Malaviya National Institute of Technology
Department of Computer Science & Engineering

Machine Learning Algorithm for Pro-active Fault Detection in Hadoop Cluster

Winter Internship of:
Agrima Seth

Advisor:
Prof. Mahesh Chandra Govil

December 2014

Acknowledgement

I would like to express my sincere gratitude to Prof Mahesh Chandra Govil for providing me this opportunity to work at Malaviya National Institute of Technology, Jaipur.
I am deeply indebted to him for his guidance and encouragement.
I would also like to express my gratitude to Mr. Mohit Gokhroo for his help and support.
A special thanks to Prof. Sangeeta Jadhav, HoD, Information Technology, Army Institute of Technology, University of Pune, without whose support, this internship would not have been possible

Contents

List of Figures **iv**

1 Introduction **1**
- 1.1 Introduction . . . 1
- 1.2 Fault Tolerance . . . 2
  - 1.2.1 Role of Machine Learning . . . 2
  - 1.2.2 Supervised Learning . . . 2
  - 1.2.3 Unsupervised Learning . . . 3

2 Hadoop Cluster **5**
- 2.1 Hadoop Structure . . . 5
- 2.2 Hadoop Architecture . . . 6

3 Ganglia Monitoring **9**
- 3.1 Ganglia Monitoring Daemon . . . 9
- 3.2 Ganglia Meta Daemon . . . 10
- 3.3 Ganglia PHP Web Front End . . . 10

4 Approach **11**
- 4.1 Feature Extraction . . . 11
- 4.2 Sparse Model . . . 12
  - 4.2.1 Learning . . . 12
- 4.3 Method of Optimal Directions . . . 13

5 Results **15**
- 5.1 The patterns Observed . . . 15

6 Future Work **19**

Bibliography **21**

List of Figures

2.1 Multi Node Hadoop Cluster 6

4.1 Sparse Model 12

5.1 Log of Bytes 15
5.2 Cpu and Memory Logs 16
5.3 Cpu Usage and memory availability 16
5.4 Cpu Parameters 16
5.5 Varying Load with input packets 17
5.6 Varying load across nodes 17
5.7 Memory Logs 17

Introduction 1

1.1 Introduction

In any system or application the reliability is the foremost requirement. In cloud the complexity increases due to its scalable and heterogeneous architecture, increasing the possibility of more failures. This needs some fault tolerance measures to provide reliable service to users in the cloud. Our scope of study is to predict the possibility of fault in cloud before it actually arrives or occurs.

The faults maybe of many types depending upon their nature of occurrence:

1. Permanent Faults.
2. Intermittent Faults.
3. Transient Faults.

In cloud generally, services offered to various users are under SLA(Service Liable Agreement) which generally specifies QoS required and penalty clause if service rendered is not as per the SLA. This requires that the service rendered is not as per the SLA. This requires that the services offered to any customer be reliable and delivered in stipulated time. Therefore, fault tolerance measures become more important in cloud environment.

1.2 Fault Tolerance

Approach to handle faults :

1. Avoidance
2. Tolerance
3. Removal

Cloud requires a good fault management strategy o predict or detect the fault in advance and mask or mitigate the same in an efficient manner. The fault prediction is a difficult task because it is difficult to know in advance that which hardware/software component will malfunction or develop a permanent or transient fault. The scope of our study is to design, develop and increment a fault prediction technique based on Machine Learning.

1.2.1 Role of Machine Learning

Machine learning explores the construction and study of algorithms that can learn from data. These algorithms operate by building a model based on inputs and using that to make predictions or decisions, rather than following only explicitly programmed instructions. Kadirvel et al. (2013)

1.2.2 Supervised Learning

Supervised learning is the machine learning task of inferring a function from labeled training data. The training data consist of a set of training examples. Each example is a pair consisting of an input object and a desired output value . A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. Mehryar Mohri and Talwalkar (2012)

1.2.3 Unsupervised Learning

Unsupervised learning is trying to find hidden structure in unlabeled data. Since the examples given to the learner are unlabeled, there is no error or reward signal to evaluate a potential solution. Tucker. (2004)

Hadoop Cluster 2

Apache Hadoop is an open source software project that enables the distributed processing of large data sets across clusters of commodity servers. It is designed to scale up from a single server to thousands of machines. Rather than relying on high-end hardware, the resiliency of these clusters comes from the softwareâĂŹs ability to detect and handle failures at the application layer. IBM

2.1 Hadoop Structure

The base Apache Hadoop framework is composed of the following modules:

1. Hadoop Common contains libraries and utilities needed by other Hadoop modules.

2. Hadoop Distributed File System (HDFS) a distributed file-system that stores data on commodity machines, providing very high aggregate bandwidth across the cluster.

3. Hadoop YARN a resource-management platform responsible for managing compute resources in clusters and using them for scheduling of users' applications.

4. Hadoop MapReduce a programming model for large scale data processing.

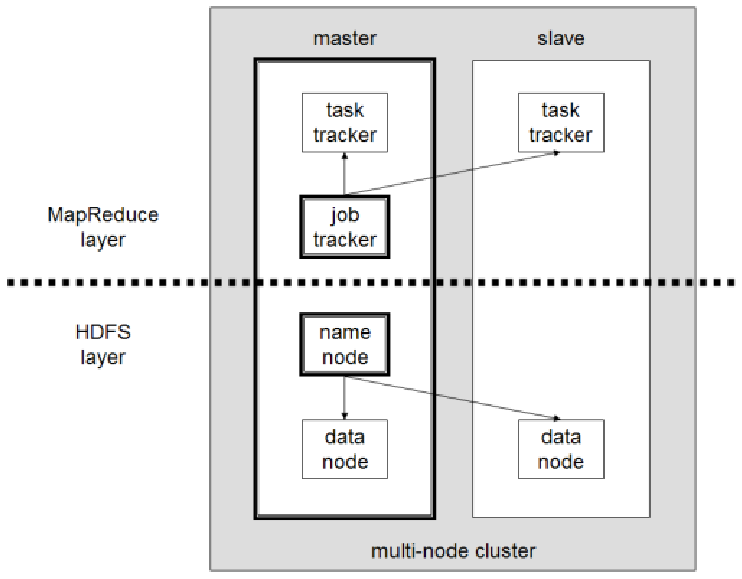


Figure 2.1: Multi Node Hadoop Cluster

2.2 Hadoop Architecture

Hadoop consists of the Hadoop Common package, which provides filesystem and OS level abstractions, a MapReduce engine (either MapReduce/MR1 or YARN/MR2) Chouraria (2012) and the Hadoop Distributed File System (HDFS). The Hadoop Common package contains the necessary Java ARchive (JAR) files and scripts needed to start Hadoop. The package also provides source code, documentation, and a contribution section that includes projects from the Hadoop Community.

A small Hadoop cluster includes a single master and multiple worker nodes. The master node consists of a JobTracker, TaskTracker, NameNode and DataNode. A slave or worker node acts as both a DataNode and TaskTracker, though it is possible to have data-only worker nodes and compute-only worker nodes. These are normally used only in nonstandard applications.

In a larger cluster, the HDFS is managed through a dedicated NameNode server to host the file system index, and a secondary NameNode that can generate snapshots of the namenode's memory structures, thus preventing file-system corruption and reducing loss of data. Similarly, a standalone JobTracker server can manage job scheduling. In clusters where the Hadoop MapReduce engine is deployed against an alternate file system, the NameNode, secondary NameNode, and DataNode architecture of HDFS are replaced by the file-system-specific equivalents.

Ganglia Monitoring 3

Ganglia is a scalable distributed monitoring system for Clusters and Grids. It is based on a hierarchical design targeted at federations of clusters. It uses XML for data representation, XDR for compact, portable data transport, and RRDtool for data storage and visualization. It uses data structures and algorithms to achieve very low per-node overheads and high concurrency. The implementation is robust, has been ported to an extensive set of operating systems and processor architectures. Sorceforgenet (2000)

3.1 Ganglia Monitoring Daemon

Gmond is a multi-threaded daemon which runs on each cluster node to be monitored.Gmond has four main responsibilities:

1. Monitor changes in host state.
2. Announce relevant changes.
3. Listen to the state of all other ganglia nodes via a unicast or multicast channel.
4. Answer requests for an XML description of the cluster state.

The two modes of information transmission are :

1. Unicasting or Multicasting host state in external data representation (XDR) format using UDP messages.
2. Sending XML over a TCP connection.

3.2 Ganglia Meta Daemon

Federation in Ganglia is achieved using a tree of point-to-point connections amongst representative cluster nodes to aggregate the state of multiple clusters. At each node in the tree, a Ganglia Meta Daemon (gmetad) periodically:

1. polls a collection of child data sources
2. parses the collected XML
3. saves all numeric, volatile metrics to round-robin databases
4. exports the aggregated XML over a TCP socket to clients.

Data sources are:

1. gmond daemons, representing specific clusters
2. gmetad daemons, representing sets of clusters.

Data sources use source IP addresses for access control and can be specified using multiple IP addresses for failover. r.

3.3 Ganglia PHP Web Front End

The Ganglia web front-end provides a view of the gathered information via real-time dynamic web pages. Most importantly, it displays Ganglia data in a meaningful way for system administrators and computer users.

The Ganglia web front-end is written in PHP, and uses graphs generated by gmetad to display history information.

Approach 4

For learning of expected behaviour of the system we established a hadoop cluster consisting of one master and four slave nodes. Further ganglia monitoring system was installed on these nodes and jobs were assigned by the master node. The logs obtained from the rrdtool of ganglia consisted of the feature vectors to consider. They were then converted to CSV format which was then subjected to sparse coding for learning in MATLAB.

4.1 Feature Extraction

Machine Learning Classifiers work according to a set of classifiers given to them. The patterns then produced through this learning is used for further processing.

The feature set used were:

1. Boot Time
2. Bytes In
3. Bytes Out
4. The Idle time for Cpu
5. Total Disk Usage

6. Used Memory of the system
7. Free Memory of the system
8. The total load on the system at an interval of 1,5,15 minutes

We analysed 250 memory logs from the hadoop cluster under various task deployment. The results of which are mentioned in Chapter 5.

4.2 Sparse Model

Sparse coding is a class of unsupervised methods for learning sets of over-complete bases to represent data efficiently.

The aim of sparse coding is to find a set of basis vectors Φ_i such that we can represent an input vector X as a linear combination of these basis vectors:

$$\mathbf{x} = \sum_{i=1}^{k} a_i \phi_i \tag{4.1}$$

The sparse coding function on a set of T tasks is defined as follows:

$$min(D) 1/T \sum_{t=1}^{T} min(h^t) \left\| X^t - Dh^t \right\|_2^2 + \lambda \left\| h^t \right\|_1$$

Figure 4.1: Sparse Model

4.2.1 Learning

Learning a set of basis vectors using sparse coding consists of performing two separate optimizations, the first being an optimization over coefficients ai for each training example x and the second an optimization over basis vectors phi across many training examples at once.

Assuming an L1 sparsity penalty, learning a $^{j}\ _{i}$ reduces to solving a L1 regularized least squares problem which is convex in $^{j}\ _{i}$ for which several techniques have been developed (convex optimization software such as CVX can also be used to perform L1 regularized least squares). Assuming a differentiable S(.) such as the log penalty, gradient-based methods such as conjugate gradient methods can also be used.

4.3 Method of Optimal Directions

Tosic and Frossard (2011) When a signal vector is approximated using a frame, the number of frame vectors to be used in the approximation has to be chosen. In the frame design algorithm presented here the number of frame vectors to be used, m, is constant for all training vectors and iterations. There are various algorithms like K-SVD, MOD to achieve optimized machine learning. We have followed the MOD approach in which fixed frame sizes are updated at the end of each iterations.

The main steps in of algorithm are as follows:

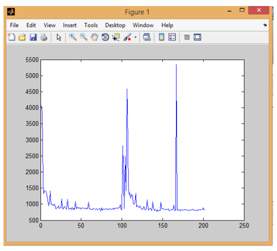
1. Begin with an initial frame Fo of size N X K, and decide the number of frame vectors to be used in each approximation, m. Assign counter variable i=1.
2. Approximate each training vector, X_I, using a vector selection algorithm: using

   where $w_l(j)$ is the coefficient corresponding to vector f_j. Find the residuals.
3. Given the approximations and residuals, adjust the frame vectors F_i.
4. Find the new approximations, and calculate the new residuals. If (stop-criterion = FALSE) + i = i + 1, go to step 3. Otherwise stop

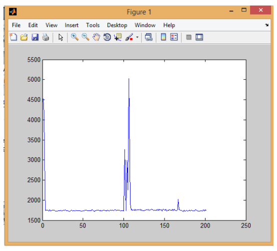
Results 5

5.1 The patterns Observed

The following patterns were observed when the nodes exhibited the expected behaviour.

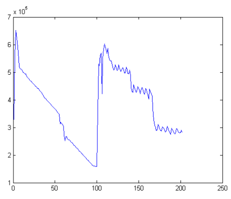


(a) Number of input bytes to the cluster



(b) Number of output bytes to the system

FIGURE 5.1: Log of Bytes


(a) Free Memory

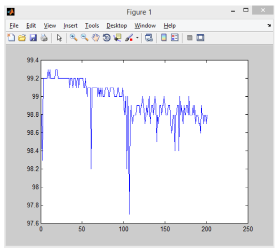
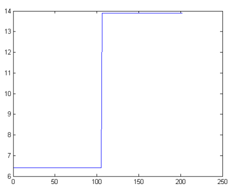

(b) Cpu Idle Time

Figure 5.2: Cpu and Memory Logs


(a) Cached Memory

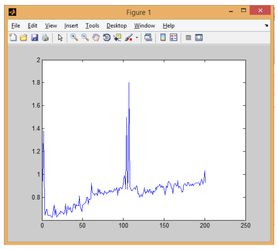
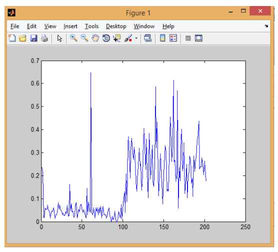

(b) Cpu Used

Figure 5.3: Cpu Usage and memory availability


(a) Cpu usage with respect to input output

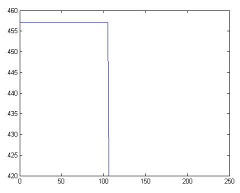
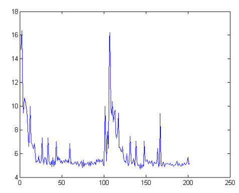
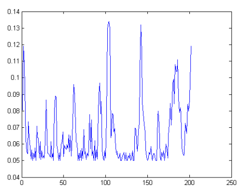

(b) Cpu Speed

Figure 5.4: Cpu Parameters

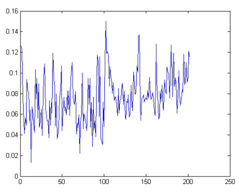


(a) Input packets

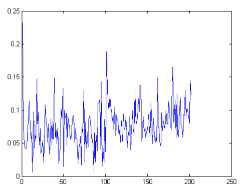


(b) Load every 1 minute

Figure 5.5: Varying Load with input packets

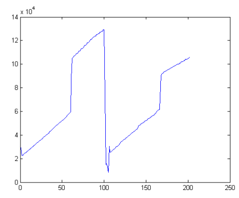


(a) Load every 5 min

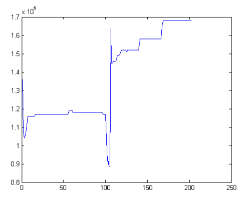


(b) Load every 15 min

Figure 5.6: Varying load across nodes



(a) Buffer Memory



(b) Cache Memory

Figure 5.7: Memory Logs

Future Work 6

Presently the sparse representations of the input log is available. This sparsely coded data will be subjected to various classifying algorithms , which will then be compared on parameters such as :

1. Time
2. Accuracy

to achieve the most optimal solution.

Once Classified the test data can then be categorised and the necessary actions for fault tolerance can be accordingly initiated. Thus producing a more robust and tolerant System.

Bibliography

Harsh Chouraria. MR2 and YARN Briefly Explained, 2012. URL "http://blog.cloudera.com/blog/2012/10/mr2-and-yarn-briefly-explained/".

IBM. What is the Hadoop Distributed File System (HDFS)? URL "http://www-01.ibm.com/software/data/infosphere/hadoop/hdfs/".

Selvi Kadirvel, Jeffrey Ho, and José A. B. Fortes. Fault management in map-reduce through early detection of anomalous nodes. In *Proceedings of the 10th International Conference on Autonomic Computing (ICAC 13)*, pages 235–245, San Jose, CA, 2013. USENIX. ISBN 978-1-931971-02-7. URL https://www.usenix.org/conference/icac13/technical-sessions/presentation/kadirvel.

Afshin Rostamizadeh Mehryar Mohri and Ameet Talwalkar. *Foundations of Machine Learning.* The MIT Press, 2012.

Sorceforgenet. Ganglia Monitoring System, 2000. URL "http://ganglia.sourceforge.net/".

I. Tosic and P. Frossard. Dictionary learning. *Signal Processing Magazine, IEEE*, 28(2):27–38, March 2011. ISSN 1053-5888. doi: 10.1109/MSP.2010.939537.

Allen B. Tucker. *Computer Science Handbook, Second Edition (Section VII: Intelligent Systems).* Boca Raton, FL: Chapman & Hall/CRC Press LLC, 2004.