RESEARCH ARTICLE OPEN ACCESS

Energy Efficient algorithm of Map Reduce for Big Data Applications

D.Revathi
Dept. of computer science & Engineering
AVN institute of technology,
Hyderabad.

Vijendar amgothu
Dept. of computer science & Engineering
University College of Engineering
Osmania University,Hyderabad

Abstract:

In recent years the data mining applications become musty and outmoded over time. Energy wastage is the major problem more in big data analytics and applications. More workload and more computational time will increase high energy cost and decrease efficiency. The Incremental computational time processing is a promising approach to refreshing mining results. It utilizes previously saved states to avoid the expense of re-computation from scratch. In this paper, we propose Energy efficiency Map Reduce Scheduling Algorithm, a novel incremental processing extension to reduce the Map, the most widely used framework for mining big data. Map reduce is a programming model for processing and generating large amount of data in parallel processing time. In this paper, Energy Efficiency reduce Map (EEMP) is algorithm provide more energy and less maps in big data. Priority based scheduling is a task will allocate the schedules based on necessary and utilization of the Jobs. For reducing the maps, it will reduce the system computational time so easily energy has improved in terms of big data applications.. Final results show the experimental comparison of the different algorithms involved in the paper.

Index Terms— Big Data, EEM, Map Reduce, incremental processing, svm.

I. Introduction

Now days waste quantity of digital data is being accumulated in several important areas, including e-commerce, social network, finance,banking, health care, education, and environment. It's become progressively popular to mine such massive data so as to achieve insights to assist business selections or to produce higher customized, higher quality services. In recent years, a large variety of computing frameworks [1], [2], [3], [5] are established for large data analysis. Among these frameworks, map reduce [1] (with its ASCII text file implementations, like Hadoop) is that the most generally utilized in production because of its simplicity, gnerality and maturity. We proposed be likely to focus on map reduce during in big data applications.It is now implemented with the bigdata applications. Big data is constantly evolving. Due to the advent of new technologies, devices, and communication means like social networking sites, the amount of data produced bymankind is growing rapidly every year. Though all this information produced is meaningful and can be useful when processed, it is being neglected Big data means it is a collection of large datasets that cannot be processed using traditional computing techniques. Big data is not merely a data, rather it has become a complete subject, which involves various tools, techniques and frameworks. As new data and updates are being collected, the input data of a big data mining algorithm will gradually change, and the computed results will become stale and obsolete over time. Incremental processing is a promising approach to refreshing mining results. It utilizes previously saved states to avoid the expense of re-computation from scratch. In this paper, we propose Energy Map Reduce Scheduling Algorithm, a novel incremental processing extension to Map Reduce, the most widely used framework for mining big data. Map Reduce to support incremental processing. However, it has two main limitations. First, Incoop method supports only task-

level incremental processing. . That is, it saves and reuses states at the coarseness of individual Map and reduce tasks. Every task generally processes an oversized number of key-valuepairs (kvpairs). If Incoop detects any data changes within the input of a task, it'll rerun the whole task. Whereas this approach simply leverages existing Map Reduce options for state savings, it should incur a large quantity of redundant computation if solely a small fraction of kv-pairs have modified during a task. Second, Incoop supports solely one-step computation, whereas important mining algorithms, like PageRank, need iterative computation. Incoop would treat every iteration as a separate MapReduce job. However, a small number of input data changes could gradually propagate to affect a large portion of intermediate states when variety of iterations, leading to expensive global re-computation after. We tend to propose i2MapReduce, an extension to Map Reduce that supports fine-grain incremental processing for each one step and iterative computation.

II. Proposed method

Main aim of proposed method is to reduce the minimum energy cost from efficient Map reducing concepts in big data applications. To optimize the mining results, we evaluate Map Reduce using a one-step algorithm and three iterative algorithms with diverse computation characteristics for efficient mining also improve the energy. In this paper we also include the algorithm for incremental processing approach named as Energy map reduce scheduling method (EEMP) is provide more energy and minimum maps. Priority based scheduling is a task will allocate the schedules based on necessary and utilization of the Jobs. For reducing the maps, it will reduce the system work so easily energy has improved. Final we shows the experimental comaprison of the different This paper involves the input files with the .arff extension, that is, artributte relation file format. An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files have two distinct sections. The first section is the Header information, which is followed the Data information.The Header of the ARFF file contains the name of the relation, a list of the attributes (the columns in the data), and their types. Hadoop plugin is implemented in the eclipse environment. Hadoop is the flexible and available architecture for large scale computation on data processing on a netwrk of commodity hardware. Eclipse is an integrated development environment (IDE). It contains a base workspace and an extensible plug-in system for customizing the environment. Here in this paper we are implementing hadoop plugin by including tha jar files in eclipse and which creates a virtual memory of 1GB. existing algorithms involved in the paper.

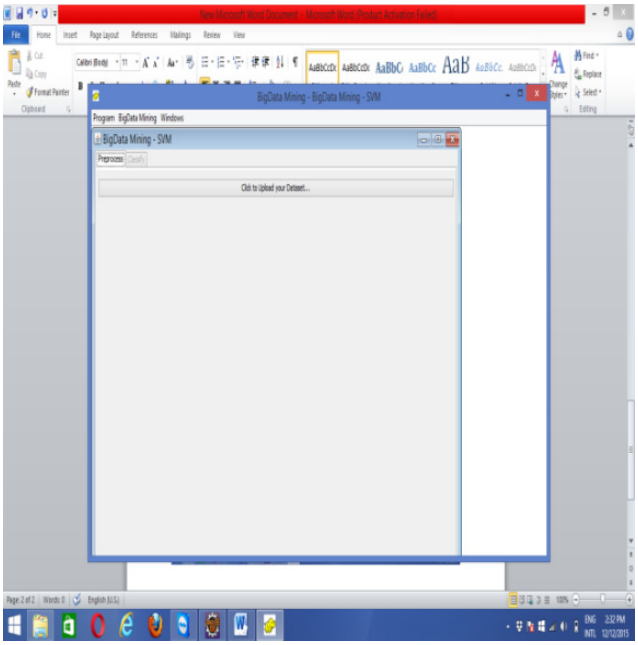

Fig.1.screen shot

II.1. Related Work

In related work we presented the proposed algorithm for enery efficiency map reduce.

Algorithm 1 EMRSA-X

1: Create an empty priority queue $\mathcal{Q}^m$
2: Create an empty priority queue $\mathcal{Q}^r$
3: **for all** $j \in \mathcal{A}$ **do**
4: $ecr_j^m = \min_{\forall i \in \mathcal{M}} \frac{e_{ij}}{p_{ij}}$, for EMRSA-I; or
$ecr_j^m = \frac{\sum_{\forall i \in \mathcal{M}} \frac{e_{ij}}{p_{ij}}}{M}$, for EMRSA-II
5: $\mathcal{Q}^m$.enqueue(j, ecr_j^m)
6: **for all** $j \in \mathcal{B}$ **do**
7: $ecr_j^r = \min_{\forall i \in \mathcal{R}} \frac{e_{ij}}{p_{ij}}$, for EMRSA-I; or
$ecr_j^r = \frac{\sum_{\forall i \in \mathcal{R}} \frac{e_{ij}}{p_{ij}}}{R}$, for EMRSA-II
8: $\mathcal{Q}^r$.enqueue(j, ecr_j^r)
9: $D^m \leftarrow \infty$; $D^r \leftarrow \infty$
10: **while** $\mathcal{Q}^m$ is not empty **and** $\mathcal{Q}^r$ is not empty **do**
11: $j^m = \mathcal{Q}^m$.extractMin()
12: $j^r = \mathcal{Q}^r$.extractMin()
13: $f = \frac{\sum_{\forall i \in \mathcal{M}} p_{ij^m}}{\sum_{\forall i \in \mathcal{R}} p_{ij^r}}$
14: $\mathcal{T}^m$: sorted unassigned map tasks $i \in \mathcal{M}$ based on p_{ij^m}

\

15: $\mathcal{T}^r$: sorted unassigned reduce tasks $i \in \mathcal{R}$ based on p_{ij^r}
16: **if** $\mathcal{T}^m = \emptyset$ **and** $\mathcal{T}^r = \emptyset$ **then break**
17: ASSIGN-LARGE()
18: ASSIGN-SMALL()
19: **if** $D^m = \infty$ **then**
20: $D^m = D - p^r$
21: $D^r = p^r$
22: **if** $\mathcal{T}^m \neq \emptyset$ **or** $\mathcal{T}^r \neq \emptyset$ **then**
23: No feasible schedule
24: **return**
25: **Output:** X, Y

In this paper involves the input files with the .arff extension, that is, artributte relation file format. An Attribute-Relation File Format file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files have two distinct sections. The first section is the Header information, which is followed the Data information. The Header of the ARFF file contains the name of the relation, a list of the attributes (the columns in the data), and their types. Hadoop is implemented in the eclipse environment. Hadoop is the flexible and available architecture for large scale computation on data processing on a netwrk of commodity hardware. Eclipse is an integrated development environment (IDE). It contains a base workspace and an extensible plug-in system for customizing the environment. Here in this paper we are implementing hadoop plugin by including the .jar files in eclipse and which creates a virtual memory of 2GB**.**

In this proposed paper we show the efficient methods of classifying the evaluvated results by using the two classification methods: one is Support Vector Machine(SVM) and Naive Bayesian.

A support vector machine is a Classification method..it is a Supervised algorithm used for Classification and Regression (binary and multi-class problem) and anomalie detection (one class problem) .An SVM training algorithm builds a model that assigns new data into one category or the other category, making it is a non-probabilistic binary linear classifier. An SVM model is a representation of the data as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. The support vector machine has been developed as robust tool for classification and regression in noisy, complex domains. The two key features of support vector machines are generalization theory, which leads to a principled way to choose an hypothesis; and, kernel functions, which introduce non-linearity in the hypothesis space without explicitly requiring a non-linear algorithm

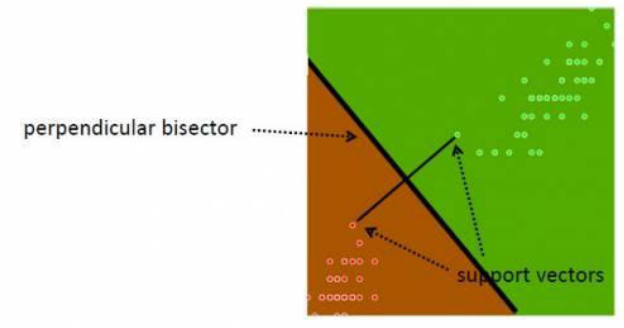

Fig.2. shows the svm classifier

The black line indicates that separate the two classes of svm. . Mathematically, the separation can be found by taking the two critical members, one for each class. This points are called support vectors. These are the critical points (members) that define the channel.The separation is then the perpendicular bisector of the line joining these two support vectors .That's the idea of support vector machine.

The second classifier method is The Naive Bayesian classifier is based on Bayes‟ theorem with independence assumptions between predictors. A Naive Bayesian model is easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. Despite its simplicity, the Naive Bayesian classifier often does surprisingly well and is widely used because it often outperforms more sophisticated classification methods. Bayes theorem provides a way of calculating the posterior probability. Naive Bayes classifier assumes that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors. This type of assumption is called class conditional independence.

The posterior probability can be calculated first, then constructing a frequency table for each attribute against the target value . Then, transforming the frequency tables to likelihood tables and finally uses the Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

II.2. Map reduce process

In the above block diagram is the map reduce processing in the big data applications**.** In this diagram three factors are considered one is map tasks, and second is reduce tasks ,final factor is reduced tasks.

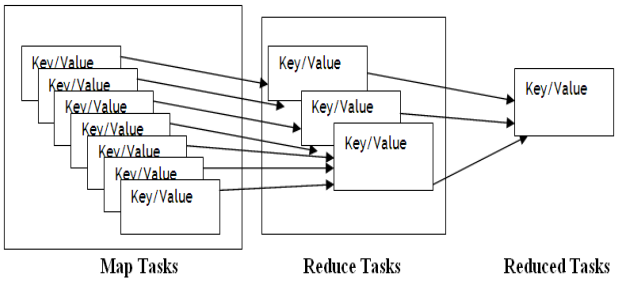

Fig.3.Map reduce processing

MapReduce is a programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster. It is this programming paradigm that allows for massive scalability across hundreds or thousands of servers in a Hadoop cluster. The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). The reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce job is always performed after the map job. Process of scheduling is an essential part of a Multiprogramming operating system. Such operating systems allow more than one process to be loaded into the executable memory at a time and loaded process shares the CPU using time multiplexing. Priority Scheduling. The basic idea is straightforward: each process is assigned a priority, and priority is allowed to run. Equal-Priority processes are scheduled in FCFS order. The shortest-Job-First (SJF) algorithm is a special case of general priority scheduling algorithm.

The probability density function for the normal distribution is defined by two parameters (mean and standard deviation)

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{Mean}$$

$$\sigma = \left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu)^2\right]^{0.5} \qquad \text{Standard deviation}$$

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad \text{Normal distribution}$$

III. Conclusion

We have described support vector machine and naïve Bayesian classification methods for effective data analysis results and a set of efficient techniques for incremental iterative processing computation. Real time experiments will show that EEMP and the described classification methods significantly reduce the run time for refreshing big data mining results compared to re-computation on both plain and iterative Map Reduce thereby reduces the workload of the system which results in the efficient and reliable energy usage. We are also classify the data by using svm and The Naive Bayesian methods.

References

[1] S. Lloyd, "Least squares quantization in PCM," IEEE Trans. Inform. Theory., vol. 28, no. 2, pp. 129–137, Mar. 1982. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] I. S R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in Proc. 20th Int. Conf. Very Large Data Bases, 1994, pp. 487–499.

[3] S. Brin, and L. Page, "The anatomy of a large-scale hypertextual web search engine," Comput. Netw. ISDN Syst., vol. 30, no. 1–7, pp. 107–117, Apr. 1998..

[4] R. Power and J. Li, "Piccolo: Building fast, distributed programs with partitioned tables," in Proc. 9th USENIX Conf. Oper. Syst. Des. Implementation, 2010, pp. 1–14

[5] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl.

[6] J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982.

[7] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in Proc. 6th Conf. Symp. Opear. Syst. Des. Implementation,2004.

[8] T. J€org, R. Parvizi, H. Yong, and S. Dessloch, "Incremental recomputations in mapreduce," in Proc. 3rd Int. Workshop Cloud Data Manage.,2011,pp.7–14.

[9] Y. Zhang, Q. Gao, L. Gao, and C. Wang, "imapreduce: A distributed computing framework for iterative computation," J. Grid Comput., vol. 10,no.1,pp.47–68,2012.

[10] D. G. Murray, F. McSherry, R. Isaacs, M. Isard, P. Barham, and M. Abadi, "Naiad: A timely dataflow system," in Proc. 24th ACM Symp. Oper.Syst.Principles,2013,pp.439–455.