010, Pig adoption continued to grow, andPig graduated from a Hadoop subproject, becoming its own top-level Apache project.

Apache Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets. At the present time, Pig's infrastructure layer consists of a compiler that produces sequences of Map-Reduce programs, for which large-scale parallel implementations already exist (e.g., the Hadoop subproject). Pig's language layer currently consists of a textual language called Pig Latin. Pig is made up of two pieces:

- The language used to express data flows, called Pig Latin.
- The execution environment to run Pig Latin programs.

A Pig Latin program is made up of a series of operations, or transformations, that areapplied to the input data to produce output. Taken as a whole, the operations describea data flow, which the Pig execution environment translates into an executable representationand then runs. Under the covers, Pig turns the transformations into a seriesof MapReduce jobs, but as a programmer you are mostly unaware of this, which allowsyou to focus on the data rather than the nature of the execution.

4. FRAMEWORK and COMPONENTS of MapR and PIG

4.1 MapR Framework and components

A MapReduce program typically acts something like this:

1. Input data, such as a long text file, is split into key-value pairs. These key-value pairs are then fed to your mapper. (This is the job of the map-reduce framework.)

2. Your mapper processes each key-value pair individually and outputs one or more intermediate key-value pairs.
3. All intermediate key-value pairs are collected, sorted, and grouped by key (again, the responsibility of the framework).
4. For each unique key, your reducer receives the key with a list of all the values associated with it. The reducer aggregates these values in some way (adding them up, taking averages, finding the maximum, etc.) and outputs one or more output key-value pairs.
5. Output pairs are collected and stored in an output file (by the framework).

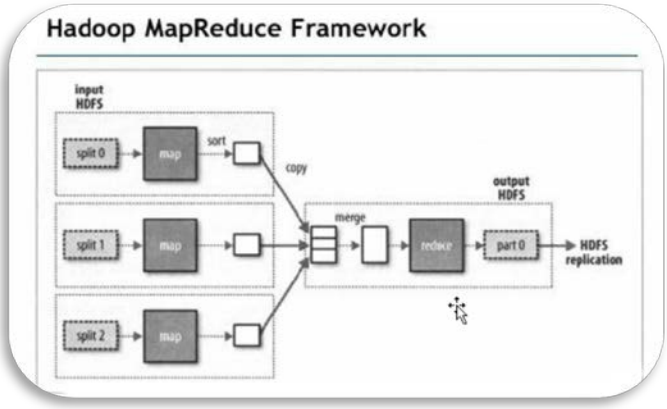

Fig1-Framework of MapReduce

4.2 Architectureof MapR

Dedicated master processidentifies worker processes/machines for map and reduce
•Master partitionsinput file into M partitions
•Partitions assignedto map workers
•Map workers output to R fileson local hard disks (by hash code), master notified
•Each reduce worker readsoneoutput file from the map workers for each input partition(by RPC) & sortsthem (many output keys per file!)
•Each reduce worker aggregatesdata per key

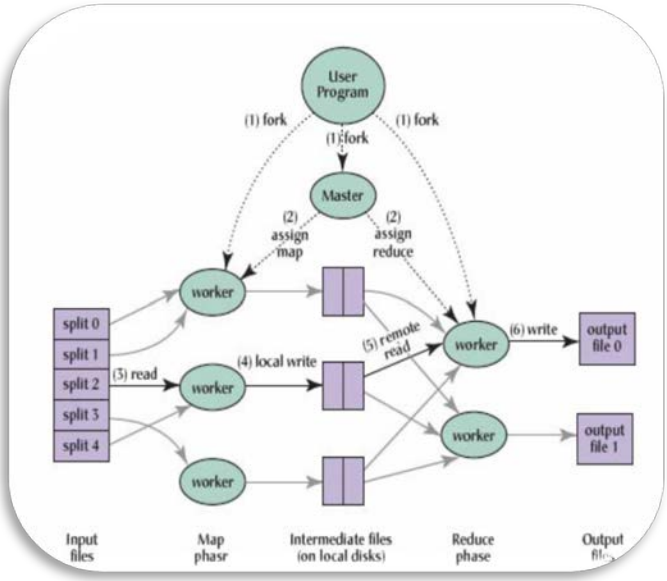

Fig2Architecture of MapReduce

4.3 Hadoop MapReduce

Hadoop MapReduce is a programming model for data processing. It follows the Master/Slave architecture. In Hadoop MapReduce, Job Tracker is the master process whereas TaskTrackers is the slave process. These are the processes which are used to process the data present in HDFS. Job tracker is running under NameNode machine whereas TaskTrackers run on Data Nodes.
MapReduce Daemons
Job Tracker and Task Tracker
• Job Tracker is the master of the system which manages the jobs and resources in the cluster (TaskTrackers). The Job Tracker tries to schedule each map as close to the actual data being processed i.e. on the TaskTrackers which is running on the same Data Node as the underlying block.
• TaskTrackers are the slaves which are deployed on each machine. They are responsible for running the map and reduce tasks as instructed by the Job Tracker.

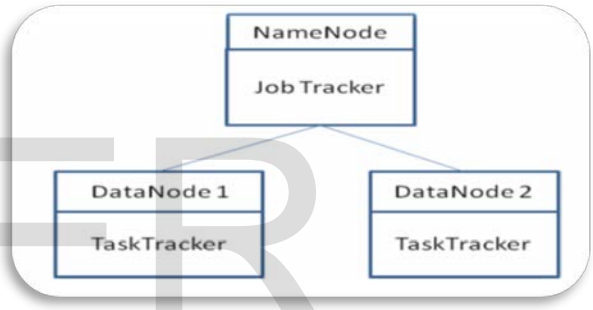

Fig3-Job Tracker and Task Tracker

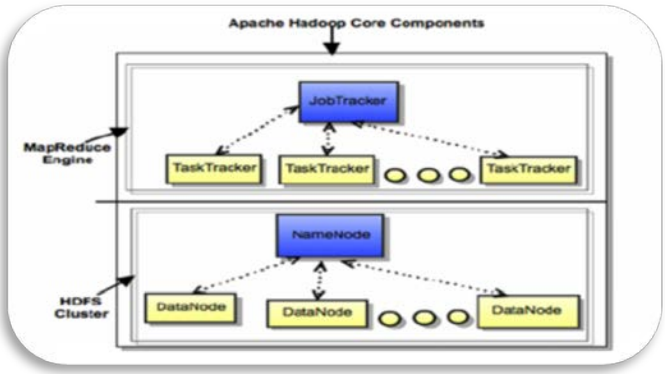

Fig4-Hadoop Daemons (HDFS and MapReduce Daemons)

4.4 PIG Components and framework

PIG Framework
As part of your development, you will want to test your Pig Latin scripts. Even oncethey are finished, regular testing helps assure that changes to your UDFs, to your scripts,
Or in the versions of Pig and Hadoop that you are using do not break your code.Pig Unit provides a unit-testing

framework that plugs into JUnit to help you write unittests that can be run on a regular basis. Pig Unit was added in Pig 0.8.

Let's walk through an example of how to test a script with Pig Unit. First, you need ascript to test: Second, you will need the pigunit.jar JAR file. This is not distributed as part of thestandard Pig distribution, but you can build it from the source code included in yourdistribution. To do this, go to the directory your distribution is in and type ant jarpigunit-jar. Once this is finished, there should be two files in the directory: pig.jar andpigunit.jar. You will need to place these in your class path when running Pig Unittests. Third, you need data to run through your script. You can use an existing input file, oryou can manufacture some input in your test and run that through your script. We willlook at how to do both.

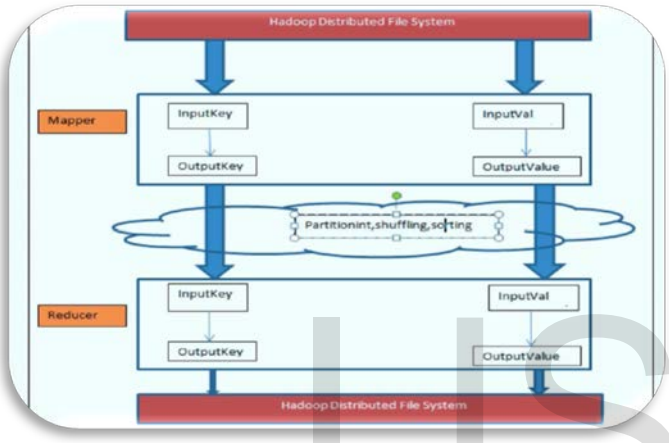

Fig5-framework of PIG on Hadoop

PIG Components

• Pig Latin
Command based language
Designed specifically for data transformation and flow expression
• Execution Environment
The environment in which Pig Latin commands are executed
Currently there is support for Local and Hadoop modes
• Pig compiler converts Pig Latin to MapReduce
Compiler strives to optimize execution
You automatically get optimization improvements with Pig updates

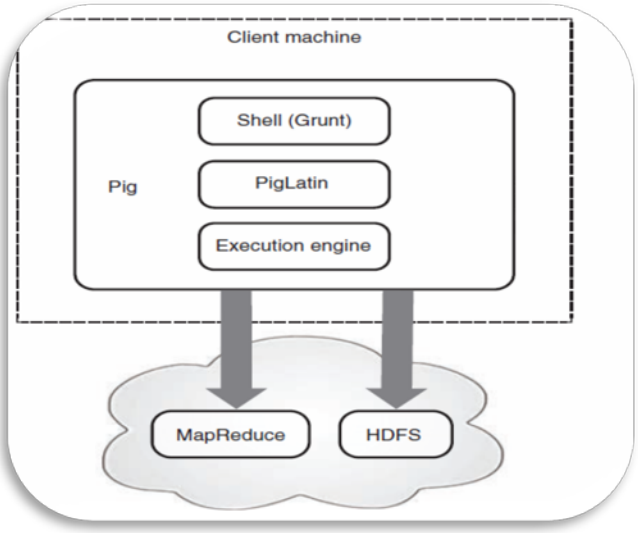

Fig 6 –PIG Components

4.5 HADOOP Mode

Use map reduce mode you first had to check that the version of pig you downloaded is compatible with the version of Hadoop. Pig releases will only work against particular version of Hadoop. This is documented in the release notes.
We can work with pig in the following modes:
1) You can run pig in inter active mode using Grunt shell, invoke the Grunt shell using the "pig" command and enter your pig Latin statements and pig commands interactively at the command line.
2) Script mode: - In case of script mode we are using editor to write script (we are writing script outside of pig (i.e. Linux)
• Vi input.pig
• Pig-x local input.pig(local Data)
• Pig input.pig(Map Reduce mode

In case of embedded mode we are creating UDF'S using java code in the form of jar file. We can use in pig.
• Pig Latin statements extract all user 10's from /etc/password file. First copy the /etc/password file to your local working directory.Next, invoke the shell by typing the "pig" command (in local or Hadoopmode).Then, enter the pig Latin statements interactively at the grunt prompt (be sure to include the semicolon after each statement.

grunt>A =load 'password' using pig Storage(':')
grunt>B= for each a generate $0 as id grunt>dump

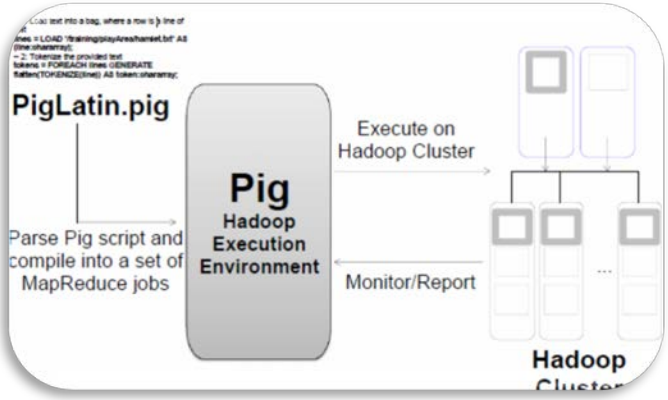

Fig7-PIG HADOOP Mode

5. INSTALLATION OF MapR and PIG

5.1 Creating AMapReduce Project

Steps To Create New Project Using Eclipse:
Go To Applications Menu → Select Programming Menu Item → Select Eclipse
Steps To Implement Application Using Eclipse:
Applications → Programming → Eclipse
Step 1: To Start New Application
Go To File Menu → New → Java Project
In the dialogue window which pops up, enter a 'Project Name' and click on Finish button.

Example:

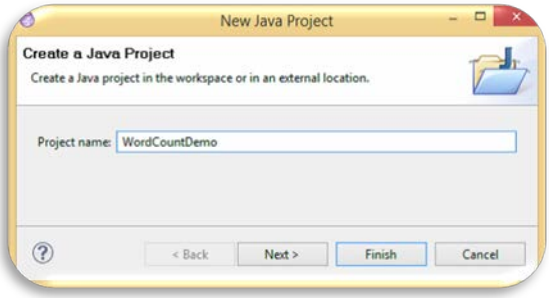

Fig8-Intallation of MapR

Click on Finish button.
Step 2: To Add Class To Project
Go To Package Explorer → Right Click On Project Name → In The Pop Up Menu → Select New → Select Class
Now, It Will Ask Class Name.
Example:

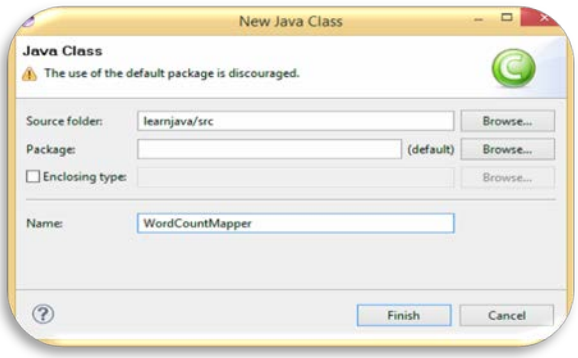

Fig9-Adding class to Project

Steps To Add Hadoop Jar Files To Application:
Step 1: Go To Package Explorer → Right Click On Project → Select Build Path → Configure Build Path → In The Pop-up Dialog Window → Click On Add External Jars → Select Hadoop Folder From The Dialog Box Window → Select All Jar Files And Click On Open Button
Example:
• Hadoop-ant-20.2-chd305.jar
• Hadoop-core-cdb30 5.jar
Jar → Java Archive

After Implementing The Complete Project, It Must Be Created In Jar Files.

To Create The Jar Files:

Go To Package Explorer → Right Click On Java Project → Select Export Option → Select Jar File → Click On Next Button This Path Can Be Any An Existing Path Followed By The Jar File Name.

Example: Path: /home/bigdataminds/desktop/anyfilename.jar
Click on finish button.

MapReduce API Configuration
It is a predefined class which is present in org.apache.hadoop.conf package. It is used to retrieve the Configuration class properties & values. Each property is named by a string & type of the value which may be one of the several types (such as boolean, int, long, etc.)

To read the properties, create an object of type configuration class:

Example: Configuration conf=new Configuration ();
Conf.addResource("xml file path");

addresource(): This is a member function of Configuration class. It takes xml file as a parameter.
Example: conf. addResource("core-site.xml")

To Get Value Of A Property:
Syntax: Variable name=conf.get("property name");
Configuration.addDefaultResource("coresite.xml");
It is a static method of Configuration class.

5.2 Installation PIG

This is the official version of Apache Pig. It comes packaged with all of the JAR filesneeded to run Pig. It can be downloaded by going to Pig's release page.Pig does not need to be installed on your Hadoop cluster.
It runs on the machine fromwhich you launch Hadoop jobs. Though you can run Pig from your laptop or desktop,in practice, most cluster owners set up one or more machines that have access to theirHadoop cluster but are not part of the cluster (that is, they are not data nodes or tasknodes). This makes it easier for administrators to update Pig and associated tools, aswell as to secure access to the clusters. These machines are called gateway machines oredge machines.
You will need to install Pig on these gateway machines. If your Hadoop cluster is accessiblefrom your desktop or laptop, you can install Pig there as well. Also, you caninstall Pig on your local machine if you plan to use Pig in local mode.

The core of Pig is written in Java and is thus portable across operating systems. Theshell script that starts Pig is a bash script, so it requires a UNIX environment. Hadoop,which Pig depends on, even in local mode, also requires a UNIX environment for itsfilesystem operations. In practice, most Hadoop clusters run a flavor of Linux. ManyPig developers develop and test Pig on Mac OS X.11

Pig requires Java 1.6, and Pig versions 0.5 through 0.9 require Hadoop 0.20. For futureversions, check the download page for information on what version(s) of Hadoop theyrequire. The correct version of Hadoop is included with the Pig download.

If you planto use Pig in local mode or install it on a gateway machine where Hadoop is not currentlyinstalled, there is no need to download Hadoop separately.Once you have downloaded Pig, you can place it anywhere you like on your machine,as it does not depend on being in a certain location.

To install it, place the tarball inthe directory of your choosing and type:tar xzf filenamewhere filename is the TAR file you downloaded.The only other setup in preparation for running Pig is making sure that the environmentvariable JAVA_HOME is set to the directory that contains your Java distribution. Pig willfail immediately if this value is not in the environment.

6. OUR PROPOSED METHODOLOGY

As stated earlier, our goal is to create a methodology through Which both workflow and goal models can be analyzed.
One of the key aspects of defining this methodology is to first have a process model which captures all relevant information as outlined below:

- Map complex and straight requirements in MapR
- Logic developments only needed when business requirements are complex and can't be driven by standard tables and Queries
- HDFS is central for storing and processing data
- Join, filters makes straight requirements, easy to implement
- If requirements can be mapped with standard tables and queries, use PIG.
- MapR is good for programming logic where requirements are complex and cannot be mapped without custom development.
- PIG will be good for analysts, data scientists and statisticians.

Fig 10-Proosed methodologies for MapR and PIG

6.1 How PIG Differs Over MapReduce

Pig provides users with several advantages over using MapReduce directly. Pig Latinprovides all of the standard data-processing operations, such as join, filter, group by,order by, union, etc. MapReduce provides the group by operation directly (that is whatthe shuffle plus reduce phases are), and it provides the order by operation indirectlythrough the way it implements the grouping. Filter and projection can be implementedtrivially in the map phase. But other operators, particularly join, are not provided andmust instead be written by the user. Pig provides some complex, nontrivial implementations of these standard data operations.For example, because the number of records per key in a dataset is rarely evenlydistributed, the data sent to the reducers is often skewed. That is, one reducer will get10 or more times the data than other reducers. Pig has join and order by operators that will handle this case and (in some cases) rebalance the reducers. But these took the Pigteam months to write, and rewriting these in MapReduce would be time consuming.In MapReduce, the data processing inside the map and reduce phases is opaque to thesystem. This means that MapReduce has no opportunity to optimize or check theuser'scode. Pig, on the other hand, can analyze a Pig Latin script and understand the dataflow that the user is describing. That means it can do early error checking (did the usertry to add a string field to an integer field?) and optimizations (can these two groupingoperations be combined?).MapReduce does not have a type system. This is intentional, and it gives users theflexibility to use their own data types and serialization frameworks. But the

downsideis that this further limits the system's ability to check users' code for errors both beforeand during runtime.All of these points mean that Pig Latin is much lower cost to write and maintain thanJava code for MapReduce.

7. Conclusion

A Test Run was conducted on an Automobile company, which specializes in making heavy duty trucks. Company generated large amount of big data though their complex business. Company had performance issues, and researchers mapped both methodologies in this company.

As mentioned in our approach, company implemented both methodologies because of their complex data. Research and implementation helped company make better business decisions by enabling data scientists and other users to analyze huge volumes of transaction data as well as other data sources that may be left untapped by conventional business intelligence (BI) programs.The company has a clearly defined workflow model for the tendering process that is mostly done in ERP and other customer and third party tools. Company had diverse system and interfaces all over places.

We started work on the PIG implementation in order to map business case with solutions. Once we started interviewing the various stakeholders, we found that several areas and department'srequirementscouldn't not be addressed through PIG implementation, as retrieving and fetching data from custom solutions and tables was complex, time consuming and error prone. Those complex KPIs and business case could not be mapped through PIG, and we recommended MapR to address those requirements. We designed Mapper and Reducers job to implement those requirements. Custom logic were written on requirements which were complex, and PIG methodology was used where requirements could be mapped with standard tables and queries.

.

REFERENCES

[1] Programming PIG, by Alan Gates, Oct 2011 edition
[2] T. White, Hadoop: The Definitive Guide. O'Reilly Media, Yahoo! Press,
[3] June 5, 2009. Reference from Hadoop Apache.org
[4] Reference from hadoop.apache.org/docs
[5] J. Dean, S. Ghemawat, "MapReduce: Simplified Data Processing on
[6] Large Clusters," In Proc. of the 6th Symposium on Operating Systems
Design and Implementation, San Francisco CA, Dec. 2004.
[7] A. Gates, O. Natkovich, S. Chopra, P. Kamath, S. Narayanam, C.
Olston, B. Reed, S. Srinivasan, U. Srivastava. "Building a High-Level
Dataflow System on top of MapReduce: The Pig Experience," In Proc.
[8] S. Weil, S. Brandt, E. Miller, D. Long, C. Maltzahn, "Ceph: A Scalable,
High-Performance Distributed File System," In Proc. of the 7th
Symposium on Operating Systems Design and Implementation, Seattle,WA, November 2006.
[9] Hadoop Handbook by Veer A Nagaraju
[10] Company's business case