UNIVERSITY OF OSLO
Department of Informatics

Investigation of Cluster and Cluster Queuing System

SAERDA HALIFU

Network and System Administration

Oslo University College

May 19, 2008

Investigation of Cluster and Cluster Queuing System

SAERDA HALIFU

Network and System Administration
Oslo University College

May 19, 2008

Abstract

Cluster became main platform as parallel and distributed computing structure for high performance computing. Following the development of high performance computer architecture more and more different branches of natural science benefit from huge and efficient computational power. For instance bio-informatics, climate science, computational physics, computational chemistry, marine science, etc. Efficient and reliable computing power may not only expending demand of existing high performance computing users but also attracting more and more different users. Efficiency and performance are main factors on high performance computing. Most of the high performance computer exists as computer cluster. Computer clustering is the popular and main stream of high-performance computing. Discover the efficiency of high performance computing or cluster is very interesting and never enough as it is really depending on different users. Monitoring and tuning high performance or cluster facilities are always necessary. This project focuses on high performance computer monitoring. Comparing queuing status and work load on different computing nodes on the cluster. As the power consumption is main issue nowadays, our project will also try to estimate power consumption on these special sites and also try to support our way of doing estimation.

Acknowledgment

First of all, I would like to express my deep appreciation and special thanks to my Supervisor Professor Mark Burgess, for his endless help, understanding, and his time for supervising me.

I also want to thank Professor Dr. Klaus Johannsen, Eirik Thorsnes at parallel lab, Bergen center of computational science. For allowing me to use their production cluster system and supporting my thesis and given me helpful advice.

Special thanks also go to Kyrre Begnum, for his inspiration and helpful discussion.

Thanks to my father and mother for their endless support and their love that encourage me to keep on tracing my dreams.

Thanks to my brother for his advice and discussion through my thesis. He knows how to encourage people to be strong.

Thanks to all my classmates, for the supportive and friendly environment that has existed throughout this master program.

Oslo, May 2008

Saerda.Halifu

Contents

1 Introduction **8**
- 1.1 Motivation . . . 8
- 1.2 Problem Definition . . . 8
- 1.3 Research Goal . . . 9
- 1.4 Outline of Remaining Chapter . . . 10

2 Background Information **12**
- 2.1 CPU and Computing . . . 12
  - 2.1.1 CPU . . . 12
- 2.2 Different Computing Method Overview . . . 13
  - 2.2.1 Serial Computing . . . 13
  - 2.2.2 Parallel Computing . . . 14
  - 2.2.3 Different Parallel Computing . . . 16
- 2.3 MPI . . . 18
- 2.4 Resource Management and Scheduling . . . 20
  - 2.4.1 Portable Batch System . . . 22
  - 2.4.2 Scheduling . . . 28
  - 2.4.3 Queue . . . 29
  - 2.4.4 Load Balancing . . . 32

3 Literature survey and discussion **36**
- 3.1 Performance Metrics and Monitoring . . . 36
- 3.2 Power Consumption and CPU Utilization . . . 37
- 3.3 About Queue . . . 37

4 Experiment Design **38**
- 4.1 Discovery on Production Cluster . . . 38
  - 4.1.1 CPU utilization and Queue . . . 40
  - 4.1.2 Job Overview . . . 41
- 4.2 CPU Utilization, Power Consumption and Temperature . . . 42
  - 4.2.1 CPU Utilization and Temperature . . . 42
  - 4.2.2 Power Consumption . . . 42

5 Results **44**
- 5.1 Queuing Property of Cluster . . . 44
  - 5.1.1 Total Jobs . . . 44
  - 5.1.2 Active Jobs . . . 45
  - 5.1.3 Idle Jobs . . . 46

5.1.4 Blocked Jobs 47
5.1.5 Cluster CPU Utilization and Queue 48
5.2 CPU Utilization VS Temperature 52
5.3 Estimation of Power Consumption of Cluster 54

6 Conclusions and Future work **56**
6.1 Conclusions 56
6.1.1 Evaluation on Cluster 56
6.1.2 Queuing Properties on The Cluster 56
6.1.3 Estimated Power Consumption of The Site 57
6.1.4 CPU Utilization and CPU Temperature 57
6.2 Future Work 57

A Appendices **63**
A.1 Script for collecting Queuing information on the master node . 63
A.2 Script for collecting CPU Utilization 63
A.3 Log Data for Queuing status 63

List of Figures

2.1 CPU Architecture 13
2.2 Serial computing architecture 14
2.3 Parallel Computing Architecture 15
2.4 Shared Memory Parallel Computing 16
2.5 Distributed Memory Parallel Computing 17
2.6 Shared and Distributed Memory Parallel Computing 17
2.7 Parallel Message passing Mechanism 19
2.8 MPI Programming Process 20
2.9 Normal Human Interactive Process 23
2.10 Batch Process 24
2.11 PBS Server-Client Communication Structure 27
2.12 First in First Out 28
2.13 Last in First Out 28
2.14 Round Robin 29
2.15 Backfill Example 30
2.16 Queuing structure 30
2.17 M/M/1 32
2.18 $M/M/1^k$ 32
2.19 M/M/K 33

4.1 System Layout 39
4.2 Job Overview 41

5.1 Total Job 45
5.2 Active Jobs 46
5.3 Idle Jobs 47
5.4 Blocked Jobs 48
5.5 Node User CPU Utilization 49
5.6 Node Idle CPU Time 50
5.7 Jobs in all status collected 51
5.8 CPU Utilization 52
5.9 CPU Temperature 53

Chapter 1

Introduction

1.1 Motivation

The more computational power we have the more we will ask. How to efficiently use the existing resources is the main issue. High performance computing normally refers to cluster. There are many component come together to construct fully functioned computational cluster. Highly intensive cooperation between all necessary components is required for computational cluster. In computational cluster generally we have two main parts, first part is master node where we queue the incoming job and do scheduling. Second part is computing node where we do actual computation. Normally we have one master node and several computational nodes. Those two parts of computational cluster must work on highly cooperative manner. In our project what we want to discover is collaboration between master node and computational node in particular computational environment. Work load analysis on both master node and computational node is necessary.

1.2 Problem Definition

In this project we are not going to solve very specific problem. Our main concern is reliability of our queuing system and job distribution. As a production environment current high performance computing facilities provides computational power especially to Bio-informatics, physics and mathematics group. We want to see real time performance of our serves.

Examine the communications and cooperations between master node and computational node of cluster, discover algorithms of queuing and scheduling jobs are our main goal. At the same time discovering the possibility to prevent dead node and dead job with auto configuration management tool will be examined.

1.3 Research Goal

The overall goal of the project is able to examine performance efficiency of real production high performance computing environment. And based on our statistical data we are be able to give site administrators some advance suggestion about site configuration and operation. However there are some certain points:

1. Evaluate our high performance cluster performances.

   In that case, to see how much time is actually spending on user task is interesting. Early research[1] shows that, different types of software have been developed for monitoring system. But we will have different concern here as we have special focus on CPU. If system is busy does not tell anything about efficiency, it might busy because of system operation, the only way to know that is check actual CPU time distribution in the computing node on the system. In our experiment we selected 22 nodes out of 86 computing nodes. There are 4 racks, rack 1,2,3 has 22 nodes rack 4 has 20 nodes , from rack 1,2,3 we selected two nodes from top , two nodes from middle, two nodes from bottom, from rack 4 we selected 2 nodes from top, and 2 nodes from bottom. The reason those nodes has been selected is that the way we get the statistical data from different part of the rack, this will also help us to illustrate if there is differences in job distribution regarding to physical location of computing nodes.

   Similar research has been done in related topic, according to research[2], they showed interesting metrics to evaluating high performance facilities, they classified the metrics and discussed correlation between those metrics and related application, that helped us to find right metrics to look into. In research[3]they showed how to manage statistical information from cluster monitoring in special cases, which brought us the idea about find reasonable explanation of our measured data after all. In research [4], different model of how to evaluating cluster performance, and their modeling method is very interesting, it helped us to build our experement model.

2. Discover Queuing Properties Cluster Site.

   Understand how queue is handled in the system, for example if there are any relation or affection between incoming jobs and CPU status of computing nods? If there is more jobs coming, will it affect normal operation or normal performance of whole cluster? Related research[5] has been done about how to handle queue storms, sudden increase of queue will stress cluster system, the idea about how to handle that queue helped us to understand queuing properties of our system.

3. Possibility of estimating power consumption according to CPU utilization.

Not all the sites have sensor instrument to measure actual power consumption. Try to find most practical way that can be used to estimating power consumption of sites according to CPU utilization is interesting. Research[6] makes us believe that estimation of power consumption from CPU utilization is realistic.

4. Extra task will be examine relationship between CPU utilization and temperature.

   That will support our belive about reflecting power consumption through CPU utilization. According to [7][8] one can say that increase of temperature can reflect increase usage of power. That will provide us the relation can be seen according to the experiment between CPU utilization and power consumption.

1.4 Outline of Remaining Chapter

In second chapter we will discuss about background information and some related research especially the core technology behind super computing industry. And will show some basic methods to achieve parallel computing. Moreover discuss software architecture about supercomputer.

In third chapter we will mainly describe our supercomputer facilities and main usage. At the same time we will discuss about our experiment design and it is reliability. As we planned experiment in three different sections, we will explain all three designs.

In chapter four we will present experiment result. Then will discuss our result to see if there is any kind of information we did not expect or we never put in consideration. And also give out explanations about our presented result. The first part of the experiment mainly focus on utilization of computing nodes and statistics of job queue. The second part of experiment is about discover the relation between CPU utilization and CPU generated heat. The third part of experiment is about power consumption. We will try to predict power consumption according to CPU utilization.

In the last chapter we will come up with some scientific conclusion. Conclusion will be based on our experiment and main theory behind all this technology. And evaluate our site configuration, necessary advices will be given and efficiency assessment of this specific super cluster operations will be given.

Chapter 2

Background Information

2.1 CPU and Computing

2.1.1 CPU

Central processing unit, also known as microprocessor or processor, is the main component of computer can be considered as the brain of the computer system. It processes everything from basic instructions to complex functions.[9]

CPU has following general components:

- ALU-arithmetic logic unit
  Mainly responsible for calculations and logical comparisons and decisions

- Registers
  Data are stored here, there are some of the registers are visible for user which means they are programmable, but some of them are invisible.

- Internal CPU Bus
  Communication between ALU, registers and control.

- Controller
  Determines who does what and when, used to control signals to control what every circuit is doing at any given clock time.

As a program is executed, data flow from RAM through external bus, which connects the CPU to RAM. Then the data are decoded by a processing unit called the instruction decoder that interprets and implements software instructions. From the instruction decoder the data pass to the arithmetic logic unit, which performs calculations and comparisons. Data may be stored by the ALU in temporary memory locations called registers where it may be retrieved

quickly. The ALU performs specific operations such as addition, multiplication, and conditional tests on the data in its registers, sending the resulting data back to RAM or storing it in another register for further use. During this process, a unit called the program counter keeps track of each successive instruction to make sure that the program instructions are followed by the CPU in the correct order.[9]

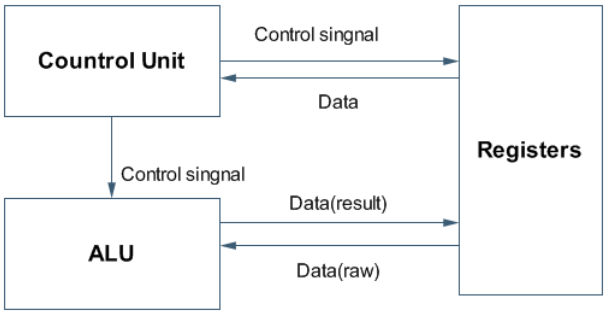

Figure 2.1: CPU Architecture-ALU is the main unit responsible for calculation, Registers are responsible for storing data, Control unit tells ALU what to do when and how.

2.2 Different Computing Method Overview

After first computer came out from 1946, the computing power is not able to fulfill the demand of scientific computing. Over six decades of period, scientist and engineers try to reach the limit of computational power, as the research going on they found many way to speed up computing power, even though all those development still far behind the request. We will briefly explain two main computing methods which are contribute the most.

2.2.1 Serial Computing

Serial (sequential) computing is very traditional way of computing. Originally all programs written to be execute on serial processing. Serial computing means solve the problem one after each other, in any single moment there is only one instruction may execute on the processor.[10]

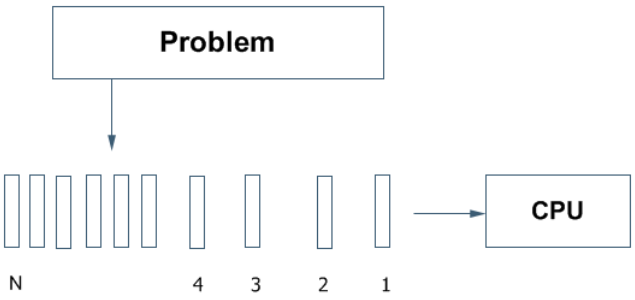

Figure 2.2: Serial computing architecture-If there is N jobs suppose to send to CPU, they will executed one after each other.

Here is the example of serial computing algorithm:

```
1: f (a, b)
2: {
3: c= a?b
4: d= 2?c
5: }
```

In this example line 4 cannot be executed until line 3 is executed, which means all problems solves one after each other. Therefore disadvantage of serial computing is the speed of computing which is highly depend on physical facilities like how fast the CPU is, and how fast the communication between memories and CPU. At the same time it is very expensive and difficult to make serial computer even faster. But the good thing about serial computing is everything is simple and transparent.

2.2.2 Parallel Computing

Parallel computing is a method where many instructions can be carried out simultaneously. In our cluster environment, almost all the jobs will be executed in parallel way. So when user submitted jobs it will be divided to different nodes to execute. Principle of parallel computing tells that large problem always can be divided to smaller ones, and can be solved concurrently. [10]

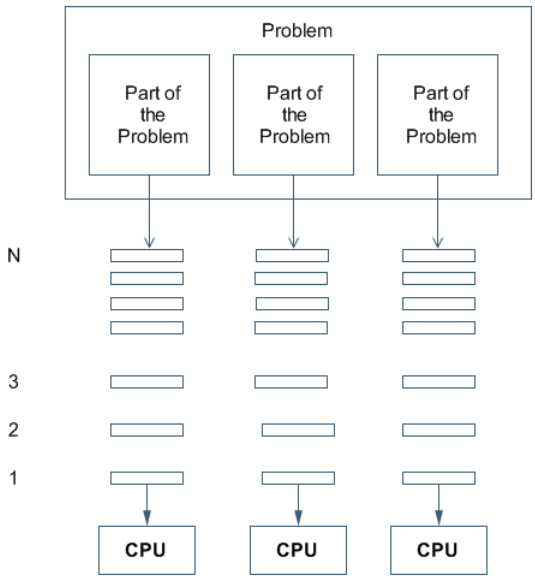

Figure 2.3: Parallel Computing Architecture-Big problem can be divided to small one , then they can be send to different CPUs to execute concurrently

If the original problem compound with some independent small parts, we can execute those small parts of the problem on different CPUs, so that we can gain speed and finish the whole process much faster than using serial computing.

Here is an example of parallel algorithm:

1: f(a, b)
2: {
3: c = a?b
4: d = 2?b
5: e = a+b
6: }

In this part of the program line 3 and line 4 are independent for each other. They both can be executed at the same time on different processor, in that way the overall program can be executed much faster than on the single processor serial computer.

The main Advantage of parallel computing is speed up the process. Especially when we try to solve large computational problem we can benefit from remote resources, cannot be limited with speed or memory of single computer or processor. But at the same time the aim to solve problem on parallel computing machine requires design of parallel programming that makes complicated programming process even difficult. Also difficult to understand and manage data locally.

2.2.3 Different Parallel Computing

As memory management and usage is very important factor of parallel computing special concern given to memory architecture in parallel computing .There are some basic classifications of parallel computing according to memory architecture. As there are more than one processing unit how to use memory is the important thing. We can share large memory between multiprocessors or we can have individual memory for every single processing unit.

2.2.3.1 Shared memory architecture

All process and computers share the same central memory. They don't have individual or spate memories. In shared memory architecture, all different processors or computers have access to global memory space. Changes made by one process or computer are visible for others.[10]

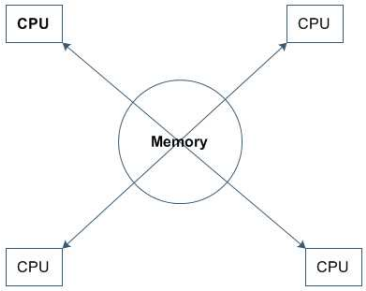

Figure 2.4: Shared Memory Parallel Computing-In parallel computing even we have more than one CPU but we still can have same memory for all the CPU at the same time, all CPUs read and modify the same memory

The advantage of above parallel computing architecture is data sharing between processors are fast and easy. Also convenient so program as all the data are in same memory. But programming process might be difficult as we want to keep integrity of the data. At the same time scalability is the main issue between CPU and memory. If we add more processor on the system it will create more traffics which will make CPU cache memory management complicated.

2.2.3.2 Distributed memory architecture

This architecture has individual memory per every single processor. As each processor has it is own memory, it operates independently, memory space on one processor doesn't match to other processors.[10]

This architecture require programmer to map overall memory managements of the program structure to existing distributed memory space, in that case programmer has do deal with all complicated process. But at the same

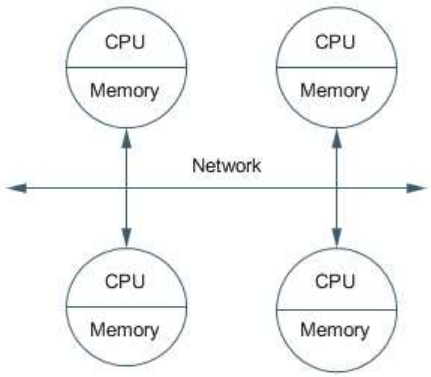

Figure 2.5: Distributed Memory Parallel Computing- Obviously all CPU engaging execution must have their own memory space individually.

time as every CPU has identical memory space memory is scalable with number of processor, add or take out any CPU doesn't make trouble. Every CPU can easily access their memory space very high speed. No interferences exist as shared memory architecture. At the same time there is another important factor which is that network connection supposes to be interconnected so that we can make sure that connection will not affect our parallel computation speed.

2.2.3.3 Shared and Distributed Memory Architectures

This is a combination of both shared and distributed memory architecture of parallel computing. Most of the super computers are using this architecture in recent days. In this way we can connect several shared memory processors together to do parallel processing, this is also the main philosophy behind cluster technology. Every group of shared memory processor is powerful computational unit, then all of them will be treated as a single processing unit with private memory, this combination is the most popular way of parallel computing.[10]

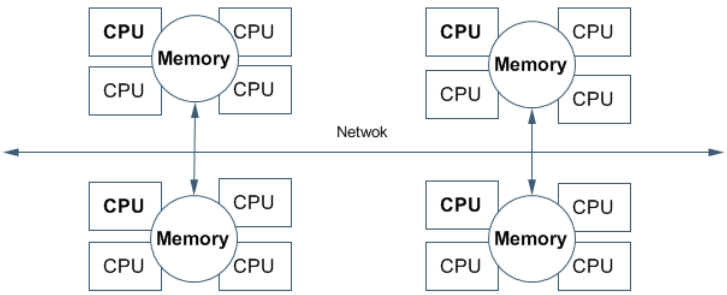

Figure 2.6: Shared and Distributed Memory Parallel Computing- This is the way to put shared memory parallel computing and Distributed memory parallel computing together.

Cluster is based on both shared memory structure and distributed memory structure, and there are different types of cluster facilities according to how we build it. We can build cluster with same specification of physical hardware or we can use different types of physical hardware, but overall goal is implement parallel computing.

2.3 MPI

As we discussed before if overall problem can be divided to small pieces and distributed to identical computing unites that will speed up computing processes than traditional way, but not all problems or computing jobs can be divided easily to independent small computing units. MPI-The Message Passing Interface is a portable message-passing standard that facilitates the development of parallel applications and libraries. [11] MPI is a library standard. The main purpose of MPI is to develop a widely used standard for writing message-passing programs. The reason for build such an interface is try to establish a practical, portable, efficient, and flexible mechanism for message passing.MPI is especially useful for problems that can be broken up into several processes running in parallel, with information exchanged between the processes as needed. The MPI programming environment handles the details of starting and shutting down processes, coordinating execution and passing data between the processes.

Reasons for using Message passing interface is[11] :

- Standardization - MPI is the only message passing library which can be considered a standard. It is supported on virtually all HPC platforms.
- Portability - there is no need to modify your source code when you port your application to a different platform which supports MPI.
- Performance - vendor implementations should be able to exploit native hardware features to optimize performance.
- Functionality
- Availability - a variety of implementations are available, both vendor and public domain.

For Message passing interface it is target platform is a distributed memory system including massively parallel machines, workstation clusters. All parallelism is explicit and the programmer is responsible for correctly identifying parallelism and implementing the resulting algorithm using MPI constructs. The number of tasks dedicated to run a parallel program is static. When comes to program Message passing interface shuld be able to used with C and FORTRAN programming languages.[11] Here we will explain some basic functionalities of message passing interface.

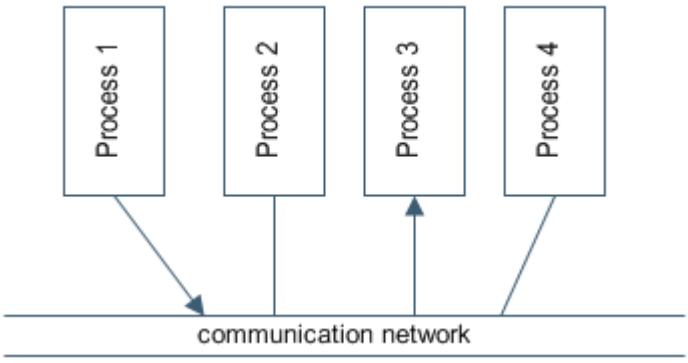

Figure 2.7: Parallel Message passing Mechanism- There are communication between different process.

1. **Message Passing**
   Message passing is the method that copy data from the memory of one processor to another. That can be down internally for instance between multi core processor or that can be down externally which means data will transfer through network, and those data has routing information.

2. **Process**
   A process is a set of executable instructions which runs on a processor. All processes communicate with each other even they are running on the same processor, because of efficiency consideration any particular time processor can only run one process.

3. **Message Passing Library**
   Message passing library is a collection of routines in the program codes which can help accomplish certain function like send, receive etc.

4. **Send / Receive**
   Message passing means there are sender and receiver. Message are transfered between processors, the process sending message usually require them to specify the data's location, size, type and the destination. The process Received message should match a corresponding sender specification.

MPI is main standard in parallel computing, major cluster and supercomputer sites all have implementation of MPI. Most popular implementation is MPICH2, is a freely available open source library and it is available for most flavors of UNIX including Linux and Mac OS X and Microsoft Windows. Moreover, MPICH is a developed program library. [11]

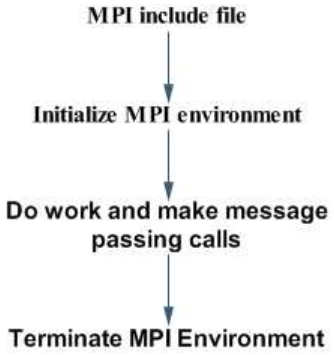

Figure 2.8: MPI Programming Process

2.4 Resource Management and Scheduling

Resource management is one of the key factors in high performance computing cluster. Efficient resource management will provide high performance. Basically resource manager is one of the important parts of our operating system. Regularly checking which process is running, and which is next process, and checking status of memory and CPU all those works going to be done by resource manager. When we talk about supercomputing all single compute nodes, master nodes they all managing their own resources, then as a whole supercomputer overall resource management should be down properly. So what is resource? What does it mean? How can we describe and measure it? Here we will explain some basic concepts about resources. Mainly there is CPU, Memory, Disk and network Throughput.

CPU is the core resources in supercomputing. And there are some factors about how to describe CPU resources.

System CPU time

System CPU time is the time the CPU spends executing system (kernel) code in order to run your program. System CPU time includes such things as reading files, moving information into and out of virtual memory, also the time spend on running such a program that responsible for keep your system up and running etc.

User CPU time

User CPU time is the time CPU spends on running user program and that is not related to operating system. And that factor is the most valuable factor we looking for. More user CPU time we have we might provide more computation resources.

CPU Input output waiting time

that time refers to the time when CPU has to wait for input output operation to continue it is current job. This input output operation is written to disk or read from disk operation. Longer CPU input output time we have, it will reduce the performance. Less input and output waiting time is always increase performance. This is mostly depending on physical hardware.

CPU Ideal time

This is when CPU has nothing to do and just waiting for next instruction to continue. If CPU keep ideal this means resource are not efficiently used.

System Load

System load average is a percentage value that can determine CPU's potential status. Load averages include all demand for the CPU not only how much was active at the time of measurement. For a single processor machine a load average of 1 means that, on average, there is always a process in the running or in legal state. Thus, the CPU is being utilized 100% of the time and is at capacity. If you tried to run another process, it would have to wait in the run queue before being executed. For multiprocessor systems, however, the system isn't CPU bound until the load average equals the number of processors in the machine.[12]

There are still some other factors related to CPU resources, like CPU nice time which is while CPU executing at the user level application with nice priority, there also a time that CPU spend to deal with service interrupts but those are not the major consumer of CPU resources compare to what we described above.

We consider CPU as the crucial rescores except CPU there are other factors which can be considered as resource consuming.

Memory usage

RAM (Random access memory) is main physical memory of computer. That is also considered as the main resources of the system. Good combination of system memory and CPU with right speed and capacity plays key rule in over performance of computing. In many cases memory shortage can be the main bottleneck of the system.

Swap

Physical memory is a finite resource on a computer system. As some times the program going to be loaded to memory is much bigger than actual physical memory, operation system will take certain amount of storage space from hard disk, make it temporary virtual memory for the system. When physical

memory demand is sufficiently low, process memory images are brought back into physical memory from the swap area on disk. Having sufficient swap space enables the system to keep some physical memory free at all times.

Hard disk space

Hard disk is not crucial for performance since storage capacity of hard disk is getting larger and price is low. But existent free space is necessary for system. When hard drive is filled up this can be deadly bottleneck for the system.

Network throughput

This is a standard to measure network activity. Specially in parallel computing, as program running is based on message passing protocol communication between different nodes needed in any second. In Another hand, all the job submission and result collection will go through network interface. In that case all time valuable network capacity is needed.

In super computer all those single pieces of resources handled by every individual computing node, and their own local operation system resource management. After all we need super resource manager to collect all those status from every single node and present overall status of the whole supercomputer. This is the key function for supercomputer as resources consumption is main concern. How to collect this information efficiently and correctly from all single computer nodes and how to present it as complete pieces is very crucial. At the same time to use supercomputer as a complete huge system we have to have mechanism to submit all kinds of tasks and jobs to the system then able to distribute those huge jobs to individual computing nodes, then still, when the job is accomplished the system has to collect all the result from different nodes and present it as whole result of original job to the system. That is not easy thing to do, that needs complicated algorithm, and engineering to achieve. Normally that can be down by combination of different kind of software. We divided this complicated task to two parts. First one is mainly resource management. Second part is scheduling.

Resource management is for information collecting (resources). We have different type of tools to achieve this goal. Scheduling is responsible for hold or keeps the incoming jobs in certain order, which we call it queuing, and find right job in the queuing pool then send it so system to process. Both of them have to work in the way of grate cooperation.

2.4.1 Portable Batch System

Before going to deep in to portable batch system we will go through some basic concept that can help us to understand portable batch file system.

2.4.1.1 Batch File

A batch file is a text file that contains a sequence of commands for a computer operating system. It's called a batch file because it batches into a single file a set of commands that would otherwise have to be presented to the system interactively from a keyboard one at a time. A batch file is usually created for command sequences for which a user has a repeated need. Commonly needed batch files are often delivered as part of an operating system. You initiate the sequence of commands in the batch file by simply entering the name of the batch file on a command line.

2.4.1.2 Batch Job

In a computer, a batch job is a program that is assigned to the computer to run without further user interaction. All input data is preselected through scripts or command-line parameters. This means that a sequence of commands to be executed by the operating system is listed in a file and is submitted for execution as a single unit. The opposite of a batch job is interactive processing in which users enter individual commands to be processed immediately. In some computer systems, batch jobs are said to run in the background and interactive programs run in the foreground. In general, interactive programs are given priority over batch programs, which run during the time intervals when the interactive programs are waiting for user requests. A program that reads a large file and generates a report, for example, is considered to be a batch job.

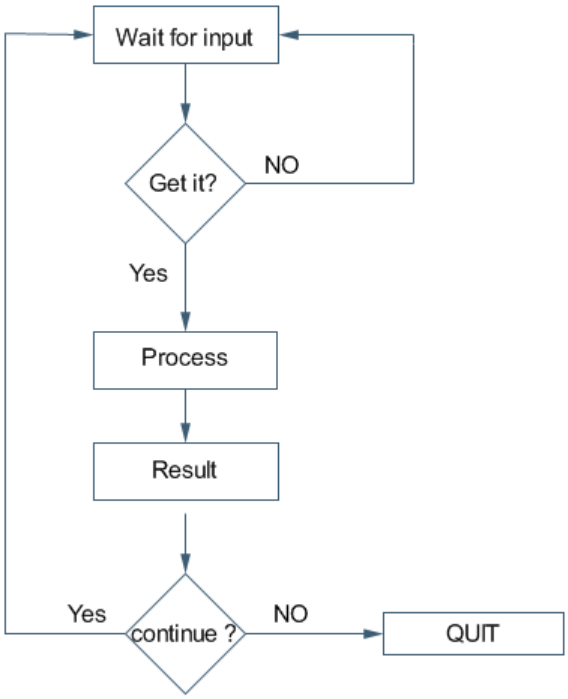

Figure 2.9: Normal Human Interactive Process

Batch jobs are usually initiated by a system user. Some are defined to run automatically at a certain time. In many cases, batch jobs accumulate during working hours and are then executed during the evening or another time the computer is idle. This is often the best way to run programs that place heavy demands on the computer. Batch jobs are typically executed at a scheduled time or on an as-needed basis.

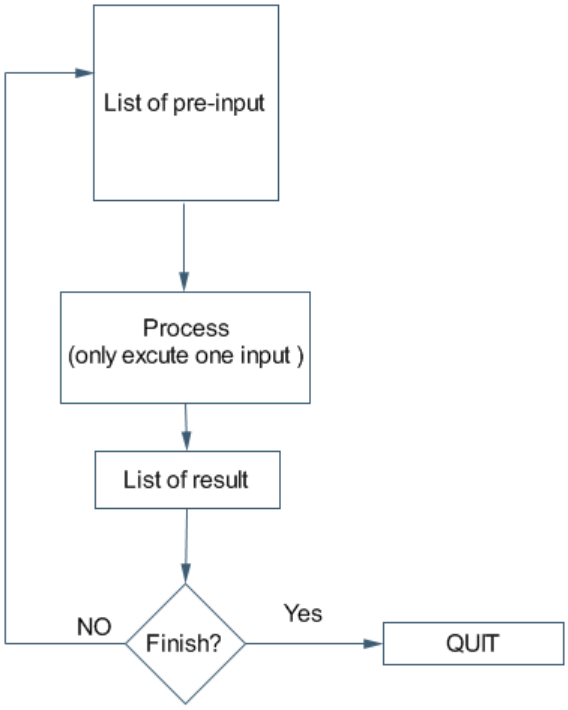

Figure 2.10: Batch Process

2.4.1.3 Batch Proccess

Batch processing is execution of a series of programs on a computer without human interaction. Batch processing is for those frequently used programs that can be executed with minimal human interaction. Batch processing has these benefits:

- It allows sharing of computer resources among many users
- It shifts the time of job processing to when the computing resources are less busy
- It avoids idling the computing resources with minute-by-minute human interaction and supervision
- By keeping high overall rate of utilization, it better amortizes the cost of a computer, especially an expensive one.

Batch processing became very popular and important, now it is getting over its mainframe origins, and is now frequently used in UNIX and Linux environments.

2.4.1.4 Portable batch system

Portable Batch System is the name of computer software that performs job scheduling. Its primary task is to allocate computational tasks,like batch jobs, in available computing resources. [13]

PBS consists of four major components[13]:

1. Commands
2. Job Server
3. Job executor
4. Job Scheduler

Commands

PBS supplies both command line commands and a graphical interface. These are used to submit, monitor, modify, and delete jobs. The commands can be installed on any system type supported by PBS and do not require the local presence of any of the other components of PBS. There are three classifications of commands[13]:

1. User commands
2. Operator commands
3. Administrator commands

Job Server

The Job Server is the main components PBS. All commands and the other daemons communicate with the Server via an IP network. The Server's main function is to provide the basic batch services such as receiving or creating a batch jobs, modifying the job, protecting the job against system crashes, and

running the job. One server manages one or more queues. A batch queue consists of a collection of zero or more batch jobs and a set of queue attributes. Jobs are said to reside in the queue or be members of the queue. In spite of the name, jobs residing in a queue need not be ordered first in, first out. Access to a queue is limited to the server which owns the queue. All clients gain information about a queue or jobs within a queue through batch requests to the server. [13].

Job Executor

The job executor is the daemon which actually places the job into execution. This daemon, pbs_mom, is informally called Mom as it is the mother of all executing jobs. Mom places a job into execution when it receives a copy of the job from a Server. Mom creates a new session as identical to a user login session as is possible.[13]

Job Scheduler

The Job Scheduler is another daemon which contains the site's policy controlling which job is run and where and when it is run. Because each site has its own ideas about what is a good or effective policy, PBS allows each site to create its own Scheduler. When run, the Scheduler can communicate with the various Moms to learn about the state of system resources and with the Server to learn about the availability of jobs to execute.[13]

The batch system fits into a client - server model, with a batch client making a request of a batch server and the server replying. This client - server communication necessitates an intercrosses communication method supportable over a network and a data exchange format. While the basic PBS system fits nicely into the client - server model, it also has aspects of a transaction system so the batch system data exchange protocol has been built on top of a reliable stream connection protocol - TCP/IP and the socket interface to the network.

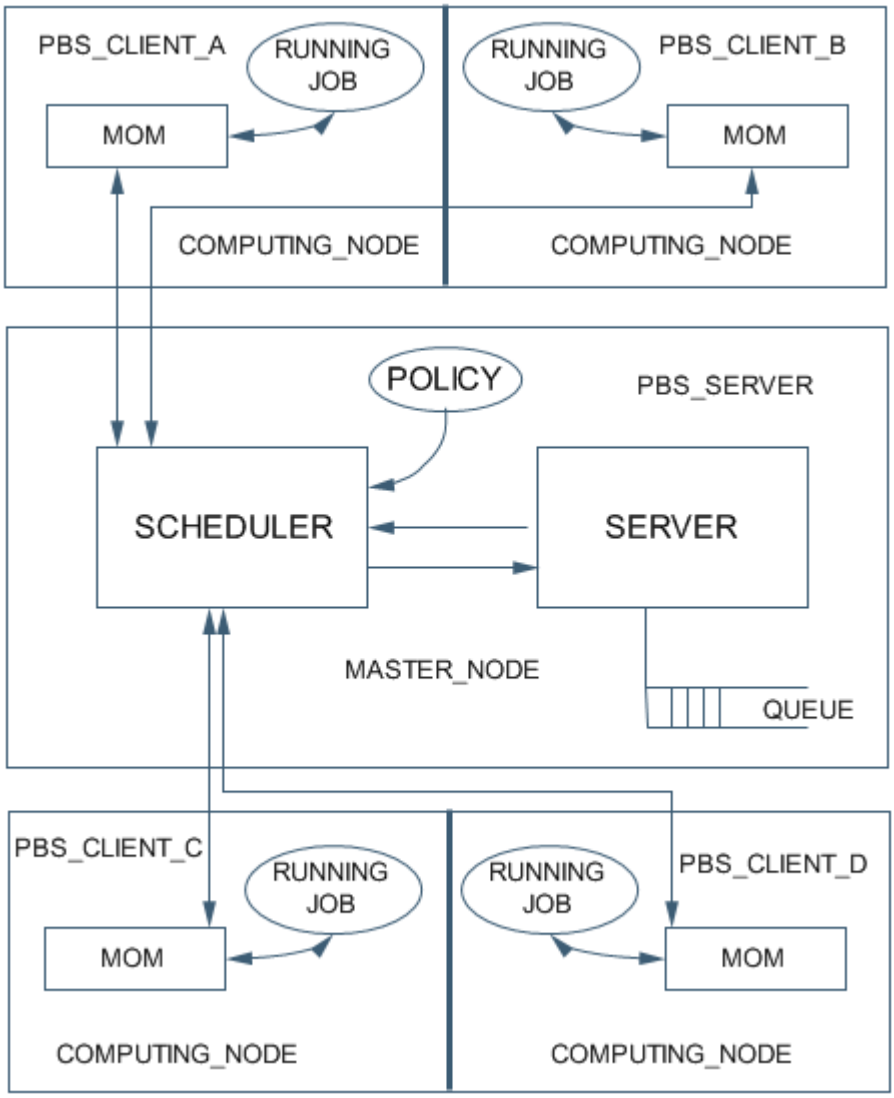

Figure 2.11: PBS Server-Client Communication Structure

Portable batch system multi host communication is complicated procedure, MOM is a part of PBS tool acting like agent on every expected PBS client or computing node. MOM is going to collect all resource related information then represent it to PBS server on master node, so that scheduler be able to decide how to send jobs to different computing nodes according resource statistic information from MOM.

2.4.2 Scheduling

Simply scheduling is a mechanism to decide how to organize your queue. On the other hand it means how to prioritize your queue. When you queuing, you have to have certain rules to keep all the incoming jobs sorted in certain order. That is what scheduler does. According to different request and demand you can use different scheduling mechanism.
Different type of scheduling:

- FIFO-First in First out

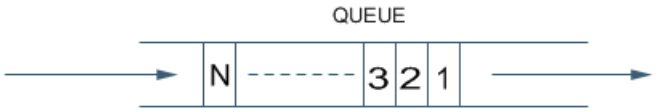

Figure 2.12: First in First Out-Incoming stream will be placed in Exactly the same order

A customer that finds the service center busy goes to the end of the queue. Forexample if incoming orders 1, 2, 3....N. When they out of the queue they will have 1, 2, 3.....N order.

- LIFO-Last in First out

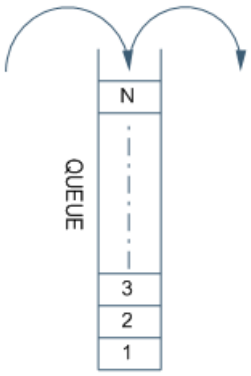

Figure 2.13: Last in First Out- Incoming stream will be placed in Reversed Order

A customer that finds the service center busy proceeds immediately to the head of the queue. In that case if incoming queue is in 1, 2, 3....N order when they out of the queue they are in N.....3, 2, 1 order.

- Random Service The customers in the queue are served in random order.

  It does not matter which one comes first, the serves will randomly select incoming job to serve.

- Round Robin Every customer gets a time slice. If her service is not completed, she will re-enter the queue.

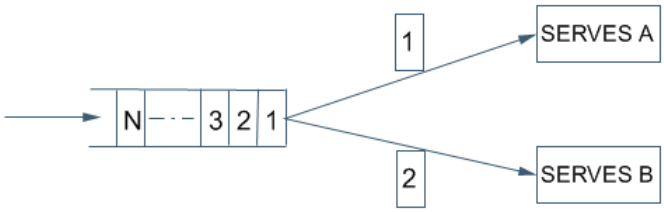

Figure 2.14: Round Robin-Incoming stream will be distributed A after B, B after A order

- Priority Disciplines Every customer has a priority. The server selects always the customers with the highest priority. This scheme can use preemption or not.
- Back fill

  Back fill scheduler used to work with other scheduler, the purpose is efficiently use current resources. Here is some example:
  Let us say there are five jobs A,B,C,D,E,X,Z. and main scheduler policy is priority disciplines. Therefore every job has their own priority,1 is the highest priority, and back up policy is backfill. Resorce requirement and priority of different jobs are like that:

  A: Number of CPU 10, Number of hours: 2, priority 1

  B: Number of CPU 2, Number of hours: 4, priority 3

  C: Number of CPU 5, Number of hours: 3, priority 4

  D: Number of CPU 4, Number of hours: 10, priority 2

  E: Number of CPU 11, Number of hours: 4, priority 5

  Z: Number of CPU 8, Number of hours: 4, priority 6

  X: Number of CPU 4, Number of hours: 4, priority 7

  According to this information, job Z has higher priority than job X but current resource condition is not able to allow job Z run, but there is an empty resource gap where X can fit, that time backfill policy going to take action put X into that empty gap.

2.4.3 Queue

In general, a queue is a line of events or things waiting to be handled. Usually it refers to sequential process that ordered from beginning to the end. In

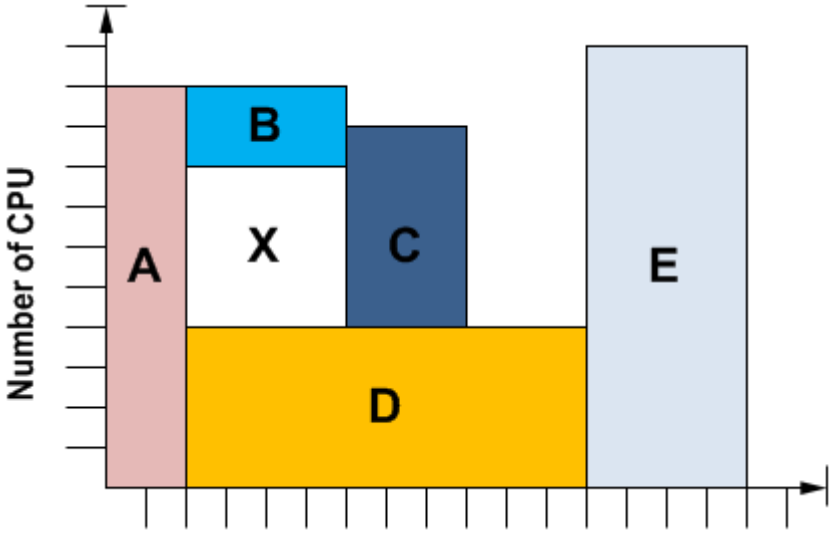

Figure 2.15: Backfill Example

computer technology, a queue is a sequence of processes, requests, jobs that are waiting to be processed.

Existence of queue means over demand of serves taking place in certain time. That causes waiting and managing, sorting etc. therefore queue is very simple structure but complicated to manage. In computer science, the process is serialized queuing taking place in any single time in system, for example network traffic, process management, database management etc.

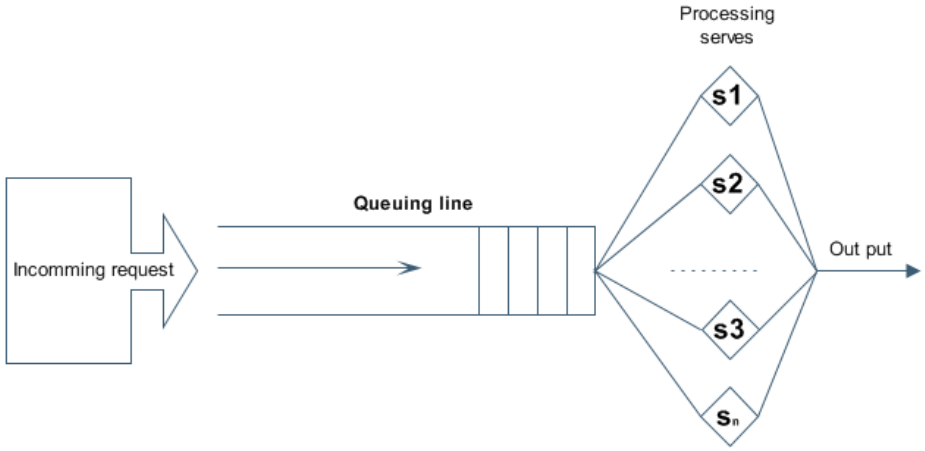

Figure 2.16: Queuing structure

There is number of choices or parameters in the queuing model[14]:

- Arrival time Is the time that new Job to arrive at queue
- Process time Is the time that serves spend to process every Jobs in the queue
- Number of serves That means how many serves we will have to process incoming Jobs
- System processing limit
- Maximum population size Length of queue
- Scheduling policy Means how to process the queue, like FIFO(first in first out).

A/B/m/N - S

A - Represent the distribution of the inter arrival time. B - Represent the distribution of the service times m - Represent the number of servers. N - Represent the maximum size of the waiting line in the finite case (if N = 1 then this letter is omitted).

S - that is optional, denotes the service discipline used (FIFO, LIFO etc.). If S is omitted the service discipline is always FIFO.[14]

2.4.3.1 Queuing Model

We characterize queuing model as their way of accepting and processing incoming request[**?**]. There are different queuing model:

1. M/M/1

   In M/M/1 queue there is only one serves processing incoming request. This is the simplest queuing system. Request comes in random time interval, and incoming request take random amount of time to be processed.[14]

2. M/M/1^k In M/M/1^k queue, we add more serves to the M/M/1 queue, incoming request distributed to the number of serves according to the status of serves or special scheduling system.[14]

Figure 2.17: Queing Model M/M/1[14]

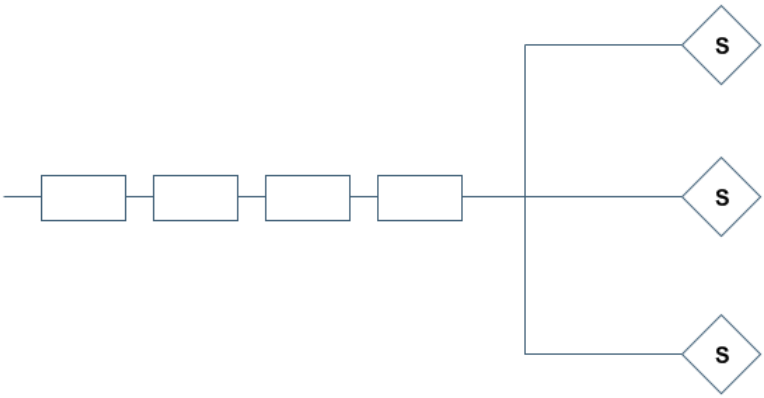

Figure 2.18: Queing Model M/M/1^k [14]

3. M/M/K In M/M/K queue we have dedicated serves for different incoming request, which can be understand as presorted, or we can say incoming request goes to special serves according to serves.[14]

In all there type of queuing model ,It is really difficult to say which one is much more efficient as it is depending on certain environment, some perform very efficient in certain environment but not all. Therefore get the best performance out from queuing system we have to put the right queuing model in the right scenario.

2.4.4 Load Balancing

In computer science and information technology, load balancing means distribute computing processes, network traffics, resource requests to different handing objects so that we can get best performance. That is because, in that way we can keep our system healthy, reliable, and efficient.

As we hope to balance the load we have to have replica group, which can share the load, or take over the original serves in case of failover which is most useful way to deploy redundancy. Therefore load distributor has to have a mechanism to deliver load to those replica groups, according those mechanism there are different type of load balancing :

- Random distribution Incoming load randomly distributed to replica group.
- Adaptive distribution Incoming load distributed according to least-loaded

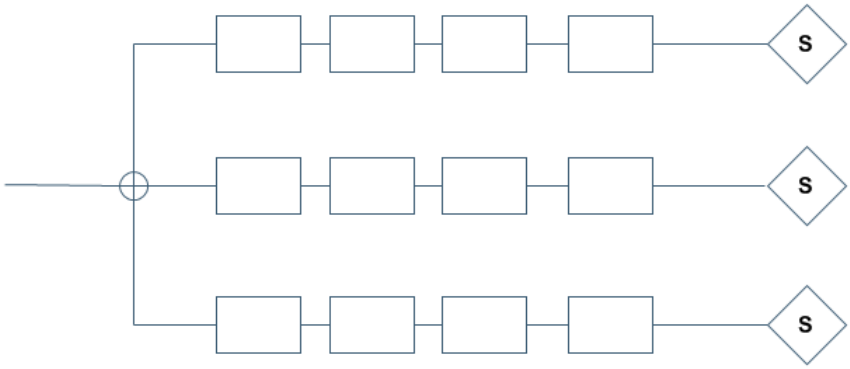

Figure 2.19: Queing Model M/M/K [14]

object, which is most common load balancing in supercomputing.

- Round Robin distribution Incoming load distributed one after each other.

- Ordered distribution Incoming load distributed according to special order with the character of request.

Except this there is failover load balancing which is replica server running side by side as the original server, if there is anything happen like server machine is out of respond or down, basically this copy of machine can take over all servers. In that way, that is you can build robust serves, you always have redundancy out there, and reliable serves can be provided.

there are many benefits for all kind of different services from load balancing technology, but specially for cluster or supercomputing there are many benefit from load balancing :

1. Availability

2. Scalability

3. Manageability

4. Lower total cost of ownership

In supercomputing main idea is distribute work load to all over computing nodes, proper scheduling mechanism and load balancing mechanism has to work properly and cooperatively together, that will mostly decide performance of all super computer.

Chapter 3

Literature survey and discussion

In last decades, several related research has been done, and papers has been published. Cluster and super computer are recently developed technology, but queuing theory has been developed over last 100 years, Danish engineer who worked for the Copenhagen Telephone Exchange, published the first paper on queuing theory in 1909.[15] According to content of the research we can discuss all those related research in two different ways.

3.1 Performance Metrics and Monitoring

According to research [2], they set up different metrics to evaluate cluster. Their main concern is job execution time. They also divided performance metrics to application metrics and machine matrices. In that case, they can evaluate cluster performance according to those metrics. But in our case, our main concern is CPU time. We want to evaluate system according to actual user CPU time, regardless of application and machine metrics.

In research [1] they developed special cluster monitoring tool called AJM-Automated Job Monitor. That's a customizable automated system for indentifying and rectifying routine system/queuing issues normally performed by system administrators.[1]. This tool will record all queuing jobs, running process and file system information. And that would bring extra load to our experiment environment that's why their method is not acceptable.

In research[3], they mainly discovered the way of analysis the collected data from cluster. As there are large amount of data, getting meaning full conclusion from that data set is difficult, what they did is, in analyzing collected data they took the environmental factors, like physical location of the rack, in to the account, combine those factors with result analysis, in that way get best understanding out of data collection, in our case, our analysis is based on data collection, what we want to achieve is, despite those physical factors we want to draw a picture just based on data collection to see how much that can explain our assumption.

3.2 Power Consumption and CPU Utilization

It is difficult to know how much relation it has between system CPU utilization and power consumption. Since we don't have instrument to really prove this we did some survey about CPU utilization and power consumption. First according to [6] they showed many proven example to say that disk power consumption and memory power consumption can be negligible as the dynamic differential between idle and fully utilized is only around 30%[6] , that makes us firmly believe that CPU utilization can be a factor to reflect power consumption.

3.3 About Queue

Queuing is the key part of the cluster. The problem about queue on the cluster system is mainly about rapid increase of jobs. That makes queue getting bigger. According to the research[5], they proposed two way of reducing rapid increase queue. One is have a wider diversity of user, that case will decrease the chance that events like project deadline will be correlated, thus reduces queuing jobs. Another approach is reduce number of user using machine, so that control job submission, but for us both of the method is difficult to achieve, in real environment it is difficult to have a wider diversity of user and control number of user, so what we want to see is what is really going to happen when queue storm is happened on the queuing machine.

Chapter 4

Experiment Design

We divided my experiment to three parts according to my research goal.

- First part is evaluation to the production supercomputer.
- Discover the relation between CPU utilization and CPU power consumption according to CPU temperature
- According to above experiment does power consumption evaluation on our production supercomputer.

4.1 Discovery on Production Cluster

The main Idea about this experiment is to see how efficient the cluster system are working, as such computing facilities, efficiency is main issue, at the same time learn the behavior of supercomputing is also interested, specially queuing status of the system will be mainly Discovered.

The experiment took place at Bergen center of computational science, as the main computational unit of Bergen University, they provide computational power to different research organization. For example bio-informatics, climate science, computational physics, computational chemistry, marine science, etc. But mainly this particular cluster has been mostly used by bio-informatics scientist.

The experiment is down at IBM e1350 cluster (fimm.bccs.uib.no), it has following specification:

- 86 e326 nodes
- 172 AMD/opteron 250 (2.4 GHz) processors (2 cups per node)
- 258 Gigabyte memory (on average 3 Gigabyte per node)

- 6880 Gigabyte disk (80 Gigabyte per node)
- Linux operating system (Redhat)
- Gigabit Ethernet on all 86 nodes, and a low latency SCI/Dolphin interconnect on 25 nodes

Here is basic system layout:

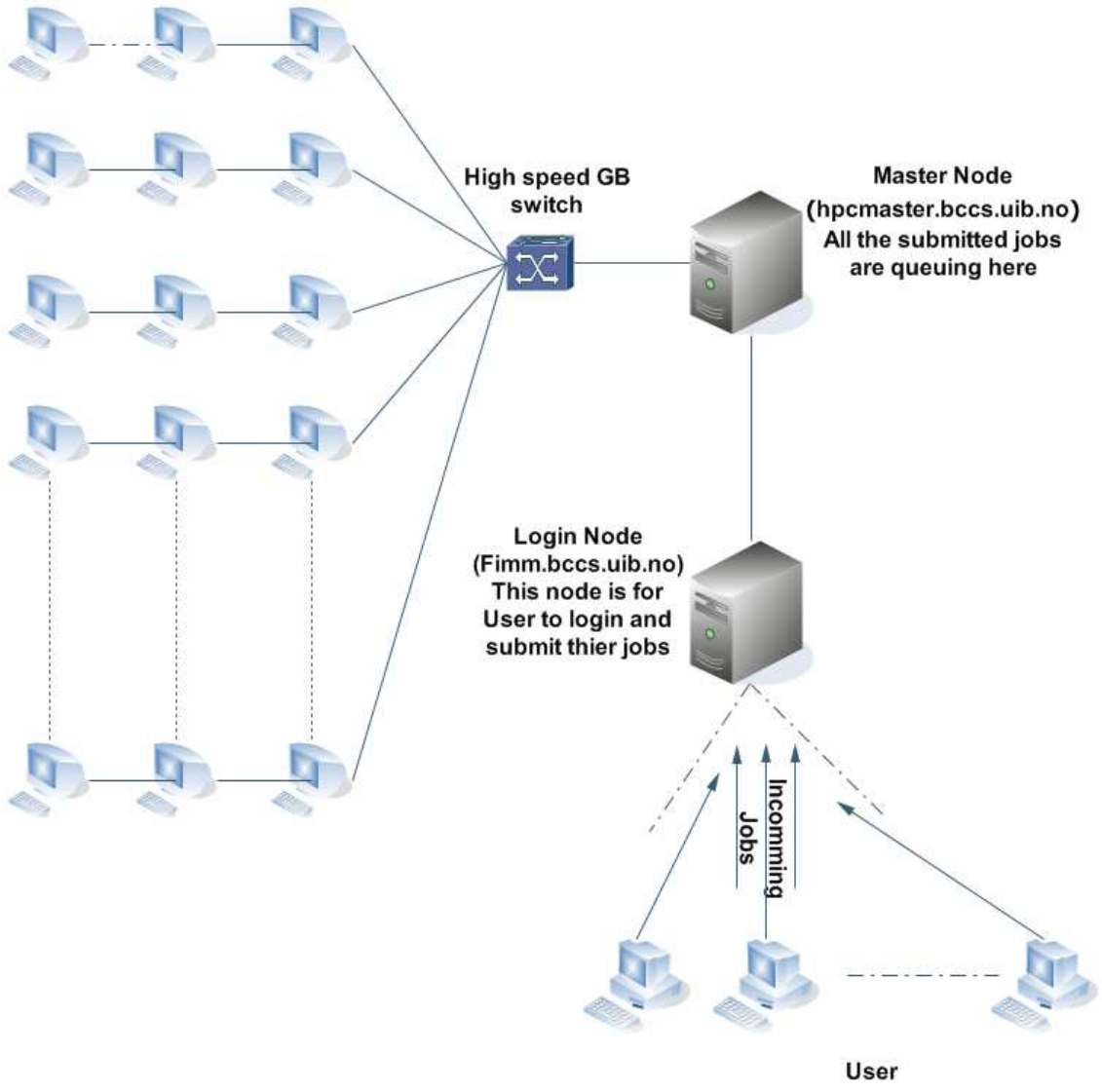

Figure 4.1: System Layout: User will login to the login node, submit thier jobs, submitted jobs will queue on master node, and then it will distribute to all computing nodes according their resource situation

This is fairly large system. So how can we see if this system is working efficiently? We turn our main concern to CPU utilization, because in such a supercomputing facilities computing power is key resources. When talking about computing power CPU time is the core issue. Therefor how to measure how CPU is working is very important, especially in high performance computing site users have to buy CPU time, which is the main concern here.

Here are two different way to define CPU utilization.[16]

$$U = \frac{CPU\ Busy\ time}{Total\ time} \tag{4.1}$$

In our case as a user normally have to purchase the CPU time, because of that, we are really interested about actual time CPU spend on user tasks, but try to find user CPU time we will subtract all other CPU times from CPU busy time, that includes CPU system operation time, CPU disk input output time.

4.1.1 CPU utilization and Queue

First of all, for collecting CPU utilization in respect of user time we found that sysstat[17]pakage is already installed on the cluster. in that way we can easily get the percentage where CPU spend on different tasks using mpstat command. we randomly selected 20 computing nodes,and run a small script to collect CPU utilization information. The reason we select 20 nodes as sample is that when we get all different number of information we need, then we can use these samples to estimate mean value of overall cluster. Since load on different computing nodes is very critical, that is the most reliable and scientific way to get the overall percents of user CPU time.

Another interesting factor we are going to take in to account is queuing status on Master node. In the master node all incoming jobs are going to queue in certain configuration. Mainly we will look in to:

- Active jobs

  Active job means the number of job actually running on the computing nodes.

- Idle jobs

  Ideal job means the number of job waiting for to be executed and not activated yet, which means on the other hand the real queuing jobs.

- Blocked jobs

  Blocked job means the number of jobs have been blocked and waiting on the master node to be release from blocked status. There are many

reasons incoming jobs or active jobs can be blocked, for example user submitted much more jobs which they can actually allow to do so.

- Total jobs

  Total jobs means the sum of active jobs, idle jobs and blocked jobs.

4.1.2 Job Overview

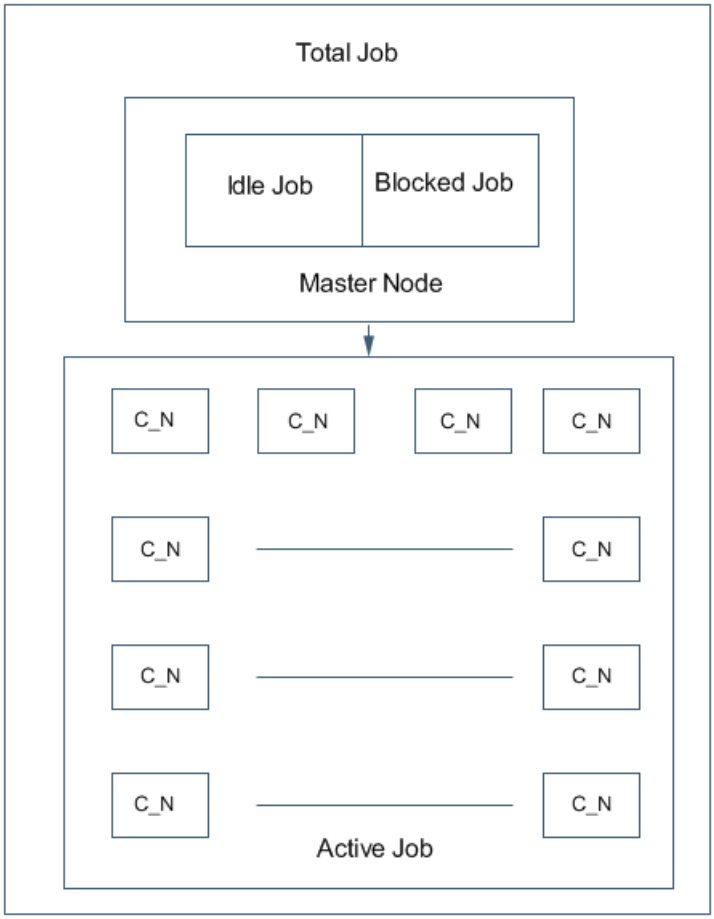

Figure 4.2: Overview of Jobs in Different Status- This is a statistics from different part of cluster. Total job is the sum of active jobs, idle jobs and blocked jobs. As explained before idle jobs and blocked jobs are statistics from master nodes, and active jobs are statistics from computing nodes

Queuing activity on the machine is very difficult to predict, because user will submit their jobs in any time that sometimes causes queuing storms[5], how our system handle this , or how it can be represent in our site ? That can help us understand queuing property of the system. , we want to see when

there is number of queue in the master node, how does rest of the computing nodes utilized. As far as there are queue, we expected that computing nodes suppose to be utilized with user CPU times in certain percentage. We also want to learn about queuing properties of this particular cluster. Specially arriving rate of those jobs even though those jobs coming in stochastic manner.[3]

4.2 CPU Utilization, Power Consumption and Temperature

As mentioned in research goal evaluating system and see CPU time distribution is one part of this research. The reason is, as usage of CPU time is discovered we will be able to distinguish how much time is actually spending on real user task. That will tell us how efficient the system is.

4.2.1 CPU Utilization and Temperature

In that part of the experiment, find relationship between CPU utilization and temperature is important. We used Cfenvd[18] to collect the data in one of our server in network experiment lab, as this is the only machine we can use Lmsensor package[19] to fetch the temperature data from the CPU. In that way we can compare the CPU utilization data with CPU temperature data.

4.2.2 Power Consumption

Power consumption is main concern in Data center recent years. As power consumption is directly lead to problem about clean energy usage and even global warming. It has been proven that CPU utilization highly related to power consumption of the system.[7][8][20][21][22][6]

According to [6],once we get maximum power draw and idle power draw of the system, in any given CPU utilization we can find correlated power consumption within $\pm 5\% of accuracy$.

Chapter 5

Results

In this section I will show my experiment result and analysis. We divided our experiment to two parts. The reason is we want to see some relation between CPU utilization and CPU temperature but our cluster system does not support this temperature reading in physical level, so we did another experiment in one of our normal server where we can read temperature date easily. Other measurement has been down on the production cluster machine.

5.1 Queuing Property of Cluster

5.1.1 Total Jobs

That graph 5.1 shows us the total number of jobs that have been appeared in the master node, X axes shows us the time series which is measured in every hour, the reason for measure it in every hour is because after long time observation it appears to that it will hardly change within one hour of time interval. As a configuration smallest job has to take at least one hour of calculation time. There is some peak which is caused by sudden boost from high amount of user job submission. It also tells us that user behavior and the time cluster spending on finishing jobs determined the total job distribution at the graph. There is no clear pattern in this distribution that we can follow as incoming rate of the job is stochastic.

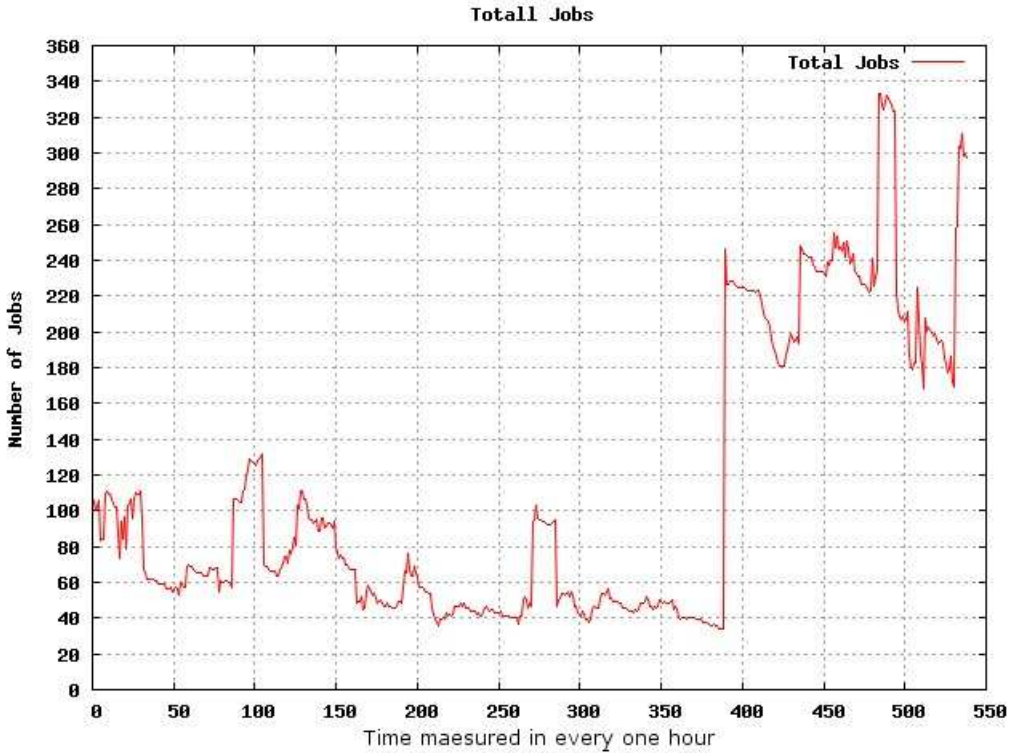

Figure 5.1: Total Number of Jobs in the System From 17th of April Until 10th of May on Master Node–That includes idle jobs, active jobs, blocked jobs. Time measured in every one hour.

5.1.2 Active Jobs

That graph 5.2 shows us that number of active jobs running on the cluster from 17th of April until 10th of May. Time is measured in every one hour which is in X axes, and Y axes shows number of jobs. In first 15 days we can see number of jobs running in the system is remain under 100, which means system is less utilized. That can be proven from rest of the graph, from day 16th the system accepted Farley large amount of jobs, until end of the measurement the number of jobs are active is raised to 160. How many jobs can be run simultaneously is not only depend on system resources, it can be also depend on how many jobs the user can run at the same time, that will decided by site administration.

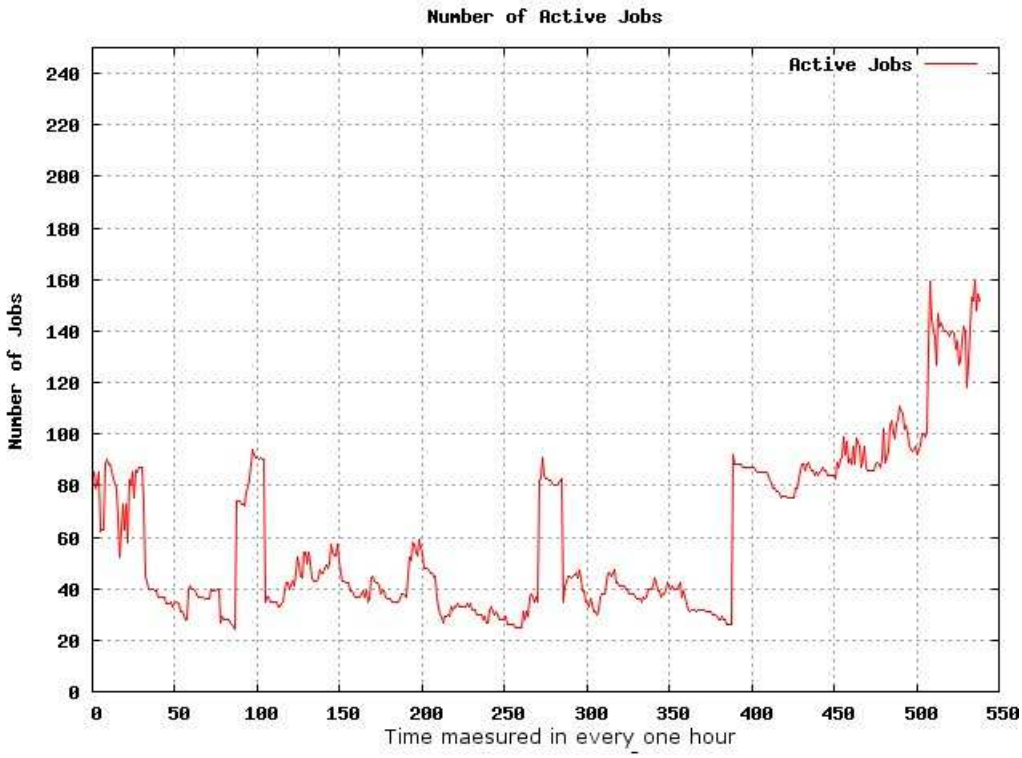

Figure 5.2: Active jobs collected from 17th of April to 10th of May on Master Node

5.1.3 Idle Jobs

That graph 5.3 shows us that number of idle jobs which is actually queuing in the system, that means those jobs are eligible to run but because of lake of resources, it has to queue. In that graph the highest number of jobs we get is 17, which is considerably small amount of number compare to number of jobs actually running in the system. Idle status of jobs only can happen when jobs eligible to run and there are no resources valuable according to it is priority. We can see some interesting relation between active jobs and idle jobs, there is a peak in time interval 100, at the same time there is also a small peak in active jobs in 100 time interval. That means at that period of time system is not fully utilized, so incoming jobs can directly turn in to active status and being executed, but still as we have more incoming jobs than we can accept as active jobs therefore system start to turn jobs in to idle status. Same thing happen at 380 time interval until 540 time interval both active job graph and idle job graph shows peak. Active jobs raised from 30 to dramatically 160, at time seems like system is full , rest of the incoming jobs turn to idle status that caused idle jobs raised from 0 to 17 .

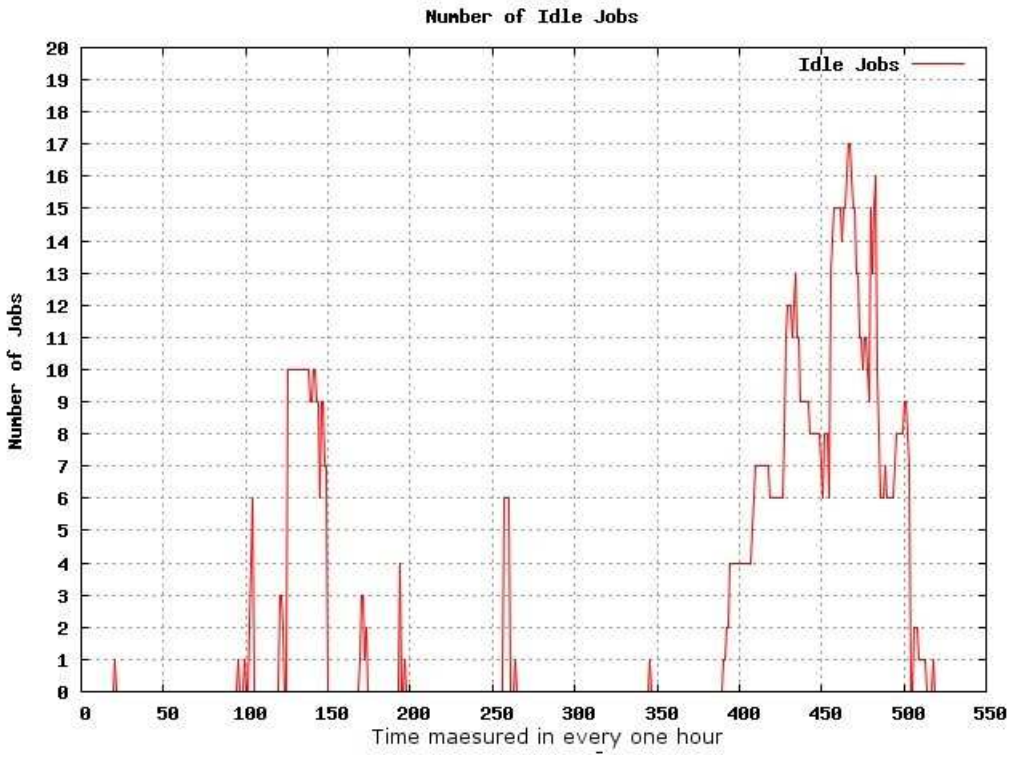

Figure 5.3: Idle Jobs collected from 17th of April to 10th of May on Master Node

5.1.4 Blocked Jobs

That graph 5.4 shows us number of blocked jobs in the system. Time is measured in every one hour in X axes, Y axes shows number of jobs. There is couple of reason that job can be blocked , mainly jobs going to be blocked because user run out of quota, or submitting much more jobs than he/she is allowed to do. Blocked jobs take a large part of total jobs in the system. There is one important reason for large amount of jobs going be blocked it is because of the quality of the program that user submitted, and some operational skill of user. For example user allowed running 5 jobs in the same time, and there are already 4 of them running in the system , user expect the at least one or two of them planed to be finished by today 5:00 pm , then user will keep submitting new jobs as in his program he planned to finish his job on time , but in the real system there might be some difference , user might request special CPU or resources due to that reason in cant be finished on time, then miscalculation of user will cause block of his new job. If we compare the graph show active jobs with graph shows blocked jobs we can see that blocked jobs also dramatically increased from 380 time interval from 10 to 170 this is the same time we get boost at active jobs graph. After 500 time interval we can see number active jobs are still keep rising, but idle job already start to decrease almost close to 0, and blocked jobs also dramatically decreased.

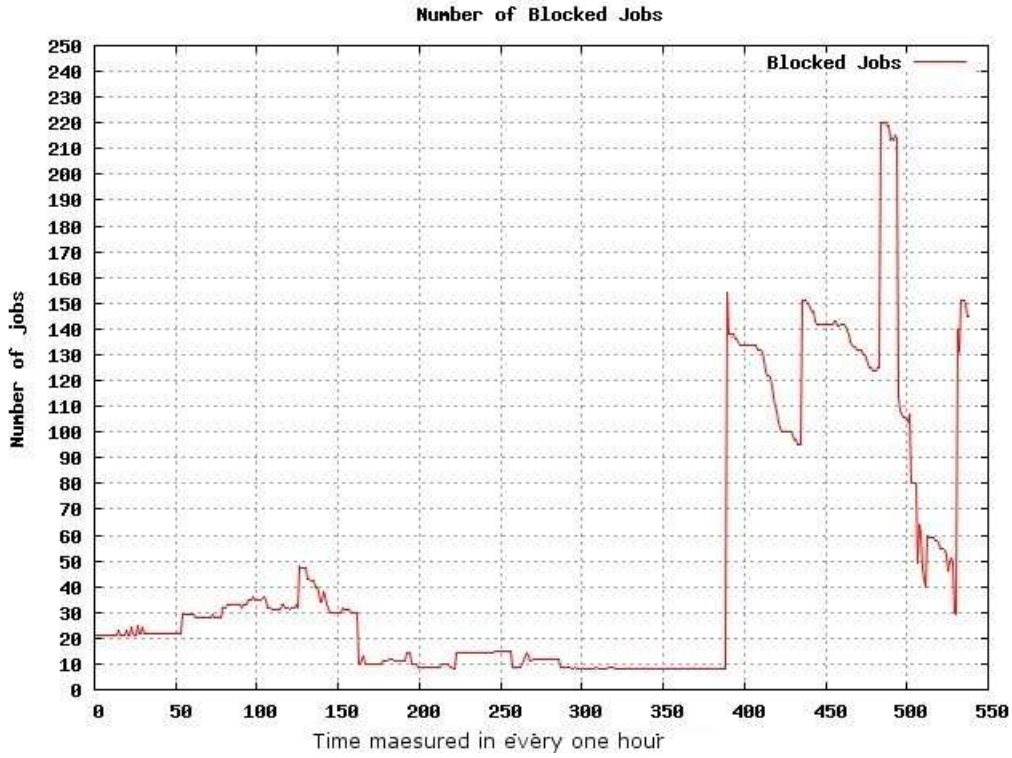

Figure 5.4: Blocked Jobs collected from 17th of April to 10th of May on Master Node

5.1.5 Cluster CPU Utilization and Queue

As described in our problem definition, evaluate cluster is one of our goal, we want to see how efficiently the system is working , in that case we are not only considering CPU busy time but also interesting about how much time CPU actually spend on user task. This is the reason even though CPU is busy it might doing non user related task. There is also another interesting compression, as we said at the problem definition, if there is more jobs coming to the system does it affect the current CPU utilization status, in other word if number of incoming jobs has anything to do with current status. Because of those questions we collected User CPU time and system CPU time and want to compare it with different job status graph.

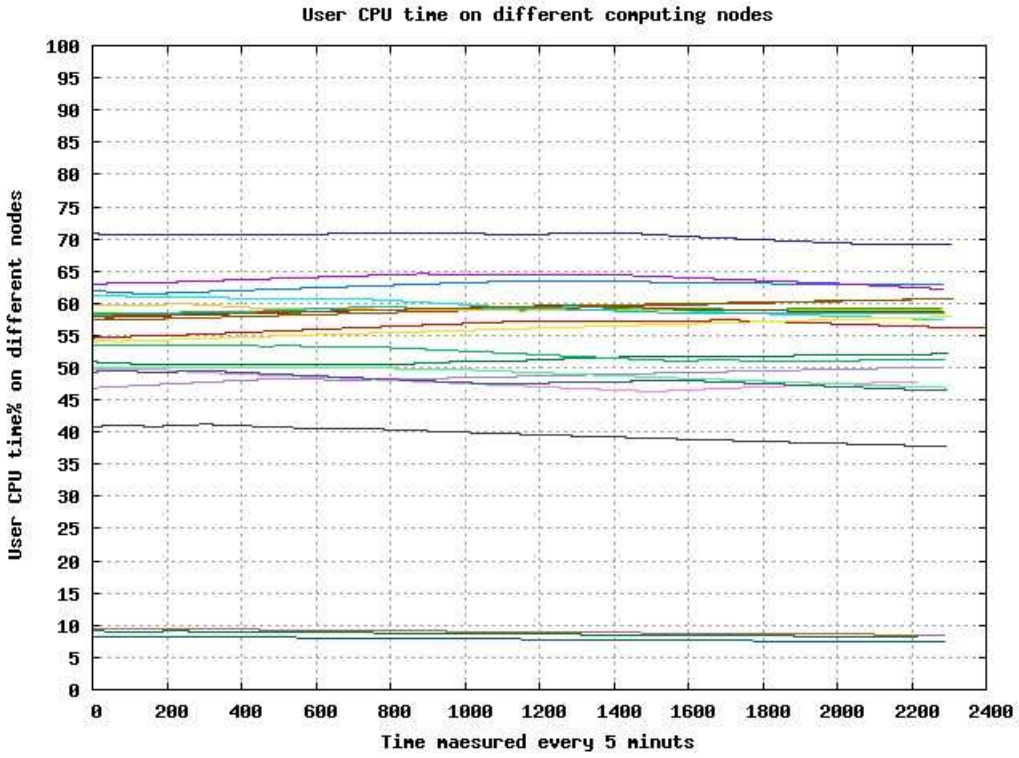

Figure 5.5: Node User CPU Utilization from 30th of April to 10th of May in 22 Computing Nodes

That graph 5.5 shows us the time CPU spends on user task from computing nodes in the cluster. X axes shows time which is measured in every 5 minutes, Y axes shows % of time CPU spend on User task. It is clear that most of the computing nodes spend between 45% till 75% of their CPU time on user task and it is constant and stable. There is couple of Nodes shows its User CPU time is around 8%-10%, after trackback the record it shows that those nodes are reserved nodes for special group of people. And that's the reason they have very small utilization. Therefore overall pictures show that cluster computing nodes spending average of 45% to 75% time on user task this is satisfactory result for efficiency.

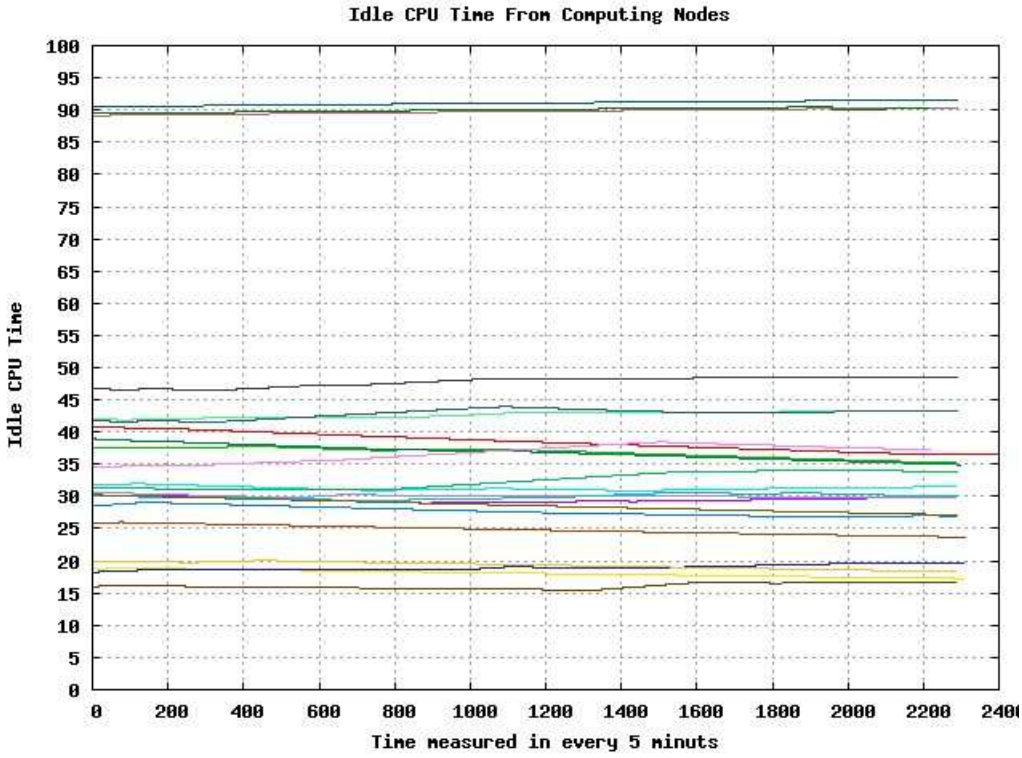

Figure 5.6: Node Idle CPU Time from 30th of April to 10th of May in 22 Computing Nodes

That graph 5.6 shows us percent of time CPU is idle on computing nodes, which means being free. X axes shows time measured in every 5 minutes Y axes shows percent of time CPU is free. We can clearly see that CPU has free time around 15% -45% on all nodes. If we can put that graph with the graph5.5 which shows user CPU time on the computing nodes, we can see both of user CPU time and idle CPU time together almost combine 100%of time, which means CPU is mainly spend it is time on user task rest of time it is free, system operation and rest of the operation like input output operation is less than 5% . Also we can see some of the nodes free for almost 90% of their time, those nodes are reserved nodes for special groups, that's just match to graph 5.5

In both of the graph there is no variation ?that can be explained in various of way , first of all it can explained as the CPU activity in all nodes is stable , and consistent , that is because the system might receive fairly large job for long time computing. In this 11 days period all those nodes might processing the same job. Actually that stable graph is what we willing to see. This is desirable the reason is that tells us CPU is either working hard or just free, not spending time or spending very little time on system operation, disk right read operation etc.

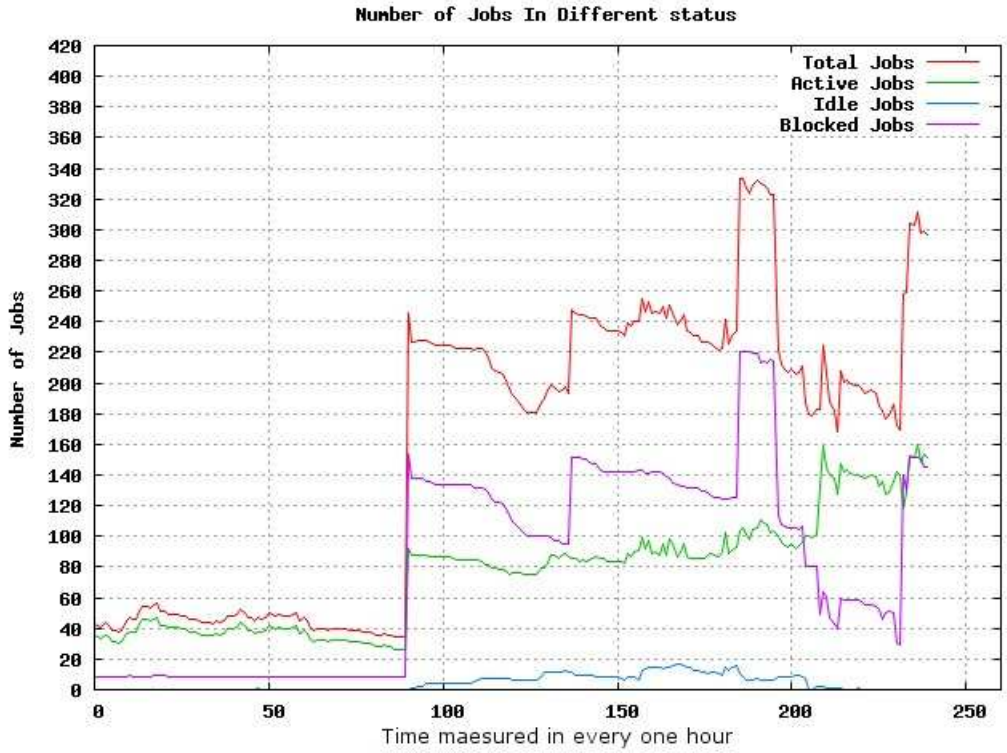

Figure 5.7: Jobs in all status collected from 30th of April to 10th of May from Master Node

That graph 5.7 shows us jobs collected in all status from 30th of April to 10th of May. X axes shows time measured in every hour, Y axes shows number of jobs. When all the status of jobs plotted in to one graph we can see that sudden boost of incoming jobs only or highly affect active jobs and blocked jobs but there is not too much affection at idle jobs. In graph we can see in 80 time intervals incoming jobs suddenly increased dramatically, followed the increase of incoming jobs active jobs and blocked jobs also increased dramatically but we don't see any obvious increase on idle jobs, that's what we didn't expected, we expected sudden increase of incoming job will cause increase of idle jobs, as it is actual queue in the system. That can tell us cluster will try to consume all incoming jobs as much as possible. But reason for increase in blocked job is interesting, that might be caused by user behavior. And at the same time blocked jobs are also decrease very soon. That can tell us that blocking job is also the way to hold incoming jobs for certain time until find proper resource when system is busy.

5.2 CPU Utilization VS Temperature

As we cannot get temperature data from cluster node easily, we did experiment on our normal server which is hosting 27 virtual machines, and also support Temperature cheapest reading. We use cfenvd[18] one of the environment demo tool from cfengine[18]. Cfenvd will collect lots of information about system but most importantly it helps us to collect CPU utilization and temperature data. That graph 5.8 shows us CPU utilization in average with

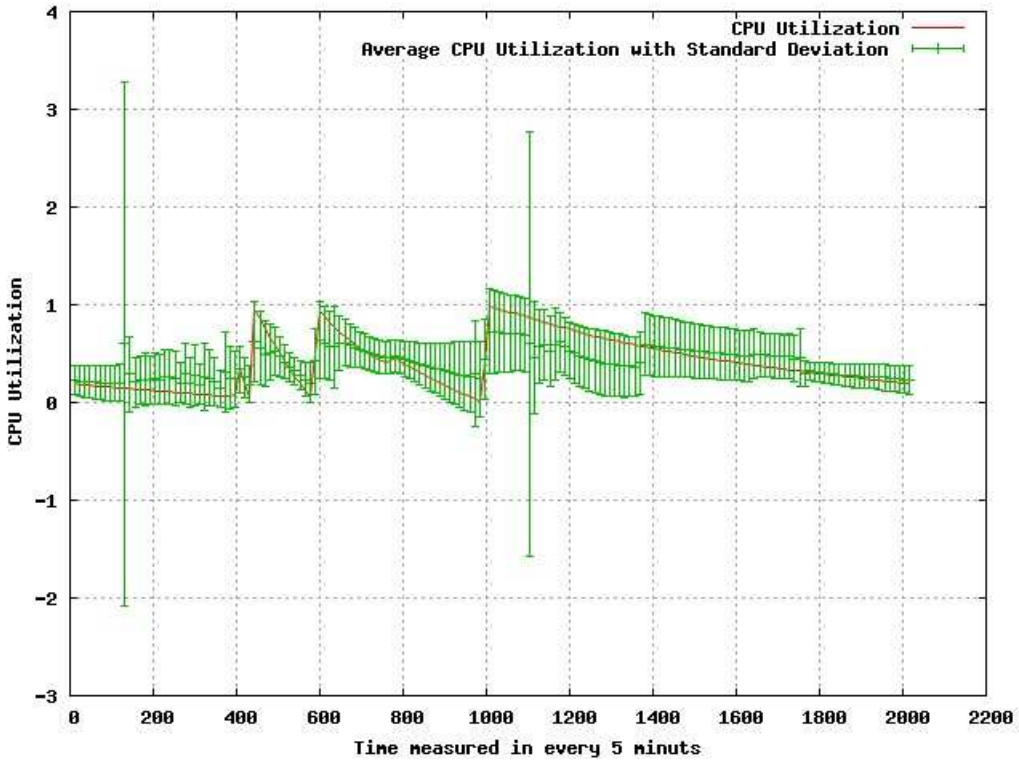

Figure 5.8: CPU Utilization From Server mln1

error bar. X axes shows time measured in every 5 minutes, Y axes shows CPU utilization measured in percent, the graph also shows standard deviation of average CPU utilization, first of all we can see CPU utilization is not consistent, it is changing dramatically regularly, as this server is hosting 27 virtual machine CPU utilization vary according to different operation going on the system not like computing node. The error bar shows us mean value of CPU utilization and it is variants, and it also keep remind us those measurement are sample date when we try to draw conclusion.

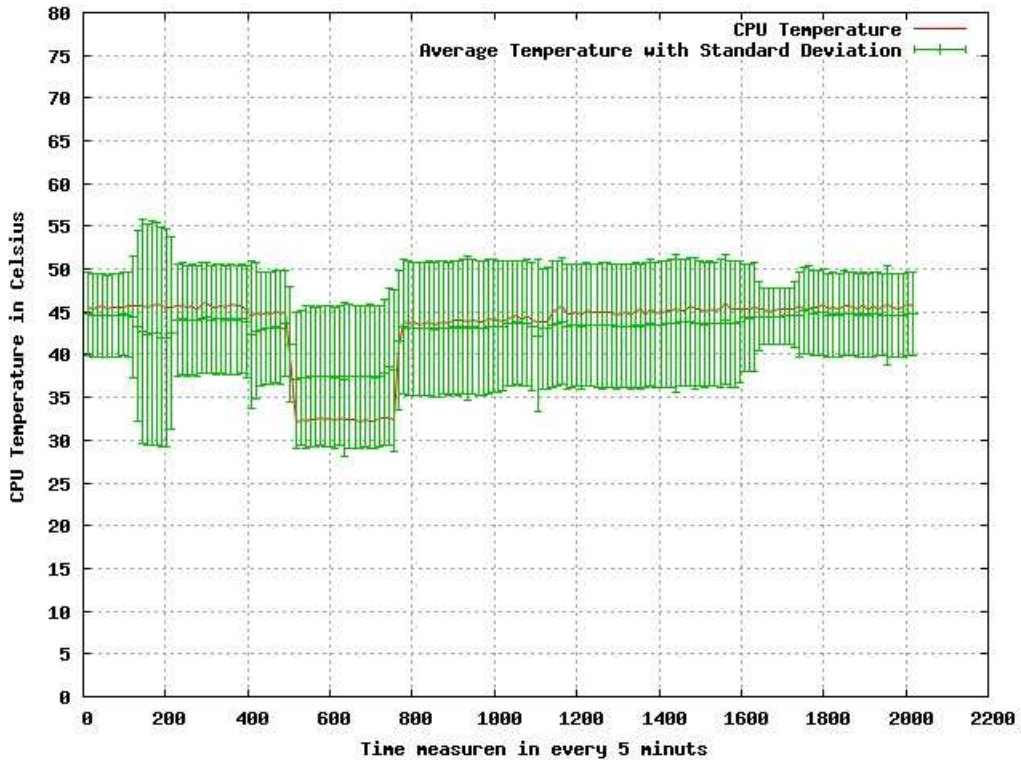

Figure 5.9: CPU Temperature From Server mln1

That graph 5.9 shows us average temperature data with standard deviation. X axes shows time measured in every 5minuts, Y axes measured temperature in Celsius. That is measured in one week time period. As we can see from 500 times interval there is big curve going to, after track back we found that serves was down for couple of hours, rest of time temperature is pretty much around 40 to 45 Celsius. If we have longer time experiment and collection that error bar we see might be around 40 -50 Celsius degree. Actually temperature is much more stable compare to our CPU utilization activity. If we put two graphs together we can easily find that there is no direct relation between CPU utilization and CPU temperature from this last measurement values. This is understandable that environmental factors dominating the temperature. That means in this special server environment, the cooling system of this server room, the position and location we put this server is actually affecting those temperature data we collected. But long time measurement is still necessary to discover if there is direct or indirect relation between CPU utilization and CPU temperature.

5.3 Estimation of Power Consumption of Cluster

As mentioned in our research goal find simple and practical way to predict power consumption is discussed. As we find many related research proved relation between power consumption and CPU utilization, using formula provided by[6][23] we can get estimated power consumption for our sites.

$$P = \{P_{max} - P_{idle}\} \times n/100 + P_{idle} \tag{5.1}$$

In that formula P is power consumption we want to find,P_{max} is max power draw of system, P_{idle} is idle power draw of the system. Once these figures are available, power consumption P at any specific CPU utilization n% can be calculated. It has been proven[6]that in estimation is accurate to $\pm$5%.

In our experiment we collected all the CPU utilization time for evaluating efficiency of the system, in 22 nodes we did 2292 times measurement within 10 days time. Now we can use those data to predict power consumption of whole cluster. We can find average mean CPU utilization of 22 computing nodes as:

$$\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i \tag{5.2}$$

After calculation we get:

$$\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i = 60.68\% \tag{5.3}$$

If we assume that those 22 computing nodes can represent all computing nodes on the cluster, which means we assume that sample CPU utilization can represent whole cluster average utilization, using formula 4.1 , assuming max power draw of the cluster is M, idle power draw is I, we will get this estimated equation:

$$P = 0.6 * M + 0.4 * I \tag{5.4}$$

This is just estimation, that equation need to be proven when there is real measurement regarding power consumption has been down. This kind of estimation will help system administrator to prepare for temporary power shortage.

Chapter 6

Conclusions and Future work

6.1 Conclusions

6.1.1 Evaluation on Cluster

To evaluate the cluster we did some measurement on CPU time of the node where we selected 22 sample computing nodes out of 86. Our experiment shows sufficient amount of usage of those computing nodes specially regarding to user CPU time. The extremely predictable user CPU time is interesting, the most possible explanation for that is in that period of time all sample nodes might computing the same job, that keeps computing node consistently in high utilization. This makes us believe that the measurement on the cluster side or supercomputing side need fairly long time of experiment to explore more interesting trends.

There are couple of test that we are really interested but we are not able to do, first of all we wanted to get temperature data from cluster. But physical layer of cluster does not support our temperature reading pakege, in that case we decide to find an other production server facilities(mln1) where the physical layer of the the computer support temperature reading pakege.

Second is we are interested about different user group. In that case to compare different CPU time utilization according to different gourp of user. That will take long time to discover user group first then try to distingwish different user process from system. As a matter of time this was not achived. But after this experement we get better underestanding of basic system that will help us to go for futher research.

6.1.2 Queuing Properties on The Cluster

Queuing property is another interesting issue. As cluster sites or supercomputing sites has their own different usage and user, the set up of the system is vary from each other, in that case different sites will have different user focus, that caused very unique queuing behavior on different sites, from the very begging we were expecting, when sudden job storm appear in the cluster most of the new coming jobs might go to idle queue, in that case idle queue will get most impact from that job storm, but our experiment shows us, in our special

environment that will end up affecting blocked jobs, there are also explanation for that , as we discovered the experiment time is considerably short ,in that case the affection to blocked jobs can be same user submitting much more jobs that they can to do so. At the same time we did not see really big impact on the idle queue. Again long time experiment is necessary for further discovery on queuing properties.

In that part of experiment tracking user job from master node to computing node and see if there is trends in distributing jobs to certain nodes was considered, But because of limited time and lacked understanding of the system didn't allowed us to do that

6.1.3 Estimated Power Consumption of The Site

In data center, be able to estimate power consumption without extra power consumption sensor is necessary and very handy, that can help for system administrator to be able to predict how long system can run under UPS without electric supply. In this part of estimation we only want to show there are possibilities to estimate power consumption of cluster. Our estimation might not be very accurate, the estimation will be examined when we get actual cluster power consumption.

6.1.4 CPU Utilization and CPU Temperature

The result from experiment about CPU utilization and temperature is also slightly different from what we expected. The reason for the experiment is try to prove and see that power consumption can be predicted from CPU utilization. Because rising temperature eventually reflects power consumption as energy moves as the form of temperature. But what we see is, there is no direct relation from the graph, still as we explained environmental factors dominating the temperature. But we still believe that long time experiment will be able to show more direct or indirect relationship.

For that part of experiment, Cfenvd is most helpful tool from the beginning to the end, when we cannot able to get the temperature and utilization together with cfenvd, updated version of cfenvd helped us.

As most of our experiment done at production environment, we couldn't get super user privilege, but we still manage to get necessary data that we need.

6.2 Future Work

Our experiment showed analytical work, furthermore it shown very interesting and practical result. But time is the main issue here, for examining this type of large infrastructure we need long time consistent experiment to extract more result. In our experiment further research is very interesting. Especially on the concern about evaluating user CPU time, one can do even more detailed research about property of different user and their job and how does that affecting user CPU time, in that case different user group doing experiment on

computing nodes, according to their subject, program, method we might see different user CPU time on the system. CPU utilization and temperature will be hot topic as more and more attention drawing to power consumption. Doing experiment in special equipped machine is necessary. Because reading temperature more rely on physical environment. The goal was achieved in this thesis work. As we discovered more interesting trends the more we able to understand our system, hopefully that thesis work can help further research to achieve more interesting facts.

Bibliography

[1] Raytheon Company Robert F.Cromp, Gilad Suberri. Automated job monitoring in high performance computing enviroment. Technical report, Proceeding of the International Conference on Autonomic Computing, 2004.

[2] Brent Gorda Jeffrey S. Vetter, Theresa L. Windus. Performance metrics for high end computing. Technical report, HECRTF White Paper, 2003.

[3] Y. M. Marzouk J.M. Brandt, A. C. Gentile and P.P.Pébay. Meanigfull statistical analysis of large computational clusters. Technical report, Sandia National Laboratories, 2005.

[4] Reda A. Ammar Ahmed M. Mohamed, Lester Lipsky. Performance modeling of a cluster of workstations. Technical report, University of Connecticut, 2003.

[5] Scott H. Clearwater Stephen D.Kleban. Quelling queue storms. Technical report, Proceedings of 12th IEEE International Symposium on High Performance Distributed Computing, 2003.

[6] MARK BLACKBURN. Five ways to reduce data center server power consumption. Technical report, The green grid, 2008.

[7] M Pidwirny. Energy, temperature, and heat. accesed 2008 May , available at `http://www.physicalgeography.net/fundamentals/6c.html`.

[8] Christos Kozyrakiz Partha Ranganathan Dimitris Economou, Suzanne Rivoire. Full-system power analysis and modeling for server environments. Technical report, 2005.

[9] Matthew Elton. How computer processors work. Technical report, 2006.

[10] High Performance Computing Group. Introduction to parallel computing. Technical report, Lawrence Livermore National Laboratory, 2008.

[11] University of Hawaii Maui High Performance Computing Center. Message passing interface. accesed 2008 May , available at `http://www.mhpcc.edu/training/workshop/mpi/MAIN.html`.

[12] What is load average. accesed 2008 May , available at `http://immike.net/blog/2007/07/27/what-exactly-is-a-load-average/`.

[13] James Patton Jones Casimir Lesiak Bhroam Mann Bill Nitzberg Tom Proett Judith Utley Albeaus Bayucan, Robert L. Henderson. Portable batch system. Technical report, 2000.

[14] M. Burgess. *Analytical Network and System Administration — Managing Human-Computer Systems*. J. Wiley & Sons, Chichester, 2004.

[15] WIKIPEDIA. History of queuing theory. accesed 2008 May , available at `http://en.wikipedia.org/wiki/Queueing_theory`.

[16] Michael T.Trader. How to calculate cpu utilization. Technical report, 2001.

[17] Sebastien Godard. Sysstat system monitoring utilities. accesed 2008 May , available at `http://pagesperso-orange.fr/sebastien.godard/`.

[18] Aleen Frisch M. Burgess. *A System Engineer's Guide to Host Configuration and Maintenance Using Cfengine*. Published by the USENIX Association, 2007.

[19] accesed 2008 May , available at `http://www.lm-sensors.org/`.

[20] Sandeep K.S. Gupta Tridib Mukherjee, Georgios Varasamopoulos. Measurement-based power profiling of data center equipment. Technical report, Arizona State University, 2005.

[21] Jitendra K. Singh. Managing power consumption based on utilization statistics. Technical report, Hewlett-Packard Development Company, 2006.

[22] Richard Russell Morrie Altmejd Evandro Menezes, David F. Tobias. Cpu utilization measurment techniques for use in power managment. Technical report, Advanced Micro Devices,Inc., 2005.

[23] James E. Hanson Jeffrey O. Kephart Malgorzata Steinder, Ian Whalley. Coordinated management of power usage and runtime performance. Technical report, 2008.

Appendix A

Appendices

A.1 Script for collecting Queuing information on the master node

```
#!/bin/bash
NUMBER=$(/opt/maui/bin/showq | tail -1 | awk '{print $3,$6,$9,$12}')
DATE=$(/bin/date | awk '{print $3, $4 }')
echo "$NUMBER $DATE" >> /local/cfengine/log.txt
```

A.2 Script for collecting CPU Utilization

```
#!/bin/bash
/usr/bin/mpstat>>/local/cfengine/CPU_U.txt
NUMBER=$( grep "all" CPU_U.txt | awk '{print $1,$3,$4,$5,$6,$7,$8,$9,$10}' )
DATE=$(grep "Linux" CPU_U.txt|awk '{print $4}')
join  $NUMBER $DATE > /local/cfengine/uti_1.txt
```

A.3 Log Data for Queuing status

Total Job	Active Job	Idle Job	Blocked Job	Day	Time
96	75	0	21	17	13:36:07
93	72	0	21	17	13:37:08
92	71	0	21	17	13:37:16
104	83	0	21	17	13:59:01
106	85	0	21	17	14:59:01
100	79	0	21	17	15:59:01
102	81	0	21	17	16:59:02
106	85	0	21	17	17:59:01
83	62	0	21	17	18:59:01
84	63	0	21	17	19:59:02
84	63	0	21	17	20:59:01
109	88	0	21	17	21:59:01
111	90	0	21	17	22:59:01
109	88	0	21	17	23:59:03
109	88	0	21	18	0:59:01
107	86	0	21	18	1:59:01
104	83	0	21	18	2:59:01
102	81	0	21	18	3:59:01
102	79	0	23	18	4:59:01
88	67	0	21	18	5:59:01
73	52	0	21	18	6:59:01
94	73	0	21	18	7:59:01
84	63	0	21	18	8:59:01
97	73	1	23	18	9:59:01
79	58	0	21	18	10:59:01
103	82	0	21	18	11:59:02
104	80	0	24	18	12:59:01
107	85	0	22	18	13:59:01
96	75	0	21	18	14:59:01
107	86	0	21	18	15:59:01
110	85	0	25	18	16:59:01
109	87	0	22	18	17:59:01
109	87	0	22	18	18:59:01
111	87	0	24	18	19:59:01
96	74	0	22	18	20:59:01
67	45	0	22	18	21:59:01
65	43	0	22	18	22:59:01
62	40	0	22	18	23:59:02
62	40	0	22	19	0:59:02
62	40	0	22	19	1:59:01
62	40	0	22	19	2:59:01
61	39	0	22	19	3:59:01
61	39	0	22	19	4:59:02
59	37	0	22	19	5:59:01
59	37	0	22	19	6:59:01
59	37	0	22	19	7:59:01
59	37	0	22	19	8:59:02
59	37	0	22	19	9:59:01

56	34	0	22	19	11:59:01
56	34	0	22	19	12:59:01
57	35	0	22	19	13:59:01
55	33	0	22	19	14:59:01
56	34	0	22	19	15:59:01
57	35	0	22	19	16:59:01
56	34	0	22	19	17:59:02
53	31	0	22	19	18:59:01
60	31	0	29	19	19:59:01
59	30	0	29	19	20:59:01
57	28	0	29	19	21:59:01
57	28	0	29	19	22:59:01
69	40	0	29	19	23:59:01
70	41	0	29	20	0:59:02
69	40	0	29	20	1:59:01
69	40	0	29	20	2:59:01
67	39	0	28	20	3:59:01
66	38	0	28	20	4:59:01
65	37	0	28	20	5:59:01
65	37	0	28	20	6:59:02
65	37	0	28	20	7:59:01
65	37	0	28	20	8:59:02
64	36	0	28	20	9:59:01
64	36	0	28	20	10:59:01
64	36	0	28	20	11:59:01
64	36	0	28	20	12:59:01
68	40	0	28	20	13:59:01
68	39	0	29	20	14:59:01
67	39	0	28	20	15:59:01
67	39	0	28	20	16:59:01
68	40	0	28	20	17:59:01
68	40	0	28	20	18:59:01
55	27	0	28	20	19:59:01
61	29	0	32	20	20:59:01
60	28	0	32	20	21:59:01
60	28	0	32	20	22:59:01
61	28	0	33	20	23:59:01
61	28	0	33	21	0:59:01
60	27	0	33	21	1:59:01
59	26	0	33	21	2:59:01
57	24	0	33	21	3:59:02
107	74	0	33	21	4:59:02
107	74	0	33	21	5:59:01
107	74	0	33	21	6:59:01
106	73	0	33	21	7:59:01
105	73	0	32	21	8:59:01
105	72	0	33	21	9:59:02
111	78	0	33	21	10:59:01
112	79	0	33	21	11:59:01

123	88	0	35	21	13:59:01
129	94	0	35	21	14:59:01
128	92	0	36	21	15:59:01
127	91	1	35	21	16:59:01
126	91	0	35	21	17:59:01
125	90	0	35	21	18:59:01
128	91	2	35	21	19:59:01
129	90	4	35	21	20:59:01
132	90	6	36	21	21:59:01
70	35	0	35	21	22:59:01
69	37	0	32	21	23:59:01
69	37	0	32	22	0:59:01
67	35	0	32	22	1:59:01
66	35	0	31	22	2:59:01
66	35	0	31	22	3:59:01
66	35	0	31	22	4:59:02
66	35	0	31	22	5:59:01
64	33	0	31	22	6:59:01
64	33	0	31	22	7:59:01
67	34	0	33	22	8:59:01
68	35	0	33	22	9:59:01
71	39	0	32	22	10:59:01
74	42	0	32	22	11:59:01
74	42	0	32	22	12:59:01
71	40	0	31	22	13:59:01
78	43	3	32	22	14:59:01
76	41	3	32	22	15:59:02
80	46	2	32	22	16:59:01
85	52	0	33	22	17:59:01
81	49	0	32	22	18:59:02
103	45	10	48	22	19:59:01
101	44	10	47	22	20:59:01
111	54	10	47	22	21:59:01
111	54	10	47	22	22:59:01
107	50	10	47	22	23:59:02
107	54	10	43	23	0:59:01
103	50	10	43	23	1:59:01
96	44	10	42	23	2:59:01
95	43	10	42	23	3:59:01
95	43	10	42	23	4:59:01
93	43	10	40	23	5:59:02
94	44	10	40	23	6:59:01
95	47	10	38	23	7:59:01
89	46	9	34	23	8:59:01
89	46	9	34	23	9:59:01
96	48	10	38	23	10:59:01
96	49	10	37	23	11:59:02
90	48	9	33	23	12:59:01
91	50	9	32	23	13:59:01

93	54	9	30	23	15:59:01
92	53	9	30	23	16:59:01
90	53	7	30	23	17:59:01
94	57	7	30	23	18:59:01
81	51	0	30	23	19:59:01
76	46	0	30	23	20:59:01
73	43	0	30	23	21:59:01
75	43	0	32	23	22:59:01
73	42	0	31	23	23:59:02
73	42	0	31	24	0:59:01
70	39	0	31	24	1:59:01
70	39	0	31	24	2:59:01
68	38	0	30	24	3:59:01
67	37	0	30	24	4:59:01
67	37	0	30	24	5:59:01
67	37	0	30	24	6:59:01
67	37	0	30	24	7:59:01
48	38	0	10	24	8:59:02
49	39	0	10	24	9:59:01
49	37	0	12	24	10:59:01
52	39	0	13	24	11:59:01
45	35	0	10	24	12:59:02
46	36	0	10	24	13:59:01
55	44	1	10	24	14:59:01
58	45	3	10	24	15:59:01
56	43	3	10	24	16:59:01
53	42	1	10	24	17:59:01
54	42	2	10	24	18:59:01
51	41	0	10	24	19:59:01
48	38	0	10	24	20:59:02
49	39	0	10	24	21:59:01
50	39	0	11	24	22:59:01
48	37	0	11	24	23:59:01
47	36	0	11	25	0:59:01
47	36	0	11	25	1:59:01
48	36	0	12	25	2:59:01
47	35	0	12	25	3:59:01
47	35	0	12	25	4:59:01
46	35	0	11	25	5:59:01
46	35	0	11	25	6:59:01
46	35	0	11	25	7:59:02
47	36	0	11	25	8:59:01
49	38	0	11	25	9:59:01
49	38	0	11	25	10:59:01
48	37	0	11	25	11:59:01
56	45	0	11	25	12:59:01
66	52	0	14	25	13:59:01
65	51	0	14	25	14:59:01
76	58	4	14	25	15:59:01

64	54	0	10	25	17:59:01
64	53	1	10	25	18:59:01
69	59	0	10	25	19:59:01
65	56	0	9	25	20:59:02
63	54	0	9	25	21:59:01
57	48	0	9	25	22:59:01
57	48	0	9	25	23:59:01
57	48	0	9	26	0:59:01
56	47	0	9	26	1:59:01
55	46	0	9	26	2:59:01
55	46	0	9	26	3:59:01
54	45	0	9	26	4:59:02
54	45	0	9	26	5:59:01
46	37	0	9	26	6:59:01
42	33	0	9	26	7:59:01
39	30	0	9	26	8:59:01
38	29	0	9	26	9:59:01
36	27	0	9	26	10:59:02
39	29	0	10	26	11:59:01
39	29	0	10	26	12:59:01
40	30	0	10	26	13:59:01
39	29	0	10	26	14:59:01
43	33	0	10	26	15:59:01
41	32	0	9	26	16:59:01
42	33	0	9	26	17:59:01
41	33	0	8	26	18:59:01
42	34	0	8	26	19:59:01
47	33	0	14	26	20:59:01
47	33	0	14	26	21:59:01
47	33	0	14	26	22:59:01
47	33	0	14	26	23:59:01
48	34	0	14	27	0:59:01
47	33	0	14	27	1:59:01
48	34	0	14	27	2:59:02
46	32	0	14	27	3:59:01
46	32	0	14	27	4:59:01
46	32	0	14	27	5:59:01
44	30	0	14	27	6:59:01
44	30	0	14	27	7:59:01
44	30	0	14	27	8:59:01
44	30	0	14	27	9:59:01
42	28	0	14	27	10:59:01
43	29	0	14	27	11:59:01
41	27	0	14	27	12:59:01
41	27	0	14	27	13:59:01
46	32	0	14	27	14:59:01
47	33	0	14	27	15:59:01
45	31	0	14	27	16:59:01
44	30	0	14	27	17:59:01

45	30	0	15	27	19:59:01
43	28	0	15	27	20:59:01
43	28	0	15	27	21:59:02
43	28	0	15	27	22:59:03
43	28	0	15	27	23:59:02
44	29	0	15	28	0:59:02
41	26	0	15	28	1:59:02
41	26	0	15	28	2:59:02
41	26	0	15	28	3:59:01
41	26	0	15	28	4:59:02
41	26	0	15	28	5:59:02
40	25	6	9	28	6:59:01
40	25	6	9	28	7:59:01
40	25	6	9	28	8:59:01
40	25	6	9	28	9:59:01
40	31	0	9	28	10:59:01
37	28	0	9	28	11:59:01
41	31	0	10	28	12:59:01
41	29	1	11	28	13:59:01
50	37	0	13	28	14:59:01
52	38	0	14	28	15:59:01
50	37	0	13	28	16:59:01
46	35	0	11	28	17:59:01
48	37	0	11	28	18:59:01
47	35	0	12	28	19:59:01
94	82	0	12	28	20:59:01
95	83	0	12	28	21:59:01
103	91	0	12	28	22:59:01
96	84	0	12	28	23:59:01
95	83	0	12	29	0:59:02
95	83	0	12	29	1:59:02
94	82	0	12	29	2:59:01
94	82	0	12	29	3:59:02
93	81	0	12	29	4:59:02
92	80	0	12	29	5:59:01
92	80	0	12	29	6:59:01
92	80	0	12	29	7:59:01
93	81	0	12	29	8:59:01
94	82	0	12	29	9:59:01
95	83	0	12	29	10:59:01
47	35	0	12	29	11:59:01
50	41	0	9	29	12:59:01
51	42	0	9	29	13:59:02
54	45	0	9	29	14:59:03
54	45	0	9	29	15:59:01
53	44	0	9	29	16:59:01
54	45	0	9	29	17:59:01
55	46	0	9	29	18:59:01
52	44	0	8	29	19:59:01