ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

Self Learning Based Optimal Resource Provisioning For Map Reduce Tasks with the Evaluation of Cost Functions

Nithya.M, Damodharan.P

M.E Dept. of CSE, Akshaya College of Engineering and Technology, Coimbatore, India

Associate Professor, Dept. of CSE, Akshaya College of Engineering and Technology. Coimbatore, India

ABSTRACT: Due to the massive improvement in the usage of data’s in the real world, it becomes more burdens to handle and process it effectively. The Map reduce is the one of the more developed technology which is used to handle and process the big data/largest tasks. Map reduce is used to partition the task into sub partitions and map those partitions into the machines for processing. This process need to be done by the considering the minimization of cost and meeting deadline to improve the user satisfaction. In the previous work, CRESP approach is used which focus on allocating the map reduces tasks in the machine with the consideration of reduction of cost and deadline. However this method does not concentrate on the skew and stragglers problem which can occur while handling the largest task. In our work, we try improve the performance of resource allocation strategy by considering the skews and stragglers problem in mind. This problem of skews and stragglers are handled by introducing the partitioning mechanism. The partitioning mechanism will improve the failure of task allocation strategy.

KEYWORDS*:* Cloud Computing, Hadoop, Map Reduce, Micro Partitioning, Self Based Learning, TeraSort, WordCount, PageRank, TableJoin.

I.INTRODUCTION

In the Development of Cloud Computing,Sensor Networks, Grid Computing the huge amount of Datasets could be Collected from the users, Applications and Environment.For Instance, nowadays users are capable of storing huge amount of Datasets in a Datacenter, where we go for usage of the Big Data. Map Reduce is a technique which is used to work with the Big Data.Big Data is nothing but a collection of huge datasets which is difficult to process using on-hand database management tools or traditional data processing applications.

On the other hand, Cloud analysts in most Organisations such as research institutions, Government institiutions have no opportunity to access more private Hadoop/MapReduce clouds. Based on the requirements Amazon introduced Elastic Map Reduce which runs on Hadoop Clusters. Apache Hadoop is the open source Software for Storing and Processing Large amounts of datasets on Clusters of Hardware. Requirements for running Hadoop cluster on Public and Private clouds varies promptly. First for each job clusters could be allocated starting from the virtual nodes to make use of “pay-as-you-use” economic Cloud model. It is difficult to maintain a constant Hadoop cluster as private Hadoop clusters because data processing requests are generally entering in Continuosly.

The Optimization of the Resource Provisioning could involve about 2 factors.
provisioning the virtual machine nodes which could consider about the monetary cost and Finishing of the job which considers about the time cost. Resource provisioning could be done based on the cost which depends on the time required

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

for resources to be used. It is complicated to use other constraints such as deadline or monetary budget to reduce the cost for usge of resources. We propose a method to help users to decide about Allocation of running MapReduce programs in public clouds. This method, Micro Partitioning is used to distribute the work load amoung the nodes.

II.CRESP APPROACH

The Cloud RESource Provisioning (CRESP) for MapReduce Programs is based on the time cost model. Considering the time cost model and the parameters, providing the users could solve optimization problems. We analyse cost model in terms of Input datasets, Specific complexity of the application and the resources that are available for the system. In this approach we consider the combination of the white-box and machine learning approaches. MapReduce programs have different time and the logical complexity. The cost functions that may be differing from application to application.

Analysing Map Reduce Tasks

MapReduce is the Combination of parallel and distributed processing.Large number of datas that could be processed which are defined as clusters. Reduce phase is executed after the execution of Map phase. The execution of Map and Reduce programs is done by using the concepts of Map/Reduce slot and Map/Reduce task. Slot is the unit used for running the tasks by allocating the resources that are available. Fixed number of slots could be allocated based on the capacity of the system. In hadoop, which consists of four Components NameNode which is the heart of hadoop file system, DataNode which stores blocks of datas and retrieve them, TaskTracker is responsible for the allocation of the number of Map slots and Reduce slots, JobTracker is responsible for the allocation of the client jobs.

Calculation of Cost For Map Task

Map phase consists of the three stages Map,Read and Sort. First we consider the calculation of cost for the input that is given to the Map phase. The input that may be in the form of data blocks i(b) that could be from the local or remote disk. Second we consider the Map function f(b) that is given by the user. After that sorting of the datas $o_m(b)$ could be done and then the output will be in the form of (Key,Value) pairs which could be give to the reduce phase.

$$\Phi_m = i(b) + f(b) + s(om(b),R) \longrightarrow \varepsilon_m \qquad (1)$$

Calculation of Cost Of Reduce Task

Reduce phase consists of the three stages Shuffle, MergeSort, Reduce and WriteResult. The execution that could be done in parallel in the reduce phase. In the execution of the reduce task the amount of data is proportional to the number of keys that are assigned. The keys given by the Map phase is distributed equally to the reduce Tasks. In the Shuffle stage the each and every reduce tasks will be assigned for its shares which could be given as k/R and the the amount of data that could be given is

$$b_R = M * o_m(b) * k/R \longrightarrow (2)$$

In the MergeSort stage the datas that could be simply merged because the sorting of the datas are done at the earlier stages. The cost of MergeSort could be given as it is $ms(b_R)$ which depends upon $b_{R.}$

Learning the Model

For the calculation of the cost function we concentrate on the number of input variables, M, m, R and then the parameter $\beta_{i.}$ First we randomly sample the variables which is considered for testing. Second collect the time and cost for the map and reduce tasks by setting the variables (M,m,R). m is nothimg but the number of samples that are considered. Third the regression model is considered which is applicable for learning of the models with the transformed variables such as,

$X_1 = m/M$, $x_2 = MR/m$, $x_3 = R/m$, $x_4 = (M \log M)/R$, $x_5 = M/R$, $x_6 = M$, $x_7 = R$.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

III.SYSTEM MODEL

Micro Partitioning

Micro Partitioning is the key technique is to run a large number of reduce tasks, splitting the map output into many smaller units of partitions than reduce machines in order to produce smaller tasks. These smaller tasks are assigned to reduce machines in a “just-in-time” fashion as workers become idle, allowing the task scheduler to dynamically mitigate skew and stragglers. With large tasks, it can be more efficient to exclude slow nodes rather than assigning them any work. By assigning smaller units of work, jobs can derive benefit from slower nodes.

These tasks are assigned to reduce machines as workers become idle, allowing the task scheduler to dynamically mitigate skew and stragglers. Running many small tasks lessens the impact of stragglers, since work that would have been scheduled on slow nodes is small which can now be performed by other idle workers.

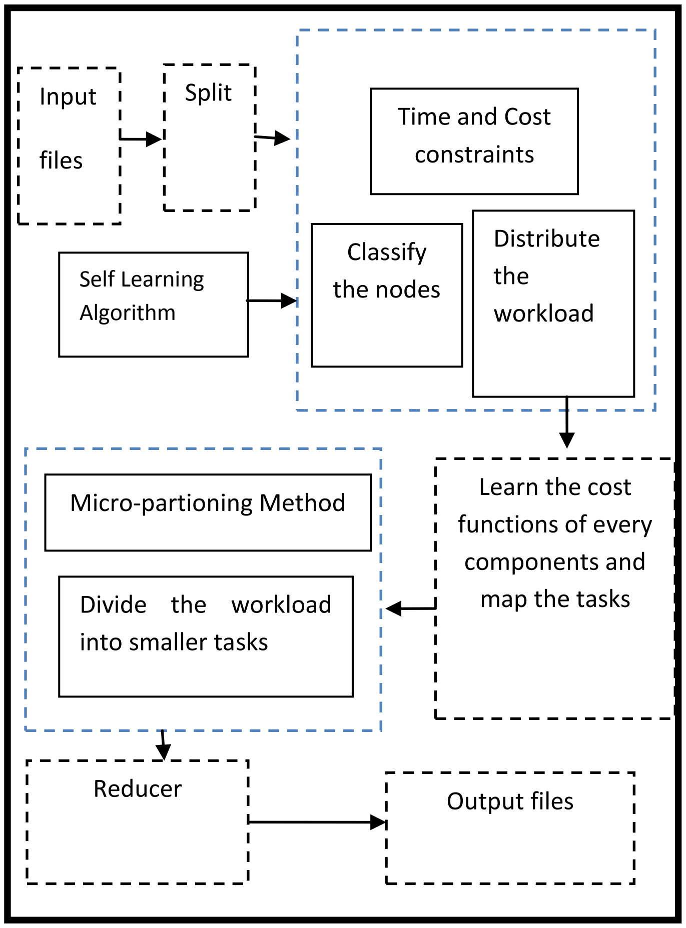
Input files
Split
Time and Cost constraints
Self Learning Algorithm
Classify the nodes
Distribute the workload
Micro-partioning Method
Divide the workload into smaller tasks
Learn the cost functions of every components and map the tasks
Reducer
Output files

Figure 3.1. Architecture Diagram

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

For inputs containing few distinct keys, fine-grained partitioning may result in many empty reduce tasks that receive no data. These empty reduce tasks are unproblematic, since they can be easily detected and ignored by the scheduler. Jobs with few distinct keys are the most sensitive to partitioning skew, since there may not be enough other work to mask the effects of a straggling task created by a key collision in the hash partitioning function.

Datasets

We use four types of testing Datasets to test the samples. The datasets could be 1000 words randomly chosen is used as samples from the dictionary. Another dataset is samples generated from the PageRank Program, next dataset is packages from the Hadoop.

The samples used are

WordCount: Used to calculate the number of words that are present in the input file that is given.

TeraSort: Sorting of the datas that are done and then given for reducers.

PageRank: The ranking for the accessing of the websites are given.

TableJoin: The joining of the wordcount, TeraSort is done.

The implemention of micro partitioning technique is as follows:

```
Step 1: Take the input samples
Step 2: Store the input samples in trie
Step 3: Build the two-level trie
Step 4: Count the occurrence of each and every prefixes
Step 5: Use cut-point algorithm to determine the cut-points
Step 6: Split points are obtained
Cut points = sum of counters/number of partitions+ 1
Step 7: Use cut points to send the keys to appropriate reducers
if (key<cutpoint1)
Send key to reducer 1
else if (key>=cut-point1&&key<cut-point2)
Send key to reducer2
else if (key>=cut-point2&&key<cut-point3)
Send key to reducer3
else
Send to the finished reducer
Step 8: Determine the slow running node by comparing the performance of each node with other.
Step 9: If there is any slow running node move the data to the free node.
```

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

IV. PROPOSED METHOD

Learning Resource Status Using State Vector Machine

SVM based learning approach is introduced for learning about the resouces that are clocated to the training sample. The cost that is required for the compution that could be reduced by the informations that are learned. In machine learning, support vector machines are the supervised learning methods which are associated with the learning algorithms, that could analyse datas and recognise patterns in order to do classification and regression analysis.

The machine learning method which could identify the pedagogical relationships. We are learning the new rules in the pedagogical relationships or the DRs in the electronic textbooks in the training phase. Support vector machines which make use of the decision planes that defines the decision boundaries. A decision plane is nothing but the separation between a set of objects having different class memberships. The SVM modeling algorithm finds an optimal hyper plane which focuses on the maximal margins to separate the two classes. This coul be required to solve the following optimization problem.

Maximize,

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j \, y_i y_j \, k(x_i, x_j)$$

Subject to,

$$\sum_{i=1}^{n} \alpha_i \, y_i = 0$$

Where $0 \le \alpha_i \le b$ $i = 1,2,\dots\dots,n$, α_i is the weight of training sample x_1, K is a kernel function, used to measure the similarity between two samples. A popular radial basis function (RBF) kernel functions. This is repeated for ‘k’ times.

Algorithm

Input: Number of the training samples (determined in existing system) with dataset w as input data point for SVM classification

Output: Classification result

```
Procedure SVM ($w$) // input training data results from the SVM classification

Begin

Initialize the value C=0 //initially the class labels should be zero
```

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

Get input file dataset w for training //the input dataset result as the example for training the user data and prediction results of the pedagogical relationships

Read the number of input training dataset W from the given original dataset

$x_i.w + b = 0$ ////Input training dataset W is represented as matrix and denoted by x_i and w is the weight value matrix whose product is summed with bias value to give the class value.

$x_i.w + b = 1$ // this above equation marks a central classifier margin. This can be bounded by soft margin at one side using the following equation.

Decision function $f(W) = x_i.w - b$ //decision function f(w) decides the class labels for the SVM classification training examples ,

If $f(W) \geq 1$ for x_i is the first class // if the f(w) is greater than or equal to the 1 is labeled as first class

Else

$f(W) \leq -1$ for x_i is the second class // if the f(w) is less than or equal to the value of -1 is labeled as second class

The prediction result for (i=1,…n) number of training samples //after the classification result are performed then check the classification result by testing phase it is check the below function

$y_i(x_i.w - b) \geq 1$

Display the result //finally we display the classification result.

V. RESULT ANALYSIS

Figure 5.1 compares the consumption of time in CRESP Approach with the Micropartitioning Mechanism. The time consumption is reduced in the Micro Partitioning mechanism. Figure 5.2 shows that uage of the size of memory is reduced in Micro Partitioning Mechanism when compared with the CRESP Approach. Figure 5.3 shows that cost is reduced in Micro Partitioning Mechanism when compared with the CRESP Approach.

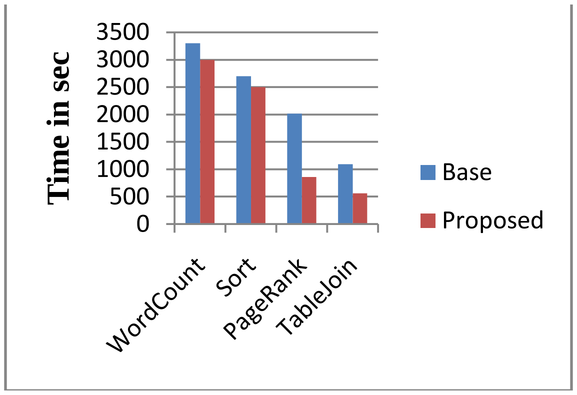

Figure.5.1 Time Consumption in ms

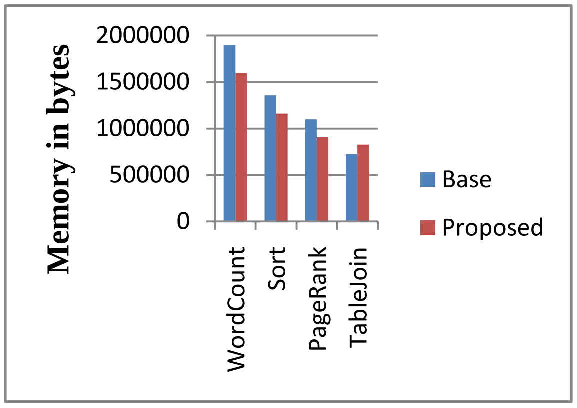

Figure.5.2 Memory consumption in bytes

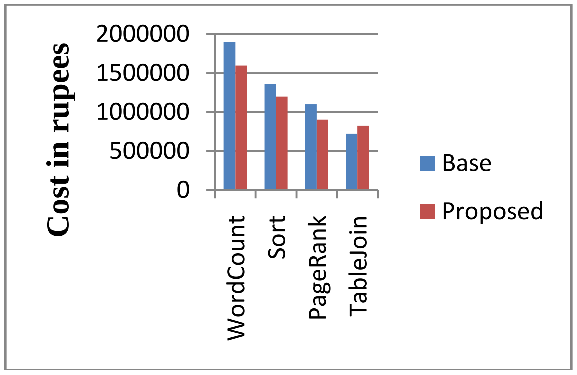

Figure.5.3 Cost in rupees

VI. CONCLUSION AND FUTURE WORK

A. CONCLUSION

In this work, we study the components in MapReduce processing and build a cost function that explicitly models the relationship among the amount of data, the available system resources, and the complexity of the Reduce function for the target Map Reduce program. The model parameters can be learned from test runs. Based on this cost model, we can solve decision problems, which could minimize the monetary cost by considering monetary budget or job finish time.

To improve the load balancing for distributed applications, micro partitioning techniques. By improving load balancing, MapReduce programs can become more efficient at handling tasks by reducing the overall computation time spent processing data on each node. In addition to that we use MapReduce For that we use node classification method and distribute the workload among the nodes according to the node capacity. After that a micro partitioning method is used for applications using different input samples. This approach is only effective in systems with high-throughput, low-latency task schedulers and efficient data materialization.

B. FUTURE WORK

In the future, we would like to implement the proposed task scheduler architecture and perform additional experiments to measure performance using straggling or heterogeneous nodes. We also plan to investigate other benefits of micro-tasks, including the use of micro-tasks as an alternative to preemption when scheduling mixtures of batch and latency-sensitive jobs.

REFERENCES

1. Candan KS, Kim JW, Nagarkar P, Nagendra M, Yu R (2010) RanKloud: scalable multimedia data processing in server clusters. IEEE MultiMed 18(1):64–77
2. Chang F, Dean J, Ghemawat S, Hsieh WC, Wallach DA, Burrws M, Chandra T, Fikes A, Gruber RE (2006) Big table: a distributed storage system for structured data. In: 7th UENIX symposium on operating systems design and implementation, pp 205–218.
3. Dean J, Ghemawat Dean S (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51:107–
4. Ghemawat S, Gobioff H, Leung S-T (2003) The Google file system. In: 19th ACM symposium on operating systems principles (SOSP).
5. Jiang W, Agrawal G (2011) Ex-MATE data intensive computing with large reduction objects and its application to graph mining. In: IEEE/ACM international symposium on cluster, cloud and grid computing, pp 475–484.
6. Jin C, Vecchiola C, Buyya R (2008) MRPGA: an extension of MapReduce for parallelizing genetic algorithms. In: IEEE fourth international conference on escience.
7. Kavulya S, Tany J, Gandhi R, Narasimhan P (2010) An analysis of traces from a production MapReduce cluster. In: IEEE/ACM international conference on cluster, cloud and grid computing, pp 94–95.
8. Krishnan A (2005) GridBLAST: a globus-based high-throughput implementation of BLAST in a grid computing framework. Concurr Comput 17(13):1607–1623.
9. Hsu C-H, Chen S-C (2012) Efficient selection strategies towards processor reordering techniques for improving data locality in heterogeneous clusters. J Super computer 60(3):284–300.