Modeling Storage Performance in a HPC System

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

Master of Engineering

IN

COMPUTER SCIENCE AND ENGINEERING

by

Sandeep Kumar

Computer Science and Automation

Indian Institute of Science

BANGALORE – 560 012

JUNE 2013

To

My Family & Friends

For

Their Unconditional Love And Support

Acknowledgements

I wish to convey my most sincere gratitude to Prof. K. Gopinath for his unparalleled support in helping me complete this project.

I am indebted with gratitude to my parents, my grandparents and brother for being a constant source of inspiration during the entire tenure of the project. I also wish to extend my thanks to my friends for providing me moral as well as intellectual support during the course of the project.

Thank you all,

Sandeep Kumar
June, 2013.

Abstract

Many programs that run on a cluster process large amount of data. Most of the times the storage requirements of such programs cannot be fulfilled by a single node of the cluster, but can be satisfied by the combined storage space of the whole cluster. This require the development and usage of parallel file system that presents a unified view of the cluster storage space.

Performance of these parallel file systems are not easy to model because the performance depends on a lot of configuration parameters, like the network, disk, under lying file system, number of servers, number of clients, parallel filesystem configuration etc. We present a Multiple Linear regression model that can capture the relationship between the configuration parameters of a file system, hardware configuration and the workload configuration (collectively called features) and the performance metrics and use this to rank the features according to their importance in deciding the performance of the file system.

With the knowledge about the importance of the features and by using the prediction model, the bottleneck in the system can be identified which will be useful for example when upgrading the hardware of the cluster.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Due to data explosion in recent years, many computer programs often involves processing large amount of data. For example Facebook processes about 500 PB of data daily[2]. In 2013 Microsoft migrated over 150 PB of data from the old Hotmail to the new outlook.com in a week [3]

One way to make this possible is to use a parallel file system, which is installed on the nodes of the cluster (called servers) and create a virtual volume and present a unified view of the total disk storage capacity of the servers used. This virtual volume can be used like a normal volume. Most of the parallel file system can do auto load balancing of the data on the servers (nodes which are used in the parallel file system), which makes sure that no single node will become a bottleneck. Using techniques like **striping** (where one file is divided into pieces of predetermined fixed size and is stored in different server), we can increase the performance of the disk operation as they can be now done in parallel, and also allows one to save a huge file (whose size is more than disk capacity of any server). Using **replication** (where there are a number of copies of a file stored on different servers), we can achieve high availability of the data.

Hadoop Distributed File System (HDFS), LustreFS, GlusterFS, [4, 5, 1] are some of the common parallel file system. All of them have the same purpose but differ significantly

in their architecture. Some of them are used for general purpose storage (Lustre Gluster), whereas some are optimized for very specific kind of workloads (HDFS).

1.1 Motivation

Using a parallel file system effectively requires an understanding of the architecture and the working of the file system plus a knowledge about the expected workload for the file system. A major problem with any given parallel file system is that they come up with many configurable options which requires an in depth knowledge about the design and working of the file system and their relationship with the performance.

Together with the hardware configuration, they overwhelm the user with the number of ways one can configure the cluster, and because of this, the user generally ends up using the filesystem with the default options and this may result in an under-utilization of the resources in the cluster and a below average performance. After experiencing low performance, user tries to upgrade the system without really knowing the cause of the performance bottleneck, and many times ends up making a wrong decision, like buying more costly hard drives for the nodes when the real cause of the performance bottleneck is the network.

Through our model we have identified some of the key configuration parameters in the Gluster File system, hardware configuration and the workload configuration (features) that significantly affect the performance of the parallel file system and, after analyzing their relationship with each other, we were able to rank the features according to their importance in deciding the performance. We were also able to build a prediction model which can be used to determine the performance of the file system given a particular scenario, i.e. when we know the hardware and file system configuration.

Usefulness of the analysis

Given the importance of each of the features in a cluster, we can weigh the positive and negative points of a given design for such a cluster system and choose a design that satisfy our requirements.

The result obtained from the analysis is used to analyze some of the design options available for an experimental cluster system for the Computer System and Architecture Laboratory (CASL) and select the best possible among them, given some initial requirements. One of the main requirement was that the cluster should be *dense and low power*, i.e. it should take less space on a rack and use less power than a traditional cluster system.

Because of this, the network connectivity option was limited to Gigabit (Infiniband connectors take a lot of space, thus violating the condition of the dense systems).

To make the system use less power, we have to choose a suitable CPU for the system, which uses less power than the conventional CPU, and is not a bottleneck in the system.

The list of possible CPUs are shown in the table 1.1.

As seen from the result in table 7.1, we can see that Infiniband is big plus point for the perfor-

CPU	Points	Power Usage
Low power Atom	916	8.5 Watt
Low power Xeon E3-1265	8306	45 Watt
Low power Xeon E3-1280	9909	95 Watt
Xeon E5430	3975	80 Watt
Xeon E5650	7480	95 Watt

Table 1.1: Choice of the CPU along with their processing power (mentioned as points as calculated by the PassMark Benchmark) and Power Usage

Source: cpubenchmark.net and cpu-world.com

mance of a distributed program and the processing power of the CPU is the actual bottleneck. However if we move to a Gigabit connection and towards a denser server cluster (12 nodes or 24 nodes in a 3U rack space), then the network performance is the limiting factor and CPUs with low processing power are well capable of handling the load.

Atom CPU have the lowest power consumption, but their processing power is also very low and even when gigabit is used the CPU will be the bottleneck, so it was not a suitable option.

After weighing the processing capability and the power usage of all the CPUs, low power Xeon E3-1265 Servers with Gigabit connectivity has been chosen.

Chapter 2

Related Work

2.1 Performance prediction using statistical machine learning

The machine learning algorithm used in [6], Kernel Canonical Correlation Analysis (KCCA) is used to predict the performance of the file system. Apart from the features of the hardware and file system, they also include some features from the application itself, since KCCA can tell us the relationship between the features of the application and the features of the file system and the hardware cluster. From some initial benchmark test we observer that only one or two of the configuration features determine the performance of the parallel filesystem, so there is no point in going for a very detailed model using complex methods. For the sake of completeness we are using more features (and not only those one which have a major impact on the performance). A more detailed model is required to analyze the interaction between these features.

2.2 Performance of Superscalar Processors

[7] proposes to use Latin hypercube and L2-star discrepancy to select relevant features from a vast pool of available features and then to select a range of values for which actual simulation needs to be done. Such sophisticated techniques are not required in our case as the set of the

possible values that our feature can take is not that large and the value set for most of the features is 0 or 1.

2.3 Queuing Model

A queuing model, models a system by forming a queue for each component of the system and the relationship between the components is represented as the dependencies among the queues. [8] However to model the system, using a queuing model requires a deep understanding of the working of the system and it also does not give closed form solution Our model does not require any in depth knowledge of the system an d aims at building a black box model. The information required to use our model is minimal and readily available like the number of cores in the Network Bandwidth, Local disk read and write bandwidth, number of servers etc.

Chapter 3

Background Work

3.1 Selection of the File Systems

There are a number of parallel file systems available for use in a cluster. Some of them can be used for general purpose storage whereas some of them are optimized for some specific type of usage.

While selecting the file system for our experiment, along with the general requirements of a file system like consistency and reliability, we also focused on the following:

- **Usage scenarios:** We checked in which scenarios this file system can be used and whether it is adaptable to the different workloads demands of the user or if it is designed for only one type of specific workload.
- **Ease of installation:** It should be easy to install, like what information it requires during the installation and whether a normal user will have access to it.
- **Ease of management:** How easy it is to manage the file system, so that it can be managed without special training.
- **Ease of configuration:** A file system will require many tweaks during its life time. So it must be very easy to configure the file system and also if it can be done online (when the virtual volume is in use) then that is a huge plus point.
- **Ease of monitoring:** An admin monitoring the usage of the file system should me able to make better decisions about load balancing and whether the cluster needs a hardware

upgrade.

- **Features like redundancy, striping etc:** The parallel file systems are generally installed on commodity hardware, and as the size of cluster grows chances of a node failing also increases. So the file system must support feature like replication, which make sure that even if some of the nodes fails there is less chance of data corruption. Striping helps in reading a file in parallel which results in a huge performance gain.
- **Point of failures and bottlenecks:** We looked at the architecture of the file system to figure out how many points of failure the file system has or what can result in a bottleneck for the performance.

We studied the features of some of the most common filesystem being used as of today. The first was the **Lustre File system** [9], which is used in some of the world's fastest Super computers [10]. Improvement in the disk operation is obtained by separating the metadata from the data. To handle the metadata there is a separate metadata server used, whereas data is stored in separate servers. This metadata server is used only during resolving a file path and in the permission checks. During the actual data transmission, client directly contacts the data storage server.
Using a single metadata server increases the performance but it also becomes a *performance bottleneck* as all the requests for any file operation has to go through this. This is also a *single point of failure*, because if the metadata server is down, technically whole file system is down (however this can be tackled by configuring a machine as the backup of the metadata server which kicks in when the main metadata server fails).
Installing the Luster file system is not an easy process as it involves applying several patches to the kernel.

The next was the **Hadoop distributed file system (HDFS)** which is generally used along with the hadoop, an open source framework for distributed computing. Hadoop is generally used to process large amount of data, so HDFS is optimized to read and write large files in a sequential manner. It relaxes some of the POSIX specifications to improve the performance.
The architecture is very similar to that of the Lustre file system. It also separates the metadata

from the data and a single machine is configured as the metadata server, and therefore it suffers from the same problem of performance bottleneck and single point of failure.

Lastly we looked into **Gluster File system**, which is a evolving distributed file system with a design that eliminates the need of a single metadata server. Gluster can be easily installed on a cluster and can be configured according to the expected workload. Because of these reasons we choose Gluster file system for further analysis.

3.1.1 Gluster File System

Gluster file system is a distributed file system which is capable of scaling upto petabytes of storage and provide high performance disk access to various type of workloads.
It is designed to run on commodity hardware and ensure data integrity by replication of data (user can choose to opt out of it).

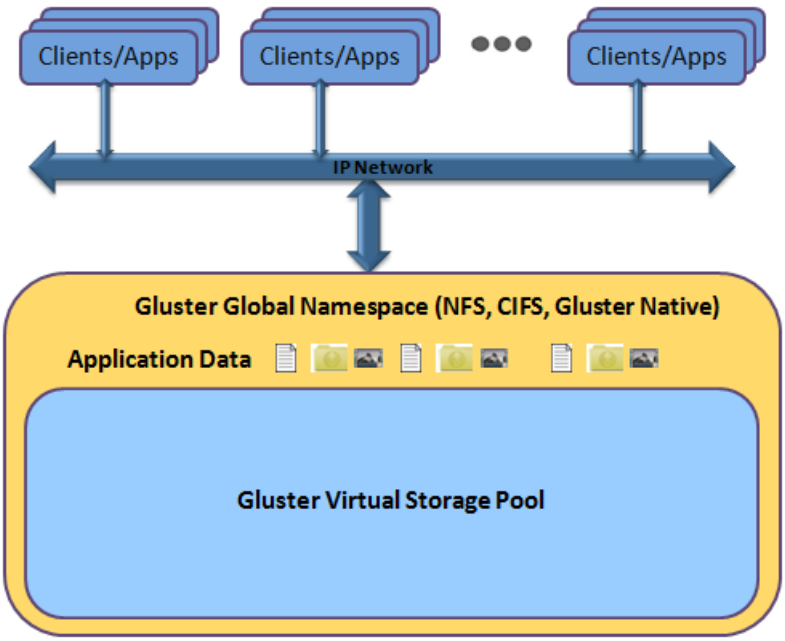

Figure 3.1: Architecture of the Gluster File system [1]

Source: http://www.gluster.org/

The Gluster file system differs from the all of the previous file systems in its architecture design.

All of the previous file systems increased the I/O performance by separating out the data from the metadata by creating a dedicated node for handling the metadata.

This achieved the purpose but during high loads, this single metadata server can be reason for a performance bottleneck and is also a single point of failure.
But in Gluster file system there is no such single point of failure as the metadata is handled by the under lying file system on top of which the Gluster file system is installed, like ext3, xfs, ext4 etc.

The main components of the Gluster architecture is:

- **Gluster Server:** Gluster server needs to be installed on all the nodes which are going to take part in creation of a parallel file system.
- **Bricks:** Bricks are the location on the Gluster servers which take part in the storage pool, i.e. these are the location that is used to store data.
- **Translators:** Translators are a stackable set of options for virtual volume, like Quick Read, Write Behind etc.
- **Clients:** Clients are the node that can access the virtual volume and use it for storage purpose.

  This virtual volume can be mounted using a number of ways, the most popular are, Gluster Native Client, NFS, CIFS, HTTP, FTP etc.

Gluster has an inbuilt mechanism for file replication and striping. It can use Gigabit as well as Infiniband for communication or data transfer among the servers or with the clients.
It also gives options to profile a volume through which we can analyze the load on the volume.
Newer version of Gluster supports Geo Replication, which replicates the data in one cluster to

an another cluster located far away geographically.

Because of these features, we chose to do further analysis on the Gluster File system.

3.2 Selection of the features

Performance of a parallel file system depends on the underlying hardware, the file system configuration and the type of the work load for which it is used.

The main configurable parameters that we have chosen to analyze are:

Configurable Parameters

Hardware configurable features:

- **Network Raw Bandwidth**
- **Disk Read Speed**
- **Disk Write Speed**
- **Base File System:**
  Which is the Base file system of the local disk, like ext3 or XFS.
- **Number of Servers:**
  Number of servers used to create the Gluster volume
- **Number of clients:**
  Number of clients simultaneously accessing the Gluster Volume

Gluster file system configurable features:

- **Striping Factor:**
  Striping Factor for a particular Gluster volume
- **Replication Factor:**
  Replication Factor for a particular Gluster volume
- **Performance Cache:**
  Size of the read cache

- **Performance Quick Read:**

  This is a Gluster translator that is used to improve the performance of read on small files. On a POSIX interface, OPEN, READ and CLOSE commands are issued to read a file. On a network file system the round trip overhead cost of these calls can be significant. Quick-read uses the Gluster internal get interface to implement POSIX abstraction of using open/read/close for reading of files there by reducing the number of calls over network from n to 1 where n = no of read calls + 1 (open) + 1 (close).
- **Performance Read Ahead:** It is a translator that does the file prefetching upon reading of that file.
- **Performance Write Behind:**

  In general, write operations are slower than read. The write-behind translator improves write performance significantly over read by using ”aggregated background write” technique.
- **Performance io-cache:** This translator is used to enable the cache. In the client side it can be used to store files that it is accessing frequently. On the server client it can be used to stores the files accessed frequently by the clients.
- **Performance md-cache:**

  It is a translator that caches metadata like stats and certain extended attributes of files.

Workload features:

- **Workload Size:**

  Size of the total file which is to be read or written from the disk (virtual volume in case of Gluster).
- **Block Size:**

  How much data to be read and written in single read or write operation.

3.2.1 Performance metrics

The performance metrics for the Gluster file system under observation are:

- Maximum write Speed

- Maximum Read Speed
- Maximum Random Read Speed
- Maximum Random Write Speed

These performance metrics were chosen because they captures most of the requirements of the application.

Chapter 4

Data Generation

4.1 Tools Used

We have used various tools to measure the different aspects of the file systems and the cluster, whose results were later use to generate the file system model.
The tools used were as follow:

4.1.1 Iozone

Iozone is a open source file system benchmark tool that can be used to produce and measure a variety of file operation.
It is capable of measuring various kind of performance metric for a given file system like: Read, write, re-read, re-write, read backwards, read strided, fread, fwrite, random read, pread, mmap etc
It is suitable for bench marking a parallel file system because it supports a distributed mode in which it can spawn multiple threads on different machine and each of them can write or read data from a given location on that particular node.
Distributed mode requires a config file which tells the node in which to spawn thread, where iozone is located on that particular node and the path at which iozone should be executed.

Example of a config file (say iozone.config):

```
node1 /usr/bin/iozone /mnt/glusterfs
node2 /usr/bin/iozone /mnt/glusterfs
node3 /usr/bin/iozone /mnt/glusterfs
node4 /usr/bin/iozone /mnt/glusterfs
```

This config file can be used to spawn 4 threads on node1, node2, node3 and node4 and execute iozone at a given path (on which the parallel file system or the virtual volume is mounted) Some of the option of the IOzone tools that was used:

-+m: run the iozone tool in distributed mode
-t : No of thread to spawn in distributed mode
-r : Block size to be used during the bench-marking (It is not the block size of the file system)
-s : Total size of the data to be written to the file system.
-i : This option is used to select the test from test suite for which the benchmark is supposed to run.
-I : This options enables the O_SYNC option, which forces every read and write to come from the disk. (This feature is still in beta and does not work properly on all the clusters)
-l : Lower limit on number of threads to be spawned
-u : Upper limit on the number of threads to be spawned
Example:

```
iozone -+m iozone.config -t <no of threads>
-r <block size> -s <workload size>
-i <test suits> -I
```

It does not matter on which node this command is issued, till the iozone.config file contains valid entries.
Iozone can be used to find out the performance of the file system for small and large files by changing the block size.

Suppose the total workload size is 100 MB, then if the block size is 2 KB then its like *reading or writing 51200 files* to the disk. Whereas if the block size is 2048 KB then its like *reading or writing to 50 files*.

4.1.2 Iperf

Iperf is an open source tool used to measure the raw network bandwidth between two nodes. To measure the bandwidth between two nodes say node1 and node2, start the iperf server on node1 by issuing the command:

```
iperf -s
```

then on the client side connect to the server by issuing the command:

```
iperf -c node1
```

(assuming there is an entry for node1 in the /etc/hosts file).

4.1.3 MATLAB AND IBM SPSS

Matlab and IBM SPSS, is used to create the model from the generated data and to calculate the accuracy of the model.

4.2 Experiment set up

4.2.1 FIST Cluster

The whole of the FIST cluster comprises of total 36 nodes in total, out of which 1 node is the login node and is used to store all the users data. The remaining 35 nodes are used for computing purposes.

The configuration of the 35 nodes are:

CPU:	8 core Xeon processor @ 2.66 GHz
RAM:	16 GB
Connectivity:	Gigabit (993 Mb/sec), Infiniband (5.8 Gb/sec)
Disk:	3 TB 7200 RPM (5 nodes), 250 GB 7200 RPM (All of them)

Out of these 35 nodes, we reconfigured the 5 of them with 3TB hard disk, and used them as Gluster servers for our experiments.

Rest of the nodes were chosen to act as the client.

This setting made sure that if a client wants to write or read data to the virtual volume (mounted via NFS or Gluster Native client from the servers), the data has to go outside the machine, and the writing of data to a local disk (which probably will increase the speed) is avoided.

4.2.2 Atom Cluster

In the atom cluster there are 4 nodes. The configuration of each of the atom nodes are as follow:

CPU:	Atom, 4 Core processor @ 2 GHz
RAM:	8 GB
Connectivity:	Gigabit (993 Mb/sec)
Disk:	1 TB, 7200 RPM

Clients are connected to the cluster from outside via a Gigabit connection.

4.3 Data generation

The data generation for the analysis was a challenge since some settings for some of the features vastly degrades the performance of the file system.

For example, when the block size is set at 2 KB, the speed for writing 100 MB of file (to disk and not in the cache, by turning on the O_SYNC) was **1.5 MB/sec**.

The largest size of the workload in our experiment is 20GB. So the time taken to write 20GB

to the disk will be around 30000 seconds, i.e **8.3 Hours**.
In the worst case when number of clients is 5 and replication is also 5, the total data written to the disk will be 500 GB (20*5*5).
The time taken to write 500 GB of data will be around 8.6 days, and including time taken for reading, random reading and random writing the data, the total time will be ,

8.6 + 3.34+3.34+8.6 = 23.88 days.

The read and write speed achieved above is maximum, when cache size is equal to or more than 256 KB, as shown in table 6.9.

If we try to model all of them together (using the normal values for the Gluster configuration), Gluster configuration is neglected as they have negligible effect on the performance when compared to Hardware and Workload configuration.
But they become important once the hardware and type of workload is fixed, Gluster configuration is used to configure the File system in the best way possible.
For this reason we decided to create two models, one for the hardware and the workload configuration and another for the Gluster configuration.

For the first model the parameters varied are listed in table 4.1. The size of the workload was dependent on the system RAM, which is 16 GB. If we try to write a file whose size is smaller than the 16 GB then the file can be stored in the RAM and we will get a high value for read and speed. To avoid this workload size of 20 GB was chosen.
But cache plays an important role when we try to read or write small files. We capture that behavior of the file system by making the workload size less than the RAM.

For the second model, the workload size was fixed at 100 MB and table 4.2 contains the list of the parameters and the values that they take during benchmark process.
Disk was unmounted and mounted before each test to avoid the caching effect

Feature	Values
Network	Gigabit and Infiniband
Disk Read Speed	117 MB/sec, 81 MB/sec
Disk Write Speed	148 MB/sec, 100 MB/sec
Base File system	Ext3, XFS
No of Servers	1,2,3,4,5
Striping	0,2,3,4,5
Replication	0,2,3,4,5
No of Clients	1,2,3,4,5
Workload Size	100 MB, 200 MB, 500 MB, 700 MB, 1 GB, 10 GB, 20 GB

Table 4.1: The Hardware and Workload configuration parameters that were varied during the data generation on **FIST cluster**

Feature	Values
Block Size	2 KB, 4 KB, 8 KB to 8192 KB
Performance Cache Size	2 MB, 4 MB, 8 MB to 256 MB
Write Behind Translator	ON/OFF
Read Ahead Translator	ON/OFF
IO Cache Translator	ON/OFF
MD Cache Translator	ON/OFF

Table 4.2: The Gluster configuration parameters that were varied during the data generation on **ATOM cluster**

Chapter 5

Analysis of the Data

As stated earlier our goal is to figure out the relationship and impact of the hardware, Gluster and workload features on the performance features.
Multiple linear regression fits our criteria well as it can be used for:

- **Prediction:** Predicting the value of Y (the performance metric), given the cluster environment and the Gluster configuration (X).
- **Description:** It can also be used to study the relationship between the X and Y, and in our case we can study the impact of the various configurations on the performance, for example, which of them affects the performance the most and which has the least effect on the performance.

5.1 Assumptions

Before applying multiple linear regression, the condition of linearity must be satisfied i.e. there should be a linear relationship among the Dependent variable and the Independent variables to get a good fit.

The relationship among the configuration features and the performance features can be seen in table 5.1.

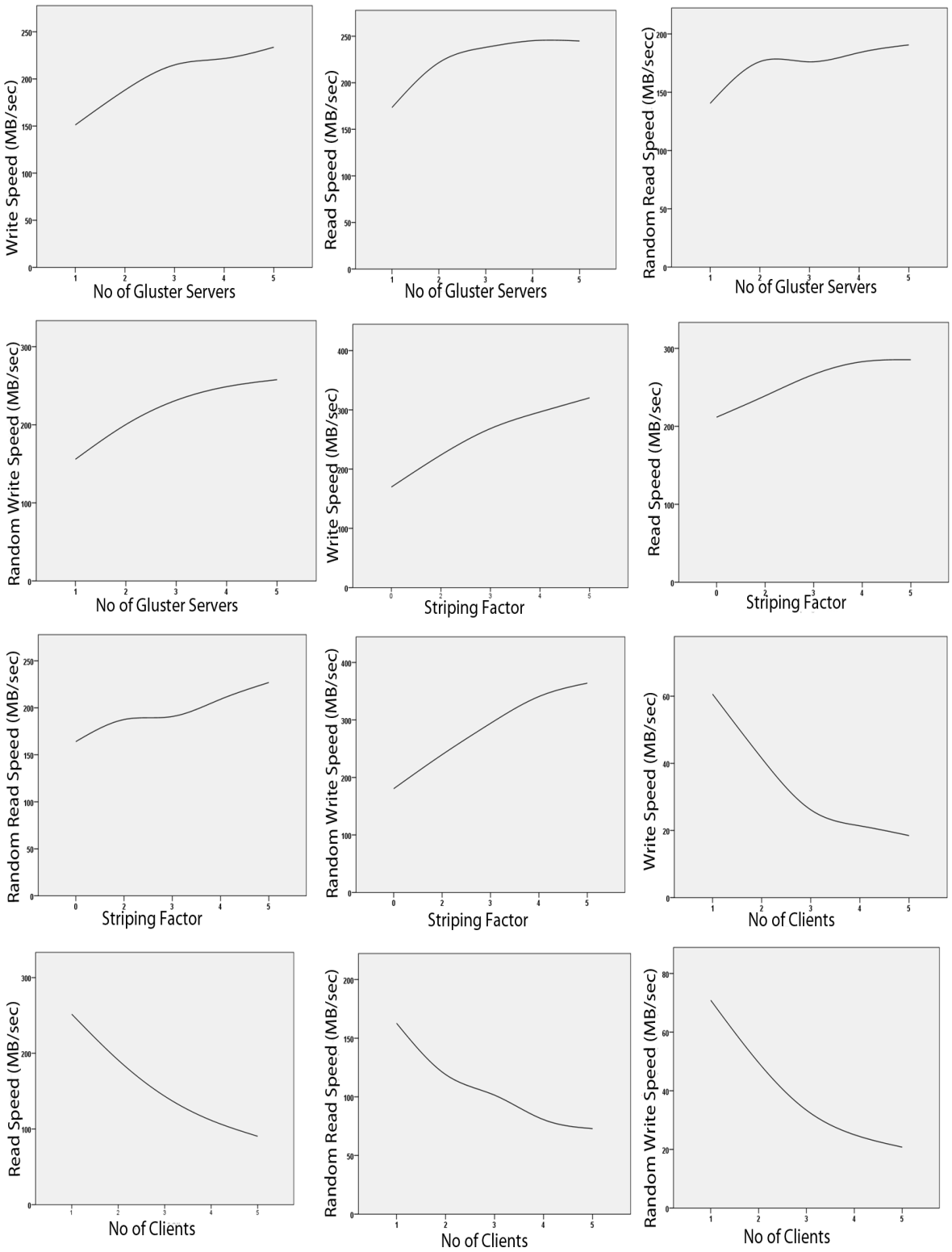

Table 5.1: Table showing an approx. linear relationship between **Number of Gluster Servers, Striping and Number of client** with the performance metrics **Write Speed, Read Speed, Random Read Speed, Random Write Speed**

5.2 Applying Multiple Linear Regression

5.2.1 Multiple Regression

To study the relationship between more than one independent variable (features) and one dependent variable (performance) multiple regression technique is used.
The output of the analysis will be coefficients for the independent variables, which is used to write equation like:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} ... + \beta_p x_{ip} + \epsilon_i, i = 1, ..., n$$

or,

$$y_i = x_i^T \beta + \epsilon_i$$

where,

y_i: Dependent variable's i^{th} sample (Performance metric).

x_i: Independent variable's i^{th} sample (Configuration parameter).

ϵ: error term

β: Coefficients (output of the analysis)

n: No of data sample we have

p: No of independent variable we have.

In vector form:

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \; X = \begin{pmatrix} 1 & X_1^T \\ 1 & X_2^T \\ \vdots & \vdots \\ 1 & X_n^T \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \ddots & & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}, \; \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}, \; \epsilon = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{pmatrix}$$

β can be calculated by the formula:

$$\beta = (X^T X)^{-1} X^T Y$$

5.2.2 Evaluating the model

The accuracy of the model can be checked by using:

- **$\mathbf{R^2}$ test to check the goodness of fit:**

  It represents the proportion of the variance in dependent variable that can be explained by the independent variables.

  So, for a good model we want this value to be as high as possible.

  R^2 can be calculated in following way:

$$TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2$$
$$SSE = \sum_{i=1}^{n}(y_i - \hat{y})^2$$
$$SSR = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$$

  where,

  TSS: Total Sum of square

  SSE: Sum of Squares Explained

  SSR: Sum of Square Residual

  y: Dependent Variable

  $\bar{y}$: Mean of y

  $\hat{y}$: Predicted value for y

  n: Total no of data samples available

$$R^2 = 1 - \frac{SSE}{TSS}$$
$$R^2 = \frac{SSR}{TSS}$$

  value of R^2 varies from [0, 1]

- **Adjusted $\mathbf{R^2}$** :

  R^2 is generally positive biased estimate of the proportion of the variance in the dependent variable accounted for by the independent variable, as it is based on the data itself.

  Adjusted R^2 corrects this by giving a lower value than the R^2, which is to be expected

in common data.

$$Adjusted\ R^2 = 1 - \left(\frac{n-1}{n-k-1}\right)(1-R^2) \tag{5.1}$$

where,

n= total no of samples

k = total no of features or independent variables.

- **Predictor importance:**

  Predictors are ranked according to the sensitivity measure which is defined as follow:

$$S_i = \frac{V_i}{V(Y)} = \frac{V(E(Y|X_i))}{V(Y)} \tag{5.2}$$

  where,

  V(Y) is the unconditional output variance.

  Numerator is the expectation E over X_{-i}, which is over all features, expect X_i, then V implies a further variance operation on it.

  S is the measure sensitivity of the feature i.

  S is the proper measure to rank the predictors in order of importance.[11]

  Predictor importance is then calculated as normalized sensitivity:

$$VI_i = \frac{S_i}{\sum_{j=1}^{k} S_j} \tag{5.3}$$

  where,

  VI is the predictor importance. We can calculate the predictor importance from the data itself.

- **Cross Validation test:**

  The R^2 test is the measure of goodness of fit on whole of the training data. Cross validation test is used to check the accuracy of the model when it is applied to some unseen

data. For this the data set is divided into two sets called training set and the validation set. The size of these set can be decided as per the requirement. We set the size of the training set size equal to 75% of the data set and the validation set was 25%. The distribution of data samples to these set was completely random.

Chapter 6

Results

Two separate analyses are done, one for the hardware and the workload configuration and another for the Gluster configuration.
In the first model, the assumption of linearity holds and hence the multiple linear regression model gives an accuracy of 75% - 80%. However, due to the huge impact of the cache and block size in the Gluster configuration, the assumption of linearity is no longer valid. So we used predictor importance analysis to rank the features according to their importance.

6.1 Model for the Hardware configuration

6.1.1 Maximum Write Speed:

The output of the Multiple Linear regression analysis, i.e the coefficients corresponding to each of the configurable feature is shown in the table 6.1 From the coefficient table we can see that the Random write performance is most effected by the Network, followed by Replication, No of clients , No of servers and Striping.
The signs tell us the relationship, for example, on increasing the Network bandwidth the write performance increases, and increasing the replication factor decreases the write speed.

Feature	Coefficient
Constant	-120.485
Network	271.936
Disk Read Speed	≈0
Disk Write Speed	≈0
Base File system	4.149
No of Servers	30.729
Striping	10.380
Replication	-59.182
No of Clients	-34.359
Workload Size	-0.16

Table 6.1: The coefficients for the model for Write performance of the parallel file system

Test	Value
Adjusted R^2	.735 (73.5%)
Cross Validation	75.4%

Table 6.2: Accuracy of the Write Model. Cross validation was done by training the model on 75% of data and then running validation on the rest of the 25% data

6.1.2 Maximum Read Speed:

The output of the Multiple Linear regression analysis, i.e the coefficients corresponding to each of the configurable feature is shown in the table 6.3

Feature	Coefficient
Intercept	-142.931
Network	237.502
Disk Read Speed	≈0
Disk Write Speed	≈0
Base File system	-4.406
No of Servers	15.890
Striping	3.508
Replication	-21.058
No of Clients	-21.094
Workload Size	-.004

Table 6.3: The coefficients for the model for Read performance of the parallel file system

From the coefficient table we can see that the Read performance is most effected by the Network followed by Replication, No of clients and No of servers.

Replication is having an negative effect on the read performance instead of no effect or some

positive effect because of the replication policy of the Gluster.

If we have specified a replication factor of say n, then when a client tries to read a file, it has to go to every server to ensure that the replicas of that file is consistent everywhere and if it is not consistent, read the newest version and start the replication process.

Because of this overhead the read performance of the file system drops with an increase in the replication factor.

Test	Value
Adjusted R^2	.801 (80.1%)
Cross Validation	82.5%

Table 6.4: Accuracy of the Read Model. Cross validation was done by training the model on 75% of data and then running validation on the rest of the 25% data

6.1.3 Maximum Random Read Speed:

The output of the Multiple Linear regression analysis, i.e the coefficients corresponding to each of the configurable feature is shown in the table 6.5 From the coefficient table we can

Feature	Coefficient
Intercept	-39.676
Network	162.003
Disk Read Speed	≈0
Disk Write Speed	≈0
Base File system	-17.888
No of Servers	14.204
Striping	-.961
Replication	-22.799
No of Clients	-16.546
Workload Size	-.003

Table 6.5: The coefficients for the model for Random read performance of the parallel file system

see that the Random read performance is mostly effected by the Network, then by Replication followed by Base file system, No of clients and No of servers.

Test	Value
Adjusted R^2	.744 (74.4%)
Cross Validation	75.2%

Table 6.6: Accuracy of the Random Read Model. Cross validation was done by training the model on 75% of data and then running validation on the rest of the 25% data

6.1.4 Maximum Random Write Speed:

The output of the Multiple Linear regression analysis, i.e the coefficients corresponding to each of the configurable feature is shown in the table 6.7

Feature	Coefficient
Intercept	-114.066
Network	230.091
Disk Read Speed	≈0
Disk Write Speed	≈0
Base File system	-2.190
No of Servers	34.241
Striping	13.538
Replication	-60.986
No of Clients	-39.060
Workload Size	-0.13

Table 6.7: The coefficients for the model for Random Write performance of the parallel file system

From the coefficient table we can see that the Random write performance is most effected by the Network, followed by Replication, No of clients , No of servers and Striping.

Test	Value
Adjusted R^2	.745 (74.5%)
Cross Validation	76.9%

Table 6.8: Accuracy of the Random Write Model. Cross validation was done by training the model on 75% of data and then running validation on the rest of the 25% data

6.2 Model for the Gluster Configuration parameters

Data samples were generated for 100 MB of file. Since the size of the file is much smaller than the RAM available (8 GB), then during the benchmarks, cache effect will be present. To avoid this, we enabled the O_SYNC option in the Iozone benchmark test which will ensure that every read and write operation is done directly to and from the disk. But we also want to see the effect of cache when we turn on some translators that are cache dependent. So we ran every benchmark test twice, one time with the O_SYNC option ON and another time with the O_SYNC option OFF.

6.2.1 With O_SYNC option ON

When the O_SYNC option is ON, i.e. every read or write operation goes to the disk, then the dominating feature is the block size, which is to be expected, as increasing cache size and turning on translators will not help as we are forcing every operation to come from the disk. We are doing it for a 100 MB file, but same kind of behavior can be seen when dealing with large files (size greater than the RAM size).

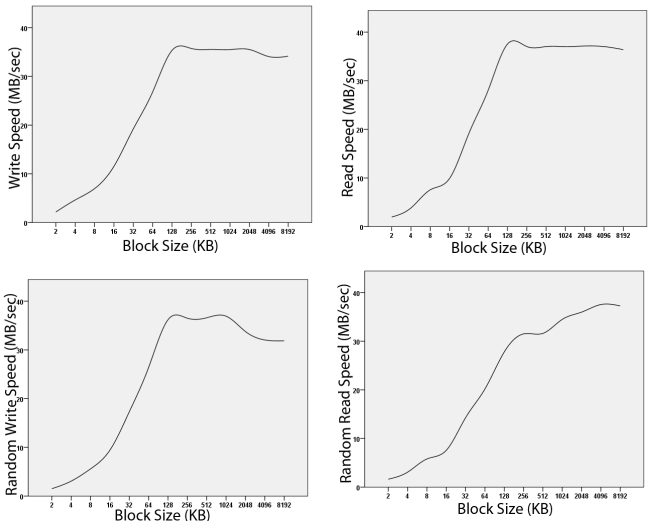

Table 6.9: Relationship of the Performance metrics with the Block Size with O_SYNC On

6.2.2 With O_SYNC option OFF

With the O_SYNC option off, the cache effect can be seen in the Read performance of the system, where as the Block size is still the dominant feature

The size of the workload is 100 MB only, and still the block size is the dominating factor but the effect of cache can be seen in the performance of read speed and random read speed as Gluster have some translator (read ahead and quick read) that uses cache to optimize the performance of read of the files, specially for the small files.

The performance of write and random write follows the same pattern as early.

We can see that the speed of the read operation remain the same with change in the block size as compared to early when the O_SYNC option was ON and everything has to come from disk.

Feature	Importance
Block Size	≈ 1
Rest	≈ 0

Table 6.10: Predictor Importance with O_SYNC ON

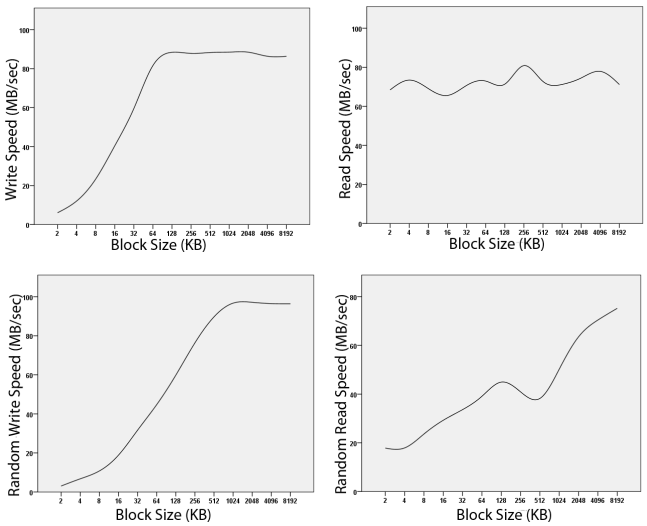

Table 6.11: Relationship of the Performance metrics with the Block Size with O_SYNC OFF

Feature	Importance
Block Size	≈ 1
Rest	≈ 0

Table 6.12: Predictor Importance with O_SYNC OFF for Write and Random Write

Feature	Importance
Block Size	0.78
Cache Size	0.22
Rest	≈ 0

Table 6.13: Predictor Importance with O_SYNC OFF for Read

Feature	Importance
Block Size	0.95
Cache Size	0.05
Rest	≈ 0

Table 6.14: Predictor Importance with O_SYNC OFF for Random Read

Chapter 7

Performance in a Real World Application

To verify the result we compared the performance of an Ocean Modeling code in different scenarios.

Ocean code is used in the modeling of the ocean behavior given some initial observation. It uses ROMS modeling tool to do the calculation and MPI to run on a cluster by spawning 48 threads. NTIMES factor controls the number of iterations inside the code and is the major factor in deciding the run time of the code The optimal value of NTIMES is 68400 (changes depending upon the requirement). During its execution it processes more than 500 GB of data. We tested its performance in situations as listed in the table 7.1

From the table 7.1 we can see that the network indeed is the most dominating factor in deciding

Configuration	Time Taken
4 Atom Servers, on Gigabit	$\approx$ 2311 Minutes, $\approx$ 38.5 Hours
4 Gluster Servers, on Gigabit	$\approx$ 1916 Minutes, $\approx$ 31 Hours
2 Gluster Servers, on Infiniband	$\approx$ 478 Minutes, $\approx$ 7.9 Hours
4 Gluster Servers, on Infiniband	$\approx$ 282 Minute, $\approx$ 4.7 Hours
10 Gluster Servers infiniband	$\approx$ 144 Minutes, $\approx$ 2.4 Hours

Table 7.1: Performance of the Ocean code in different scenarios

the performance of the code. On gigabit network, the performance of more number of servers falls behind the performance of less number of servers on infiniband and as we increase the number of servers on infiniband the time taken to complete the code decreases.

We can see from the time taken to run the code that if Gigabit network is used, the performance of the Atom CPU is very close to that of Xeon because the network was the bottleneck here. This fact was helpful in deciding which low power servers to be bought for CASL lab.

Chapter 8

Conclusion and Future work

For a first order model, Multiple Linear Regression is good enough, as the performance is mainly decided by only one or two features (network in case of the gluster file system). For a much more detailed model which can explain the interaction between the various of the features, a much more sophisticated tool is required.
The cache of the system also plays an very importance role in deciding the performance but requires support of the file system (gluster file system optimizes the read speed using the cache of the system).

- **Unified Model:**
  Because of the huge difference between the impact of the hardware configuration features and the Gluster configuration features, a separate model was required.
  A unified model can be developed by adding some bias to the Gluster configuration features which will increase its importance and the we can incorporate all of the configuration features in one model.
- **Automate the whole process:**
  Build a tool which can do necessary things like generate the data and then do appropriate analysis on it automatically and give us the desired result.

Bibliography

[1] Developers, GlusterFS. The Gluster web site. *http://www.gluster.org/about/*.

[2] Donna Tam. Facebook processes more than 500 TB of data daily. *http://news.cnet.com/8301-1023_3-57498531-93/facebook-processes-more-than-500-tb-of-data-daily/*, August 2012.

[3] Dick Craddock. Outlook.com: 400 million active accounts, Hotmail upgrade complete and more features on the way. *http://blogs.office.com/b/microsoft-outlook/archive/2013/05/02/outlook-com-400-million-active-accounts-hotmail-upgrade-complete-and-more-features-on-the-way.aspx/*, May 2013.

[4] Dhruba Borthakur. The hadoop distributed file system: Architecture and design. *Hadoop Project Website*, 11:21, 2007.

[5] Daniel Pilaud, N Halbwachs, and JA Plaice. Lustre: A declarative language for programming synchronous systems. In *Proceedings of the 14th Annual ACM Symposium on Principles of Programming Languages (14th POPL 1987). ACM, New York, NY*, volume 178, page 188, 1987.

[6] Archana Sulochana Ganapathi and David Patterson. *Predicting and optimizing system utilization and performance via statistical machine learning*. PhD thesis, University of California, Berkeley, 2009.

[7] PJ Joseph, Kapil Vaswani, and Matthew J Thazhuthaveetil. A predictive performance

model for superscalar processors. In *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 161–170. IEEE Computer Society, 2006.

[8] Eno Thereska and Gregory R Ganger. Ironmodel: Robust performance models in the wild. In *ACM SIGMETRICS Performance Evaluation Review*, volume 36, pages 253–264. ACM, 2008.

[9] Lustre File System Architecture. *http://wiki.lustre.org/index.php/Main_Page*.

[10] Wikipedia. Lustre use. *http://en.wikipedia.org/wiki/Lustre*.

[11] Andrea Saltelli, Stefano Tarantola, Francesca Campolongo, and Marco Ratto. *Sensitivity analysis in practice: a guide to assessing scientific models*. Wiley, 2004.