Data Center Specific Thermal and Energy Saving Techniques

by

Tausif Muzaffar

A dissertation proposal submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 1, 2015

Keywords: Thermal, Modeling, Hadoop, Storage Systems, Energy Efficiency

Approved by

Xiao Qin, Associate Professor of Computer Science and Software Engineering
Cheryl Seals, Associate Professor of Computer Science and Software Engineering
Alvin Lim, Associate Professor of Computer Science and Software Engineering
Jeffrey Overbey, Assistant Professor of Computer Science and Software Engineering

Abstract

Data centers are ever increasing as we become more reliant on web based transactions. The benefits of such massive computing are obvious by the speed and ease we can get most media or information. A challenge is that new large data centers introduce a level of energy consumption that the world has not seen before. The obvious energy cost of running the computers is a billion dollar problem, but there are hidden costs like running cooling systems as well. Moreover, data centers are getting more concentrated on specific tasks, be it SQL or Hadoop or anything else an organization needs. To help combat the problems of large data centers, we aim at developing solutions that can work for each type of data center. This could entail creating tools that are generic enough to work for all data centers, or focusing on specific tools the type of software running in the data center. Our dissertation study works in both ways. We build a thermal model that is flexible enough to be applicable for all data centers; within our thermal model research we even show how it can be used to save energy. We also create energy saving techniques for Hadoop clusters specifically, where we focus on very data centric implementations of Hadoop to gain a significant energy savings. Lastly we propose a Spark specific process that takes what we have learned from Hadoop and thermal research and developed techniques that offer large energy and thermal savings within Spark clusters.

Acknowledgments

"Seeking knowledge is a duty on every Muslim." (Bukhari)

That quote that really resonated in my time as a graduate student, not only was the work I did to expand myself as an academic but it was a responsibility for me to always have a curiosity and thirst for knowledge. I hope that my time in Auburn helps to please Him.

I want to take a moment and thank all those who have helped me everyday in my pursuit in knowledge. Thank you to my family, my peers, my professors, and my committee members for all your direct and indirect help because.

I also want to make sure to specifically thank Dr. Qin, with out his input and encouragement none of this would be possible. Thank you!

Table of Contents

Abstract ii

Acknowledgments iii

List of Figures viii

List of Tables x

1 Introduction 1

2 iTad 3

2.1 Introduction 3

2.1.1 Reducing Monitoring Cost 4

2.1.2 Reducing Monitoring Time 4

2.1.3 Benefits of Thermal Model 4

2.1.4 Contributions 5

2.2 Related Work 6

2.2.1 Energy-Efficient Data Centers 6

2.2.2 Thermal Aware Data Centers 7

2.2.3 Thermal Simulations 7

2.2.4 Thermal Models 8

2.3 Methodology 9

2.3.1 Determine Recirculation Factors 9

2.3.2 Determine Hardware Factors 12

2.4 Modeling 12

2.4.1 Assumptions and Notation - 12

2.4.2 Modeling impacts of heat on temperature 14

2.4.3 Modeling impacts of workload on temperature 15

2.5 Experimental Parameters 17
2.5.1 Set up 17
2.5.2 Period 18
2.5.3 Determining a baseline temperature 20
2.5.4 Impact of I/O utilization on temperature 21
2.5.5 Impact of CPU utilization on temperature 22
2.5.6 Shared I/O and CPU Utilization 23
2.5.7 Determining Constants 23
2.6 Usage 27
2.6.1 Verification 28
2.6.2 MPI 28
2.6.3 Hadoop 30
2.7 Experience 30
2.7.1 Improvements 30
2.7.2 Extension 31
2.8 Conclusion 32
3 NAP 33
3.1 Introduction 33
3.2 Related Work 35
3.2.1 Energy Aware Datacenters 35
3.2.2 Green Hadoop 36
3.2.3 Disk Management 36
3.3 Motivations 37
3.3.1 Hadoop Architecture 37
3.3.2 Disk Consumption 38
3.3.3 Hadoop Disk Management 40
3.3.4 Dynamic Disk Power 41

3.4 Methods and Materials . . . 42
3.4.1 Equipment and Software . . . 42
3.4.2 Algorithm Design . . . 45
3.5 Experiments and Results . . . 47
3.5.1 Block Size . . . 47
3.5.2 File Size . . . 48
3.5.3 Map vs Reduce Routine . . . 49
3.5.4 Map Heavy vs Reduce Heavy Programs . . . 51
3.6 Evaluation . . . 53
3.6.1 Model . . . 53
3.6.2 Usage . . . 55
3.7 Future Work . . . 57
3.7.1 Reactive and Proactive algorithms . . . 57
3.7.2 Savings model . . . 57
3.7.3 Heterogeneous disk arrangement . . . 58
3.8 Conclusion . . . 58
4 Sparke . . . 60
4.1 Introduction . . . 60
4.2 Related Work . . . 61
4.2.1 Spark Benchmarks . . . 61
4.2.2 Memory Consumption . . . 61
4.2.3 Mobile Consumption . . . 62
4.2.4 Hadoop Power Management . . . 62
4.3 Motivation . . . 62
4.4 Spark-e . . . 63
4.4.1 Algorithms . . . 63
4.5 Setup . . . 65

4.6 Observations 65
4.6.1 Thermal Observations 65
4.6.2 Energy Observations 66
4.7 Model and Simulations 69
4.7.1 Thermal and Standard Energy Model 69
4.7.2 Thermal and Standard Energy Simulation 73
4.8 Future Work 73
4.9 Conclusion 74
5 Savings 76
6 Conclusion 77
Bibliography 79

List of Figures

2.1 Temperature of Processor when CPU utilization 100% vs Time [56]. 6

2.2 Thermometer used in testing 7

2.3 An Overview of a data center. 9

2.4 Model Overview 10

2.5 Three factors affect the outlet temperature of a single blade server. 11

2.6 Three factors affect the inlet temperature of a single blade server. 11

2.7 Radiant heat equals convective heat 15

2.8 Visual representation of workload effects outlet temperature 16

2.9 Utilization of Components at fixed utilization 19

2.10 Measured Area of Temperatures 20

2.11 Surface Temperatures 22

2.12 Utilization Temperature 24

2.13 Relationship between Utilization and Outlet Temperature 25

2.14 Values of Z in all experiments 27

2.15 Verification of Model 29

2.16 Sample MPI Usage 29

3.1 Hadoop Data Flow Chart [40] 38

3.2 Showing the importance of hard drives on energy 39

3.3 Hadoop Disk Configurations 41

3.4 Disk State Energy Utilization [25] 42

3.5 Powermetere 44

3.6 Effect on speed 44

3.7 Results how file size effects the energy savings 50

3.8 The max percentage of disk consumption we could save 56

4.1 Spark vs Hadoop (Group 4) Temperature 67

4.2 Spark vs Hadoop (Group 4) Energy 68

4.3 Spark vs Group Size 68

4.4 Spark vs Group Sizes vs Memory 69

4.5 Spark Energy Savings Model 74

List of Tables

2.1 Model Notation 13
2.2 Server Specifications 18
2.3 Temperature Zones 23
2.4 Compilation of all the values gathered 26
3.1 Recommended Hadoop configurations [44] 39
3.2 Server Specification 42
3.3 Hadoop Application Profiles 51
3.4 Model Variables 54
3.5 Western Digital WD1200JB consumption model 55
4.1 Model Variables 70

Chapter 1

Introduction

Energy efficiency and energy conservation are efforts that becoming important issues for society at whole. The amount of power needed to continue to run our daily lives isn't sustainable. As we scale into a world of more power consumption we will need to find creative aways to fill that power. A good way to get ahead of this problem is to make everything more energy efficient, and today one of the fastest growing energy hogs are data centers. In fact in 2013 data centers contributed to 91 billion kilowatt-hours of electricity, enough electricity to power all the households in New York City twice over [43]. The NRDC states that much of the power used by these data centers is in "under-utilization of data center equipment and the misalignment of incentives, including in the fast growing multi-tenant data center market segment." [43]. Since the data centers are growing and there is opportunity to create more efficient energy aware data centers we simply had to decided to what level of granularity do we want to effect the data centers.

Not only is energy important recent studies show that thermal management is an important issue to data centers due to ever-increasing cooling cost [57]. Cooling costs contribute to a significant portion of the operational cost of large-scale data centers; therefore, increasing the size of a data center leads to huge amount of energy consumed by the center's cooling system. An efficient way to combat the high cost of cooling systems is to develop thermal-aware management techniques that place jobs and data on servers to minimize temperatures of data centers.

Its the ubiquity of data centers that makes energy and thermal management so interesting. Our research focuses on finding creating ways to manage and save energy within

specific data center environments. We create these tools and processes then go on validate them so they can be implemented on real life systems.

Contributions.

1. We created iTad, which is a thermal model for single server nodes. This model if light weight so any data center can make smart decisions on thermal energy.

2. We created NAP, which is a process for Hadoop clusters to manage its disk energy. This process saves energy at only a minimal cost to performance.

3. We propose a Spark energy profile which we use to develop Sparke which is an energy/thermal aware application for Spark clusters that will use the architecture of spark to find either energy savings or thermal savings.

Organization. Each contribution is under the umbrella of data center optimizations, but vary from perspective of thermal management, Hadoop energy management, or Spark energy and thermal management. So we decided to organized this paper is organized by our contributions. Each chapter will detail the related work, motivations and algorithms for each contribution. We will use those contributions to show how data centers can be optimized with real life data and models as evidence.

Chapter 2

iTad

2.1 Introduction

Recent studies show that thermal management is an important issue to data centers due to ever-increasing cooling cost [57]. Cooling costs contribute to a significant portion of the operational cost of large-scale data centers; therefore, increasing the size of a data center leads to huge amount of energy consumed by the center's cooling system. An efficient way to combat the high cost of cooling systems is to develop thermal-aware management techniques that place jobs

Thermal management aims at reducing cooling costs of data centers; thermal management mechanisms largely rely on thermal information to make intelligent job and data placement decisions. Thermal information can be acquired in the following three means:

1. Temperature sensors measure inlet and outlet temperatures of servers
2. Computational fluid dynamics simulators (see, for example, Flovent) simulate temperatures of servers in data centers
3. Thermal models estimate a server's temperature based on the server's workloads.

After looking through these options we decided to create an CPU and I/O aware thermal model called iTad. iTad standing for I/O Thermal Aware Data center. The reason we decided to make such a model than use the other two options will be explained in the following subsections.

2.1.1 Reducing Monitoring Cost

The first approach is to monitor server inlet temperatures by deploying sensors in a number of locations in a data center [50]. This approach faces a dilemma; while high levels of accuracy can be achieved by increasing the number of sensors, this leads to an expensive monitoring solution. Reversely reducing the number of sensors may cause inaccuracies, and an algorithm would need to be developed to extrapolate the heat from individual nodes thus taking away the simplicity that makes this route so appealing. For large-scale data centers, this approach is not very practical for two reasons. First, it is prohibitively expensive to deploy hundreds of thousands of sensors to offer accurate temperature measurements. Each server needs at least two sensors; each sensor may cost up to $100 [8]. Second, wiring and maintenance cost of the large number of sensors can further increase the operational cost of data centers.

2.1.2 Reducing Monitoring Time

To reduce the high cost of deploying an excessive number of sensors, data center managers can make use of the computational fluid dynamics simulators to simulate and collect inlet temperatures of servers [42]. Although this simulation approach offers accurate thermal information at low cost without employing any sensor, it is time consuming (e.g., several hours) to run each simulation study. Thus, the simulation studies must be conducted offline, indicating that thermal management mechanisms are unable to retrieve thermal information from the simulators at run-time.

2.1.3 Benefits of Thermal Model

Thermal models are arguably a more promising approach to providing thermal management mechanisms; they can provide temperature information of servers at run-time without incurring any cost to purchase and maintain sensors. Thermal models offer the following four major benefits for data centers. First, thermal models significantly reduce thermal

monitoring costs. Second, unlike thermal simulators, thermal models offer temperature information to thermal management schemes in a real-time manner. For example, our iTad thermal model is able to profile the thermal characteristics of a data center in a matter of seconds. Third, thermal management powered by thermal models helps cut cooling costs and boosts system reliability. Last, thermal models allows data center designers to quickly make intelligent decisions on thermal management in an early design phase.

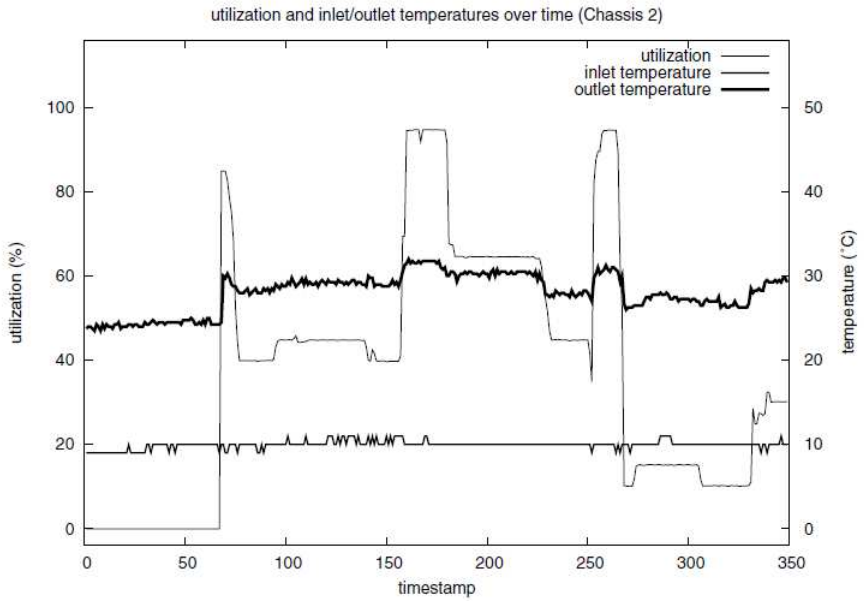
Most existing thermal models in the market treat servers as a uniform black box because it is unclear what all factors are involved in the heat distribution of a data center [37]. There are a few thermal models (see, for example, [56]) that can derive power consumption and necessary cool power from inlet and outlet temperatures of servers. However, implementing these models requires (1) many thermometers and (2) the management of thermal information in real time (3)only based on CPU work. CPU Workload has been known to affect thermal load, for example Figure 2.1 shows that when CPU utilization is increased, there is a large increase in temperature [56], but it neglects I/O workload. I/O-intensive activities in servers are commonly overlooked in these models. One of the goals of this research is to demonstrate that I/O utilization plays an important role in a server's thermal dissipation. We believe I/O-intensive applications running in data centers impose heavy load on servers, making disks of the servers hot-spots.

2.1.4 Contributions

Contributions. The major contribution of this study are summarized as follows:

1. We develop the iTad thermal model that provides outlet temperatures of servers in a data center. We show that both CPU and I/O thermal outputs can be extrapolated from radiation heat and convection heat applied to a server. With iTad in place, thermal management schemes can quickly make workload management decisions at run-time based on I/O and CPU utilizations.

Figure 2.1: Temperature of Processor when CPU utilization 100% vs Time [56].



2. We validate the accuracy of the iTad model using a server's real-world temperature measurements obtained by an infrared thermometer (Figure 2.2).

3. Our experimental results suggest that iTad is an accurate model to derive server outlet temperatures according to I/O and CPU activities.

4. We show that this model is easily can be plugged into any data centers.

5. We analytically study the relationship between I/O load and server outlet temperatures. Our analysis confirms that I/O-intensive workloads have significant impact on temperatures of servers.

2.2 Related Work

2.2.1 Energy-Efficient Data Centers

Large data has become on the hottest topics in computer. With refocus there has been more interest in energy efficient data centers [11] [12], because a recent study shows that 1.2% of all energy consumption in U.S. is attributed to data centers [33]. To minimize the

Figure 2.2: Thermometer used in testing

effect of the data centers on the national consumption there has been many energy-saving approaches, one that relates to this research is the work of Bieswanger et al. where they deploy sensors to analyze the power consumption instantaneously, our research deals with real-time thermal management has some overlap [9].

2.2.2 Thermal Aware Data Centers

Energy aware data centers has been the classical way of thinking about reducing the effect of data centers on the environment. Another school of though is if we manage the thermal outputs of the data centers, thus reducing the cooling cost we can have the same impact that as energy efficient data centers. [58]

2.2.3 Thermal Simulations

Most of the research related to thermal management in data centers use a commercial simulation software *FloVent*, which provides detailed 3D visualization of airflow and temperature throughout the server room [2]. It can get very accurate heat recirculation results. The downside is that, it is very complicated to setup or configure and it takes huge amount of time to run each simulation. Such software is very useful for machine learning because of the time needed to implement machine learning techniques but not very effective on split

second decision making. We use iTad to implement a low cost and less time consuming management technique.

2.2.4 Thermal Models

Eibeck *et al* [17] developed a model to predict the transient temperature profile of an IBM 5-1/4-in. fixed disk drives by experimentally determining the thermal characteristics of the disk drive. Tan *et al* presented a 3D finite element modeling technique to predict the transient temperature under frequent seeking [55]. Gurumurthi *et al* investigated the thermal behavior of the hard disk and presented an integrated disk model. Their model calculates the heat generated from the physical components of the disk drive like spindle motor, voice-coil motor and disk arms [22]. Kim *et al* studied thermal behavior of disks by varying the platter types and number of platters and established a relationship between seek time and the disk temperature [30]. However, the impact of the disks utilization on the disk temperature and contribution of disks to the outlet temperature of nodes have not been investigated. Even though clearly thermal footprints of computing has a breathe of research.

Microsoft research and Carnegie Mellon University [37] presented a model which predicts the future temperature of servers through machine learning. As this model relies on the sensor data, it will be costlier for large data centers to buy large amount of sensors. In our research, instead of predicting future temperature we want a model to calculate the current temperature based on the workload without using sensors.

Tang et al [59] [56] developed an interesting model demonstrating the effect of heat recirculation on the inlet temperature of servers in a data center, and in turn, on the efficiency of cooling system. They calculate inlet temperature of servers based on the temperature of the air supplied by the cooling room air condition (CRAC) and CPU utilization. Li et al [36] showed that CPU intensive applications cause dramatic heat change for processor. We believe that data intensive applications running in data centers will have the similar effect on the disks of storage nodes, which has to be taken into account while calculating the total

heat generated by the node. Kozyrakis [16] studied the effect of different application and observed the power consumption of the nodes. It showed that disk and memory consumes significant amount of power, even as compared to CPU (as shown in Example 2). As power consumption has direct impact on heat generated, there is a need to investigate the thermal load of I/O intensive applications on the nodes in data centers.

2.3 Methodology

2.3.1 Determine Recirculation Factors

We achieve the aforementioned goal by focusing on heat recirculation of active data centers. Figure 2.3 depicts a general model for a data center, where each blade server's outlet temperature affects room temperatures. The outlet temperature of the server depends on its inlet air that enters the front of the server's rack. The inlet air temperature is the computer room temperature cooled by an air conditioning system.

Figure 2.3: An Overview of a data center.

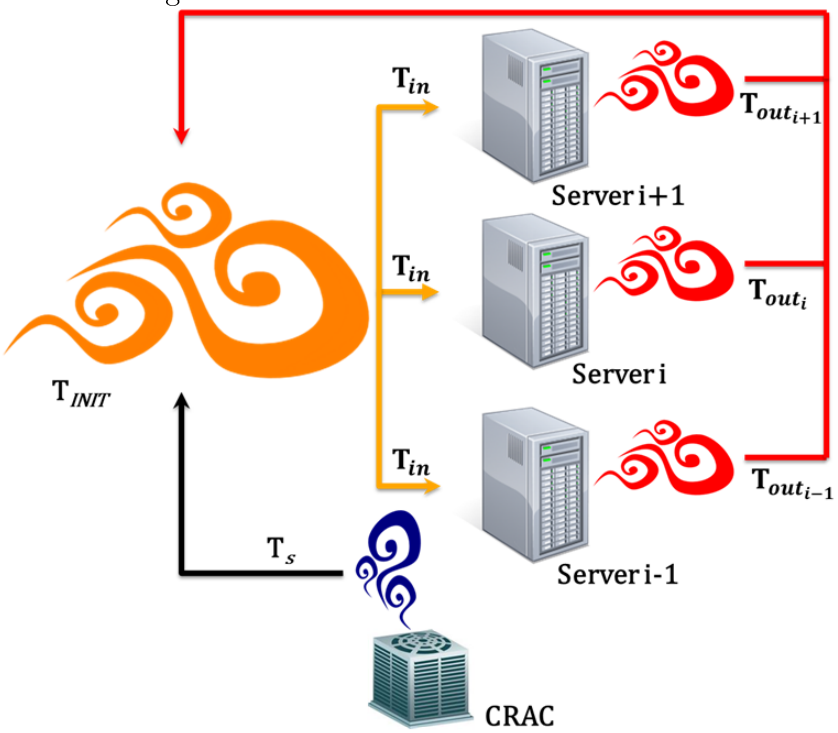
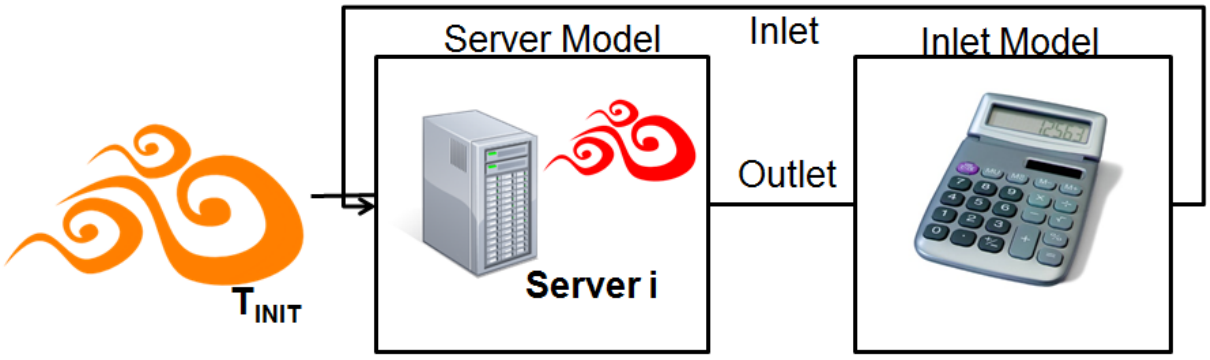


Figure 2.3 shows that heat recirculation in a data center can be derived as the sum of each server's outlet temperature. To build a model representing the heat recirculation of a data center (see Figure 2.3), we start this study by paying attention to constructing a thermal model for each individual server. Here we are using the assumption that since recirculation is the sum of single servers, then if we can model a single server, we just need to apply it to all the servers in the data center and add it all together. Essentially we will modeling two different things, the first will be the server heat transfer and the the inlet temperatures value before the server heat transfer. Figure 2.4 shows how the initial temperature will feed into our first model and they feeds into our inlet temperature model.

Figure 2.4: Model Overview



In our iTad model, there are three components (see Figure 2.5, where T_{out} denote outlet temperature) affecting the outlet temperatures of a blade server. These three affecting factors are inlet temperature, CPU utilization, and I/O workload.

Our iTad model makes use of these factors to estimate the outlet temperature for server i, thereby enabling thermal management schemes to place workloads to control outlet temperatures. The iTad model is orthogonal to existing thermal management schemes; iTad can be seamlessly integrated with any thermal management scheme to either minimize outlet temperatures or minimize heat recirculation in a data center. In this study, we focus on the accuracy of iTad by validating it against real-world temperature measured by an infrared thermometer.

Figure 2.5: Three factors affect the outlet temperature of a single blade server.

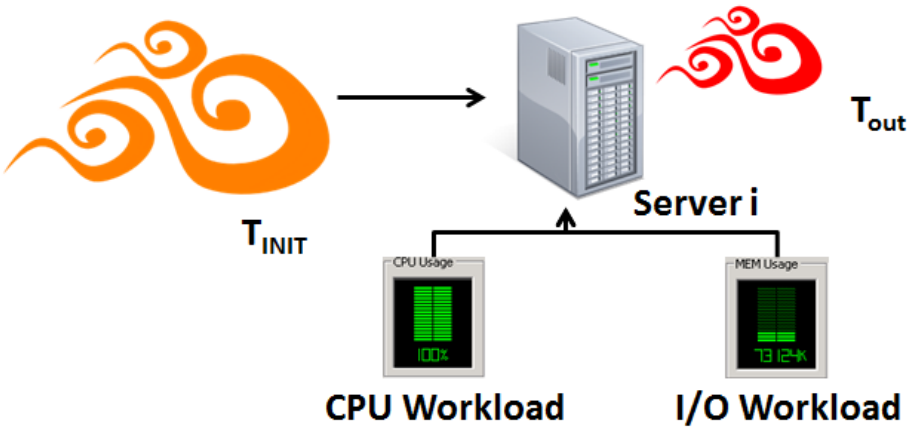
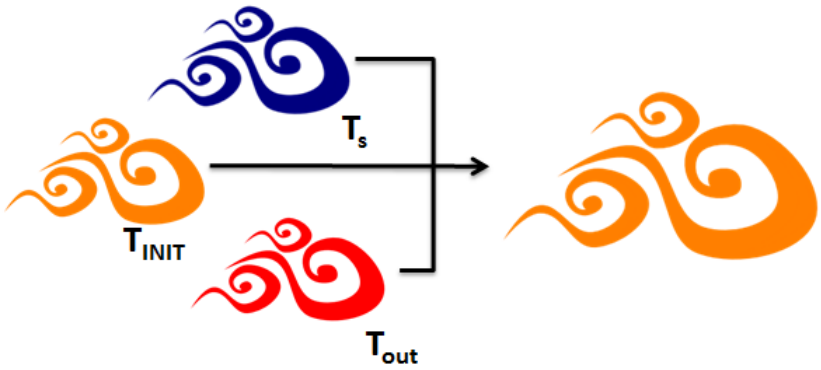


Figure 2.6: Three factors affect the inlet temperature of a single blade server.



A challenge in the development of iTad is the measurements of inlet temperatures of servers. More specifically, Figure 2.3 indicates that the air entering the servers is not equivalent to initial temperature. Rather, the inlet temperature equals to the initial temperature subtracted by some factor of air supplied by the air conditioning system. The inlet temperature of a server is affected by three factors, namely, computer room temperature, cooling supply air temperature, and the outlet temperatures of other servers (see Figure 2.6). For this model we decided to model only the current server outlet temperature an instantaneous moment so its the only one that affects input temperatures. In one of our current studies, we are extending the iTad model to investigate the heat recirculation effect by considering the impact of all nodes outlet temperatures on inlet temperatures.

2.3.2 Determine Hardware Factors

After dealing with actual inlet temperatures, we incorporate I/O and CPU workloads into iTad. In this part of the study, we show how the outlet temperature of a server changes based on I/O-intensive activities. The iTad model has to deal with heat transfer, especially convection heat transfer. Convection heat transfer [60] is based on temperature and specific heat, all of which have a linear relationship. A study conducted by Barra and Ellzey demonstrates how a wide range of shapes affect heat transfer [10]. iTad is the first model that attempts to incorporate I/O-intensive workload therefore, we consider cases where all the components in a data center have the same transfer rate. Nevertheless, we do not imply by any means that all the components have an identical transfer rate. In our future work, we will extend iTad to consider multiple heat transfer rates to further improve the accuracy of iTad.

The iTad model helps in improving the energy efficiency of data centers because thermal information offered by iTad assists dynamic thermal management to reduce the energy consumption in cooling systems in data centers. We show that thermal management mechanisms can quickly make workload placement decisions based on thermal information facilitated by iTad.

2.4 Modeling

2.4.1 Assumptions and Notation -

We described the plan of our model as well as the basic components necessary for the model. In this subsection, we will present the assumptions and the notations we used in the model. Following are the assumptions :

1. Initial temperature is always consistent throughout the data center.

2. The air flow is static in all parts of the data center.

3. Supplied temperature strength is linearly proportional to the distance from the vent.

4. Our model is models temperature at an instantaneous moment so nothing is being circulated in our model.

5. The adjacent nodes will not heat up enough to cause an effect to the node in question.

6. PC components are all similar in shape so the heat transfer is consistent.

7. The entire experiment is based on the premise that taking a single node from a cluster and running our experiments we can grasp the important factors in thermal change in computers. With this information we will able to model large scale environment.

After laying out the assumptions, the notations used in the model are described in Table 2.1.

Table 2.1: Model Notation

Variables	Description
i	Number of Server Node
Q	Heat generated (J)
p	Density of air (kg/m^3)
f	Flow rate (m^3/s)
c_p	Specific Heat (J/kg/c)
T_{out}	Outlet Temperature (c)
T_{in}	Inlet Temperature (c)
T	Change in temperature (c)
h_r	Heat Transfer Coffiecnt (J/s*m^2*c)
A	Surface area of PC components(m^2)
Z	Percent of added temperature after workload
R	Ratio of distance
k	The amount outlet affects inlet temperature
d_i	Distance of the server from AC vent (m)
d	Height of room (m)
T_{INIT}	Room Initial Temperature (c)
T_s	Supplied temperature from CRAC (c)
$T_{workload}$, T_w	Surface Temperature at a workload (c)
T_{idle}	Surface Temperature at a idle (c)
W	Workload supplied (%)
T_{Max}	Max Temperature the components(c)

This equation can be reorganized to solve for outlet temperature.

2.4.2 Modeling impacts of heat on temperature

The heat transfer in a data center node can be expressed by Equation 2.1 [37] [56] [57]. There are two kinds of heat transfer in this system: convective heat transfer adn radiant heat transfer. We organized Equation 2.1 to solve for outlet temperature. In Equation 2.1, Q_i is the convective heat transfer of server i, which means as the inlet air passes through the amount of heat is builds up is Q_i.

$$\begin{aligned} Q_i &= pfc_p(T_{out_i} - T_{in_i}) \\ T_{out_i} &= \frac{Q_i}{pfc_p} + T_{in_i} \end{aligned} \tag{2.1}$$

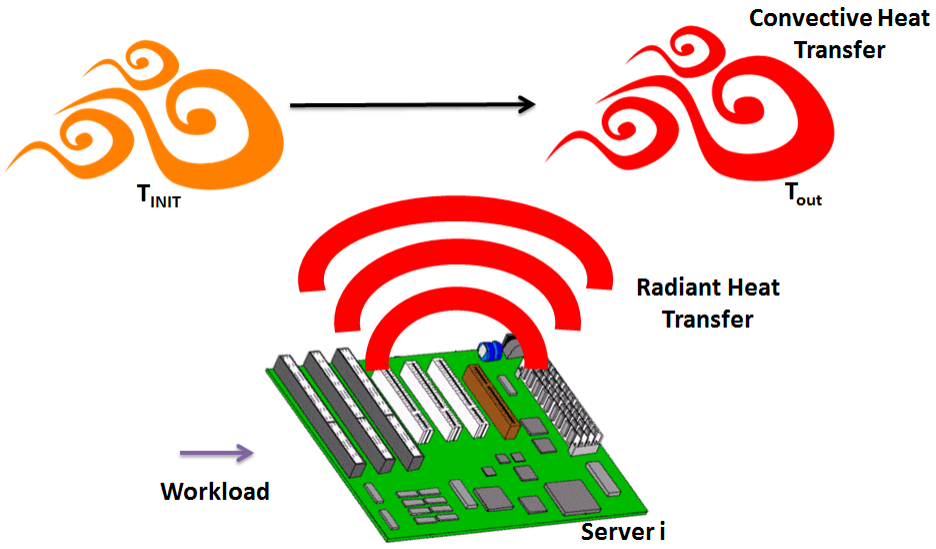
The heat generated in the chassis is actually the heat being radiated from the components of a server, which also know as radiant heat. So, in this case, the convective heat transfer of inlet temperature and outlet temperature is equal to the radiant heat transfer of the PC components. Figure 2.7 shows you the model how all the heat that radiates off the components mixes into the air to form the outlet temperature and its convective heat gain.

The equation 2.2 shows the formula for radiation heat transfer [1]. The radiant heat is dependent on the surface of the object and the heat it generates on its surface.

$$Q_i = h_r A \triangle T_i \tag{2.2}$$

In Equation 2.3, the $\triangle T_i$ is the change of temperature caused by the PC components, which we modeled as the change in temperature of the server at the specified workload ($\triangle T_{workload}$) plus difference between inlet and outlet temperature of server at idle state ($T_{out_{idle}} - T_{in_{idle}}$).

Figure 2.7: Radiant heat equals convective heat



$$\triangle T_i = \triangle T_{workload_i} + (T_{out_{idle}} - T_{in_{idle}}) \tag{2.3}$$

To help simplify what we need to find we set Equation 2.1 and Equation 2.2 equal to each other to give us the variable Z, thus letting us relate T_{out} to T_{in} and $\triangle T_i$, as shown Equation 2.4

$$\begin{aligned} h_r A \triangle T_i &= p f c_p (T_{out_i} - T_{in_i}) \\ Z = \frac{h_r A}{p f c_p} &= \frac{T_{out_i} - T_{in_i}}{\triangle T_i} \\ T_{out_i} &= Z \triangle T_i + T_{in_i} \end{aligned} \tag{2.4}$$

2.4.3 Modeling impacts of workload on temperature

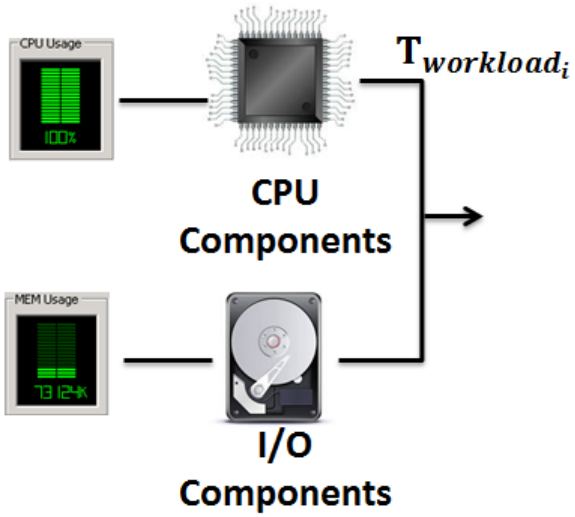
In the article [57], they define T_{in} as dependent on T_s and a vector which models the exact strength of T_s at each height. We simplified the model further by declaring the T_{in} as the room temperature subtracted by the a percentage of the temperature supplied by the CRAC as shown in Equation 2.5.

The amount that T_{out} effects the inlet temperature is proportional to k which is something that is outside the scope of our paper. That being said the way it is implemented now, only the a current server outlet temperature will effect the outlet temperature.

$$R = \frac{d_i}{d}$$
$$T_{in} = T_{INIT} - RT_s + kT_{out} \tag{2.5}$$

Also in Equation 2.3, the other variable that defines T_{out} is $\triangle T_i$, and we modeled $\triangle T_i$ after the Figure 2.8.

Figure 2.8: Visual representation of workload effects outlet temperature



The theory behind our proposed model is that some components of the server get more heated by I/O intensive applications while others get more heated by CPU intensive applications; and based on the percent of CPU or I/O utilization the components will get to some percentage of its maximum temperature. After the calculations of Equation 2.5 we are given $\triangle T_{workload_i}$ which is the increase in the temperature as compared to idle server.

$$\triangle T_{CPU}^{MAX} = T_{workload_{MAXCPU}} - T_{idle}$$
$$\triangle T_{I/O}^{MAX} = T_{workload_{MAXI/O}} - T_{idle}$$
$$\triangle T_{workload_i} = \triangle W_{CPU} \triangle T_{CPU}^{MAX} + \triangle W_{I/O} \triangle T_{I/O}^{MAX} \quad (2.6)$$

In the end all of these equation are the components needed in modeling a single server node. This is important because, as we discussed before, getting each single server outlet temperature can help to model a data center thermal profile. Before we can do that we need to verify that these equations are accurate.

2.5 Experimental Parameters

In this subsection we will be determining the parameters for the model we created in the modeling subsection for server. Do this we need to prove that all the factors describe in the model will indeed have an effect, and then solve for the constants described in the previously in the modeling subsection.

2.5.1 Set up

Since our models, described in the previous subsection, model a single server node, we decided to verify the equations by setting up an experimental machine. The machine we used is an OptiPlex 360 whose specifications are listed in Table 3.2. So in this subsection we will define the characteristics of our machine and later use those constants to verify how accurate our models are.

To test our server we used a command called "stress" in Ubuntu, which can spawn multiple CPU workers or I/O workers. This process would allow us to get a estimate how a computer would act under such a load. To get an estimate of CPU utilization impact we just used "stress" to call only CPU workers [6]. To get an estimate of I/O utilization impact

Table 2.2: Server Specifications

Dimensions	15.65 x 4.59 x 14.25
RAM	1GB
Chipset	Intel G31/ICH7
DC Power Supply	255 W
Processor Type	Intel Core 2 Duo
Memory	800 MHz DDR2 SDRAM

we just used "stress" to call only I/O workers. Finally to find a mixture of I/O and CPU utilization impacts we called a ratio of CPU workers to I/O workers. (i.e 80 CPU workers and 20 I/O workers will be 80% CPU utilization and 20% I/O utilization).

After running our stress tests, we used 3 different tools to help design experiment. First, we used the Linux command "iostat", which gave us details about server usages. The most important pieces of information in "iostat" were the "CPU user%" which displays the percentage of CPU utilization, "system%" which displays the percentage of I/O utilization [4] [5].

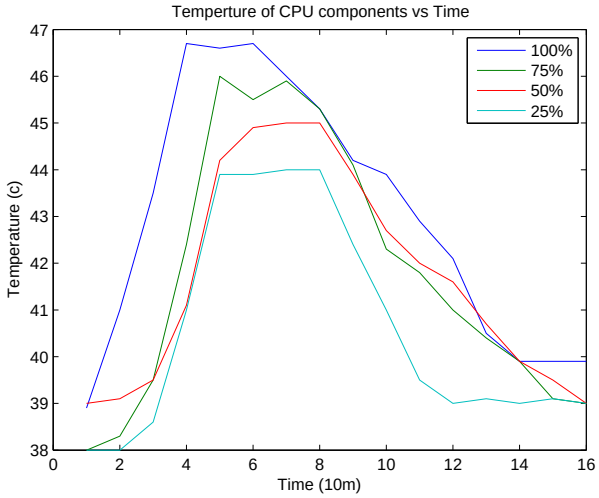
Another tool we used "HDDTemp"; a software that can measure the temperature of the hard drives [3]; more of as a reassurance, to make sure our thermometer was working.

When we say thermometer, we are referring to the HDE Temperature Gun Infrared Thermometer w/ Laser Sight. This thermometer measures the surface temperature of whatever surface it is pointed on. The thermometer has a reading ratio that is 12:1, which means for every 12 cm away we have a 1 cm radius of temperature. We used this tool to measure all kinds of temperatures used for verification.

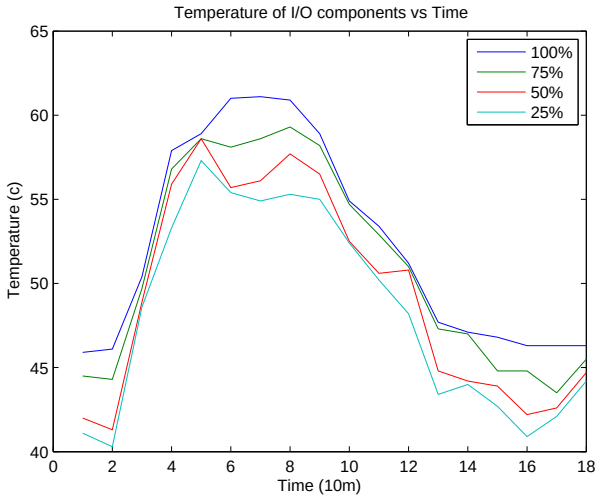
2.5.2 Period

Since it takes time for different components to heat up to its max temperature, we needed to test how long it would take for our each application to reach it hottest point. For CPU intensive application we can assume that the processor would be the most highly active component. So we ran our stress test at 100% CPU utilization and periodically checked the processors temperature. We plotted the temperatures over time as shown in Figure 2.10(a).

Figure 2.9: Utilization of Components at fixed utilization



(a) Temperature of Processor when CPU utilization is changed vs Time

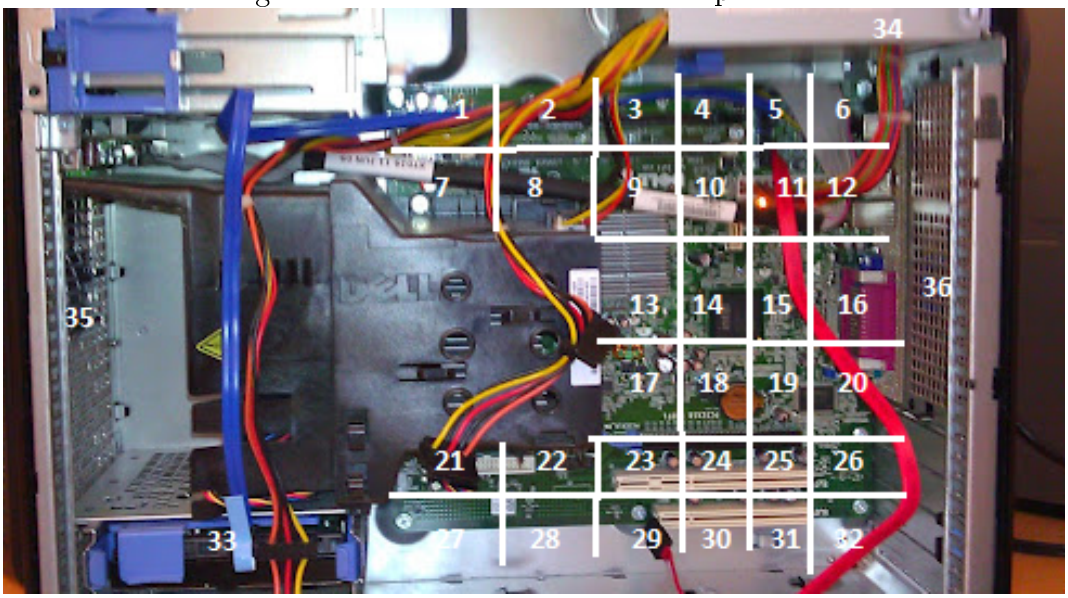


(b) Temperature of Processor when CPU utilization is changed vs Time

If you look at Figure 2.10(a), you can see the temperature plateaus around 30 minutes, but we wait till the blue line to turn off the stress test.

In Figure 2.10(a), we did similar procedure, but we used I/O intensive application. During the I/O intensive application, instead of monitoring the processor we monitored the I/O controller. From Figure 2.10(b), it is clear that I/O application takes longer to heat up and once we turned it off, at the blue line, it takes longer to cool down. So for all our other

Figure 2.10: Measured Area of Temperatures



tests we run them for 1 hour before taking temperature readings, to give the CPU and I/O components ample time to heat up. We waited 1.5 hours between tests to allow server to cool down.

2.5.3 Determining a baseline temperature

After finding out how long it takes to run an experiment we were able to run our tests. The first experiment we needed to run was one to figure out the thermal impact of the idle machine. In order to do this, we decided to take an array of temperatures and extrapolate the information we need.

Figure 2.10 shows the insides of the our server; the numbers 1-32 are areas where we measured the temperature, 33 is the place we measured the hard drive, 34 is the place where we measured power supply, 35 is the place where measured inlet temperature and finally 36 is the outlet temperature. So once we determined what to measure, we measured each grid area with our thermometer for the server at the idle state. The measurements are given in Figure 2.12(a).

In Figure 2.11, we arranged the gathered data to graphically match Figure 2.10. In Figure2.11, the inlet temperatures are temperatures in the middle on the far left. All the numbers in the middle row are the temperatures of grid spots of 1-32. The two temperatures on the bottom left are the temperature of the disk drive measured by two different methods, one using *HDDtemp* and other using thermometer. After gathering the temperatures we calculated the average, which we labeled off to the bottom on the far right. This gives us a baseline value, which is called T_{idle} in our model. We compared this baseline value with the other values. Another number that we needed to keep for later calculations is the difference between the idle inlet and idle outlet temperatures ($T_{out_{idle}} - T_{in_{idle}}$), which is 1.8^{o}C.

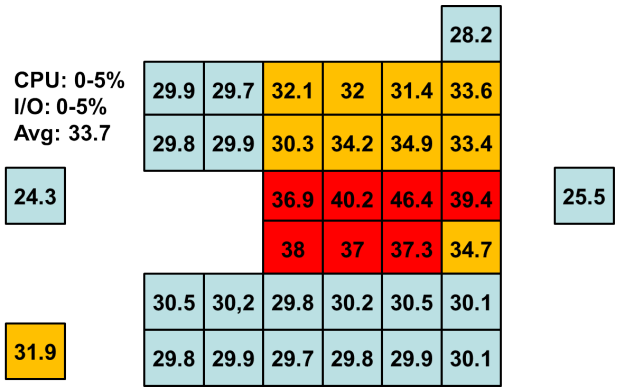
2.5.4 Impact of I/O utilization on temperature

Once we have the idle data as a reference we started to test an active machine. We started with an I/O intensive machine, and to make the machine I/O intensive we use the following command:

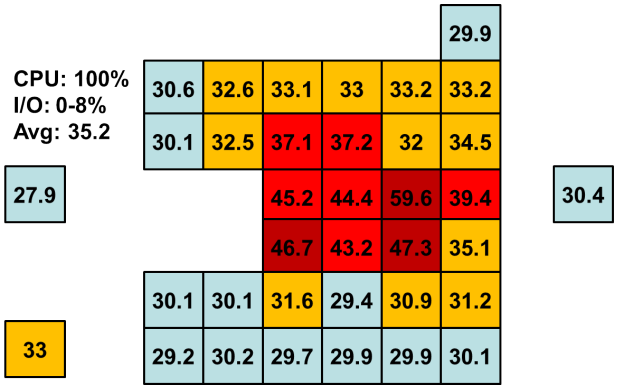
$stress âĂŞio 3 âĂŞvm 7

The ratio of –io to –vm and –io is 30% which means that the I/O is set to 30% utilization. Using this pattern we created the a figure similar to 2.11. We then allowed the computers to cool down and then rechecked the temperatures at 60% utilization, then once more for 100% utilization. Although we kept and organized the data for these runs similar to the format of the idle calculations above for simplicity to show how the utilization effected the heat with Figure 2.12. Each bar represents a group of points from Figure 2.10. Table 2.3 lists each group to help interpret the results.

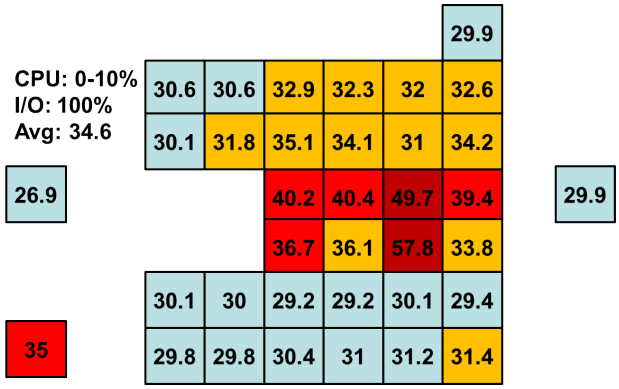
Figure 2.11: Surface Temperatures



(a) Idle Temperatures



(b) Max CPU Temperatures



(c) Max I/O Temperatures

2.5.5 Impact of CPU utilization on temperature

After conducting the I/O tests it was time to test CPU utilization impacts. To do this we used the stress command in the following way:

$stress âĂŞcpu 3 âĂŞvm 7

Table 2.3: Temperature Zones

Group Number	Number of points from Figure 2.10	Reasoning
1	34	Power Supply
2	1-12	Rarely changing
3	13,14,17,18	CPU Controls
4	15,16,19,20	I/O Controls
5	33	Hard drive

The ratio of –cpu to –vm and –cpu is 30% which means that the CPU is set to 30% utilization. Using that pattern we created a figure similar to 2.11. Just as we did for I/O we checked the temperatures at 3 different utilization levels; making sure to give the machine time to cool down before each experiment. We found the average temperature, like we did for the idle case and I/O, and also created a group heat graph as seen in Figure 2.13(b). After plotting the average temperatures we were able use curve fitting techniques which we discuss in subsection 2.5.7.

2.5.6 Shared I/O and CPU Utilization

Finally we decided to test a mixture of both I/O and CPU utilizations. We did this by calling the command:

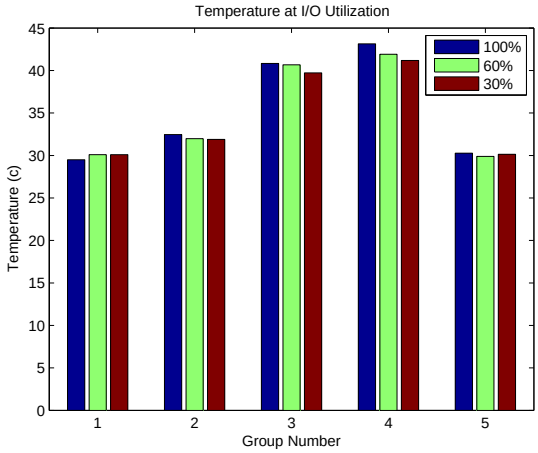
$ stress âĂŞcpu 50 âĂŞio 50

This would set utilization for each component to 50%. So we did the same procedure as we did on the I/O intensive and CPU intensive. After running the experiments for the mixed utilization we created the Figure 2.13(c), which we will discuss in subsection 2.5.7.

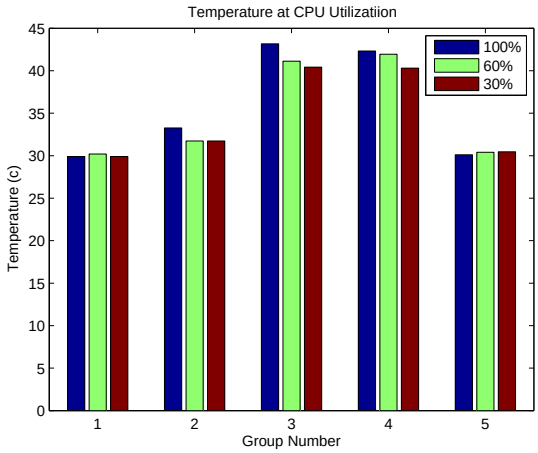
2.5.7 Determining Constants

After our experiments, we are left with many impressions about iTad. First of all, there is a clear relation between utilization and temperature which is shown in Figures 2.13(a), 2.13(b). This relationship seemed to be linear relationship shown by the curve fitting

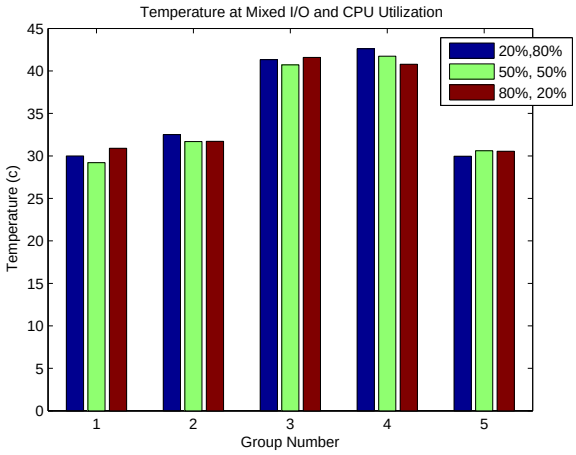
Figure 2.12: Utilization Temperature



(a) I/O Utilization



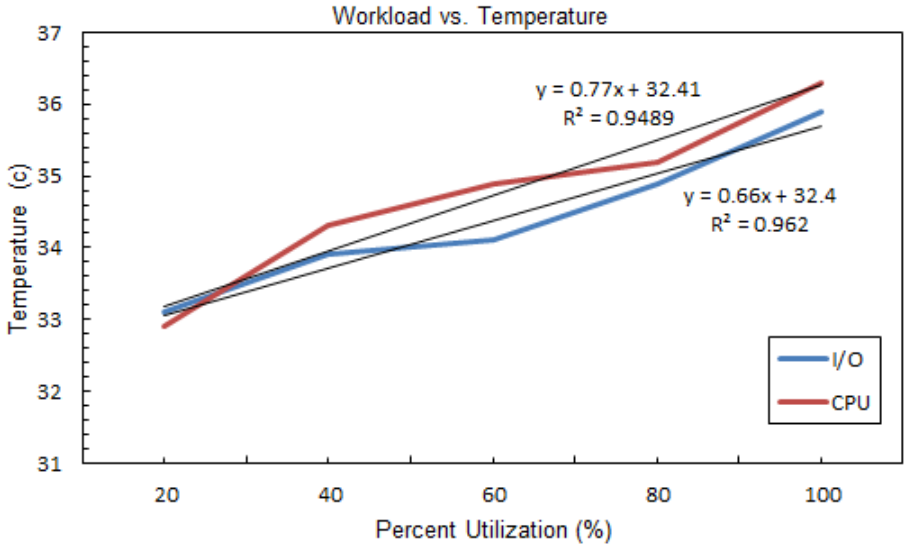
(b) CPU Utilization



(c) Mixture of CPU and I/O Utilization

techniques we used on the Figure (2.13) where we graph the change in outlet temperature change over the percent utilization.

Figure 2.13: Relationship between Utilization and Outlet Temperature



After using the data from the I/O runs we were able to calculate, the slope of the line is 2.7 which represents the speed with which the temperature was increasing with R^2 value of 0.981, where R^2 represents the accuracy of the slope. In the case of CPU data, the slope of the line is 3.5 which represents the speed with which the temperature was increasing with a R^2 value of 0.961. Which indicates that a CPU intensive application will get server hotter than an I/O intensive one. This is nearly confirmed by the mixed data because it clearly shows that when CPU is higher than I/O the server is warmer.

Once we accept that the relationship is linear we can start to figure out some of the values on constants in the equation that we proposed. The proposed Equation 2.4 has consolidated all the constants of the experiment into one variable and with all our data readings we can solve for Z.

In Table 2.4, we consolidated all the information we gather while trying to determine the experimental parameters process. In Table 2.4, column 2 is the average surface temperature of the machine at that utilization, while column 3 is the average surface temperature with

Table 2.4: Compilation of all the values gathered

I/O Intensive

W_{io}	T_W	T_{Idle}	$\Delta T_{Workload}$	$T_{idle_{In-Out}}$	Q	T_{in}	T_{out}	T_{In-Out}	Z
30%	34.021	33.692	0.315	1.800	2.115	26.700	28.700	2.000	0.946
60%	34.237	33.692	0.630	1.800	2.430	24.700	26.900	2.200	0.905
100%	34.742	33.692	1.050	1.800	2.850	26.900	29.900	3.000	1.052

CPU Intensive

W_{cpu}	T_W	T_{Idle}	$\Delta T_{Workload}$	$T_{idle_{In-Out}}$	Q	T_{in}	T_{out}	T_{In-Out}	Z
30%	34.039	33.692	0.442	1.800	2.242	26.100	28.300	2.200	0.981
60%	34.400	33.692	0.884	1.800	2.684	26.900	29.700	2.800	1.043
100%	35.166	33.692	1.474	1.800	3.274	27.900	31.400	3.500	1.069

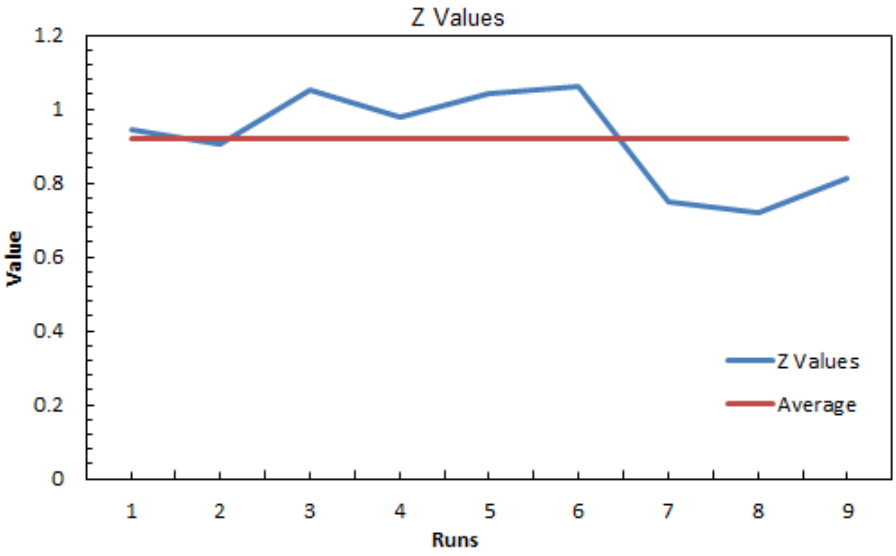
I/O and CPU Intensive

W_{cpu},W_{io}	T_W	T_{Idle}	$\Delta T_{Workload}$	$T_{idle_{In-Out}}$	Q	T_{in}	T_{out}	T_{In-Out}	Z
20%,80%	34.347	33.692	1.135	1.800	2.935	27.700	29.900	2.200	0.750
50%,50%	34.326	33.692	1.262	1.800	3.062	26.600	28.800	2.200	0.718
80%,20%	34.639	33.692	1.389	1.800	3.189	26.200	28.800	2.600	0.815

no utilization; the difference in column 2 and 3 is the observed difference in the values. This shows how much extra heat is generated once the server is pushed to that specific utilization. Column 4 shows what the extra temperature generated should would be, from iTad. As you can see the values are closely related. Any difference could be accounted by the change in air flow of the room or own server fans.

Column 5 of Table 2.4 is $T_{workload}$ we calculated plus the difference between inlet and outlet temperature for the idle server. This is essentially $\triangle T$ from Equation 2.4. And since we were able to actually measure the final inlet and outlet temperature for server, we were able to calculate the value of Z. The value of Z is ratio of $\triangle T$ from column 9 to the surface heat listed in column 6. As you can see, in Figure 2.14, the value of Z has an average just under 1.

Figure 2.14: Values of Z in all experiments



This simply means that the current arrangement of hardware has a one to one relationship between heat exuding from the server and outlet temperature. The change in Z for different utilization maybe an indication that the air flow is changing, but our assumption is that the air flow stays constant and since the values don't vary that much it doesn't contradict iTad.

Through our results the accuracy seems close enough to say that the equations used to model the outlet temperature can work as a basis to for thermal management based on workload, or even just a starting point for future expansion.

2.6 Usage

In the verification subsection we took our server and determined all the constants of the machine. In the following subsection we will discuss how accurate the numbers from the verification process are. First we would simply like to explain a use case how this model can be used a in data center.

As we you can tell by Equation 2.4 the most important value you need to solve for the the Z variable. So when building a data center you should find the Z for every machine then you can plug it into equations.

After gathering the Z values like we did in the verification subsection for all the machines we will need to run another task that ill keep track of CPU and I/O utilization of each machine

After setting up a monitor you take the values and plug them into Equation 2.6 and the outcomes will give the individual outlet temperature. Now this is where the model is lacking there is no heat circulation model, that will need to extended research to help control thermal output.

2.6.1 Verification

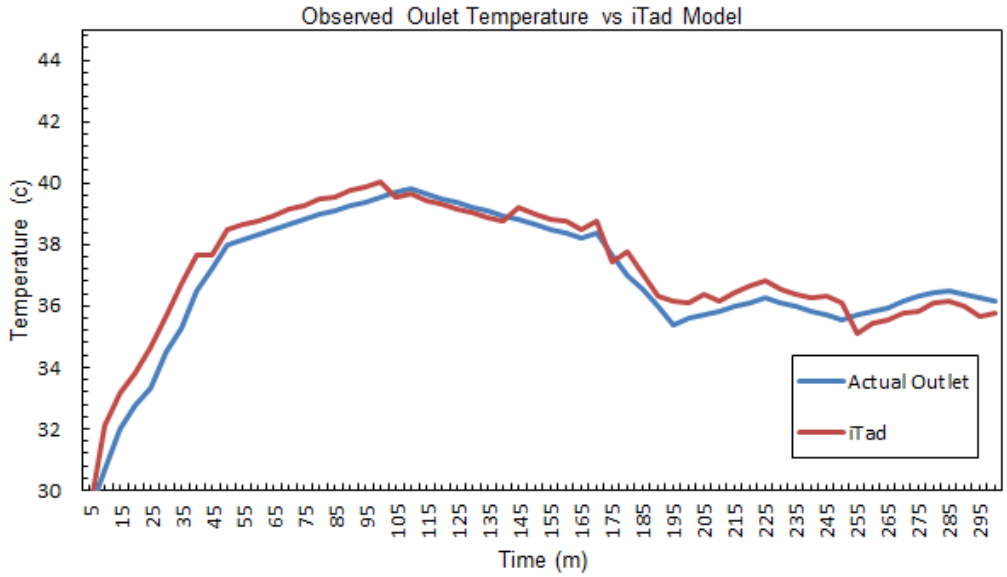
Now that we developed the model and got all the constants of the experiment solved the only thing left to do was to actually run iTad model on a server with random amount of CPU and I/O utilization. In Figure 2.15 you can where we ran a server for 5 hours, with utilization variation. The model had a tendency to over estimate the the temperature especially at the beginning of the process.

We would account for this problem to a poor recirculation constant and the fact that the model isn't time variant. That being said the longer the machine ran the better the results were because this would return back to the state how we tested our machines to find Z value. The variation of the model and actual is a not perfect because at points our model is over 2 degrees off, but the good thing is the trend stays very close to the original so we have a model that is going to err in the safe side of calculations.

2.6.2 MPI

One of the goals of this experiment was to make that this model could work for any kind of data center. So if the data center was to using a C version of MPI, message passing

Figure 2.15: Verification of Model



interface, then iTad should work for them. So we set up a method called "iTad" inside an toy MPI project. iTad would return an outlet temperature, and based on that value it would make a decision about data movement. The method iTad would pull the utilization from the OS and get the other values from system to determine the its output, and all of this happened seamlessly without any noticeable change in performance.

Figure 2.16: Sample MPI Usage

```
if(iTad() < 29)
{
    if(myid != 0)
    {
        MPI_Recv(&number, 1, MPI_INT, myid-1, 0, MPI_COMM_WORLD,MPI_STATUS_IGNORE);
    } else {
        MPI_Recv(&number, 1, MPI_INT, world_size, 0, MPI_COMM_WORLD,MPI_STATUS_IGNORE);
    }
} else {
    while(iTad() > 29){
        sleep(10);
    }
    if(myid != 0)
    {
        MPI_Recv(&number, 1, MPI_INT, myid-1, 0, MPI_COMM_WORLD,MPI_STATUS_IGNORE);
    } else {
        MPI_Recv(&number, 1, MPI_INT, world_size, 0, MPI_COMM_WORLD,MPI_STATUS_IGNORE);
    }
}
```

2.6.3 Hadoop

We also wrote the model for JAVA, this would allow the model to used for other data centers that would use JAVA. This version had to use a runtime shell to get CPU and I/O utilization because the JAVA virtual machine doesn't have direct access to the OS information like C does. This version of the model would work for a JAVA MPI implementation, but for Hadoop each node is not responsible for its own data so we needed to see if our model could be used for an Hadoop data center. In the work of our group member paper [35] shows that the scheduler and heartbeat of Hadoop can be updated to take account of CPU and I/O utilization to make thermal decisions. So implementing our model is as straight forward as replacing there thermal method with out iTad method.

2.7 Experience

2.7.1 Improvements

Our experiment is not beyond criticism, but no criticism that is so heinous that will crush the results of this experiment. The first critic that can be made about this experiment is that the stress command is not pushing the computer enough, especially the I/O commands seems to be inaccurate so the temperatures we are reading are off. This criticism is one that has some true concerns but the fact that when I/O intensive the I/O controller is the hottest spot proves that the I/O is being pushed. Some people may see the fact that our machine is an isolated machine not actually on a rack as concern. This in fact a true issue, while other reports [23] an increase 10 degree different in inlet and outlet with CPU utilization our machine shows a max of 1.7. This concern is invalid because our computer is prototype to monitor trends to prove our model. The next concern is the method of gathering temperature readings, in which we use the surface temperature using IR thermometer. This concern is slightly justified because of the cables and clutter on a machine the reading can be off. But each subsection in Figure 2.11 is actually a sample of points in that block thus giving us a

fairly good representation of the machine, and since most of the materials stay the same and we average out the hotspots out with the cold spots before using any of the data it takes care of small temperature reading issues. The last concern may hold the truest of all the concerns, the way we created a recirculation variable k which without doing much research on how that variable must be used. In our experiment we pick a very small number for k because our setup had alot of room so the outlet temperature would dissipate very easily so a improvement would be to aggregate all the servers and layouts and update the k variable in real time.

2.7.2 Extension

The next step would be to run these similar results in a full blown data center, or at least one with AC unit and multiple servers. Also for the future research we would like to use more sophisticated pieces of hardware for testing temperatures. The follow up research should have a flowing air thermometer for the inlet and outlet temperatures, some kind of heat measurement to keep from extrapolating heat from surface temperatures, and lastly mounts for our sensors and programmatic way to gather these multiple pieces of data without manually scanning them to get more accurate results. Future research should look at neighboring nodes and heat recirculation more closely to see how they will affect the output in a more exact fashion. Equation 5 is a possible enhancement to the model that takes in account of the neighboring nodes.

Another place for future research could be to look at the model as function of time. This could be useful because some application may run short while other run long. Since this model was made to help facilitate quick modeling of thermal pattern of data centers, a model that will run in real time could help in develop even more robust algorithms. As a quick preview equation 6 explains what a time based model could look like.

2.8 Conclusion

Growing evidence show that cooling costs contribute a significant portion of the operational cost of large-scale data centers. Therefore, thermal management techniques aim to reduce energy consumption and cooling costs of data centers. A thermal management mechanism relies on thermal information to make intelligent decisions. Thermal information can be acquired in three ways: 1) using "Computational Fluid Dynamics" simulators (e.g., Flovent is a commercial product), 2) deploying temperature sensors to measure inlet and outlet temperatures of servers, and 3) applying a thermal model to estimate temperatures based on specific workloads.

We advocate that thermal models are a cost-effective and practical approach to providing information on server temperatures to thermal management mechanisms. In this study, we develop a thermal model - iTad - that enables thermal management techniques to quickly make management decisions based on intensive I/O activities. We show that in light of iTad, both CPU and I/O thermal outputs can be extrapolated from radiation heat and convection heat applied to a server. The iTad model helps in improving the energy efficiency of data centers, because thermal information offered by iTad assists dynamic thermal management to reduce the energy consumption in cooling systems in data centers. We validate the accuracy of the iTad model using a server's real-world temperature measurements obtained by an infrared thermometer. Our experimental results suggest that I/O-intensive workloads have significant impacts on temperatures of servers. We demonstrate that thermal management mechanisms can quickly make workload placement decisions based on thermal information facilitated by iTad.

Chapter 3

NAP

3.1 Introduction

Energy efficiency and energy conservation are efforts that becoming important issues for society at whole. The amount of power needed to continue to run our daily lives isn't sustainable. As we scale into a world of more power consumption we will need to find creative ways to fill that power. A good way to get ahead of this problem is to make everything more energy efficient, and today one of the fastest growing energy hogs are data centers. In fact in 2013 data centers contributed to 91 billion kilowatt-hours of electricity, enough electricity to power all the households in New York City twice over [43]. The NRDC states that much of the power used by these data centers is in "under-utilization of data center equipment and the misalignment of incentives, including in the fast growing multi-tenant data center market segment." [43]. Since the data centers are growing and there is opportunity to create more efficient energy aware data centers we simply had to decided to what level of granularity do we want to effect the data centers.

When it comes to energy efficiency of data center there are a wide range of things we could do. It can range from using more power efficient hardware all the way to changing the clock speed of each servers CPU. One area we determined to be a interesting area to research is the data storage at data centers. According to Baseline [41] data is projected to double every year for the foreseeable future. By 2020 we should expect 40,000 exabytes of data. This seems far-fetched until you hear the statistic that nearly 90% of all the data we have stored to today has been created in the last 2 years. Directly proportional to the amount of data being collected and processed is the amount of energy used. Google talked about storing a 1 petabyte of data in 2008 on 48,000 different disks of different sizes [14].

At aggregate the energy consumed by these disks are not insignificant. Since the nature of hard drives are storage more than processing there is opportunity to save energy even if a data center is highly active.

After focusing our attentions to storage and specifically hard disks in data center nodes we narrowed our focus to data centers that used Hadoop. Hadoop is Yahoo's implementation of Google's groundbreaking work in "MapReduce". Google's MapReduce introduced a new shift in big data processing which would take the power of commodity hardware to process unorganized data in two stages the "Map" phase and the "Reduce" phase [15]. Another aspect introduced with "MapReduce" was the Google File System (GFS) [15], which Hadoop implements as Hadoop Distributed File System (HDFS), which creates metadata to manage where data is located within a cluster and creates replicas to ensure more availability. Hadoop version of MapReduce is becoming the go-to big data platform, in fact market forecasts from 2013-2020 predict that Hadoop will grow at the rate of 59.2% year over year because of it popularity [46]. The reason Hadoop is so popular is because its open sourced and it design for the "unsupervised landfill" theory for data, which means Hadoop encourages a lot of data storage, therefore plenty of hard disks to be managed. This popularity and reliance on data makes Hadoop an ideal platform to develop energy saving techniques.

Hadoop was design in terms of performance, so it still have plenty of optimization for energy that can be made [18]. Especially when it comes to disk energy Hadoop relies on JVM and OS management more than its own systems. So in this paper we discuss our novel approach to bringing disk energy management to a more visible position within Hadoop. We call our approach NAP (eNergy Aware hadooP) because we make Hadoop more energy aware by manually checking which disks are in use in Hadoop and putting all inactive nodes to sleep for a short period (ie a nap). We will go over our methodology implementing these techniques, discuss its performance in a variety of scenarios, and how to make it a viable solution for all Hadoop clusters.

3.2 Related Work

3.2.1 Energy Aware Datacenters

Energy Aware Scheduling

There has been many different research for distributed systems line in papers by Kong et al [32] or Sheikh et al [51]. In these paper they use knowledge of things like the hardware, the energy profiles, and power infrastructure to make smarter decisions when it comes to scheduling tasks. They introduce ideas like batching tasks so the data center uses energy more efficiently so they can put the servers in to sleep. There were even ideas that if we know that some green energy is being created we could schedule our task for when there is enough green energy to use. All of this is important to the research we are doing because it shows that when we are energy aware scheduling we may have to wait on resources to gain any savings.

Energy Aware Sensors

Recent study shows that 1.2% of all energy consumption in U.S. is attributed to data centers [33]. The way that study was able to get that information was in small part due to sensors. Also to minimize the effect of the data centers on the national consumption there has been many energy-saving approaches, one that relates to this research is the work of Bieswanger et al. where they deploy sensors to analyze the power consumption instantaneously [9]. While we dont employ active sensors for our algorithm, we do use sensor to monitor our results and recognize that a power feedback loop would help make good energy savings profiles.

Brown Energy Aware

Brown energy is the typical kind over energy we think of then we think of power. It is energy made from none renewable resources and contrast to green energy. Data centers have

to pay large cost for all the energy they take out of the power grid. So many times these data centers will make deals about when to take energy away from the grid so the cost per kilo watt is much cheaper. The real dollar cost brown energy has created work like Govindan et al [19] where they try to set utilization around different hours to conserve cost. This research is applicable to our research because we to are trying to map jobs to lower energy cost, where we are trying to lower the amount of wattage we use and this research simple changes when that energy is used.

3.2.2 Green Hadoop

Hadoop specific energy saving techniques are growing because of the popularity of Hadoop. One of the first papers on working on Hadoop and energy efficiency was Goiri et al [18] called Green Hadoop. This paper looked at Hadoop's batching pattern as place for energy savings. It waits for either cheap brown energy or green energy because running each section of its batch programming. This is applicable to us because it shows that Hadoop batch programming can lend itself to energy savings.

Another greening of Hadoop was introduced in Krish work [34] where they use the fact that Hadoop clusters can and tend to be heterogeneous, so if they make the scheduler aware to which machines have the best energy efficiency and schedule more tasks to them. This research is important to our work because helped highlight the important of scheduler in Hadoop, in that it controls how active a Hadoop node is, which in turns controls how much energy that node uses.

3.2.3 Disk Management

The first thing we need to learn about disk management is that parallelism in disks means more disk will be used per server node [48] [7]. This is important to our research because focuses on I/O being important, and its increase in speed will keep disks from being a

big bottleneck. That doesn't mean that disks bandwidth is unlimited because of parallelism and the wall of I/O reads and writes means we need to creative in our disk management [7].

The key research that helps our paper is the work on dynamic power management [21]. This is the concept of changing the state of disks to a state where it uses less energy. Disk have this ability because there are many different states for disk including active, idle or standby. You can see in Hylick [25] that these states vary greatly in power. This is important to our research because its this concept but a smarter switching algorithm that can maximize the lower power states.

3.3 Motivations

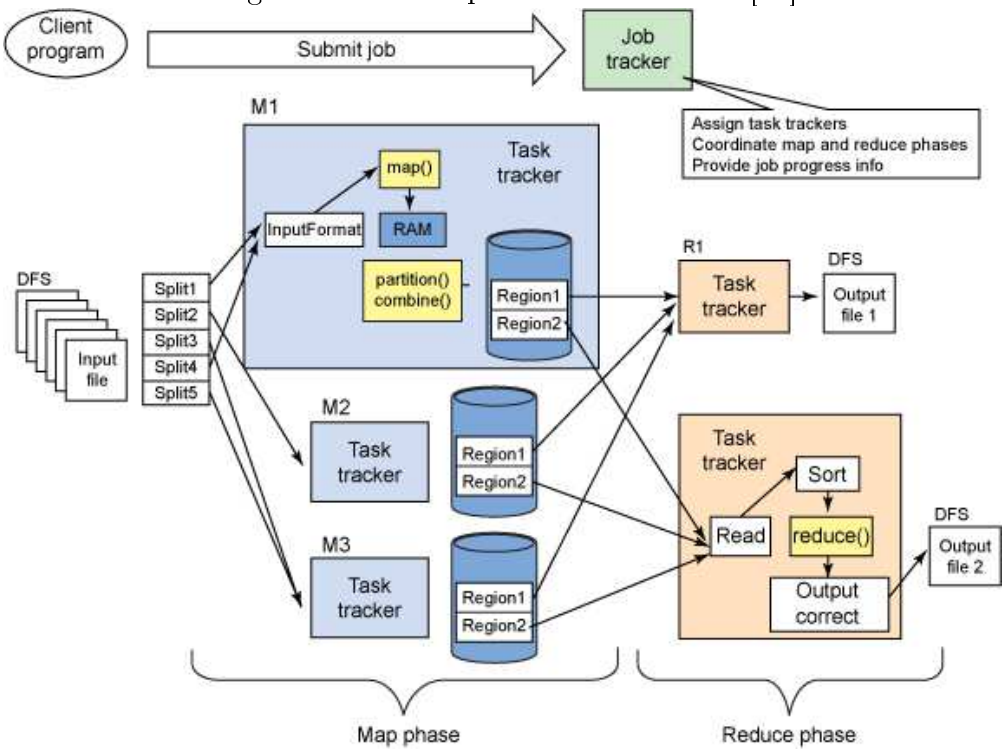
3.3.1 Hadoop Architecture

While looking for ways to save costs and energy in data centers we explored many ideas, but decided to focus on Hadoop because of its popularity but also because its architecture lends itself to some energy savings. As you can see in Figure 3.1 Hadoop has a staged approach, where we first split the files and replicate them throughout the cluster, then mapper agents are released throughout the cluster to run the map phase, then reduce agents are run throughout the system. As you can see there are atleast three stages where Hadoop runs in a different way which could mean that you could develop different policies to get different energy utilization.

Another reason why Hadoops' structure makes it an interesting candidate for energy savings is because, as stated before, it creates replicas of data which means much of the data in the Hadoop cluster will not be accessed unless of some kind of problem. If we are able to isolate that variable we could use it to our advantage to gain energy savings.

Lastly a reason Hadoop is appealing in terms of energy savings is because the implementation itself was not looking to save energy. The whole idea of this kind of distributed architecture in Figure 3.1 is so data can be highly available and quickly processed. Slowing down and actively monitoring energy is not the active goal in Hadoop. That means there

Figure 3.1: Hadoop Data Flow Chart [40]



could be potential energy hogging activities. As we mentioned before Hadoop increasing popularity makes it very costly to not explore every avenue for energy savings.

3.3.2 Disk Consumption

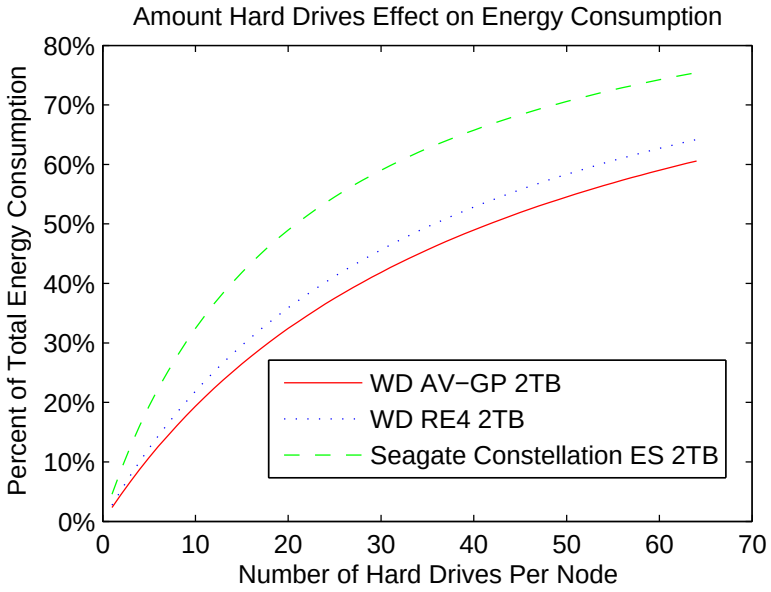
As we went through the related work on Hadoop and disk consumption we needed to see if it was critical for Hadoop disk energy to be managed better. One of the first things we needed to see was what are the general Hadoops' disk hardware configurations. Cloudera manages a very popular Hadoop distribution called CDH. On the Cloudera website they have some recommendation for Hadoop configurations, as you can see in Table 3.1, which show that an Hadoop node is recommended to have between 4 and 24 disks depending on its use [44]. Even though those are the recommend configuration Hadoop disk configuration can be managed by RAID or JBOD which theoretically have limits in the hundreds of disks per node. The price of storage is falling so quickly [31] that Hadoop teams will be temped to add many more drives to increase the functionality of each node and saving costs.

Table 3.1: Recommended Hadoop configurations [44]

Configuration	**CPU**	**# of Disks**	**Size of Disk**
Light Processing	Two hex-core	8	1TB or 2TB
Balanced Compute	Two hex-core	12 âĂŞ 16	1TB or 2TB
Storage Heavy	Two hex-core	16-24	2TB âĂŞ 4TB
Compute Intensive	Two hex-core	4-8	1TB or 2TB

After figuring out how many a disks Hadoop cluster will likely be in each single server node we started to look towards what effect does the disks has on the server as an whole. To do this we simulated the amount of extra energy that will be used per added disk. As you can see if Figure 3.2 depending on the type of disk on the low end disks contribute to about 8% - 15% of total energy of a node, but on the recommended high end the disk contribute between 25% - 50% of the total energy. Figure 3.2 clearly shows that at just the recommended configuration the disk energy is not negligible, and if a data center was inclined to go over the recommendation it could be even greater. Therefore the managing Hadoops' disk energy throughput is a not a meaningless task. With the billions of dollars energy costs that data center have now every percentage of savings we achieve will cut costs by millions.

Figure 3.2: Showing the importance of hard drives on energy

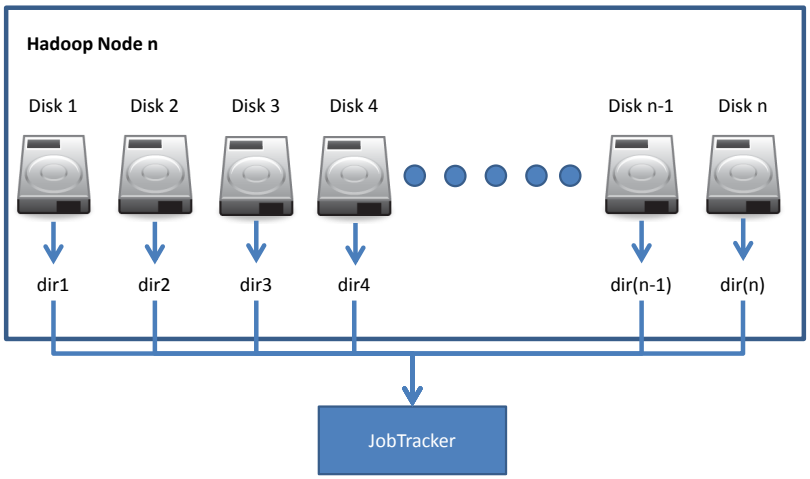


3.3.3 Hadoop Disk Management

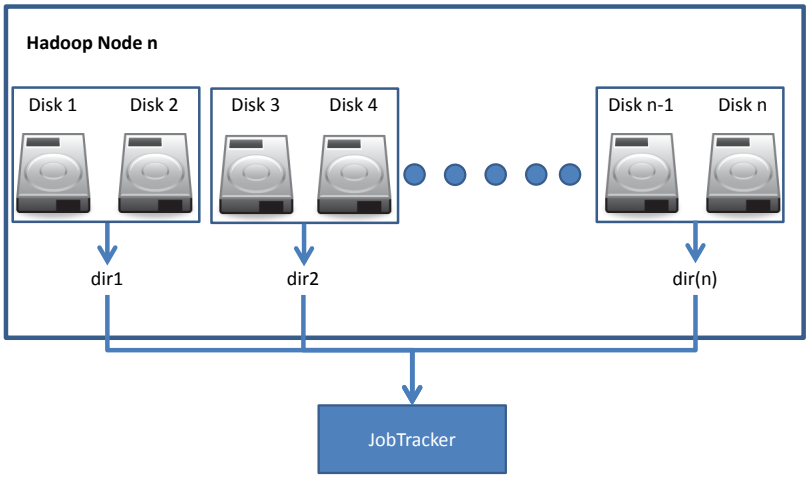
After establishing that disk management can be a viable mean to energy savings we needed to see if Hadoops management would lend itself to some kinds of savings. The first trade-off we become aware of is the cost benefits of adding more disk vs. the speed of processing. Kambatla et al [28] clearly show that there is a direct increase in performance as more disk drives are added. This is because the more disk in a server node the more data can be processed at once because of parallelism in disk reads. The thing that Kambatla et al doesn't address what is the limit to this increase is. The speed increases he sees will decline because for every added disk for two reasons: I/O interconnects, and processing power. To use disks in parallel your CPU needs to be handle the flow of data, while CPU's are very powerful and generally its the I/O bus that is limitation, as more disk are added in parallel a CPU bus will not get bogged down especially for very large data sets that Hadoop may see. The more important factor to the decline in data parallelism is the I/O interconnects. Many I/O interconnects are not rated to handle 20+ hard drives, so on data intensive Hadoop configurations that could be cause a slow down.

Another phenomenon discovered in Kambatla et al [28] work is the fact that splitting solid state drives (SSDs) into multiple directories increased speed dramatically. That was because of the nature of SSDs they don't need to seek which means if the Hadoop has more ways to access the data in parallel the job would get better performance. The same although can not be said about standard hard disk drives (HDDs). HDDs need to seek to retrieve data so random access would hurt performance. Using that logic and the fact that there is a limit to how many disks can be read in parallel we propose that HDDs could be grouped together as a single directory to limit strain on I/O and CPU bus but maximizing the amount of data per server. In Figure 3.4(a) you see a general implementation of Hadoop where disks are all a single directory that the JobTracker will call, but we propose Figure 3.4(b) because it will allow more disks with less strain, but also may give us opportunity for energy savings.

Figure 3.3: Hadoop Disk Configurations



(a) Every disk a separate directory to called



(b) Disks are grouped as single directories

This method is adaptive from the idea of Intra-Disk Parallelism [48] where they were able to place a larger disk array with a smaller and achieve similar performance.

3.3.4 Dynamic Disk Power

The main motivation of looking into Hadoop disks for a source of energy savings is because of the research that has ready been done with hard drives. All modern disk drives have modes that carry a certain wattage. In Hylick et al [25] the discuss looking at many drives and figuring out how the disk energy changes as the state changes.

Figure 3.4: Disk State Energy Utilization [25]

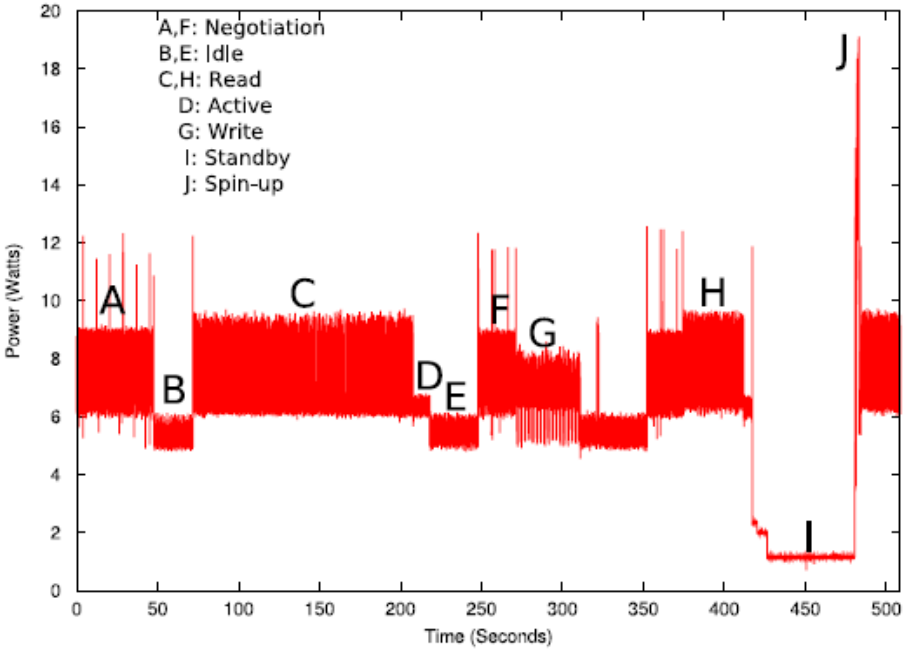


Table 3.2: Server Specification

Server Type	Acer Veriton M661
Number of Disks installed	4
Disk Types	Western,Digital WD1200JB
RAM Upgraded	4gb

Figure 3.4 clearly shows a big change of energy utilization based on state. Active, idle, and standby are all disk states that can be manipulated manually. Generally disks are quickly sent to the idle state when its not being accessed because there is low turn up cost, there would be far greater savings if a disk could go to sleep more than idle.

3.4 Methods and Materials

3.4.1 Equipment and Software

This section we will discuss all the things we did to set the experiments we discuss later in this section. We used 3 server that connected by a private hub with gigabyte rated wires. Each server has default specification of a Acer Veriton M661 with the customization seen in Table 3.2.

The drives above are connected as JBOD (Just a Bunch of Disks) which means the only data redundancy will be done through Hadoop. In Hadoop we declared a folder in each drive for inputs and outputs so all drives would be used connected with a symbolic link rather than doing a separate folders per drive. In our algorithms we talk about "disk groups" the number of symbolically link folders are considered a disk group, for this research we left the group size as 1 disk. The type of Hadoop installed is the distribution known as CDH, or Cloudera Hadoop, this distribution installs many things like MapReduce, Hbase, Hive and more. We deal exclusively with MapReduce in these experiments, and Cloudera helped to gather results for our conclusions. Unless specified other wise we used famous open sourced books text with randomly generated files sizes ranging from 100 - 250mb, with the default block size of 128mb. For any experiment where the program we used is not specified we used the default word count program that comes with Hadoop.

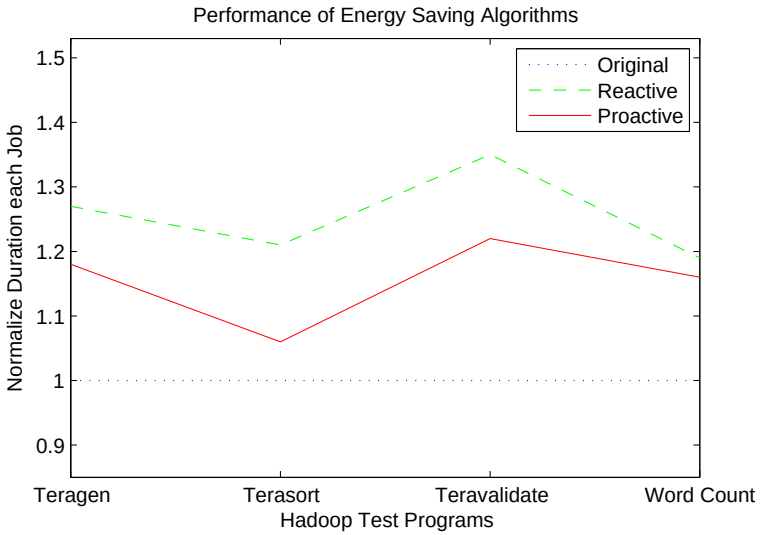
We connected the servers in to a power strip and plug that power strip into a power meter which we used for these results. The monitors and bridges were plugged into a different plug outlet. The power meter would record kWh for how long the power meter was turned on, so for our experiments we would turn on the meter when we hit start on our programs. This way we didn't get the energy needed for uploading the input and getting the work started. We recorded the kWh averaged over the number of machines and the duration and high and lows wattage for our calculations.

To legitimize our data we ran a minimum of 35 runs for each variable and each algorithm and used the averages in the data in the following sections. This choice was to make sure any outlying data would be obvious. While not in the exact scope of this research it is important to note that despite an aggressive disk management we use the slowdown of our algorithms is very low. Figure 3.6 has the original, non tampered with, Hadoop algorithm normalized as 1 and with the other algorithms Figure 3.6 shows that with some test programs that the worst time increase happens at teravailidate with a 1.3x increase in time. The program we use for most of experiments "Word Count" is right in the middle of time increase, and for

Figure 3.5: Powermetere

most map reduce environments this loss of performance might be worth the energy savings we demonstrate later in this paper.

Figure 3.6: Effect on speed



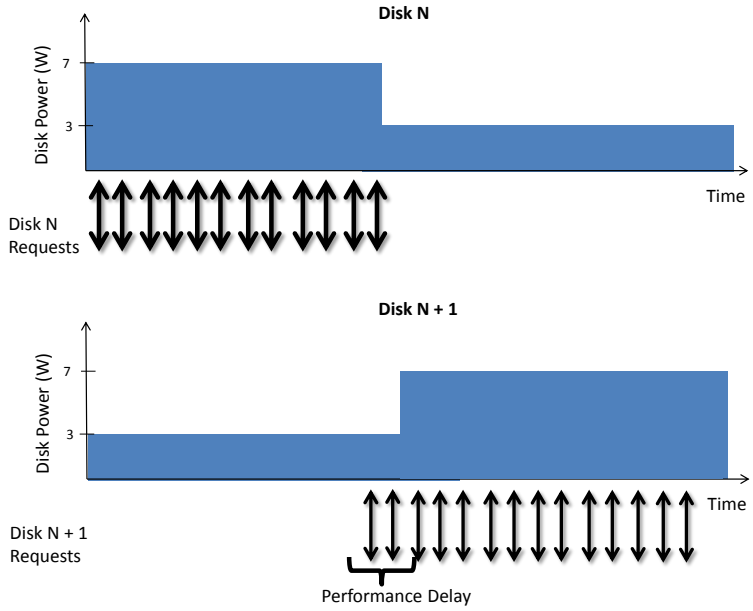
3.4.2 Algorithm Design

From the related work you can see that there has been many energy saving techniques used in data centers. In terms of hard disk one of the most common techniques to save power is switching from active to idle or standby modes. Remembering from Figure 3.4 such techniques can be quite advantageous in saving energy.

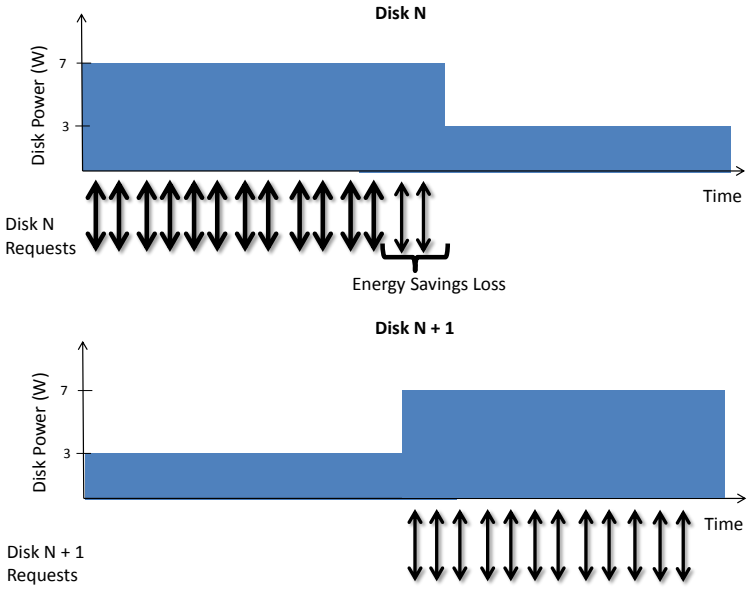
The thing about such power schemes are they are reactive measure. Meaning that once the conditions are met then it takes a specific actions. The benefits of such reactionary measures are that it easy to set up and if the data it reacts to is perfectly tuned then you can a max energy savings. The problem a reactive power scheme has if it mistakenly puts a disk to sleep it will have a big effect on energy savings, and if it does it too much it could even cause more use of energy. The other problem with this kind of reactive approach to disk management is there slow down time that is noticeable, because the algorithm only knows to turn off a disk and waits for a system to wake the disk up like in Figure 3.7(a).

We introduce a second method for disk management. In this method we use Hadoop information to tell us which disks will be active and when the next disk will be active. This way we are being being predictive in the way we manage the disks.It is easy to set thresholds and tweak those thresholds to set a point where power should be switched off. This tweaking and pinpointing of disk transition could allow data centers to create very accurate times to spin down and spin up disks. On the other hand a problem that occurs is that you need to have a lot of information about previous runs of an applications to get a very exact time to turn off a disk. If the system does not have enough data on the process we can either have disks that are on too long or disks that have to keep transitioning back and forth. As you can see in Figure 3.7(b) the advantage of this system we wont have the performance drag that the reactive has. There could be a energy savings loss as compared to a reactive strategy but that is only if you assume a reactive strategy never mistakenly puts an important disk to sleep.

We implement an Hadoop version of both algorithms to show that there is benefits over the a non-controlled Hadoop node.



(a) Reactive Disk Management



(b) Predictive Disk Management

3.5 Experiments and Results

Our experimentation is based on real life clusters connected whose energy usage is monitored with a power meter. We use the measurements to determine the average power in watts during each run of our experiment. In cases where we represent power took the energy readings and divided it but how long each run took and used the experiments time divided by the control experiments time to normalize our numbers. The reason for this is so we can have a clear understanding what the cost of each experiment without the length of execution to be a factor. This real life data gives alot of insight that simple forecasting can not give us because it shows us how real life machines react where things are not perfect and where our assumptions are justified them or misguided. Later on we will discuss how to expand this small scale energy models for larger clusters .

The next few sections discuss all the experiments we ran and the reason these choices were made. We wanted to make sure to test out power saving scheme as many ways as possible so we could get an accurate idea of how well our approaches really worked.

3.5.1 Block Size

One of Hadoop advantages is its file system Hadoop File System (HDFS). What good about this file system, it takes files and replicates them over the cluster and process each files as little chucks know as blocks. Block processing is good for parallel computing because you are able to load data quicker and finish chucks out of order so as the processing power become available it can be assigned a task. If the tasks get too small a new problem emerges of not enough power to run them all and the schedule has to be wait and do more work scheduling future tasks. The trick to performance is to find a block size that gives the best trade off between processing and availability for the the type of machines and data you are processing. [49]

Since we know how the block size effects the performance is likely to assume that the energy used might changed based on block size. Cloudera Hadoop(CDH), the distribution

of Hadoop we tested on, recommended block size is 128mb. If you look to Figure 3.8(a) you can see that 128mb has a clear energy savings as compared to 32mb. Lower than 128mb block size is not recommended because a node will process smaller blocks very quickly so their will be alot of competition for open slots thus creating overhead that slows down performance. That overhead and CPU utilization that slows down the performance also contributes to the higher energy consumption. Another factor that makes block sizes smaller than recommended more of energy hogs is the fact that there are more disk reads. The disk reads effect our algorithm energy efficiency rather than the original because original Hadoop has very few disk exceptions. The multiple disk reads effect the reactive algorithm negatively because if it calls on a sleeping disk it may try to look for other copies waking up more disk than needed. The proactive model suffers also because it gets harder to predict when a drive needs to be spun up so it either wakes too early or wakes late and we add the penalty that we got from the reactive process. Lastly you can see as the the block size increase the energy savings also get larger, but with minor gains. This gains can be explained by more predictable memory accesses and fewer times blocks have wake up disks for changing. The 128mb seems to be a good trade off in speed and energy savings, making it a very good recommended block size.

3.5.2 File Size

Hadoop file system although very robust it has a weakness in regards to file size and data locality. [62] The problem being that the way Hadoop creates blocks and store files on a servers is not optimized for heterogeneous servers. For instance if some machines were to process tasks faster than other then there will be extra cycles looking for blocks to process. This would be a bigger problem with larger files uploaded because if Hadoop has to do some correlation between blocks there is a bigger chance the blocks are not local. So in work by Xie [62] they explain different patterns of data locality to decrease the scheduler in Hadoop to search for appropriate blocks.

Our theory in designing this experiment was using this idea of data locality in Hadoop and how that might effect energy consumption. We assumed any loss in performance shown with would impact the energy efficiency, and we though perhaps our implemented algorithms would be hurt energy efficiency more aggressively because if the scheduler has to do more search then it would have to wake up disks more aggressively.

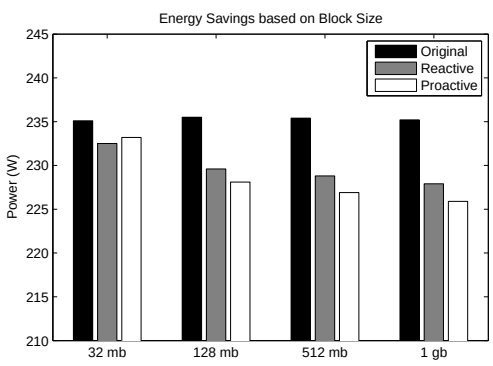
Figure 3.8(b) shows our results from this experiment. We were able to change the input size of files, but keeping the block size the same. Aside from the change we see in typical reruns we are presented with data that suggests that file size has no effect on the energy efficiency. One reason is that the research that we design this experiment around explains the need for heterogeneous cluster and ours was closer to homogeneous, but that being said the discovery of energy efficiency of input size on homogeneous Hadoop clusters still needed to be explored. Lastly we may have over estimated the cost of a more active scheduler on energy efficiency because when the file sizes are bigger the utilization will be high from processing data and when the energy efficiency is low the utilization will remain high for scheduler logic. Combining theses concepts its easy to understand why the input file size didn't effect energy efficiency of the machine processing data.

3.5.3 Map vs Reduce Routine

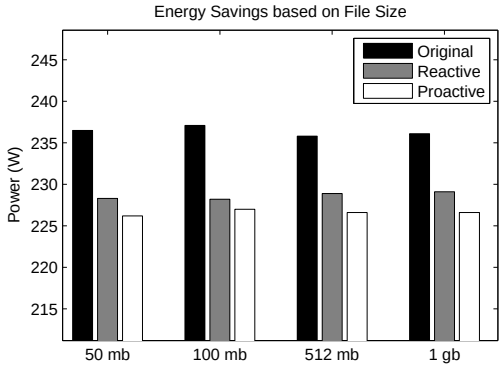
As we talked about earlier Hadoop is based on Google seminal paper on MapReduce, which discusses a new parallel programming paradigm where tasks are either map or reduce routines. Map routine is generally where the data is organized locally and lightly processed into intermediate data, and reduce routine is where this intermediate results are compiled in an order and processed again for its final results. We wanted to design experiments and see how each phase is effected by our proposed energy saving techniques.

While map and reduce routine are explicitly called in the way we program files, and there are "mappers" and "reducers" elements they do not run strictly separately. Generally 100% of map tasks run before any reduce task run, but there are cases where they may

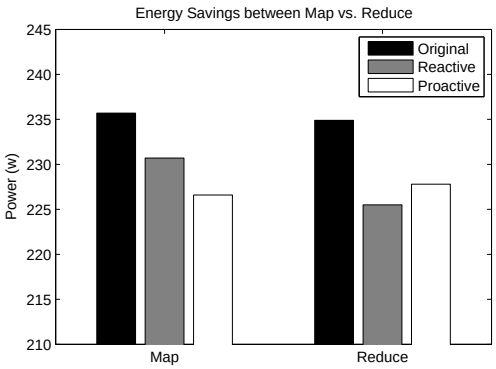
Figure 3.7: Results how file size effects the energy savings



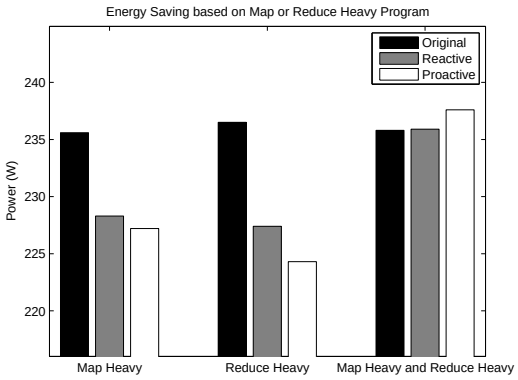
(a) We used a variety of block sizes to see how our energy saving techniques would match up



(b) We used a variety of file sizes to see how our energy saving techniques would match up



(c) We compare the map phase vs reduce phase for word count



(d) We compare different Hadoop program design patterns

run simultaneously. The figure 3.8(c) took a weighted percentage of mappers vs reducers to determine the energy cost on display.

On to the actual results of this experiment you can see that both the map and reduce section benefited from our algorithms. The reactive process was better suited in the reduce phase rather than map phase. The reason being the program we used to test map vs. reduce, creates smaller output than input which means fewer disk will have to be woken up total so there is less waiting for disks to wake up. To that same point the predictive model did worse than reactive one in the reduce because it is harder to predict which drives will be turned and in which order during reduce because the amount of drives needed are not 100%. So a consequence the predictive model constantly made the wrong call to the drives.

3.5.4 Map Heavy vs Reduce Heavy Programs

In the previous section we discussed how the energy efficiency of the map process compared to the energy efficiency of the reduce process. Although that experiment gives us insight to how each process could save energy, those two process can differ greatly in the amount of time either process is executed, how much data it might use, or even what utilization it uses. This is a main reason why when creating benchmarks for Hadoop there are many considerations that people use. The Intel China Software Center research team discusses a variety of benching marking techniques covering small benchmarks and large real life application benchmarks. [24] The part we found most interesting in their research was the workload data access patterns they discuss because of the way it directly effects our energy saving algorithm design.

As China Software Center research team showed Hadoop programs can be categorized in many ways but the way that makes sense in terms of experimentation is the size of its input and output data. The amount of data that we given to process at the start of an execution directly effects the amount of time the mapper agents will be running in Hadoop. Mapper agents take the input chunks and create the intermediate data that will later be processed by reducer agents. Then those same reducer agents directly effect how big output of an program will be. So we have given the terminology of a program that takes in large data set then return a small data set a "Map-Heavy program", if the program rather takes in a small input then return a large data set we called that a "Reduce-Heavy program", and lastly if we take in a large data set then return a large data set we called that "Map and Reduce Heavy program".

Table 3.3: Hadoop Application Profiles

	Input Size	**Output Size**	**Example**
Map Heavy	Large	Small	Word Count
Reduce Heavy	Small	Large	Teragen
Map and Reduce Heavy	Large	Large	Terasort

Table 3.3 gives a quick look at the information we explain above, and we list out application that for each group. Strictly speaking word count doesn't always guarantee a "small" output we knew our data had many repeat words which gave a relatively small output. Teragen can create massive amounts of data with simple parameter change so its a very good test for reduce heavy outputs. Lastly all sorts are good examples of map and reduce heavy programs because the size of input will the same size as the output.

The results of testing each of these program characteristics are represented in Figure 3.8(d). The word count or map heavy results are the results we are most familiar with from other experiments because it has been the benchmark test so far. Word count has good predictability and long processing time in both map and reduce phase which allows for energy savings.

The reduce heavy results are interesting when you compare it to the results from Figure 3.8(c). The reduce heavy application actually had an average energy consumption lower than the map heavy consumption, but the energy savings of map phases was shown to but higher than the reduce phase. The way to reconcile these results are to look at the reasoning why map was more energy efficient compare to reduce in Figure 3.8(c). Word count, the bench mark used in Figure 3.8(c), while map heavy it has a much larger reduce phase than teragen has map phase. This explains why the overall energy consumption is lower in Figure 3.8(d) for reduce heavy applications, but not why the trends for reactive and proactive algorithms don't match the reduce phase trends. Teragen map phase is very short so uses virtually no energy compared to its reduce phase so no real savings will happen in the map phase. The reduce phase for teragen is much more systematic and predictable than word count, which means the disk transitions will be much more reliable in the predictive model; thus explaining the trends accurately.

Lastly comes the map and reduce heavy applications, whose results are the most disheartening but not unpredictable. As you can see in Figure 3.8(d) the map and reduce heavy application, terasort, had no energy savings, in fact in general case you can see we

actually used additional energy. This can be explained from the fact that if a resource is being used very heavily its very difficult to get any savings. Sort as a concept requires looking at everything over and over so disks will remain active as will scheduling resources, and this is exacerbated by the fact there will be no compression between input and output. The mapping portion of this experiment would see the similar gains we'd expect from mapping in any other application, with caveat that sort requires a few more file checks than the other algorithms. The reduce phase in a sorting problems requires alot of random accesses, this unpredictability caused an increase in disks to spin up and use energy, anecdotally this unpredictability causes disk failures and scheduling problems causing terasort to crash very often. This over head for failures adds greatly to overall energy consumption. The proactive proved to be more aggressive than the reactive algorithm, spinning up and down disk very often, and like in the case of reduce phase in Figure 3.8(c) it created a worse energy effectiveness.

3.6 Evaluation

In the Experiments and Results sections we discuss why our experiments and results are valid, but we don't discuss the actual savings demonstrated. On the average of all our experimental data we find that per server we save between 8 - 10 watts or about 4% of the total energy utilization or about 20% of the max disk utilization. These results may seem small but Hadoop is a parallel highly distributed framework that can be implemented on thousands of server, which could make this research very profitable. This section we will cover what the theoretical savings we could see and how it compares to our experimental data.

3.6.1 Model

We have already demonstrated in Figure 3.2 that as the number of disks increase the percentage of the total energy disks used can be significant. No algorithm or process will ever

Table 3.4: Model Variables

Variables	Description
E_{total}	Energy of the entire system
E_{disks}	Energy from all the disks
E_{disk}	Energy from single disk
$E_{transition_i}$	The energy from a specific transistion for a specific disk
$E_{standby_i}$	The energy of a standby disk for a specific disk
E_{idle_i}	The energy of a idle disk for a specific disk
E_{active_i}	The energy of a active disk for a specific disk
D	Number of disks
N	Size of disk group

reduce that energy used by disk down to zero for Hadoop, so to get a clear understanding on the max energy we could theoretical save we developed a model. Based on the work of Manzanares prefetching for parallel I/O buffer disks [39] but we make the simplification where he uses time disk are active and standby based on the request per job we assume all requests are the same in how long they take to be accomplished. These are assumptions are simplifications but they will over estimate the max which is good for a maximum.

$$E_{total} = E_{server} + E_{disks} \tag{3.1}$$

In Equation 3.1 we separate the rest of the server components from the disks. So all need to do is calculated how much energy the disks will use.

$$E_{disks} = \sum_{i=1}^{D/N} \frac{N}{D} E_{group} \underset{i*}{\frac{D}{N}} + \sum_{i=1}^{D} \frac{D-N}{D} E_{standby_i} + E_{transitions_i} \tag{3.2}$$

The next thing we need to figure out is how the disk energy is going to be determined. Equation 3.2 takes the number of active disks and get its weighted average to total number of disks. Then we get energy cost from each transition, the number of transitions is based on how aggressive our algorithm is so its considered a static variable for each disk. Last we

the weighted average of standby disk to total energy where each disk standby contributes an equal amount. The theory behind this calculation is that in the group of disks that are in parallel whatever that is not active will in standby.

$$E_{group_n} = \sum_{i=n}^{N+n} E_{active_i} \tag{3.3}$$

To fully illustrate the point about how the amount of active disks are inversely proportional Equation 3.3 shows that max energy an active group of disk can have is when group size is equal to number of disks, but the larger the group size is in Equation 3.2 the small the standby energy becomes. Since active energy consumption is much greater than standby energy consumption in disks then to achieve maximum savings we want to make group size as small as possible.

3.6.2 Usage

After creating model and doing our experiments we can now compare how our algorithm stack up to the theoretically limits. To find the max savings possible for experiments we need to think of the most ideal case for Hadoop using our algorithms. That would be a single drive active at a time while the others are in standby, and once one drives data is perfectly processed we can move to the next drive and turn the previous drive to standby mode. In this scenario our group size is 1 and the number of transition per disk is 2 using the information from Table 3.5 we are able to the max savings from Equation 3.2 we can find the max energy savings to be 12 w.

Table 3.5: Western Digital WD1200JB consumption model

Active Power Consumption	8 w
Idle Power Consumption	7.25 w
Standby Power Consumption	1.2 w

Our experimental data gave up average of 7.6 w savings for our reactive algorithm and about an average of 9.1 w of savings for our proactive algorithm. These results show that

our implementation is about 70% of full savings. The difference can be marked by that data isn't perfectly partitioned and there are more state transition for the reactive algorithm or in the case of the predictive algorithm it could be guessing to turn on disks too early loses some benefit. Also because we had a disk grouping of 1 the number of active disks was 1 and the results were very slow cause empty cycles to increase energy used. All that being said at the scale that we ran our experiments the savings under 4% of our total sever, but as the a number of disk increases the results could be more interesting.

Figure 3.8: The max percentage of disk consumption we could save

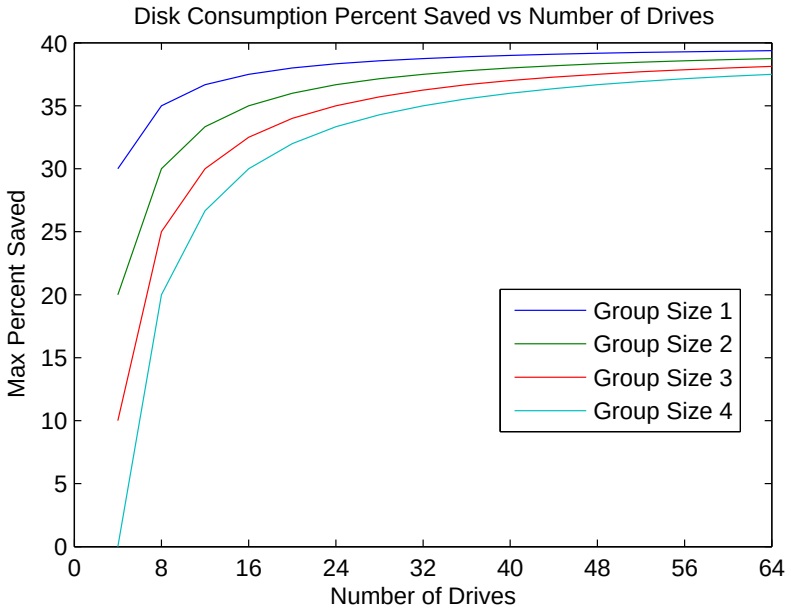


Figure 3.8 shows what is the possible amount of energy that could be saved at different number of disk. it also shows with different groups sizes so our results could be faster. If you combine the results from Figure 3.8 and Figure 3.2 with the knowledge that our current implementation is about 70% of ideal, at 64 disks we are looking at 15% savings across the board as a conservative estimate. Which is a significant result especially talking in terms of dollars.

3.7 Future Work

3.7.1 Reactive and Proactive algorithms

We developed and implemented two separate algorithms to save energy consumed by disks. These algorithms had specific problems for example the reactive model would get caught making lots of disk state changes that may not be needed, this slows down performance, causes scheduler to work harder, and cause an increase energy consumption. The predictive algorithm calculated when to spin-up and spin-down disk more intelligently which took away a lot of lost speed up an energy loss, but in cases where it had to spin up disks in a way that was not predictable it failed with great loss in energy savings and time. Future research could devoted into merging the strengths and weaknesses of these two algorithms.

The reactive algorithm seem to work better for reduce phase while predictive worked better in a map phase so an hybrid model could switch between these two algorithm depending on the mapper or reducer agent that is called. You could add a predictive agent into current reactive agents that could more knowledgeably spin down to standby for previous disk so it would have to wake disk less often.

3.7.2 Savings model

Our current modal for total savings is a little simplistic it makes a lot of assumption about run time that are not true for all Hadoop use cases. Future research can invest efforts to finding how time effect the weighted average of each active or standby disk. Also the transitions model is very broad, we could develop a model based of specific parts of Hadoop programs, like reducer agent and mapper agent deployed or input size, to determine how many transition we are to expect to more accurately present the savings we can achieve.

3.7.3 Heterogeneous disk arrangement

This research dealt the identical disk with identical machines with group sizes that were also identical. This is generally how servers are purchased in industry so the results are useful for most accounts. Although we could explore the idea of grouping disks for energy purposes in more depth by creating variation in the types of disk (HDD, SSD), the sizes of those disk, the groups we put the disks and the performance of the machines.

In the case of SSD vs HDD, solid state is much faster and less energy consuming compared to hard disk drives. So if were to create disk groups that could use some SSDs for processing in the map and HDD for storing reduce phase and control the state transitions we could create a very cost affordable green solution. The same goes for the other variables in the experiment, we could see if we could leverage their unique properties for more savings.

3.8 Conclusion

The concern with the use of energy is not just an environmental issue that but a huge factor in the economics of running a data center. While its energy is getting greener and cheaper all the time the best solution to save the money and world is to create more energy efficient data centers. Managing power supplies and computing utilizations are the big factors in the energy consumed by data centers, but the disks in data centers are not negligible. We showed that the trend towards bigger data requires a trend towards bigger storage, and the most affordable storage tend to be the least efficient in terms of energy. We discussed managing those inefficient disks in two different ways, an approach that reacts to which disks are in use and keep the other disk to a lower power state or a second model that predicts when disks are needed and manages when those power state changes should happen. We used Hadoop as an industrial standard with access pattern that could favorable shown to gain energy savings. We were able to show that these models save energy for servers in many cases, but not all cases will benefit from these models. This is to be expected because the conditions in which disk energy savings can happen are not great for very active disks

in a very active server. Even with this imperfect solution the gains are not minor and can easily save data centers thousands. This kind of though into energy savings can be expanded throughout different parts of data centers and save even more.

Chapter 4

Sparke

4.1 Introduction

AMPLab at UC Berkeley developed a new method to process big data, called Spark, to compete directly with Hadoop implementation of MapReduce. The process they proposed can be conceptualized as a micro batch processing compared to Hadoop's macro batch processing. With the shorter batches Spark is able to move all calculations onto internal memory, it does that by creating Resilient Distributed Dataset (RDD). These RDDs are immutable distributed collection of records that live in the volatile memory that can be called iteratively, so instead of mapping everything creating intermediate data on disks then reducing that data from the disks Spark takes a small batch creates intermediate data in-memory then takes the next batch of data and updates its data. [63]

There has been plenty of research looking at the speed up that Spark provides over most general map reduce programs, they mostly focus on performance in terms of time or memory activity but the thermal and energy efficiency of Spark has been overlooked. This oversight is because the objective of Spark has always been speed as seen through their all their documentation and advertisement [63], but now that it is a a top-level apache project the uptake of Spark is increasing and these factors will become more important.

Our research outlines the current thermal and energy profile of Spark and compare them to Hadoop's Map Reduce. We create a package of algorithms called Sparke that will manage disks power states in Spark. We then do an array of benchmarks on Sparke using real world data, we develop models specifically for Sparke, and used models we have already developed like iTad with Sparke.

4.2 Related Work

4.2.1 Spark Benchmarks

Spark is maturing as platform but the industry standard of benchmarks have yet to be determined. Almost all research you read about Spark will cite that is can get 100 times the performance of Hadoop [54][52][63][13] but that is a specific scenario where Spark is optimized and not the truth across the board. There has been researching look at multiple other ways to benchmark Spark from memory consumption [20] to CPU utilization and throughput [27]. This idea of benchmarking Spark is important for the next iteration of research of Spark to continue. Where Kaewkasi [27] starts to benchmark the energy consumption of Spark we will delve a deeper and with more focus.

4.2.2 Memory Consumption

Spark will always be compared to Hadoop, this research will also do that, and when it comes to speed Spark always wins. The trade offs it makes is in memory consumption. [20] A lot of what makes Spark appealing to research is how it handles memory, for one it tries to avoid disks as much as possible but the nature of RDD is that it has to create alot of intermediate data, in most cases more than Hadoop. [20] This work is seminal to the start of our Spark research because it discusses how Spark performs at different sizes of inputs as compared to Hadoop. Large data sets need to replacement policy, and which could be another place to do some disk management. The work by Shi and team on Mammoth also show that for iterative processes Spark is very fast and memory handling is very straight forward but in other cases it can be chaotic full of merges. [52] It will be useful to know which applications are straight forward versus chaotic when looking for places to save energy.

4.2.3 Mobile Consumption

One of the few places you hear about Spark and power consumption is in the work by Kai Sun called "Energy Efficient Mobile Cloud System for Deep Learning" called M2C, where they implement a deep learning tool with Spark because of its speed to process data for near real time results. [54] They talk about how Spark finishing faster allows them to not waste as much battery. Spark is only used here to speed up the backend [45] to process things very quickly saves them battery on receiver side, our research will deal with the reverse how to save Spark energy and generally faster something process the more energy it uses.

4.2.4 Hadoop Power Management

Hadoop is a more mature platform than Spark so there has already been some good background into the world energy efficiency for Hadoop. Our own work with NAP was focused on Hadoop disk management, which should provide a good starting point for how to adequately test Spark. Much research has gone into profiling different jobs in Hadoop [38] that can provide good background on how to approach Spark. The Hadoop profiles along with our iTad thermal model can help could help get an idea of what kind of thermal savings Hadoop could get and see if any of that can be inherited by Spark.

4.3 Motivation

Almost all Spark literature makes mention of Hadoop. Hadoop is now the big name in big data, but it suffer from the problem that it has to uses so much of its processing. Spark, on the hand, uses the faster cache memory to manipulate process the same HDFS filed data in a far shorter period of time. Our research with NAP and parts of iTad dealt with Hadoop and how we can save lower it energy/thermal footprint and we want to extend this research for Spark.

Spark has an interesting concept from the sense of saving energy. While doing computations on the main memory the disk are left alone, and if the disk are left at full power then

there is an opportunity to save some energy. Even though Spark tries to do all computation in memory but for large datasets it usually uses more reads and writes than Hadoop, but that being said it manages its bandwidth better so it I/O utilization stays low [26] and fewer bursts are good for energy savings [61]. Fewer busts also mean more predictability which we want to leverage to get some energy savings.

4.4 Spark-e

Spark while is very optimized for speed its energy output has been neglected. In this section we will introduce the our designs for algorithms to 1) lower energy consumption 2) lower outlet temperature that will lower air conditioning cost that will ultimately lead to lowering energy consumption. These algorithms we call "Sparke", 'e' for energy aware. After explaining our algorithms we will use them in actual Spark clusters to help to profile the thermal and energy output of spark.

4.4.1 Algorithms

The following sections are the algorithm that we used for our experiments. We designed these algorithms to be effective to lower the energy consumption and thermal output of a Spark cluster. The concept behind these designs are that we want disk to be used less which would lead to lower out put in thermal output and energy output.

Reactive Algorithm

Spark uses disks as little as possible so we want to leverage the fact that disks don't have to be active at all times for the calculations to happen. We repurposed the ideas we developed for Hadoop here in Spark creating disk groups. The size of the group will remain active at all times while the rest of the disks can go to sleep. Theory behind this is that once all the data from these disks are read then we will spin up other disks and the current disks group to sleep. So instead of all the disks being active only the disks in use are active, like

in Figure 3.7(a) in the NAP algorithms. There are added costs like spinning a disk up that add energy and time costs. For these experiments we are simply trying to discuss Sparks the energy impact so we disregard the time costs. If a disk that is asleep has to be woken up then the whole group will be woken up and will not be turned to standby until another group is turned to active, so in this case there are high costs to mistakenly turning a disk to standby. In terms of thermal output, the most more disks we can put to standby the lower our outlet temperature, but Spark is known for its high CPU utilization, so the number of disks needs to be massive to see gains.

Proactive Algorithm

As we discussed in the reactive algorithm there is a large cost to turning a disk to sleep if its not finished with data. To prevent this cost we have to known for sure when a disk will be called and when it can go to sleep. So in that case we implement a proactive algorithm that will wake a disk group up early and keep a old disk group awake until it crosses a threshold, like in Figure 3.7(b) in the NAP algorithms. If data is perfectly tasked by disk this algorithm will be inefficient in saving energy, but otherwise can be very useful. In terms of thermal footprint this will be equivalent to the reactive algorithm.

Empty Algorithm

This algorithm uses a unique feature of Spark that many other services don't have, the fact that all calculations happen in memory. This means when processing happens then no drive is actually needs to be active until the memory calls it. So in a perfect case while the other two algorithms have to have atleast one drive active this algorithm could have zero. The trade off that it has to transition the disk state very often, but if the memory is large enough this could yield enough time for real savings. In terms of thermal energy this algorithm could be better than the other algorithms and long as the disk are needed to transition so often. Also to facilitate this we turned on the "fair" job scheduler which sends

jobs fairly throughout Spark cluster so it lowers the chances the same disk will be called after the current job ends. [63]

4.5 Setup

Just as we did for NAP we used 3 server that connected by a private hub with gigabyte rated wires. Each server has default specification of a Acer Veriton M661 with the customization seen in Table 3.2.

We model this setup like we did in NAP, the drives above are connected as JBOD (Just a Bunch of Disks) with folders in each drive for inputs and outputs, and group size is equal to how many disks have to be active in reactive or proactive algorithm. Cloudera Hadoop (CDH) has Hadoop MapReduce and Spark already in the package so we reuse our setup from NAP. The advantage of this is we can use the same data to test both Hadoop and Spark for our experiments. We also use the word count program in both Hadoop and Spark as the benchmark for our results.

We gather the actually power readings the same way we did for NAP, with a power meter. As for the thermal readings we use thermometers like we did for iTad and also use the iTad model to model outlet temperature.

4.6 Observations

Since Spark has not been examined for its thermal patterns nor its energy consumption patterns our first contribution is a record account of real life data of a Hadoop vs Spark cluster. We will later describe these trends and the reason behind why we got those results.

4.6.1 Thermal Observations

Using the information about the servers we were able to find the constants needed to use the iTad model we developed to take utilization values from the cluster and extrapolate outlet temperature. The Figure 4.1 shows how Hadoop and its NAP algorithms compare to

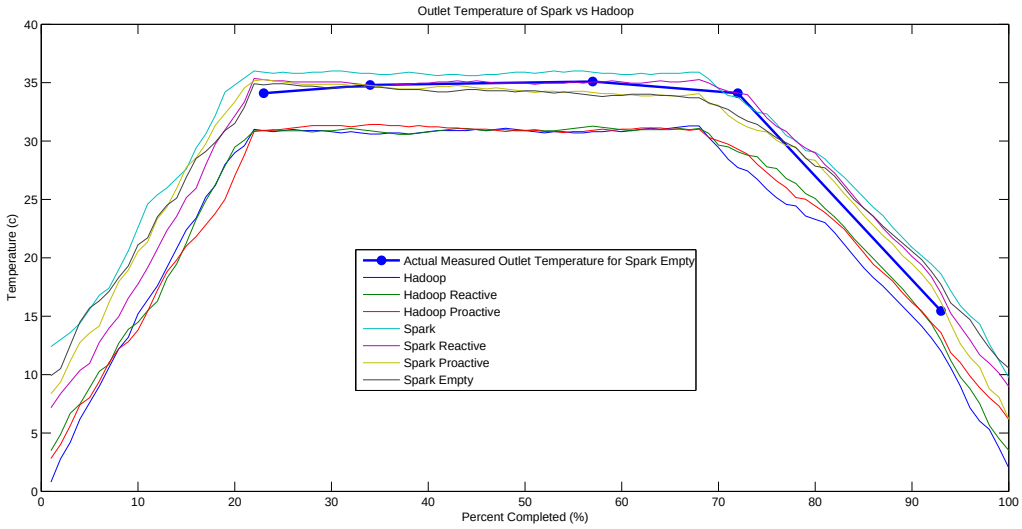
Spark using the Sparke algorithms we proposed; we achieved this by using the model but also added probed the actually temperature a few times for verification. The figure shows that outlet temperature change throughout a lifecycle of a word count. Overall you can see Hadoop overall uses the same amount of energy for all the algorithm and is less energy expensive than that of Spark. This can be explained by simple utilization, because Spark uses all main memory for its calculations which make it become over 85% CPU utilization which heats up components very quickly, while Hadoop rest around 40% utilization. The disk are just not biggest heat distributors at this scale to add any temperature variance. We can deduce that the disk have no effect because the precision of the Hadoop temperatures are very high this and if disk had any great effect we would atleast see in the Hadoop case the reactive or proactive have an noticeable difference. We already have seen Hadoop energy savings from NAP so we know that the algorithms save energy which in turns should save heat but from these results you can see that the heat saved was negligible. This can be attributed to the fact that the disks' physical size being very large compared to the other components in a the server, as we know [10] the bigger something is to dissipate heat the less it actually does. Also since we know Spark is CPU intensive compared to I/O then the heat from the CPU components over matched the heat from the disks.

After seeing these results of Figure 4.1 we concluded at our scale that controlling for any other variables would not change the thermal output in meaningful way because the nature of Spark being CPU intensive that the thermal output will be roughly the same no matter what so continued onto the energy observations.

4.6.2 Energy Observations

The thermal observations showed that CPU utilization of Spark was too great to see any real thermal energy change using Sparke or any disk management at a small scale. Thermal energy is usually a good indicator for energy used [35] we can assume that Spark uses more energy than Hadoop based on Figure 4.1. We were optimistic we would expect

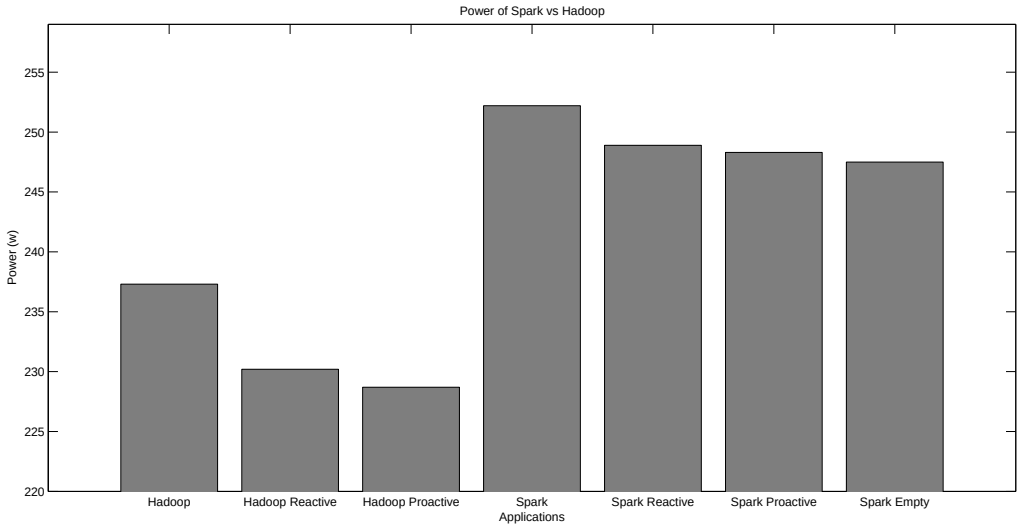
Figure 4.1: Spark vs Hadoop (Group 4) Temperature



better energy savings versus thermal savings is because after our work with NAP and Spark thermal profiling implies the portion of disks contribute to energy of servers is greater than the heat disks contribute to servers. You can see in Figure 4.2 Spark uses much more energy as compared to Hadoop which is to be expected results if you considered that the CPU utilization numbers of Spark as compared to Hadoop. Then you notice Hadoop has more energy efficiency between its algorithms compared to Spark and its Sparke algorithms, this can be explained because Hadoop being less I/O intensive so us managing Hadoop disks would have more energy savings than managing the more active Spark disks. Lastly if you look at the different algorithms in Sparke the reactive and proactive disks seems very close in savings while the empty algorithm had the most savings. An explanation of why the algorithms performed as they did include our main memory was large enough/slow enough to allows disks to sleep in all the algorithms, but for the reactive and proactive the amount of disk on was greater than the number of transitions.

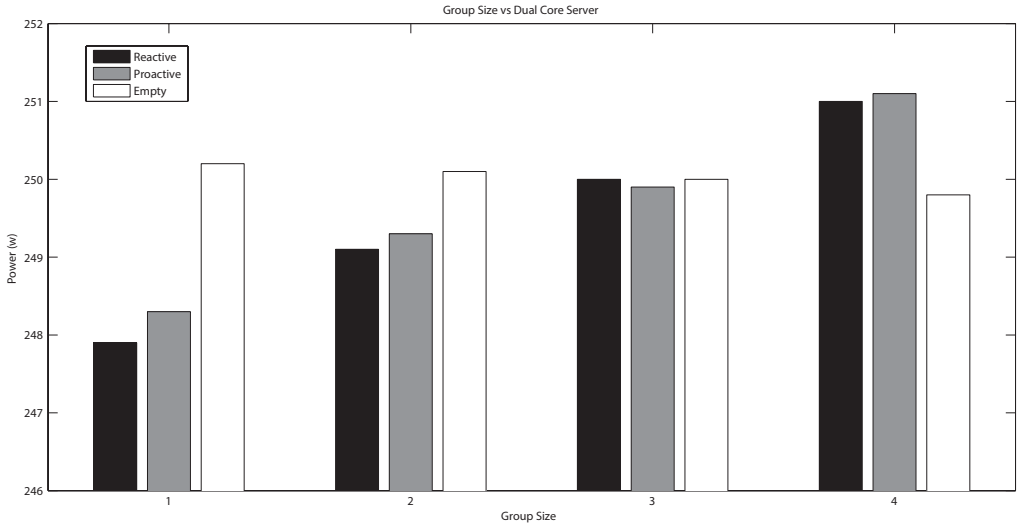
We then ran an experiment to determine how group size effected energy for the reactive and proactive algorithms. The results in Figure 4.3 were interesting because when the group size was 3 and under it was lower energy output than our empty algorithm. This results

Figure 4.2: Spark vs Hadoop (Group 4) Energy



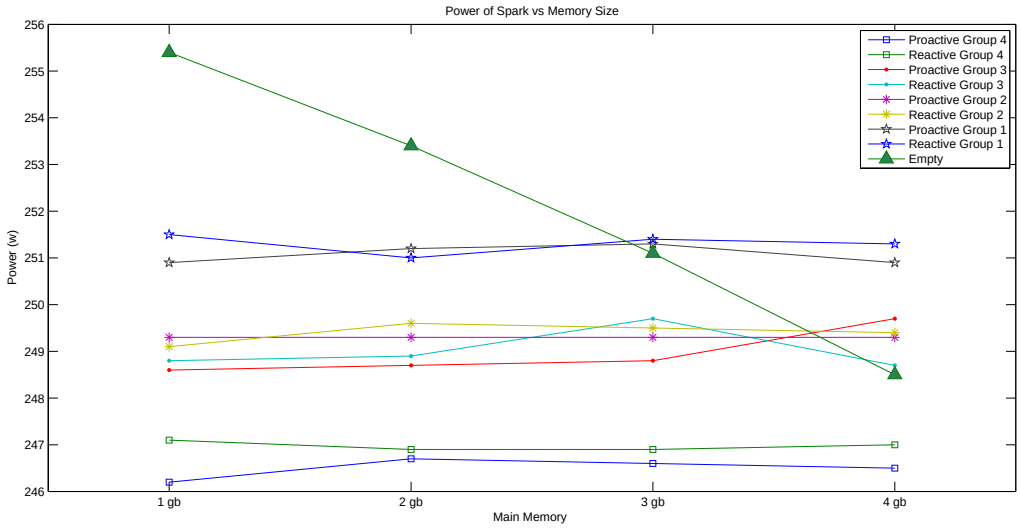
can be explained by the number of cores this machine was running with. These machines being dual core machines make data reads in parallel, since unlike Hadoop, Spark only wants to read the data it can use in main memory it lowers the main memory calls based on the number of cores as explained by the research at Box [29].

Figure 4.3: Spark vs Group Size



The last characteristic we profiled the size of the ram we used, this is because Spark is highly dependent on how much on board memory there is we concluded this could be a major factor in its energy. Spark does all its calculations in memory and with the smaller memory we used in Figure 4.4 had to make more disk calls so energy went up for the empty algorithm because the disks were not getting long enough to sleep to make any savings. The other algorithm would leave their drives on no matter what so the main memory didn't effect how it used its energy.

Figure 4.4: Spark vs Group Sizes vs Memory



4.7 Model and Simulations

4.7.1 Thermal and Standard Energy Model

Our observations and knowledge of Sparks architecture [64] shows very clearly that CPU utilization will stay high for all Spark use cases. This means any savings we want to gain can only easily happen on the disks, which is luck for us because all of Sparkes proposed algorithm deal with disk management. The fact that CPU utilization is so high and CPU utilization is key in both energy efficiency and thermal efficiency any savings energy or thermal we could

Table 4.1: Model Variables

Variables	Description
E_{total}	Energy of the entire system
E_{disks}	Energy from all the disks
E_{disk}	Energy from single disk
$E_{transition_i}$	The energy from a specific transistion for a specific disk
$E_{standby_i}$	The energy of a standby disk for a specific disk
E_{idle_i}	The energy of a idle disk for a specific disk
E_{active_i}	The energy of a active disk for a specific disk
D	Number of disks
N	Size of disk group
P	Number of Processing cores
S_M	Size of Main Memory
S_D	Size of total disks

gain will be proportional to each other. This proportional gain means our thermal efficiency and energy efficiency have the same trend lines, but the scale in number of disks need to gain a significant thermal gain is very large compared to have many disk we would need to get that same gain through energy efficiency.

We also learned that proactive and reactive algorithms have no difference in Spark because of the structure of Spark and the every disk read or write is predictable and iterative for word count so the order reactive algorithm reads disks is the way Spark does naturally and proactive can easily predict when to start a disk.

To gain the most savings we want to have the fewest drives on for the longest amount of time. We know the key factors to energy in Spark are the size of main memory and number of disk used for data parallelism which has in Spark is managed by the amount cores on each machine. The empty algorithm is effected by the size of the main memory more greatly than reactive or proactive because it does not ever let a disk stay asleep, because there will be alot more disk transitions causing a loss of energy/thermal savings. The reactive and proactive algorithms are more effected by the number of disks needs in its data parallelism, because once the size of the group outweights the size of cores there will be disks that are active but not needed, and it is at that point where the empty algorithm benefits over them.

In equation 4.1 we take the model we developed for NAP's reactive and proactive algorithms, that says that each disk has a weighted sum to the total energy, and repurpose them for Sparke's reactive and proactive algorithms. This assumption holds true still because Spark uses the same file system and file come in as uniformed blocks for calculations only difference is it happens at a faster same speed. The part that we changed in the case of Sparke is that, unlike Hadoop NAP, because of Sparks disk parallelism is limited by on memory cores of the computer more than Hadoop. Unlike the I/O stream on Hadoop, Spark has to do transfers from disk to main memory which is done by "one buffer per core" [63], this is why we multiplied the energy by the number of disks active divided by core, if the disk active number is bigger than available cores then we will have energy loss, but if there are more cores than groups no energy should be lost.

$$N/P \geq 1 \tag{4.1}$$

$$E_{disks_{RP}} = \frac{N}{P} * (\sum_{i=1}^{D/N} \frac{N}{D} E_{group} {}_{i*}\frac{D}{N} + \sum_{i=1}^{D} \frac{D-N}{D} E_{standby_i} + E_{transitions_i}) \tag{4.2}$$

Sparke has one last algorithm that doesn't function like the other ones, the "empty" algorithm doesn't have a disk group number because it closes all the disks down and open just the ones need for every memory call. Which is essentially saying the group size is equal to the max data parallelism. The other change this algorithm does changes how often things are transitioned are calculated. There will be a transition anytime the main memory cannot continue doing its calculation and needs to dump intermediate data. The amount it will call the disk is proportional to the size of disk divided by the size of main memory as you can see in 4.3.

$$D/P \geq 1 \tag{4.3}$$

$$W = S_D/S_M \tag{4.4}$$

$$E_{disks_E} = \sum_{i=1}^{D/P} \frac{P * W}{D} E_{group} \underset{i*}{\frac{P}{D}} + \sum_{i=1}^{D} \frac{D-P}{D} E_{standby_i} + \frac{P}{S_M} \sum_{i=1}^{W} E_{transitions_i} \tag{4.5}$$

To generalize the trends of these trends we can turn the summation into functions. For the reactive and proactive algorithms the first summation in equation 4.1 has active disk energy multiplied by a number less than 1 we know this because of the proof in equation 4.3. Given an Egyptian fraction [53] it can be expressed a function y/x like we describe in equation 4.6 where y is the group size and x is the number of cores in a machine. The standby energy is also variable of the group that we want to test for so we add it to our general form. The other variables like transition are static so they are expressed like as a the constant e.

$$N/P \geq 1 \tag{4.6}$$

$$N \geq P \tag{4.7}$$

$$\therefore E_{disks_{RP}} \simeq f(X) = D(N/X + cN + e) \tag{4.8}$$

We used the knowledge we learned from equation 4.6 and apply the same thing to the empty algorithm. There are no groups in this algorithm so the function will not change according to it. In equation 4.9 you can see that there is a new term that that incorporates the main memory. As the memory gets bigger there will be more less energy consumed so it is another inverse function.

$$D/P \geq 1 \tag{4.9}$$

$$D \geq P \tag{4.10}$$

$$\therefore E_{disks_E} \simeq f(X) = D(W * c/X + d/S_M + e) \tag{4.11}$$

4.7.2 Thermal and Standard Energy Simulation

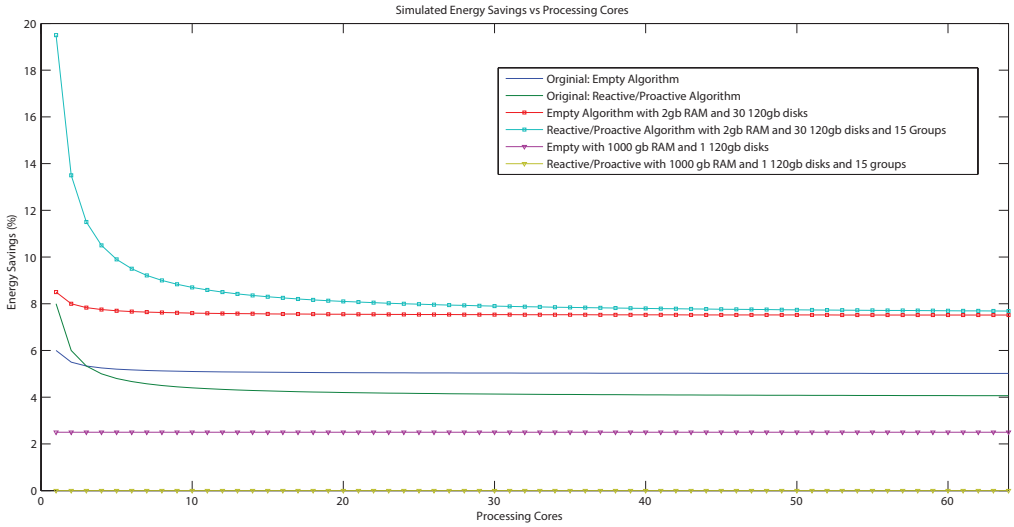
After creating our model and even a more generalize model in the previous section we wanted to simulate some possible outcomes of our model and explain what they mean. Then used our observations to fill-out the unknowns in Equations 4.6, 4.9 then plotted them in a few figures in this section.

In Figure 4.5 we first used our observations which is label original, then we created 2 more possible outcomes to show how energy savings can be effected in Spark. For the original you can see reactive/proactive and empty lines meet then veer off this suggests that the number of cores in our machines are low because the reactive and proactive algorithms are not able to utilize there data parallelism. The figure also shows the lower the processor size the better the energy savings. The 2gb lines on Figure 4.5 show that with an increase in disks we can start getting real savings, also at some point the size of main memory will be a drain on energy savings for the empty algorithm. Finally the 1000gb test shows that at the most memory intensive case if we don't pick a proper group size we could lose any energy savings at all.

4.8 Future Work

This work was simply a survey of Spark energy and thermal efficiency, and while we implemented some energy savings techniques they were not very effective overall. On further examination there should be research into the different primitives that spark employs and whether those primitives have areas for energy savings.

Figure 4.5: Spark Energy Savings Model



We also focuses our efforts into a the word count application, which would generalize for small-medium data calls in terms of Spark disk interaction. Future research could use our techniques for other Spark applications.

The RDD that Spark employees is very active especially if it fits inside the main memory. Some research further research could delve into how it would change the process if RDD was mostly called from disk, and whether we use the disk copy to save manage energy.

Lastly in term of thermal efficiency we were looking at the node level to save thermal energy and found there is very little room to do that, but perhaps at a cluster level we can organize the data to create hot spots and cold spots like previous Hadoop research [35].

4.9 Conclusion

Apache Spark is getting alot of notoriety these days for it improvements over Hadoop while being flexible and easily sharing resources with Hadoop through HDFS. Spark as the name implies is about speed, a lot of speed and so leveraging the fast cache memory is great idea for throughput but causes very high CPU utilization. If Spark becomes a prevailing application on data centers you will see spikes in cooling cost and power bills. Our proposed

algorithms, Sparke, look at Spark through a energy conservationists eyes. This research combined with the previous work of iTad showed that pulling down outlet temperature with in Spark is very hard because the CPU utilization is very high and the I/O utilization is low but constant and CPU utilization dominates most thermal environments. As for energy efficiency we found that using the dynamic power of disks in spark can give some gains, it still isn't ideal. Spark is very much an up incomer in big data but it still very new and without dramatic uptake that lead to rise in huge data center costs, it will be hard shift the priority from speed to energy.

Chapter 5

Savings

In the previous chapters we talked about how the changes we implemented could help reduce costs. Our work was very much practical using real machines so we could estimate real data center savings in terms of energy. So in this chapter we want to discuss savings in terms of real dollars.

Assuming the average cost per kWh is $0.10 then an average data center could cost around $5.2 million a year.[47] Using iTad you could start balance your thermal output, which has been shown that, could to save 15% - 30% in cooling cost. [47] That could be between $250,000 and $500,000 in savings.

Saving energy is a more powerful way to cut costs in a data center. Assuming a Hadoop data center NAP we showed that although we would need 10% more time to run applications we could still have 15% on cost which is close to $0.75 million dollar savings.

If the data center was an Hadoop data center that could be migrated to Spark data center than we could implement Spark-e. This will give us up to 20% savings in time and on top of that a 10% energy savings is quite easy to obtain as shown through work with Spark-e. Which would have a savings to the tune of 1.43 millions dollars.

Combining energy savings with thermal savings we are looking at 20% - 33% savings every year for moderately well optimized energy aware data center, which shows that there is huge incentive to continue this line of research.

Chapter 6

Conclusion

Our research has proven that data center are a drain on resources but one that will be growing as we enter a world in which more people relies on cloud services. Instead of trying to reduce the number of data center that we have the goal should be to optimize them to be efficient in everything we can including energy efficiency and cost.

In this work we showed that thermal energy is a important indicator on air conditioning power used and energy efficiency of a data center. We showed discussed how tools could help to allow companies and programmers to more efficiently monitor and manage thermal energy. To that end we proposed a model of data centers that would be low cost to use and in terms of computing resources. The model we proposed was dubbed iTad, I/O Thermal Aware Datacenter, and it contribution to the world of thermal efficiency was two fold: it created an easy way to model outlet temperature for a single node, it was both I/O and CPU model that allowed for more accurate temperature prediction. This model was able to model outlet temperature for small scale server very accurately, over time with more research this model could be scaled up to take in more information to correctly predict the temperatures and complex thermal model.

Our next contribution was to show find a novel approach to lower energy consumption in a data center. We did that by choosing a specific data center configuration, in this case Hadoop, and look for ways to cut down on its disk energy cost. Our solution was to leverage the Hadoop architecture and target the disks that didnt need to be in an active state during any given task. We implemented to energy conservation algorithms which we packaged together and called NAP, energy aware Hadoop. NAP contain two algorithms one which reacted to which disks were being called on and turned other disks to standby until they

were needed; the second algorithm would predict when the next disk will need to be spun up and woke it up early so it wouldn't run into a loss in performance. We were able to prove that at scale we could save over 15% of the energy with the algorithms in still a primitive state.

Lastly we wanted to take our previous research and extend it to profile a relatively under researched cluster computer framework, Spark. Spark is often compared to Hadoop because of the way it works and even can run on the same file structures. So we extended our NAP algorithms to create an energy aware Spark algorithm set called Sparke. Sparke contains the reactive and proactive algorithms from NAP but also introduces the idea of keeping no disks active while Spark does all its computation in memory. We also though because of the lower disk energy from Sparke we might get some thermal benefit as well, but using iTad profile Spark thermal usage and show that the Sparke is better suited for energy efficiency. Even the energy benefits we saw with Sparke were much lower than NAP for Hadoop. We were still able to contribute some rough models and energy profiles for a more in-depth Spark analysis.

These three research products were formed together to show that data centers have plenty of different energy problems but can be solved in an array of ways. The only way to fix those problems is to identify the energy weaknesses, develop tools to analyze the weakness, and systems to fix those weakness; thus why we introduced Sparke, iTad, and NAP respectively.

Bibliography

[1] Convective and radiant heat transfer equations. http://blowers.chee.arizona.edu/cooking/heat/convection.html.

[2] Flovent. http://www.mentor.com/products/mechanical/products/upload/flovent-data-center.pdf.

[3] hddtemp. http://manpages.ubuntu.com/manpages/natty/man8/hddtemp.8.html.

[4] iostat. http://sourceforge.net/apps/mediawiki/filebench/index.php?title=Main_Page/.

[5] iostat. http://linux.die.net/man/1/iostat.

[6] stress. http://www.unixref.com/manPages/stress.html.

[7] S. R. Alam, H. N. El-Harake, K. Howard, N. Stringfellow, and F. Verzelloni. Parallel i/o and the metadata wall. In *Proceedings of the Sixth Workshop on Parallel Data Storage*, PDSW '11, pages 13–18, New York, NY, USA, 2011. ACM.

[8] Amazon. http://www.amazon.com/pyle-pma90-anemometer-thermometer-temperature/dp/b009tq6ilq. *Amazon*, 10 March 2012.

[9] H. F. H. Andreas Bieswanger and H.-D. Wehle. Energy efficient data center. Technical Report 1, 2012.

[10] A. J. Barra and J. L. Ellzey. Heat recirculation and heat transfer in porous burners. *Combustion and Flame*, 137(1):230 – 241, 2004.

[11] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, Dec. 2007.

[12] D. J. Brown and C. Reams. Toward energy-efficient computing. *Commun. ACM*, 53(3):50–58, Mar. 2010.

[13] L. Cui. Parallel pso in spark. 2014.

[14] G. Czajkowski. *Sorting 1PB with MapReduce*, 2008.

[15] J. Dean and S. Ghemawat. Google research publication: Mapreduce. *Google Research Publication: MapReduce*, 2004.

[16] D. Economou, S. Rivoire, and C. Kozyrakis. Full-system power analysis and modeling for server environments. In *In Workshop on Modeling Benchmarking and Simulation (MOBS*, 2006.

[17] P. Eibeck and D. Cohen. Modeling thermal characteristics of a fixed disk drive. *Components, Hybrids, and Manufacturing Technology, IEEE Transactions on*, 11(4):566 –570, dec 1988.

[18] I. n. Goiri, K. Le, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini. Greenhadoop: Leveraging green energy in data-processing frameworks. In *Proceedings of the 7th ACM European Conference on Computer Systems*, EuroSys '12, pages 57–70, New York, NY, USA, 2012. ACM.

[19] S. Govindan, A. Sivasubramaniam, and B. Urgaonkar. Benefits and limitations of tapping into stored energy for datacenters. In *Computer Architecture (ISCA), 2011 38th Annual International Symposium on*, pages 341–351, June 2011.

[20] L. Gu and H. Li. Memory or time: Performance evaluation for iterative operation on hadoop and spark. In *High Performance Computing and Communications*, pages 721–727, Nov 2013.

[21] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Franke. Drpm: dynamic speed control for power management in server class disks. In *Computer Architecture, 2003. Proceedings. 30th Annual International Symposium on*, pages 169–179, June 2003.

[22] S. Gurumurthi, A. Sivasubramaniam, and V. K. Natarajan. Disk drive roadmap from the thermal perspective: A case for dynamic thermal management. *SIGARCH Comput. Archit. News*, 33(2):38–49, May 2005.

[23] U. Hoelzle and L. A. Barroso. *The Datacenter As a Computer: An Introduction to the Design of Warehouse-Scale Machines*. Morgan and Claypool Publishers, 1st edition, 2009.

[24] S. Huang, J. Huang, J. Dai, T. Xie, and B. Huang. The hibench benchmark suite: Characterization of the mapreduce-based data analysis. In *Data Engineering Workshops (ICDEW), 2010 IEEE 26th International Conference on*, pages 41–51, March 2010.

[25] A. Hylick, A. Rice, B. Jones, and R. Sohan. Hard Drive Power Consumption Uncovered. ACM SIGMETRICS Performance Evaluation Review, Dec. 2007. Poster.

[26] T. Jiang, Q. Zhang, R. Hou, L. Chai, S. Mckee, Z. Jia, and N. Sun. Understanding the behavior of in-memory computing workloads. In *Workload Characterization (IISWC), 2014 IEEE International Symposium on*, pages 22–30, Oct 2014.

[27] C. Kaewkasi and W. Srisuruk. A study of big data processing constraints on a low-power hadoop cluster. In *Computer Science and Engineering Conference (ICSEC), 2014 International*, pages 267–272, July 2014.

[28] K. Kambatla and Y. Chen. *The Truth About MapReduce Performance on SSDs*, 2014.

[29] L. C. T. KEH. Apache spark in resource constrained states @ONLINE, 2014.

[30] Y. Kim, S. Gurumurthi, and A. Sivasubramaniam. Understanding the performance-temperature interactions in disk i/o of server workloads. In *High-Performance Computer Architecture, 2006. The Twelfth International Symposium on*, pages 176 –186, feb. 2006.

[31] M. Komorowski. *a history of storage cost*, 2009.

[32] F. Kong and X. Liu. A survey on green-energy-aware power management for datacenters. *ACM Comput. Surv.*, 47(2):30:1–30:38, Nov. 2014.

[33] J. G. Koomey. Estimating total power consumption by servers in the U.S. and the world. Technical report, Lawrence Derkley National Laboratory, Feb. 2007.

[34] K. R. Krish, M. S. Iqbal, M. M. Rafique, and A. R. Butt. Towards energy awareness in hadoop. In *Proceedings of the Fourth International Workshop on Network-Aware Data Management*, NDM '14, pages 16–22, Piscataway, NJ, USA, 2014. IEEE Press.

[35] S. Kulkarni. *Cooling Hadoop: Temperature Aware Schedulers in Data Centers.* PhD thesis, Auburn University, 2013.

[36] J. Li, G. Peterson, and P. Cheng. Three-dimensional analysis of heat transfer in a micro-heat sink with single phase flow. *International Journal of Heat and Mass Transfer*, 47(190):4215 – 4231, 2004.

[37] L. Li, C.-J. M. Liang, J. Liu, S. Nath, A. Terzis, and C. Faloutsos. Thermocast: a cyber-physical forecasting model for datacenters. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 1370–1378, New York, NY, USA, 2011. ACM.

[38] F. Liang, C. Feng, X. Lu, and Z. Xu. Performance characterization of hadoop and data mpi based on amdahl's second law. In *Networking, Architecture, and Storage (NAS), 2014 9th IEEE International Conference on*, pages 207–215, Aug 2014.

[39] A. Manzanares, X. Qin, X. Ruan, and S. Yin. Pre-bud: Prefetching for energy-efficient parallel i/o systems with buffer disks. *ACM Transactions on Storage (TOS)*, 7(1):3, 2011.

[40] S. C. Markey. *Deploy an OpenStack private cloud to a Hadoop MapReduce environment*, 2012.

[41] D. McCafferty. *Surprising Statistics About Big Data*, 2014.

[42] J. Moore, J. Chase, and P. Ranganathan. Consil: Low-cost thermal mapping of data centers. In *the First Workshop on Tackling Computer Systems Problems with Machine Learning (SysML)*, 2006.

[43] NRDC. *America's Data Centers Consuming and Wasting Growing Amounts of Energy*, 2014.

[44] K. O'Dell. *How-to: Select the Right Hardware for Your New Hadoop Cluster*, 2013.

[45] A. J. Oliner, A. P. Iyer, I. Stoica, E. Lagerspetz, and S. Tarkoma. Carat: Collaborative energy diagnosis for mobile devices. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, page 10. ACM, 2013.

[46] L. Person. *Global Hadoop Market (Hardware, Software, Services, HaaS, End User Application and Geography).*

[47] E. N. Power. Five strategies for cutting data center energy costs through enhanced cooling efficiency.

[48] S. Sankar, S. Gurumurthi, and M. R. Stan. Intra-disk parallelism: An idea whose time has come. *SIGARCH Comput. Archit. News*, 36(3):303–314, June 2008.

[49] J. Shafer, S. Rixner, and A. L. Cox. The hadoop distributed filesystem: Balancing portability and performance. In *Performance Analysis of Systems & Software (ISPASS), 2010 IEEE International Symposium on*, pages 122–133. IEEE, 2010.

[50] R. Sharma, C. Bash, C. Patel, R. Friedrich, and J. Chase. Balance of power: Dynamic thermal management for internet data centers. *IEEE Internet Computing*, 9(1), 2005.

[51] H. F. Sheikh, H. Tan, I. Ahmad, S. Ranka, and P. Bv. Energy- and performance-aware scheduling of tasks on parallel and distributed systems. *J. Emerg. Technol. Comput. Syst.*, 8(4):32:1–32:37, Nov. 2012.

[52] X. Shi, M. Chen, L. He, X. Xie, H. Jin, Y. Chen, S. Wu, et al. Mammoth: Gearing hadoop towards memory-intensive mapreduce applications. 2014.

[53] R. Stong. Pseudofree actions and the greedy algorithm. *Mathematische Annalen*, 265(4):501–512, 1983.

[54] K. Sun, Z. Chen, J. Ren, S. Yang, and J. Li. M2c: Energy efficient mobile cloud system for deep learning. In *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, pages 167–168. IEEE, 2014.

[55] C. Tan, J. Yang, J. Mou, and E. Ong. Three dimensional finite element model for transient temperature prediction in hard disk drive. In *Magnetic Recording Conference, 2009. APMRC '09. Asia-Pacific*, pages 1 –2, jan. 2009.

[56] Q. Tang, S. Gupta, D. Stanzione, and P. Cayton. Thermal-aware task scheduling to minimize energy usage of blade server based datacenters. In *Dependable, Autonomic and Secure Computing, 2nd IEEE International Symposium on*, pages 195 –202, 29 2006-oct. 1 2006.

[57] Q. Tang, S. Gupta, and G. Varsamopoulos. Thermal-aware task scheduling for data centers through minimizing heat recirculation. In *Cluster Computing, 2007 IEEE International Conference on*, pages 129 –138, sept. 2007.

[58] Q. Tang, S. Gupta, and G. Varsamopoulos. Thermal-aware task scheduling for data centers through minimizing heat recirculation. In *Cluster Computing, 2007 IEEE International Conference on*, pages 129 –138, sept. 2007.

[59] Q. Tang, S. Gupta, and G. Varsamopoulos. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach. *Parallel and Distributed Systems, IEEE Transactions on*, 19(11):1458 –1472, nov. 2008.

[60] W. Wechsatol, S. Lorente, and A. Bejan. Dendritic heat convection on a disc. *International Journal of Heat and Mass Transfer*, 46(23):4381 – 4391, 2003.

[61] W. Wei, D. Jiang, J. Xiong, and M. Chen. Exploring opportunities for non-volatile memories in big data applications. In *Big Data Benchmarks, Performance Optimization, and Emerging Hardware*, pages 209–220. Springer, 2014.

[62] J. Xie, S. Yin, X. Ruan, Z. Ding, Y. Tian, J. Majors, A. Manzanares, and X. Qin. Improving mapreduce performance through data placement in heterogeneous hadoop clusters. In *Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*, pages 1–9, April 2010.

[63] M. Zaharia. Apache spark configuration@ONLINE, 2015.

[64] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. Mccauley, M. Franklin, S. Shenker, and I. Stoica. Fast and interactive analytics over hadoop data with spark. *USENIX; login*, 37(4):45–51, 2012.