Research Journal of Applied Sciences, Engineering and Technology 12(9): 900-906, 2016
DOI:10.19026/rjaset.12.2807
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: September 21, 2015 Accepted: January 18, 2016 Published: May 05, 2016

Research Article

An Extended Form of MATLAB To-map Reduce Frameworks in HADOOP Based Cloud Computing Environments

[1]T. Tamilvizhi, [2]B. Parvatha Varthini, [3]K. Manoj and [4]R. Surendran
[1]Research Scholar, Faculty of Computing, Sathyabama University, Chennai, Tamil Nadu, India
[2]Professor and Dean, St. Joseph's College of Engineering, Chennai, Tamil Nadu,
[3]Faculty of Computing, Sathyabama University, India
[4]Asst. Professor (IST), Sur University College, Sultanate of Oman

Abstract: Aim of study to extend the implementation of Matlab to Mapreduce translation based on the M2M translation technique. Cloud computing is a service which provides services by handling massive amount of data. To handle it effectively it needs some technology like Hadoop. Hadoop is an open source project written in java. It is optimised to handle massive amount of data (structured, unstructured, semi-structured) through parallelism. Thus to achieve this parallelism Hadoop Distributed File System (HDFS) uses Mapreduce as a programming index. Here proposing a translator which converts Matlab commands to Mapreduce commands especially concentrated in executing some basic commands in Mapreduce environment to access large datasets. Matlab is a very effective tool for numerical computing since executing this command in a platform independent, distributed environment makes it more efficient.

Keywords: Cloud computing, hadoop, mapreduce, matlab, parallel computing

INTRODUCTION

Mapreduce is a programming model which is used to handle massive amount of data through parallel computing. Design to process huge datasets for certain kind of distributable problems using a large number of nodes. From the Fig. 1 eight major steps of Mapreduce model has been explained by White (2011):

- The Mapreduce program tells the JobClient to run a MapReduce Job by Tamilvizhi *et al*. (2015).
- Then message will be sent to JobTracker which creates a unique id for each job.
- The Job Client is responsible for copying the Jar file with java code which has been written to implement Map or reduce function in to a centralized shared file system for e.g., Hadoop Distributed File System (HDFS).

After this file sharing the Jobclient will tell the Job Tracker to start the job:

- The Job Tracker does its own initialization for the job. It calculates the no.of.splits to split the mapper jobs.
- It retrieves these "input splits" from the distributed file system.

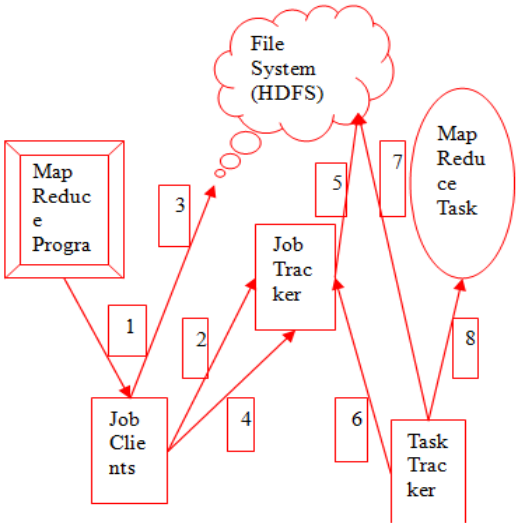

Fig. 1: Workflow of Mapreduce

- The Task Trackers are continuously sending heartbeat messages to the Job Tracker. Now that the Job Tracker has work them, it will return a map task or a reduce task as a response to the heartbeat.
- The Task Trackers need to obtain the code to execute, so they get it from the shared file system.

Corresponding Author: T. Tamilvizhi, Research Scholar, Faculty of Computing, Sathyabama University, Chennai, Tamil Nadu, India

- Then they can launch a java virtual Machine with a child process running in it and this child process runs your map code or reduce code are provided.

M2M translation technique follows the following procedure was provided by Thusoo *et al*. (2009).

The first stage is a lexical analyzer, used to the token generation. The scanner begins the analysis of the Matlab source code by reading the input, character by character and grouping characters into meaningful words and symbols (tokens) defined by a grammar of regular expressions was implemented by Chen *et al*. (2013).

The second stage is a syntax analyzer, used for parsing. Given a formal syntax specification (typically as a Context-Free Grammar (CFG)), the syntax analyzer reads tokens and groups them into units as specified by the productions of the CFG being used. Here, top-down parsing is adopted in our method was developed by Olston *et al*. (2008).

The third stage is a sematic analyzer (translator), used for semantic parsing or analysis. It is working out the implications of the expression just validated and taking the appropriate action. Here, aiming at the Matlab operations, a simple Math Operation Library based on MapReduce (MOLM) is built. When the translator processes a math command, it connects to the MOLM, gets the corresponding MapReduce code and combines with a main function. It is worth noticing that the same commands only need one MapReduce code was designed by Zhang *et al*. (2013).

Objective of this study is to implement the Matlab operation in the Mapreduce environment specifically to access a dataset with the functions available in Matlab. By using the M2M technique above mentioned the Matlab operations have been implemented in the mapreduce environment to access a dataset.

LITERATURE REVIEW

Generally cloud computing is simply defined as a service by Wang *et al*. (2010). It handles massive amount of data virtually. To handle such kind of massive, unstructured, semi-structured data efficiently it needs technology. So Hadoop is such technology which helps handling data efficiently and it uses MapReduce programming model to index the data, data in the sense dataset provided by Deyhim (2013). Majority of the Mapreduce codes have been written to index the data effectively. To index the data it needs some operations or methods that can help in building the efficiency of the code. For example a word count program in Mapreduce is used to know the details about the number of words present in a document. To handle a dataset which is totally unstructured need operations recently facebook invented Hive which is declarative language which uses SQL queries to handle datasets easily was proposed by Lee *et al*. (2011). This study proposes a service which converts some of the basic operations with loop commands to operate the dataset in the Mapreduce environment.

There are several languages which have been converted to Mapreduce code to increase the productivity of Mapreduce explained by Chen *et al*. (2013). Table 1 explains the survey of the X-to-Mapreduce concepts which has been done previously to enhance efficiency of both Matlab and Mapreduce environments.

Table 1: Survey of related works

Previous works	Merit	Demerit
Pig Latin: Foreign language for data processing	Handle data analysis and unstructured data	Support only external functions and safe optimizer not good
YSmart: Yet another SQL to Map reduce translator	Reduce redundant computation, IO operations and network transfer compared to existing translators	Merging multiple types of jobs causes overload and restriction in the input jobs
Preliminary results from a parallel MATLAB compiler	Solve large level problem	Implement small number of MATLAB Functions, Due to the complexity in large datasets processing can't enable library functions
Hive- A warehousing solution over a MAP-reduce framework	Used for data exploration and query optimization	Optimizer with limited number of rules
Jaql: A scripting language for large scale semistructured data analysis	Reusability, Higher level of abstraction , Used for intranet data analysis and log processing	Error handling not effective No Physical transparency No adaptive and robust optimization
Automatic conversion of functional sequences to mapreduce with dynamic path selection	Provides mapping for functional codes that effectively translates in to mapreduce codes.	No Input dependent algorithms Tokens vary for different mediums from java
A Matlab to modelica translator	Converts dynamically typed modelica language to help in analysis	Execution time taken to compile Matlab code is longer than the ordinary compilation
Pydoop: A python mapreduce and HDFS API for Hadoop	Framework is similar to java and access to more mapreduce components	No property access for Keys/values. No python style iterations
A Matlab to fortran 90 translator and its effectiveness	FALCON's Matlab compiler is 1000 times faster than Matlab	No parallelizing compiler No high level data distribution with the semantics of array lanuage
M2M: A simple Matlab to mapreduce translator for cloud computing	Provides environment to access matlab commands in a distributed , platform independent environment	Only basic commands are accessible Limited Library functions

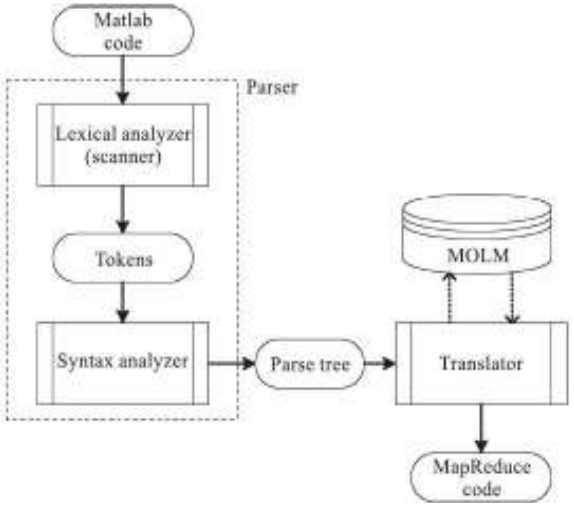

Fig. 2: Workflow of M2M translator

Declarative languages like SQL has been converted to Mapreduce to increase the performance and to handle unstructured was provided by data Zhang *et al*. (2013). Like this Mapreduce and Hadoop has some advantages like platform independent, secure, distributive environment and also it is used handle unstructured data where in any of the present technologies can't afford its attributes to handle such massive amount of unstructured data. So it has the option of X-to-Mapreduce. The languages can use the properties or advantages of this technology to improve its efficiency. Recently yahoo developed Pig which is a dataflow language and facebook developed HIVE which is a declarative language So let us see some of the high level languages that has been converted to Mapreduce codes to use its full of its merits.

Hive is a technology developed at facebook that turns haadoop into a data warehouse complete with a dialect of SQL for querying. Being a SQL dialect, HiveQL is a declarative language.It figures out how to build a dataflow to achieve it. The schema is required, but you are not limited to one schema. HiveQl on its own is a relationally complete language but not a turing complete language. It can be extended to make it has a turing complete language by Thusoo *et al*. (2009).

Jaql is developed at IBM and it is a dataflow language and its native data structure format is Javascript Obeject Notation or JSON, Schemas are optional and the jaql language itself is Turing complete on its own. It's a JSON-based query language translates in to hadoop Mapreduce jobs. JSON is the data interchange standard that is human-readable like XML but it designed to be lighter-Weight was developed by Beyer *et al*. (2011).

Pig was developed at Yahoo Research Around 2006 and moved into the Apache Software Foundation in 2007. Pig's language, called PigLatin, is a dataflow language-This is the kind of language in which you program by connecting things together. Pig can operate on complex data Structures, even those that can have levels of nesting. Unlike SQL, pig does not require that the data have a schema, so its well suited to processing unstructured data. However, pig can still leverage the value of a schema if you choose to supply one. Like SQL PigLatin is relationally complete, which means it is at least as powerful as relational algebra. Turing completeness requires looping constructs, an infinite memory model and Conditional constructs. PigLatin is not Turing complete on it s own, but is Turing complete when extended with User-Defined Functions was provided by Olston *et al*. (2008).

Dumbo is a package or library of hadoop used for Python.Dumbo is clear mapping to the java. It decreases the amount of code need to be written for use of map reduce .

Pipes is a C++ library which reduces the amount of code to be used due to the usage of C++ templates. This project reduces the amount of code to be written and It does not stress on the fact of writing the Separate Hadoop code since it does the conversion automatically.

This research is the initiative of bringing together two extremely powerful tools Mapreduce and Matlab. Matlab is software that offers a technical computing environment. It is used frequently by researchers in many fields for numerical manipulations, simulations and data processing. Mapreduce is programming model which is used to index large amount of data using parallel computing in the hadoop environment. In Deyhim (2013), Mapreduce is an emerging powerful environment which is written in java so it has some benefits like it can be compiled in any type of operating system. It provides security and more often an distributive environment. This M2M has initiated to invoke Matlab commands to Mapreduce using M2M compiler.

Figure 2 explain the compiling and the translation process of M2M and it undergoes translation through three aspects:

- Translate single matlab commands to Mapreduce code
- Translate multiple matlab independent commands to Mapreduce code
- Translate multiple Matlab commands to Mapreduce code.

The very first stage is lexical analyser which is used to read Matlab codes and to generate tokens by reading all the characters and grouping them in to meaningful tokens and words defined by regular expression grammar. Second level is syntax analyser which is used for context free grammar (parsing). Third level is semantic analyser which is responsible for delivering the parsed to code to MOLM (Math operation library based on Mapreduce).

Single input commands are easily converted to Mapreduce code. The Matlab commands like Min which returns the smallest elements along different dimensions of an array. M2M helps in allocating a single job for single command min in the Run function by Surendran and Samhitha (2014). Then by combining all the hadoop packages and imports, the corresponding hadoop Mapreduce code is generated. In the second level M2M only translates to Mapreduce codes and generates the Hadoop jobs one-by-one and do not care about the relations between jobs was designed by Quinn *et al*. (1998).

MATERIALS AND METHODS

The previous M2M translator is used to handle the operations parallel without dependency between each other. They were doing parallelization with the jobs created with help of Matlab basic commands. These basic commands like min, max, sum were handled as jobs. Operations which have dependency are controlled with the help of job control and the operation which doesn't have dependency is accessed parallel.

How Mapreduce works with loop commands: To understand Mapreduce, we need to break into its components operations map and reduce. Both of these operations come from functional programming languages. These are languages that let you pass function as arguments to other functions for loop. Want to double every element in an array. Figure 3 clearly explains how the loop commands have been processed as explained before.

```
Var a = [1, 2, 3];
For(i = 0;i<a.length;i++)
a[i] = a[i]*2;
```

The variable "a" enters the For loop as [1, 2, 3] and comes out as [2, 4, 6]. Each array element is mapped to a new value that is double the old value. The body of the for loop, in which does the doubling, can be written as a function. Define fn as a function that returns its argument multiplied by 2.

```
Var a = [1, 2, 3];
For (i = 0; i<a. length; i++)
a[i] = fn(a[i]);
The value a is
Var a = [2, 4, 6];
```

This will allow us to generalize this code. Instead of only being able to use this code to double numbers could use it for any kind of map function.

```
Var a = [1, 2, 3];
For(i = 0;i<a.length;i++)
a[i] = fn(a[i]);
```

Call this function "map" and pass the function fn as an argument to map.

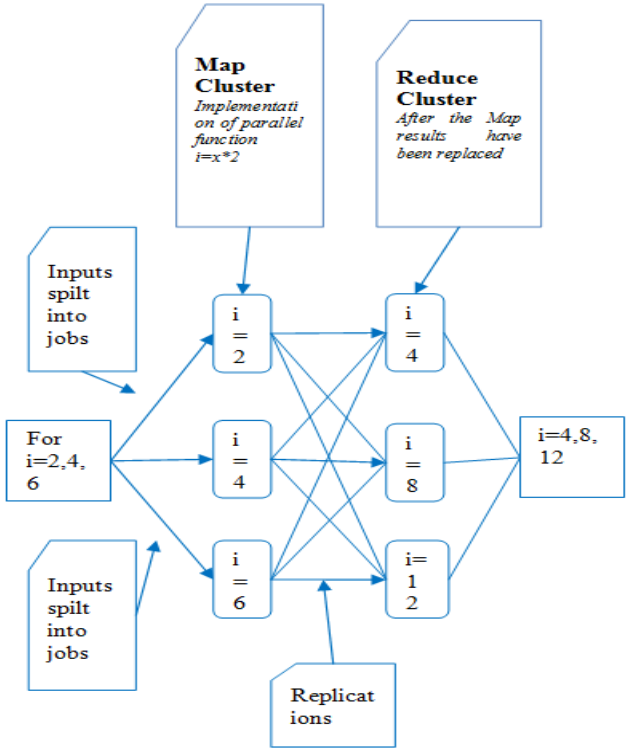

Fig. 3: Simple workflow of for loop in Mapreduce

```
Function map (fn,a)
{
For(i = 0;i<a.length;i++)
A[i] = fn(a[i]);
}
```

Map function:

```
map(function(x),{return x*2;},a);
```

Now have a general function named map and can pass our "multiply by 2" function as an argument. Writing the function definition in one statement is a common idiom in functional programming languages.

Thus rewrite for loop as a map operation taking a function as an argument. If decided to add this parallelism to the original program, we would need to rewrite the whole program. But if we wanted to parallelize the program written as a call to map, we wouldn't need to change our program at all. However M2M (Matlab-Mapreduce) converts some basic commands it will not be more efficient with those basic commands. Loop command in matlab play a very major role in accessing large datasets and in number of numeric calculations. Thus finding the loop commands inside the Matlab code and converting it to Mapreduce code provide more efficiency in accessing Matlab code in a parallel, distributive and platform independent environment. For and While are the two loop commands helps in accessing several important operations in Matlab.

Loops are generally used to iterate a condition for specific set of values until it is satisfied. In Matlab loops are of two types one is for….end construct or while…end construct. Both are exclusively used for repeating a series of calculation.

For Loop Syntax
Loop counter incremented by one:

```
For i = startvalue:endvalue
X = ….
Y = ….
.
.
End
```

i is the index where intial value starts with start value and end with end value. When i>endvalue then loopexits through end keyword:

Eg 1: The squared root of 10 integers

```
for i = 1:10
fprintf('the squared no are',i,sqrt(i));
end
```

Inccrementation in For Loop
Syntax

```
For i = startvalue:increment:endvalue
X = ….
Y = …
.
.
end
```

Eg 2: The square root of odd integers up to ten

```
For i = = 1:2:10
fprintf('the squared numbers are', i, sqrt(i));
end
```

Here in this example i value starts with 1 and incremented by 2 and then square value function will be applied to each iteration process. When I>end value the loop exits.

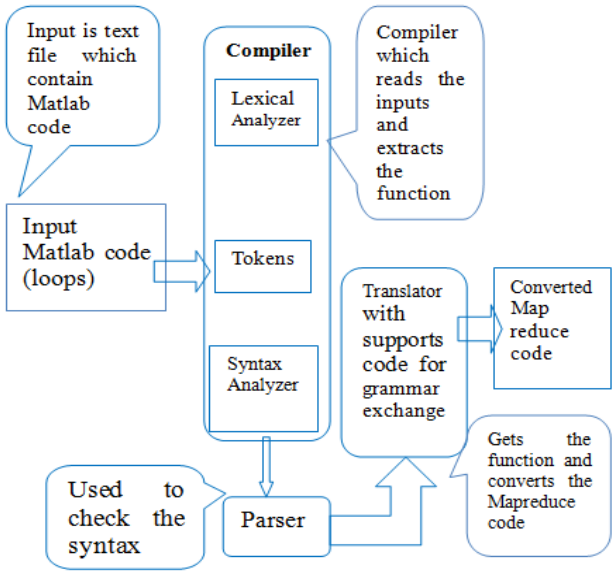

Fig. 4: Workflow of EM2M translator

Startvalue, increment value and end value may have other types rather than integer and can have negative values also.

```
Matalb for loop syntax
For var = startvalue:endvalue
Condition
..
End
Mapreduce for loop syntax
For (initialization; termination; increment)
{
Statement(s);
}
```

The Syntax used for both the programming tool has been clearly stated and there is a difference with symbols used for the functioning. Figure 4 clearly explains how the translation has been processed with Matlab commands. So our translator has been designed in such a way that the corresponding Matlab code which has been given has input will be converted to Mapreduce by based on this symbol conversions that has been written for each program inside the translator.

Since java is a platform independent environment our converted code will be executed in different platforms and their performance level will be showcased with a tabular column. Mapreduce is only a programming model which can process the given data and will store in a file system which is a Hadoop Distributed file system.

RESULTS AND DISCUSSION

Hadoop is basically a file system which is used to index data with the help of Mapreduce programming model. Let us discuss about the storage structure of Hadoop how it stores and how much memory it occupies for a particular data to be indexed in the file system. Eq. (1) the formula to estimate Hadoop storage is:

$$St = Cr * Re * Si/(1 - If) \quad (1)$$

where,
St : Hadoop Storage
Cr : Average compression ratio (Default Cr = 1)
Re : Replication factor (Default Re = 3)
Si : Size of data to be moved to hadoop
If : Intermediate factor (If = 1/3 or 1/4)

So if we calculate the hadoop storage with the default values provided we will be ended up with estimation of four times the size of the initial data size. The formula to find the number of data nodes is:

$$N = St/D \quad (2)$$

where,
St : Storage we estimated using the above formula.
D : Disk space available per node.

For example, 10TB is the allocated diskspace for each node. Then for a 700TB initial data size. N = 700/10 = 70 data nodes are needed.

The Matlab code which has been converted can be run in multiple platforms. A basic word count or a Auto min program which has been converted from Matlab to Mapreduce code can run in platforms like Windows from Chappell (2014) and Linux with appropriate Hadoop setup. Here have used IBM's BigInsight linux which is a vmware image with hadoop setup to test the code which has been converted. Matlab execution time is not a static one. The execution time totally depends upon the code which we are executing. Here have converted some basic and unique commands from Matlab which has been tested with Matlab and the converted code is executed successfully with hadoop Environment. Hence the purpose of the code will totally differ when execute in Hadoop. Here the operations are related with accessing, modifying and managing data with operation or methods available for the appropriate requirement

CONCLUSION AND FUTURE WORK

This is a very initiative approach towards implementing the matlab basic operations in hadoop environment to acces large datasets with its special functions. So it has very basic commands which has been converted and implemented in the Mapreduce code to operate a dataset. As the future extension several unique Matlab functions can be converted and implemented with Mapreduce (HDFS) environment.

ACKNOWLEDGMENT

The authors would like to thank Sathyabama Univesrsity (India) and Special thanks go to Sur University College (Sultanate of Oman) for providing us with various resources and an unconditional support for carrying out this study.

REFERENCES

Beyer, K.S., V. Ercegovac, R. Gemulla, A. Balmi, M. Eltabakhy, C.C. Kanne, F. Ozcan and E.J. Shekita, 2011. Jaql: A scripting language for large scale semistructured data analysis. Proc. PVLDB Endowment, 4(12): 1272-1283.

Chappell, D., 2014. Microsoft Azure Data Technologies: An Overview. Chappell & Associates, Sponsored by Microsoft Corporation, pp: 1-15.

Chen, C.C., M.H. Hung, N.H.T. Giang, H.C. Lin and T.C. Lin, 2013. Design and implement a

mapreduce framework for executing standalone software packages in hadoop-based distributed environments. Smart Sci., 1(2): 99-107.

Deyhim, P., 2013. Amazon Web Services-Best Practices for Amazon EMR. pp: 1-38. Retrieved from: https://media.amazonwebservices.com/AWS_Amazon_EMR_Best_Practices.pdf.

Lee, R., T. Luo, Y. Huai, F. Wang, Y. He and X. Zhang, 2011. YSmart: Yet another SQL-to-MapReduce translator. Proceeding of the 31st International Conference on Distributed Computing Systems (ICDCS, 2011). Minneapolis, MN, pp: 25-36.

Olston, C., B. Reed, U. Srivastava, R. Kumar and A. Tomkins, 2008. Pig Latin: A not-so-foreign language for data processing. Proceeding of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD '08), pp: 1099-1110.

Quinn, M.J., A. Malishevsky, N. Seelam and Y. Zhao, 1998. Preliminary results from a parallel MATLAB compiler. Proceeding of the International Parallel Processing Symposium, pp: 81-87.

Surendran, R. and B.K. Samhitha, 2014. Energy aware grid resource allocation by using a novel negotiation model. J. Theor. Appl. Inform. Technol., 68(3): 485-492.

Tamilvizhi, T., B. Parvatha Varthini and R. Surendran, 2015. An improved solution for resource management based on elastic cloud balancing and job shop scheduling. ARPN J. Eng. Appl. Sci., 10(18): 8205-8210.

Thusoo, A., J.S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff and R. Murthy, 2009. Hive: A warehousing solution over a map-reduce framework. Proc. VLDB Endowment, 2(2): 1626-1629.

Wang, L., G. von Laszewski, A. Younge, X. He, M. Kunze, J. Tao and C. Fu, 2010. Cloud computing: A perspective study. New Generat. Comput., 28(2): 137-146.

White, T., 2011. Hadoop: The Definitive Guide. 2nd Edn., O'Reilly, Farnham, pp: 1-625.

Zhang, J., D. Xiang, T. Li and Y. Pan, 2013. M2M: A simple matlab-to-mapReduce translator for cloud computing. Tsinghua Sci. Technol., 18(1): 1-9.