An Analysis of Conventional and Heterogeneous Workloads on Production Supercomputing Resources

Jonathan Allen Berkhahn

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

Ali Raza Ashraf Butt, Chair
Byoung-Do Kim
Yong Cao

May 20, 2013
Blacksburg, Virginia, USA

Keywords: Workload Synthesis, Resource Provisioning, Heterogeneous computing

An Analysis of Conventional and Heterogeneous Workloads on Production Supercomputing Resources

Jonathan Allen Berkhahn

(ABSTRACT)

Cloud computing setups are a huge investment of resources and personnel to maintain. As the workload on a system is a major contributing factor to both the performance of the system and a representation of the needs of system users, a clear understanding of the workload is critical to organizations that support supercomputing systems. In this paper, we analyze traces from two production level supercomputers to infer the characteristics of their workloads, and make observations as to the needs of supercomputer users based on them. We particularly focus on the usage of graphical processing units by domain scientists. Based on this analysis, we generate a synthetic workload that can be used for testing future systems, and make observations as to efficient resource provisioning.

Acknowledgments

Firstly, I would like to express my deep gratitude to my advisor Dr. Ali R. Butt. Ali is the reason I even considered graduate school as an option, and has continued to support me throughout my various struggles.

Next, I would like to thank Dr. B.D. Kim. BD has been instrumental in getting the access required to even attempt this study. Without him this analysis would not have been possible.

I would also like to thank Dr. Yong Cao. His class first introduced me to the concept of GPGPU programming, which lead to my interest in analyzing the usage of it here at Virginia Tech, which lead directly to this study.

I would also like to express my gratitude to Dr. Henry Monti, for his guidance and assistance in this work.

Finally, I would like to thank my parents for their support and encouragement. They made me who I am today

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Production-scale supercomputing setups requires a huge investment of resources and personnel to create and maintain. A thorough understanding of workload is vital for optimal system design and efficient resource provisioning to get the most performance out of these systems. As these workloads can change and evolve overtime, a true understanding of them requires knowledge of their characteristics over the lifetime of a deployed system. Recently, heterogeneous accelerators such as graphical processing units (GPUs) have become extremely popular in large-scale cluster setups [9], adding an additional layer of complexity to workload analysis . An institution interested in maintaining production cluster setups is thus behooved to fully understand the needs of its supercomputing users and the traits of the workloads they generate. Additionally, simulations based on traces [8, 10] are one of the main methods for efficiently testing the viability of new system setups. Therefore, a knowledge of the workload specific to a particular system is necessary not only to efficiently utilize that system, but to accurately provision future systems to service that workload as well.

The recent interest in GPUs as accelerators in a supercomputing environment has led to a rapid adoption and investment in them among research communities. While much computer science research has been done in their various uses [1, 7], it remains to be seen if they have seen a corresponding increase in usage by domain scientists. With these users being the primary users of many production supercomputing systems, their usage or lack thereof is of great interest to organizations that build and maintain such systems.

Virginia Polytechnic Institute and State University maintains a variety of production supercomputers of different architectures [4] through its Advanced Research Computing division. These systems represent an investment of millions of dollars by the university that serves a community of tens of thousands of students and faculty, providing for a wide variety of scientific and research applications. For this paper, we have focused on two specific clusters

maintained at Virginia Tech, Athena and HokieSpeed.

1.2 Contribution

1) Traces and trace analysis - In this work, we have collected three traces covering over 20 months of usage of production level supercomputing clusters. We have created tools to assist in the analysis of these and future traces. The information garnered has been used to make observations as to the nature of the workload on ARC systems. This is useful not only to further understanding of the needs of supercomputing users at Virginia Tech, but is also generalizable to the needs of the domain researchers elsewhere.

2) Synthesized workload - Based on the traces available, we have created a synthetic workload to represent the usage of ARC clusters. This workload can be utilized to run simulations for the testing of future setups to ascertain how well they service the workload unique to Virginia Tech's systems.

3) Policy recommendations - Based on our observations of the traces, we have made recommendations for actions that can be made to better utilize the resources we have in place, as we as the potential provisioning of future resources.

1.3 Outline

The remainder of this paper shall be organized as follows. Chapter 2 shall go over background information about the systems Athena and HokieSpeed, as well as the traces themselves. Chapter 3 will go over the approaches used to analyze the traces. Chapter 4 will cover the analysis of the traces. Chapter 5 will cover the synthetic workload built on data gleaned from the analysis. Chapter 6 will cover the observations and recommendations based on the trace analysis, and Chapter 7 will conclude.

Chapter 2

Background

2.1 Advanced Research Computing

Advanced Research Computing (ARC) is the professional division at Virginia Tech that manages university supercomputing resources. Their mission is to provide computing and visualization resources, support, and leadership to advance computational research at Virginia Tech. In addition, they aim to encourage the development of knowledge and skills in computational tools and techniques among undergraduate, graduate, and research faculty and staff [4]. They are a leader in high-performance computing, presenting research at conferences such as the International Conference for High Performance Computing Networking, Storage, and Analysis and collaborating with national centers for HPC research such as Oak Ridge National Laboratory. This paper focuses mainly on two systems maintained by ARC, Athena and HokieSpeed.

ARC makes use of two pieces of software to schedule and manage jobs on their systems that I have used to collect data for these traces. Terascale Open-Source Resource and QUEue Manager (TORQUE) is an open-source distributed resource manager that is used to allocate resources for user jobs on Athena and HokieSpeed [3]. Running on top of TORQUE is the Moab Cluster Scheduler, which users use to request cluster resources for jobs [2]. The majority of the information in this paper was collected using these two programs.

2.2 Athena

Athena is a production 42 node hybrid CPU-GPU system running CentOS Linux 5.5. Each of the nodes is equipped with four octa-core 2.3 GHz AMD Magny Cour processors for a total of 1,344 cores across the entire system. Each node has a large memory footprint with 64 GB of memory, or 2 GB per core. 16 of the nodes are further equipped with two NVidia S2050

Fermi GPUs. The nodes are connected with quad-data-rate InfiniBand (40 Gb/sec) and have 40 TB of local stable storage. This system is intended for computation and visualization of large data sets, with the GPUs primarily intended to accelerate rendering tasks, and serves the computing needs of the thousands of research students and faculty at the university.

Athena has some restrictions in place on the number and size of jobs that can be run. The general queue of 26 nodes restricts normal users to a maximum of 256 cores at one time, while the GPU queue has a maximum of 192 cores per use. Both queues have a maximum walltime restriction of 300 hours.

2.3 HokieSpeed

HokieSpeed is a 204 node GPU-accelerated cluster running CentOS 6.3. Each node is equipped with two six-core 2.4 GHz Intel Xeon E5645 processors for a total of 2448 cores. Each node has a total of 24 GB of memory, equating to the same 2 GB per core. Each is node is outfitted with two NVidia M2050 GPUs. The nodes are connected with quad-data-rate InfiniBand (40 Gb/sec). It has two separate job queues, the general queue for normal jobs and the long queue for particularly long running jobs. Preference is given to jobs in the long queue, but unlike the Athena queues, resources are shared between them. HokieSpeed is currently a research system not available to the university at large, currently only accessible by principle investigators pertaining to the research project. It is undergoing testing and is expected to become available to the wider Virginia Tech community in 2014 [4].

HokieSpeed also has job restrictions in place. In the normal queue, users are limited to a maximum of 32 nodes and 384 cores. Normal jobs have a runtime limit of 72 hours. The long queue imposes harsher restrictions on job size, with a maximum of 8 nodes and 96 total processors. However, long queue jobs may run for up to 144 hours.

Chapter 3

Approach

3.1 Analyzing Traces

We are are mainly concerned with analyzing the traces to extract useful data about the characteristics of the jobs running on Athena and HokieSpeed. While the exact nature of the applications used is not known, the statistics such as processor usage or memory requirements can be used to characterize the supercomputing workload of ARC users. Additionally, these statistics can be used for the simulation of the workload on future cloud setups.

We divide jobs into three categories. *Seconds jobs* are jobs that have a runtime of up to five minutes in length. These are small, bursty jobs as well as test jobs submitted by users that do no actual work. *Minutes jobs* are slightly longer in length, between five minutes and one hour. These are medium length jobs that perform actual work but do not run for long periods of time. *Hours jobs* are the final category, comprised of all jobs with a duration longer than one hour. This catches the long running jobs that have significant uptime requirements.

3.2 Overview of Traces

Athena TORQUE Trace: The first trace is a collection of TORQUE logs taken from the first 20 months of Athena's lifespan, from October 2010 to May 2012. It shows the resources that were allocated to run submitted user jobs during that time. It includes information such as the run times, node allocation, processors per node allocated, and CPU utilization. CPU utilization is expressed as a total amount of time all the CPUs alloacted spent running user code. We further divide this trace into two groups, separating the jobs run on the 26 node general queue (athena_q) and those run on the 16 node GPU-enabled GPU queue (athenagpu_q). Note that due to this trace only having records of resources that were actually allocated for user jobs, it shows the system being down and the system being idle as identical

states.

Athena GPU Utilization Trace: The second is a week long trace we collected to gauge the utilization of the GPUs on the 16 nodes of the athenagpu_q. This trace was collected using the Moab scheduler and the NVidia System Management Interface (nvidia-smi), a component of the CUDA Source Developer Kit. Moab was used to collect statistics about the number of processors allocated for the jobs, while nvidia-smi was used measure the utilization of the GPUs. Utilization figures were expressed as a percentage of time the GPU spent running user kernels.

HokieSpeed Moab Trace: The final trace is a week long trace from the Moab scheduler running on HokieSpeed. While HokieSpeed is still in testing, it is open to those working on the HokieSpeed project itself as well as "Power Users" from other departments in the university. This trace contains figures about job resource allocations and runtimes, and is compared with the Athena TORQUE traces to elucidate the differences between testing and production ARC systems.

Figure 3.1 details the different steps of the various pipelines used to transform the various logs and traces into usable data.

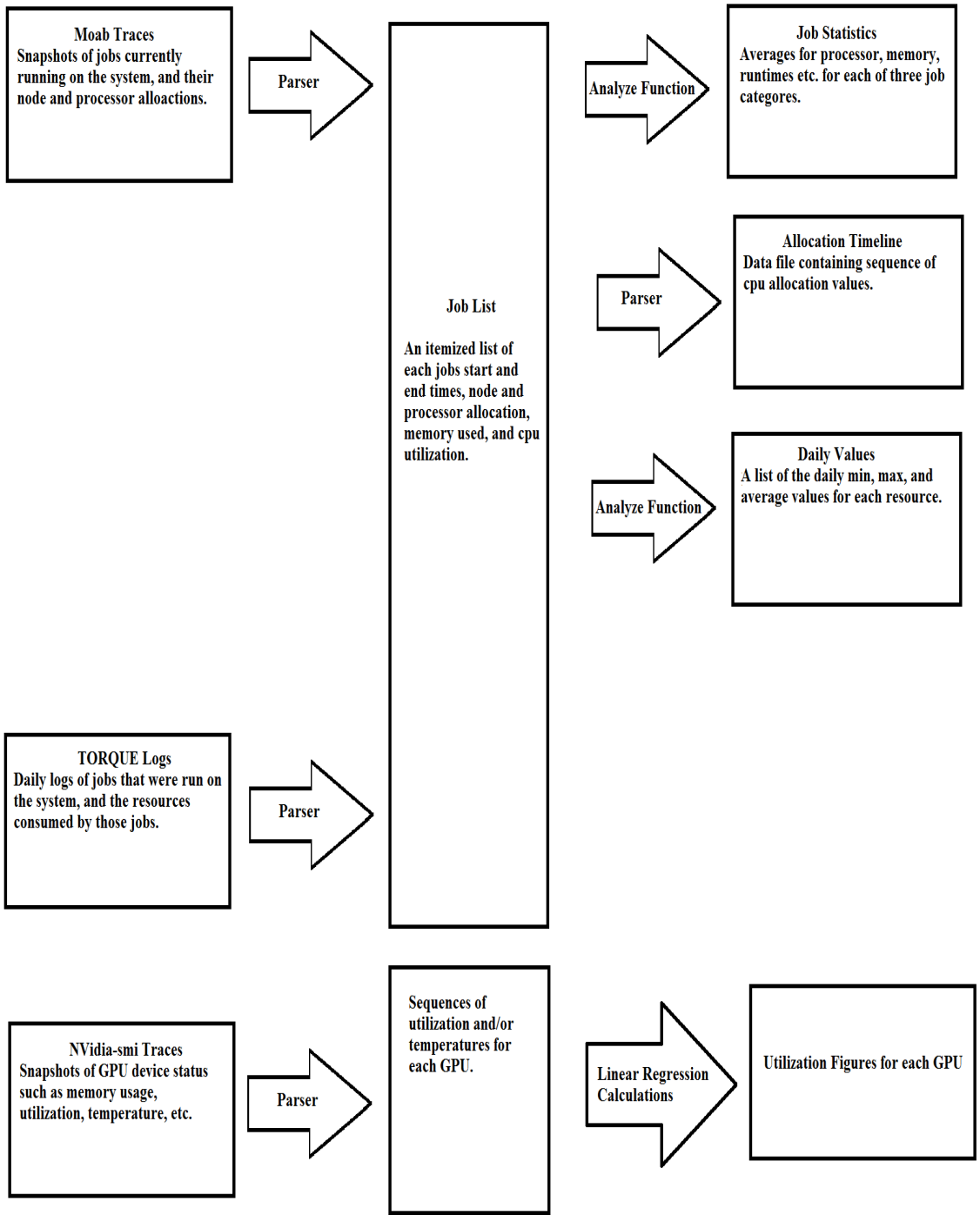

Figure 3.1: Trace Analysis Pipelines

Chapter 4

Trace Analysis

4.1 Athena TORQUE Logs - Athena_q

The trace represents a total of 37,849 jobs over a period of 20 months. Figure 4.2 shows the allocation of CPU cores over the entire duration of the logs.

We see that for the first four months of operation, the system was barely used. The majority of the jobs were vary small, with the system being completely unused most of the time. In February 2011, usage of the system picks up suddenly, showing regular usage that spikes to above 80% of the resources available. Usage dies somewhat around the end of the school year in May. Starting in June, however, we see a consistent low to middling usage that extends through the first week of August. Aside from a constant period of usage for the majority of September, the system sees only slight usage for the rest of the calender year. We see a sudden flurry of activity at the very beginning of 2012, followed by spare usage with occasional spikes in usage until the middle of February. From there we see a steady increase in constant usage until the middle of March, where usage falls off buy remains constant through the end of May. Overall, the system had a usage uptime, or time when at least one core is allocated, of 25.6%.

Table 4.1: Athena_q Job Statistics

Job Category	# of Jobs	Runtime (sec)	Processors	Memory (GB)
Seconds	12069	142, σ=99.2	10.5, σ=57.6	19.8, σ=237
Minutes	9914	1608, σ=1086	11.5, σ=59.1	28.5, σ=277
Hours	15863	16155, σ=21322	4.02, σ=16.7	4.86, σ=22.4
>8 Hours	1861			

Job Statistics: We examine the runtime, processor, and memory usage of the jobs divided into the previously mentioned categories. Table 4.1 shows the statistics of the various job categories. Although the standard deviation is quite large, the average job size in terms of processors allocated is actually quite small for all three job categories at less than an entire nodes worth of cores. Accounting for the standard deviation, most jobs are still well below the hard core limit imposed on users. Figure 4.1 shows the distribution of jobs of all processor allocations. We see that vast majority of jobs are clusters below 100 cores.

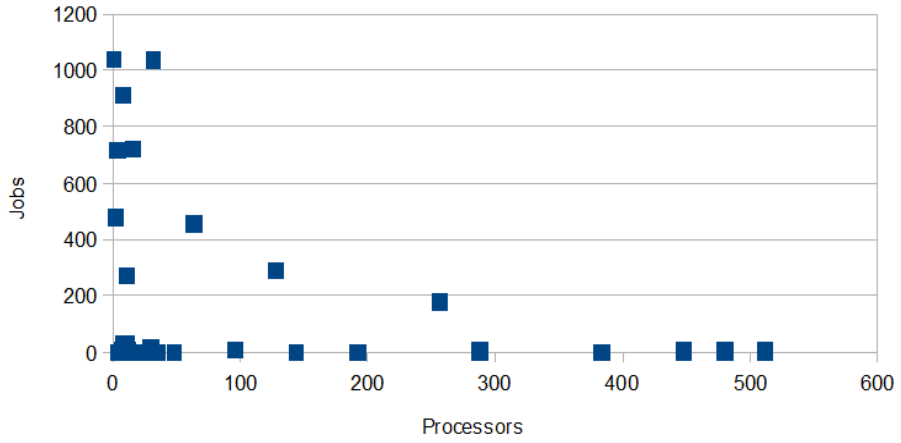

Figure 4.1: Number vs. Size of Jobs on Athena general queue

Curiously, minutes jobs utilize a much larger amount of memory, proportionally, then jobs of the other categories. Seconds jobs use roughly 2 GB/core, following the amount provisioned on Athena. Hours jobs use even less than that, at roughly 1.2 GB/core.

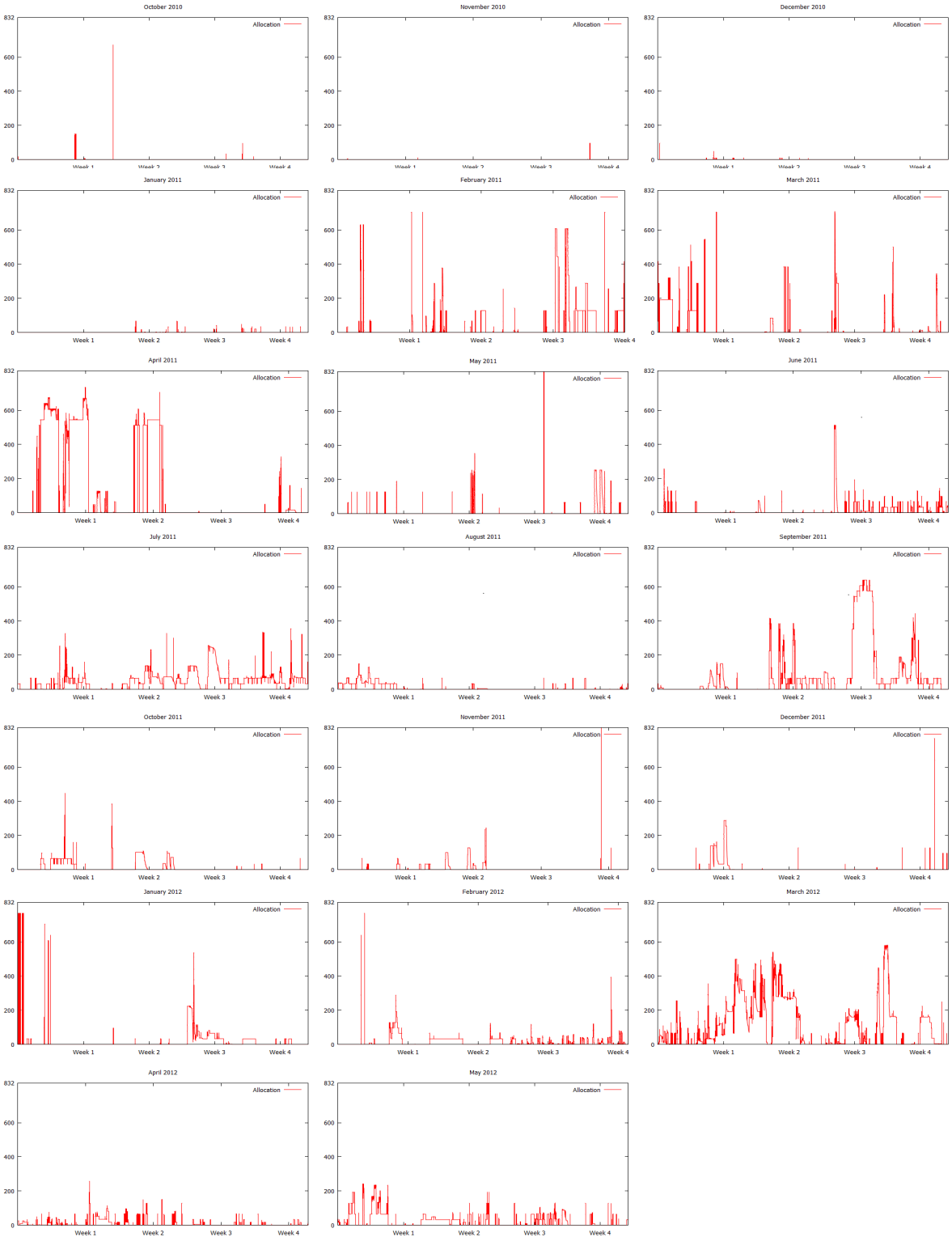

Figure 4.2: CPU Allocation on Athena general queue

Usage over Time: We are also interested in how usage of the system has changed and evolved over time. As the first three months of the trace are largely empty, we omit them from this section of the analysis. This increases the usage uptime of the system to 29.2%. We then compute the daily minimum, maximum, and average CPU allocations. Figure 4.3 shows these figures over the remaining months.

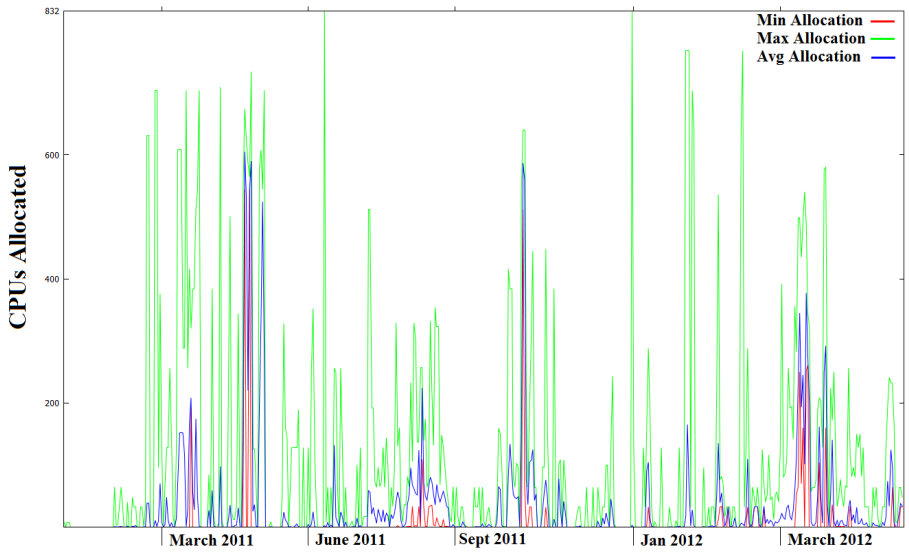

Figure 4.3: Daily CPU Allocation

We see that while there are often large spikes that take almost all of the cores on Athena, these jobs are short lived enough to not significantly impact the average allocation. Only on three occasions does the average allocation get near or above 50%. This shows that system resources are often under-requested, relative to the amount of resources available.

To see if the actual utilization of the system shows a similar drop, we compute the daily average utilization of the system as follows. For each job, we compute an *effective CPUs utilized* by multiplying the walltime by the number of processors requested by the job. We then divide the CPU utilization time by this figure to get a percentage of time the processors requested were actually being used to execute user code. We then multiply this by the number of cores requested to get an effective number of cores used by the job over its runtime. Figure 4.4 shows the daily average allocation vs. the daily average utilization.

We see that just as the average allocation is often much smaller than the maximum allocation, so too is the average utilization usually much smaller than the average allocation. While this is mostly due to factors that are beyond the control of the ARC such as poorly optimized user code, it is important because it allows certain decisions to be made regarding provisioning system resources, which shall be discussed later on.

Overall, we see that most jobs on the general queue are small jobs that use 12 cores or less. The vast majority of jobs run for less than eight hours. Memory use is generally in line with the 2 GB/core available on Athena. System resources are requested sporadically with long periods of complete idleness, and when resources are requested they are often underutilized.

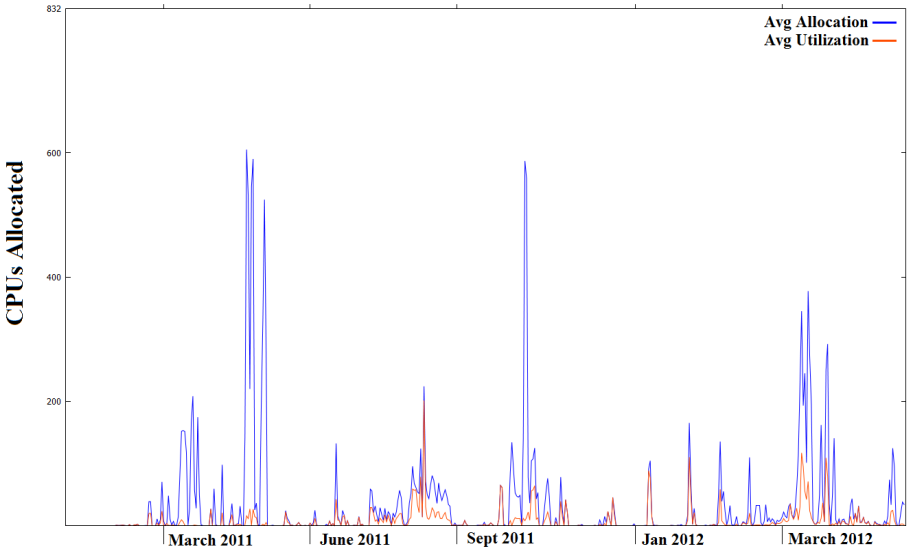

Figure 4.4: Daily CPU Allocation vs. Utilization

4.2 Athena TORQUE Logs - Athenagpu_q

This data is taken from the same 20 months of TORQUE logs as the previous section. It covers a total of 6151 jobs, with a usage uptime of 14.2%. While these logs do not contain information about the allocation or utilization of the GPUs themselves, it does serve as a useful indicator of general activity or lack thereof on the GPU queue. Figure 4.5 Shows the CPU allocation on the GPU over the course of the logs. Note that the GPU queue is smaller than the general queue, consisting of 512 cores on 16 nodes.

We see a significant usage as early as December 2010, much earlier here than the general queue. We see fairly consistent heavy usage through February 2011, and then it begins to drop off. Aside from a single spike in May 2011, usage of the GPU queue is extremely spare until October 2011. There we see some consistent small usage, before dropping again early into November. We see a large spike job later that month, and then only sporadic middling or low usage through the end of the trace.

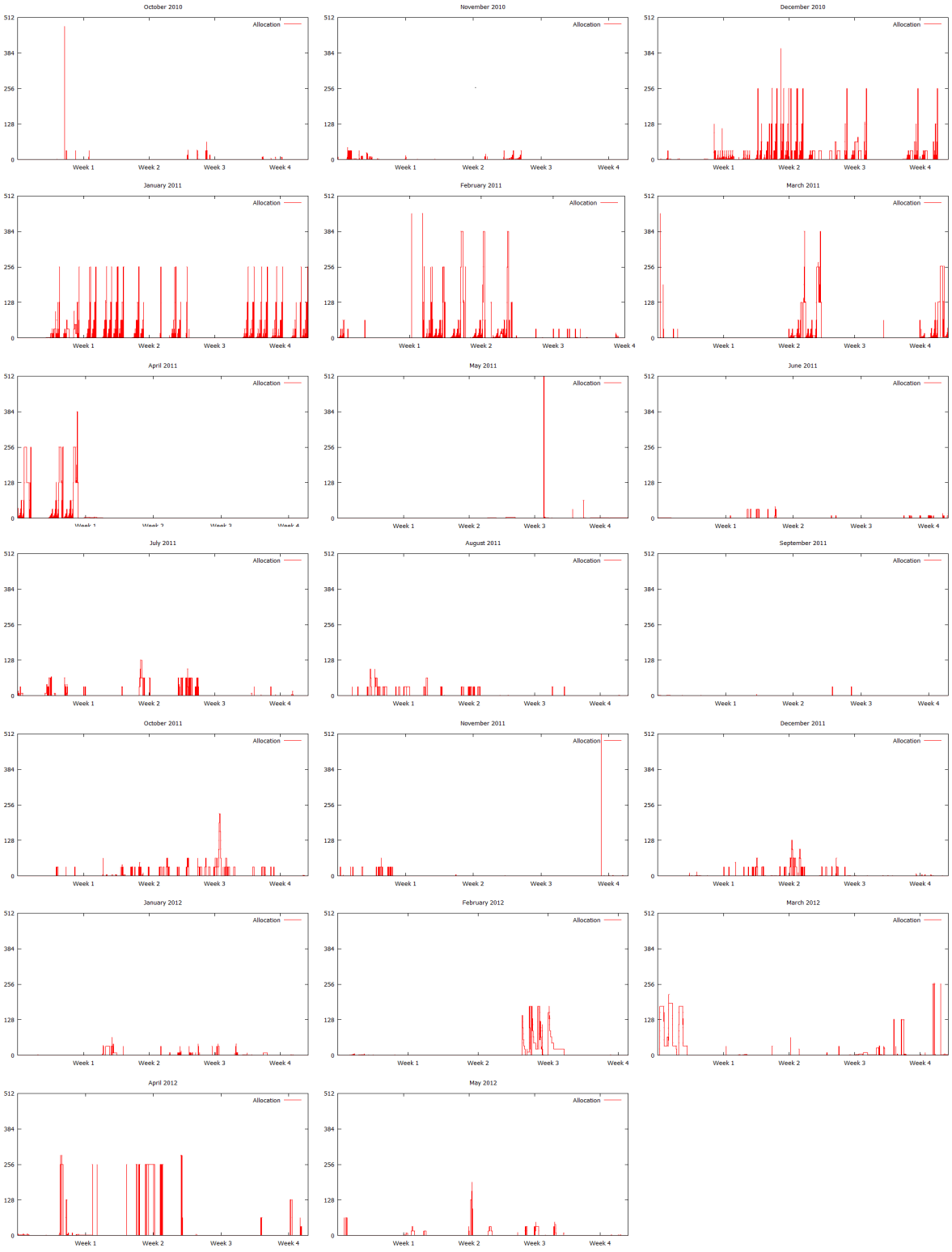

Figure 4.5: CPU Allocation on Athena GPU queue

Table 4.2: Athenagpu_q Job Statistics

Job Category	# of Jobs	Runtime (sec)	Processors
Seconds	2804	128, σ=81.5	20.8, σ=47.0
Minutes	2646	1113, σ=758.6	37.2, σ=57.9
Hours	701	16485, σ=24260	24.7, σ=38.2

Job Statistics: We examine the statistics of the jobs in the GPU queue using the same job categories as previously. These statistics are summarized in Table 4.2. We see that there are far fewer hours jobs compared to the general queue, 11.4% compared to 41.2% in the general queue. Runtimes are comparable, with the average seconds and minutes jobs being slightly shorter and the average hours jobs being slightly longer. The average number of processors allocated per job is substantially larger in all categories compared to the general queue. This is curios given that with the GPUs being the main draw, the queue having fewer cores overall, and the maximum number of cores available to a single user being smaller than on the general queue one would expect the jobs on the GPU queue to be smaller. This is discussed further on in section 4.3, where we consider the actual utilization of the Athenagpu_q GPUs.

Overall, we see less consistent usage on the Athena GPU queue. While it saw heavy use in the first six months of Athena's lifespan, since then it has mostly only seen sporadic usage. Jobs on the GPU queue are often quite short in runtime, with a vary small percentage lasting longer than one hour. Most jobs on the GPU queue are larger in terms of processors cores than their general queue counterparts.

4.3 Athneagpu_q GPU Utilization

GPGPU has been a hot topic in the computer science research community in recent years. We are interested in the actual utilization of the GPUs on Athena primarily to see if their use has actually been adopted by domain scientists for their supercomputing needs. Towards that end, we collected a week long trace on the nodes of the Athnea GPU queue. Note that due to one of the nodes being down for maintenance, this data represents the 30 GPUs on the remaining 15 nodes.

Data Collection: Utilization data was collected using the NVidia System Management Interface, or nvidia-smi. Nvidia-smi is a command line utility included as part of NVidia's CUDA Source Developer Kit to aid in the management and monitoring of NVidia GPUs. It allows users to query device statistics such as memory usage, temperature and utilization. Utilization is expressed as a percentage of time that the device spent running user kernels. Statistics about the CPU allocation and user jobs were collected from the Moab scheduler running on the Athena GPU queue.

Challenges: Collection of GPU utilization data was made difficult by the production nature of Athena. As it has been in near continuous operation for the previous two years, software packages are updated only when absolutely necessary. At the time of the trace, 11 of the nodes were running CUDA 3.1, 2 of the nodes were running CUDA 3.2, and 2 of the nodes were running CUDA 4.0. Support for collection of explicit utilization figures was added in CUDA 3.2.

We observe that the temperatures of processors are directly correlated with their utilization [6]. We have utilization data for only 8 GPUs but temperature data for all 30. The ambient temperatures of the two GPUs in each node were different, with those in slot 0 running at an average of 58 °C while idle and those in slot 1 running at an average of 51 °C while idle. Given the four sets of data we have for each slot, we would be able to compute a function to correlate temperature and utilization. However, given the nature of the results we found, we were unable to calculate actual equations.

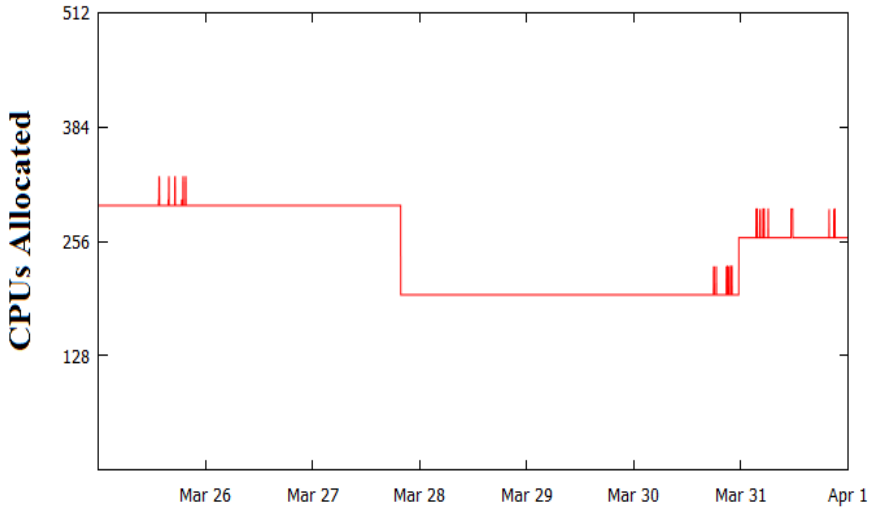

Figure 4.6: CPU Allocation on Athena GPU queue, Utilization Trace

We see that over the course of the trace the queue is seeing healthy usage. The majority of the resources consumed were due to a series of large jobs submitted by two unique users. However, we see that there are also a collection of smaller, shorter jobs submitted over the course of the trace. Figure 4.7 shows the average utilization of the 8 GPUs we have explicit utilization information for, while Figure 4.8 shows the averaged temperatures of the slot 0 and 1 GPUs over the course of the trace.

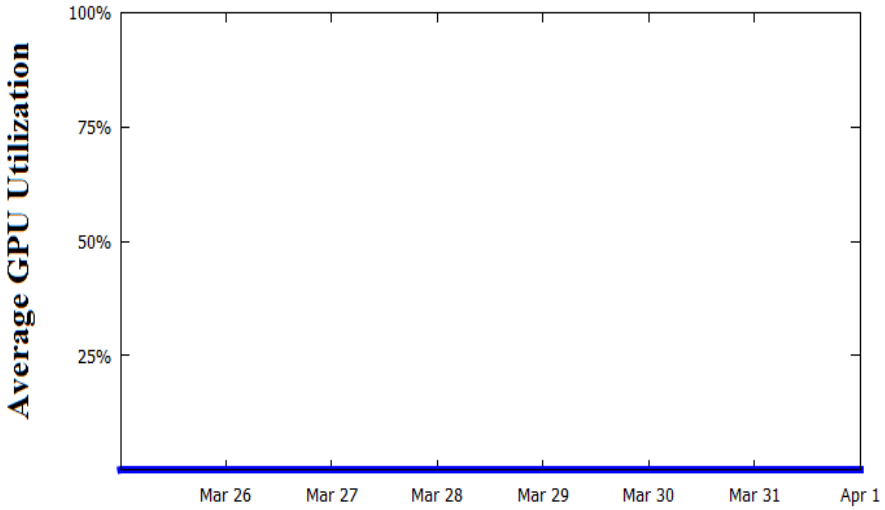

Figure 4.7: Average GPU Utilization

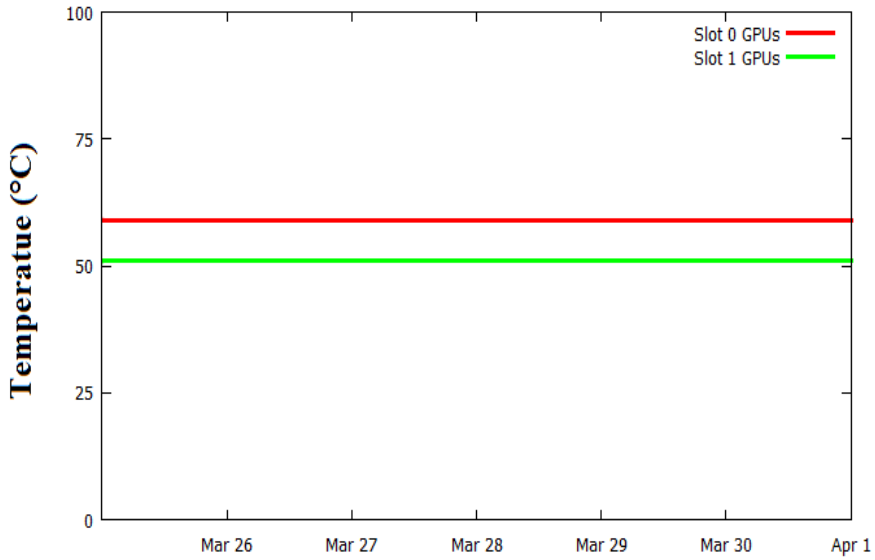

Figure 4.8: Average GPU Temperature

We see that despite the CPUs being allocated over the course of the week, the GPUs were completely unused. The potential reasons for this are many. Despite the unique capabilities that GPUs present that has caused their use to skyrocket among the computer science community, the domain scientists that represent the workload on Athena have not been utilizing them. This could be due to the difficulty in GPU based programming compared to

conventional programming. GPU data-parallel programming is a different paradigm that is often confusing, and the target audience for ARC systems, which is mostly nuclear engineers, chemical engineers, physicists, and mathematicians may be unfamiliar with the languages necessary to write GPU code. Regardless of the reason, the GPUs on the Athena GPU queue are not currently being utilized.

4.4 HokieSpeed Moab Trace

To compare the activity on the production Athena with the activity on the test environment of HokieSpeed, we collected a week long trace of user jobs using information gathered from the Moab scheduler running on HokieSpeed. As this is a user-space tool, it includes only statistics about CPU allocation and job runtimes. It is used primarily to draw a comparison about the relative levels of activity and types of jobs run on the two systems.

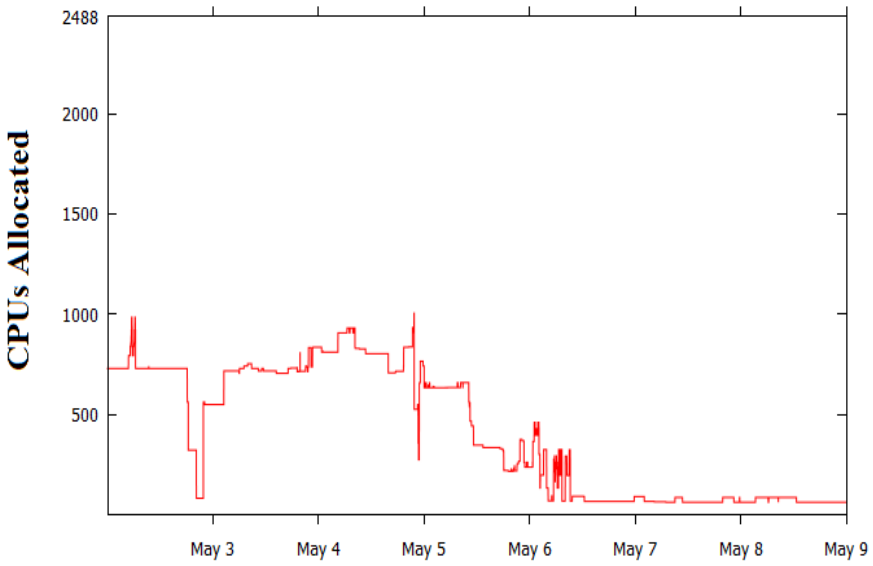

Figure 4.9: HokieSpeed CPU Allocation

Figure 4.9 shows the allocation of CPUs over the course of the week on HokieSpeed. Despite the test status of the system, it shows constant usage over the duration of the trace. During the first half of the trace, the allocation of the system rapidly changes, before settling down during the final three days. A total of 210 user jobs were recorded over the course of the week. Table 4.3 shows the statistics of the jobs broken down into the seconds, minutes, and hours job categories.

Table 4.3: HokieSpeed Job Statistics

Job Category	# of Jobs	Runtime (sec)	Processors
Seconds	75	120, σ=105	20.8, σ=47.0
Minutes	76	960, σ=726	37.2, σ=57.9
Hours	59	114240, σ=122078	24.7, σ=38.2

Several of the job characteristics are very different from those on Athena. Jobs of all categories use significantly more processor cores than on Athena, particularly the hours jobs, with an average hours job on HokieSpeed using more than six times as many cores. The hours jobs on HokieSpeed are also much longer on average than on Athena, with an average runtime over seven times longer. This is interesting because the current walltime limit on HokieSpeed long queue is only 144 hours, compared to 300 hours on Athena's general queue.

Overall, the system saw consistent usage over the entire week. However, the system was lightly used, with less then 50% allocation over the entire duration of the trace. It remains to be seen if the system will continue to see such consistent usage once opened to the university community for usage.

Chapter 5

Workload Synthesis

5.1 Selecting a Candidate Timeframe

Athena has seen vastly different usage levels over the course of its lifetime. The first few months it was almost completely unused, and has varied wildly over the months after that. To provide a workload that is useful for testing simulated future system setups, we endeavored to select a period that showed the heaviest, consistent usage of Athena that was relatively recent, to best represent the hypothetical workload that a new system would face from current ARC users. Given the relatively low usage of ARC systems at present, this candidate workload would likely be used to test a potential setup smaller in scale than current ARC systems. It needs to be recent enough to accurately represent the needs of current ARC users, and heavy enough that it represents the peak workload a future system will reasonably have to endure. Towards that end, we selected the first two weeks in March 2012 of the Athena TORQUE logs from the general queue as the timeframe we would draw our data from.

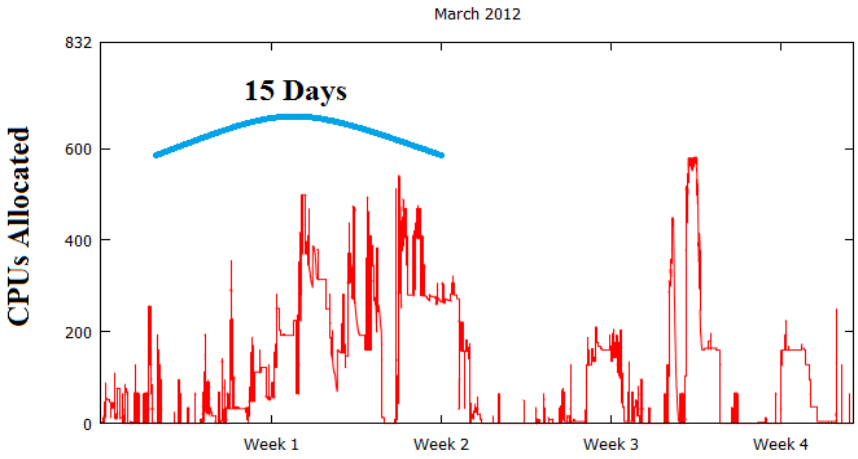

Figure 5.1: Timeframe selection for workload synthesis

5.2 Workload Statistics

We perform a analysis on the identified period similar to previously done on the logs as a whole. We separate the jobs into the same three categories, and calculate the average runtimes, processors allocated, and memory utilized. These statistics are summarized in Table 5.1.

Table 5.1: Timeframe Job Statistics

Job Category	# of Jobs	Runtime (sec)	Processors	Memory (GB)
Seconds	282	59.8, σ=65.6	30.7, σ=56.8	13.8, σ=100.6
Minutes	689	1533, σ=741.2	12.9, σ=31.8	12.3, σ=35.7
Hours	801	42057, σ=36608	4.98, σ=15.1	5.39, σ=29.36

To generate a workload that can actually be used to run simulations, we need more specific information. A figure of the rate of job submissions is needed to accurately reflect the stochastic nature of system usage. We break the 15 days of the selected time frame into 360 hour-long sections. We then calculate the average jobs submitted per hour for each of the three job categories. Additionally, as memory usage is likely correlated to the size of a particular job rather than being wholly independent, we calculate an average amount of memory consumed per core for each job category as well.

Table 5.2: Synthetic Workload Statistics

Job Category	Jobs/Hour	Runtime (sec)	Processors	Memory/Core (GB)
Seconds	.783, σ=2.3	59.8, σ=65.6	30.7, σ=56.8	.553, σ=3.98
Minutes	1.91, σ=13.31	1533, σ=741.2	12.9, σ=31.8	1.22, σ=.592
Hours	2.23, σ=9.92	42057, σ=36608	4.98, σ=15.1	1.48, σ=.535

These are all the statistics needed to create a synthetic workload of variable length for simulation-based testing of potential future system setups. As ARC procures new systems on a semi-regular basis, and these systems are an enormous investment of time and money, having the capability to determine how well they would service our unique workload is immensely valuable.

Chapter 6

Policy Recommendations

In this chapter we have compiled a selection of actions that could be undertaken to address some of the problems with the usage of ARC systems. Athena in particular is underused to the point where downsizing the system in some manner would save ARC and the university substantial amounts of money due to decreased power consumption and maintenance requirements.

6.1 Low Usage of Athena

As shown in section 4.1, the average allocation of resources on Athena is often very small when compared to the size of the system as a whole. Furthermore, the actual utilization of the allocated resources is often even smaller.

To account for the under allocation of resources, a subset of the system could be allocated to be running all the time, similar to how the entire system is currently operated. The remainder of the nodes would be brought online only when a user has a specific need for a large job.

Due to the under utilization of CPU resources by jobs on Athena, our workload serves as a prime candidate for hosting user jobs on virtual machines. Multiple virtual machines could be run on each node, increase the overall utilization of each node. Individual jobs are unlikely to see an impact on performance due to their low utilization of the resources allocated to them. This would also serve to alleviate the decreased maximum of available resources that would be caused by only having a subsection of the cluster active around the clock. Additionally, as demand for resources rises or falls, more nodes could be brought online, and user jobs could be live-migrated between them by freezing their virtual machines, moving them to the newly active nodes, and thawing them.

6.2 Utilization of GPUs on Athena

As shown in section 4.2, while the GPU queue is being used for its conventional CPU resources, the GPUs on are currently completely unused. With the looming release of HokieSpeed to general usage, it is important that ARC attempts to ascertain why users are not using them, and to educate users on how to use them.

While the ARC website currently has resources about basic use of NVidia CUDA, many users are likely unfamiliar with its use due to the relative novelty of GPGPU programming, particularly to domain scientists. It is our recommendation that ARC hold seminars to teach interested users the basics of CUDA or OpenCL programming. The purpose of these seminars would be two-fold. Firstly, they would provide the resources for users to get started in developing or porting their applications to utilize the GPU resources available to them. Secondly, it would allow ARC to gauge the interest among domain scientists in GPGPU computing. If a demand for GPU resources does not exist in the research community, that is useful information for provisioning future systems.

A second thing that could be done is moving existing programs that ARC currently supports on conventional CPU resources to make use of GPUs. Matlab in particular is a prime candidate for this. Matlab jobs constitute a significant percentage of the workload on ARC systems, to the point that there exists a dedicated queue for Matlab jobs on Ithaca, another of ARC's clusters. There exist easily compatible libraries for Matlab GPU programming that don't require knowledge of the specifics of GPU programming to use [5].

6.3 Provisioning Future Systems

Based on the observed characteristics of the workload present on Athena, we would recommend that future systems are scaled down from the current trend of larger and larger systems. A system as small as Athena is not currently fully being used, and more recent systems such as BlueRidge or HokieSpeed are several times larger. ARC's users would be better served by clusters with fewer nodes with fewer, faster processor cores. Given the low utilization of user jobs, this would still allow the virtual machine solution mentioned earlier to be run on top of such a system. Memory usage is roughly in line with the provisioning of 2 GB per core on Athena. However, newer systems such as BlueRidge have an even higher memory provisioning 5.3 GB per core. Barring a drastically different workload, it is unlikely this extra memory will see much use.

Chapter 7

Conclusion

7.1 Contribution

In this paper we have collected and analyzed three traces of the workload on production level supercomputing systems. We have analyzed these traces to show the characteristics of the workload present on supercomputing resources used by domain scientists. We have created a synthetic workload based on these traces that is characteristic of production systems in use by research scientists. This workload can be used for simulation based testing of future systems for a similar user base. Specific to the situation of ARC at Virginia Tech, we have made policy recommendations for increasing utilization of existing systems and resources, and for provisioning future systems to better suit our workload.

7.2 Future Work

In the future, we would like to take a deeper look into the other systems maintained by ARC, particularly HokieSpeed. The system admins that manage HokieSpeed were extremely hesitant to allow us unfettered access to the compute nodes themselves, which is why we were only able to gather basic statistics about job runtimes and processor usage. With the support of ARC, work could be done to deeper analyze the specifics of CPU and GPU utilization on HokieSpeed as it is released to the public. Additionally, an analysis of the workload on all of ARC's systems would show the relative rates of usage and utilization of the different architectures supported by ARC and how they are used by the research community.

Finally, we would like to take the synthetic workload we have developed and leverage it to run simulations to test future ARC systems. This would allow ARC to tailor future systems more closely to the specific needs of their users.

Bibliography

[1] A. Khasymski, M. M. Rafique, A. R. Butt, S. S. Vazhkudai, and D. S. Nikolopoulos. On the Use of GPUs in Realizing Cost-Effective Distributed RAID. In *Proceedings of the 20th Annual Meeting of the IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Washington D.C., August 2012.

[2] Adaptive Computing. Moab HPC Suite, `http://www.adaptivecomputing.com/products/hpc-products/moab-hpc-basic-edition/` Acessed May 2013

[3] Adaptive Computing. TORQUE Resource Manager, `http://www.adaptivecomputing.com/products/open-source/torque/` Accessed May 2013.

[4] Advanced Research Computing. Advanced Research Computing, `http://www.arc.vt.edu/resources/hpc/index.php` Accessed May 2013.

[5] Mathworks. MATLAB GPU Computing Support, `http://www.mathworks.com/discovery/matlab-gpu.html/` Accessed May 2013.

[6] F. J. Mesa-Martinez, E. K. Ardestani, and J. Renau. Characterizing Processor Thermal Behavior. In *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Pittsburgh, PA, March 2010.

[7] M. Moeng, S. Cho, and R. Melhem. Scalable Multi-Cache Simulation Using GPUs. In *Proceedings of the 19th Annual Meeting of the IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Singapore. July 2011.

[8] A. C. Murthy. Mumak: Map-Reduce Simulator. In *MAPREDUCE-728, Apache JIRA, 2009. [Online]*. Available: `http://issues.apache.org/jira/browse/MAPREDUCE-728`

[9] M. M. Rafique, A. R. Butt, and D. S. Nikolopoulos. Designing Accelerator-Based Distributed Systems for High Performance. In *Proceedings of the 10th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, Melbourne, Victoria, Australia. May 2010.

[10] G. Wang, A. R. Butt, P. Pandey, and K. Gupta. A Simulation Approach to Evaluating Design Decisions in MapReduce Setups. In *Proceedings of the 17th Annual Meeting of the IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Longdon, UK. September 2009.