SAND2015-2733C

Enabling Advanced Operational Analysis Through Multi-Subsystem Data Integration on Trinity

J. Brandt*, D. DeBonis*, A. Gentile*, J. Lujan†, C. Martin†,
D. Martinez*, S. Olivier*, K. Pedretti*, N. Taerat‡, and R. Velarde†
**Sandia National Laboratories*
Albuquerque, NM
Email:(brandt|ddeboni|gentile|dmart|slolivi|ktpedre)@sandia.gov
†*Los Alamos National Laboratory*
Los Alamos, NM
Email:(jewel|c_martin|ronv)@lanl.gov
†*Open Grid Computing*
Austin, TX
Email:narate@ogc.us

Operations management of the ACES Trinity platform will rely on data from a variety of sources including System Environment Data Collections (SEDC); node level information, including high speed network (HSN) performance counters and high fidelity energy measurements; scheduler/resource manager; and plant environmental facilities. The water-cooled Cray XC platform requires a cohesive way to manage both the facility infrastructure and the platform due to several critical dependencies. We present preliminary results from analysis of integrated data on the Trinity Application Readiness Testbed (ART) systems as it pertains to enabling advanced operational analysis through the understanding of operational behaviors, relationships, and outliers.

***Keywords*-High Performance Computing; Monitoring**

I. Introduction

High Performance Computing facilities operations at Los Alamos National Laboratories will be utilizing a combination of facilities and platform based environmental run-time data to manage Trinity's (Cray XC) physical infrastructure. Historically, HPC platforms (e.g. Road Runner and Cielo) and the physical plant supporting the platform have been managed independently. Platform internal environmental data such as voltages and temperatures had been accessible to the system administrators, but not facilities personnel.

The collection of a common set of data that includes both system and plant information becomes more critical with the water-cooled Cray XC platform. There must be a cohesive way to manage both the facility infrastructure and the platform due to several critical dependencies. Additionally, this platform will be the first on which we have the ability to measure fine-grained energy use at the node level. We are exploring utilizing this capability to characterize the energy footprints of applications and dynamically managing our peak power footprint through power aware scheduling.

In this work we utilize data from a variety of sources including System Environment Data Collections (SEDC), node level information, scheduler/resource manager, and plant environmental facilities. The SEDC data provides information about voltages, currents, and temperatures of a variety of components at the cabinet, blade, and node level. This data also includes water temperature, pressure and valve position information as well as dew point, humidity and air velocity. While the system utilizes many of these measurements to identify out of spec, and hence unhealthy, components it relies on fixed thresholds being crossed to trigger knowledge of an unhealthy situation. The node level information provides high fidelity energy measurements, OS level counters, and high speed network performance counters. Scheduler/resource manager information provides time windows and components associated with user applications. Plant environmental data provides us with coarse grained power draw, information about noise on the power feeds, and water temperatures and flow rates.

This paper describes the data elements, sources, collection frequencies and mechanisms for collection. It also describes the tools used and how the information is analyzed and utilized. We present preliminary analyses of SEDC data taken in conjunction with energy and application characterizations to identify the operational behavior of individual components. This type of analysis provides us with the ability to identify significant deviation of individual components from their particular normal operational behavior. It also enables identification of the range of operational behaviors for a component type across a system including

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

early identification of significant outliers. By tracking the operational behaviors of systems and components over time we will be able to evaluate the strength of their correlations with various failure types (e.g. application, component) and their causes (e.g. dirty power, aging component, water temperature/flow). High correlation coefficients can be utilized to identify actionable characteristics and mitigating actions.

II. Power, Thermal, and Facility Considerations

Modern HPC systems are comprised of tens of thousands of compute, network, memory, and storage elements that must all work together. Understanding the complex interplay between these elements, system software, and applications in the face of errors, failures, CPU throttling, and network congestion is increasingly necessary in order to continue to realize the increases in performance that is the driver for the increase in size and complexity.

Such understanding is of increasing importance as we continue to push the edge of the facilities supporting the HPC systems. Large scale HPC platforms can have a power draw in the 20MW range which can in turn stress a facilities power infrastructure. When the power demand exceeds certain thresholds the ability to prioritize and manage power allocations becomes essential. If the power draw drops to quickly it can cause facility and even site power outages. Thus active management of a platforms average and peak power draw through processor frequency management and power capping techniques has become a high priority for both HPC facilities and vendors and is currently a hot research topic. Additionally, the increase in power density of HPC components has necessitated the use of water based solutions for heat transport rather than traditional air cooling solutions. This in turn requires feedback mechanisms to maintain proper water temperature, pressure, and flow rates as well as active fan control in the case of hybrid solutions.

In order to maximize the value of delivered computing under aggregate facility constraints a more integrated management approach that tightly integrates facilities and platforms is required. The ability to control facilities infrastructure dynamically based on platform job, power, and environmental information will become necessary as we move towards exescale computing.

In this work then, we consider advanced operational analysis through data integration, with a particular eye to power, thermal, and facilities data.

III. System Configuration and Monitoring Setup

Los Alamos National Laboratory (LANL) and Sandia National Laboratories (SNL) purchased two identical Application Readiness Testbed (ART) systems Trinitite and Mutrino respectively. While most of the testing and measurements presented in this paper were performed on Trinitite, we will present some comparisons between behaviors of the two in Section VI. In this section we first describe the basic configuration of our (ART) systems. We also provide insight, at a high level, about our integrated monitoring philosophy, what information we are collecting, and our information aggregation and processing approaches. Finally we discuss the mechanisms used for transport of data from the various sources to a common system for processing. Note that this will become distributed on Trinity.

A. System Configuration

Our ART systems are single cabinet water cooled Cray XC40's and are populated with 100 compute nodes and 18 service nodes. The service nodes have the following functionalities: 1 logins, 6 burst buffer, 2 moms, 2 DVS, 3 Lnet routers, 2 sdb (1 failover) and 2 boot (1 failover). Additionally linux white boxes are provided for: external login, System Management Workstation (SMW), and Power Management Node (PMN). The PMN is currently being utilized as a `Monitor` host. All compute nodes and service nodes utilize the Cray Aries interconnect in a dragonfly configuration [1]. Each compute node is configured with dual 16 core Intel Haswell processors running at 2.3GHz and 64GB of memory. The external connections consist of FDR Infiniband to Lustre storage (Sonexion), and 40/10 Gigabit Ethernet to SMW, sdb, external login, and PMN. The public network connections are 10 Gigabit Ethernet.

B. Data Integration

As described in Section I our HPC platforms, both current and past, have had stove-piped monitoring operations where information rarely, if ever, crossed the boundaries of responsibility (e.g. facilities monitored power and cooling at a plant level and HPC system administrators monitored platform level variables. Only if it looked like the platform environment was causing problems would there be communications between the two groups about their respective data and how it was pertinent. This lack of communications is largely due to the relatively small adverse impact a platforms operational parameters have on the physical plant. As described in Section II upcoming pre- and exescale platforms will have the potential to incur large monitary cost and even cause site wide disruption to power if operated blindly. Additionally, with increasing scale is also coming greater heterogeneity in computational, storage, and networking technologies. The amount of information is becoming overwhelming and it will become impossible to efficiently manage these platforms without tools that perform run-time analysis on `all` available data continiously and take appropriate mitigating acction as soon as problems are detected. An example of the complex interplay that advanced analysis tools can be used to aid in understanding is variation in application performance due to any of: network congestion, contention for shared parallel file system bandwidth, contention for burst buffer bandwidth, automatic power capping, unintentional failure of power capping changes leaving some nodes

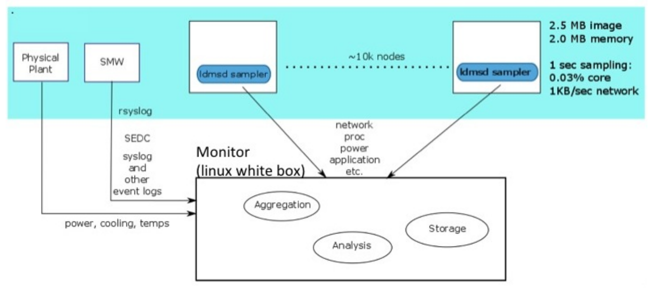


Figure 1. High Level Monitoring Diagram showing ART related information sources and their data feeds to a `Monitor` host

in a different state, thermally related CPU throttling, and working but faulty hardware/firmware. Any or all of the above mentioned conditions can cause wide variation in application performance but without simultaneous access to monitored system data, system logs, console logs, event logs, etc. correct diagnosis and problem resolution will be difficult if not impossible and manual exploration of this mass of information will become impossible.

Beyond just day to day problem analysis and resolution is the reality that we don't really understand the ageing process of many of the platform elements. Monitoring and archiving the data will enable us to discover how these elements age with use and how that process differs with differing environmental conditons.

C. *Monitoring Setup*

Figure 1 depicts the current platform components in the colored band and physical plant monitors all sending their data to a `Monitor` host that takes care of data aggregation, analysis, and both short and long term storage of the data. This scenario will change with the full system in that there will be several `Monitor` hosts for both scalability and redundancy with the 10K nodes depicted in the figure being an upper bound on what a `Monitor` host would be expected to process. As shown all monitored data will be aggregated to a set of `Monitor` hosts which will serve as data aggregation, storage, and both run-time analysis and post processing. This is the approach we have taken with data gathering for this paper where the `Monitor` box here is represented by the `PMN` block in Figure 2. The exceptions were that our facilities monitoring was performed out-of-band as described in Section IV-E and we supplimented the SEDC and power data as described in Sections IV-B and IV-C.

LDMS samplers are shown in Figure 2 running on every host (including login). Aggregators for this information run on `service` nodes (a login node in this case) and collect data at regular time intervals using RDMA to minimize compute node CPU overhead. Aggregators running on the `Monitor` hosts collect this data from the aggregators running on the `service` nodes and store the data. The

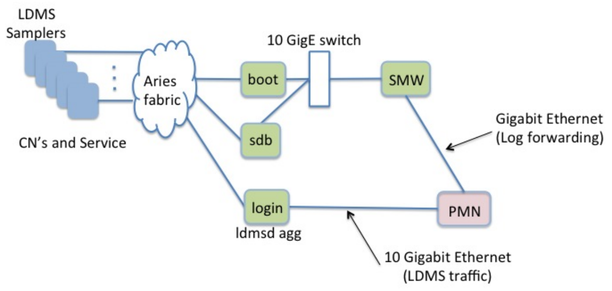


Figure 2. Application Readiness Testbed (ART) Connectivity Diagram showing how the major components of the platforms are interconnected

aggregators running on the `Monitor` hosts are also capable of doing analysis on the data as it is streaming through and will ultimately be able to provide notification of outlier behavior. Note that the out-of-band Hardware Supervisory System (HSS) network exists but is not shown. All log files and SEDC data are forwarded to the `Monitor` from the SMW. On the full Trinity platform each `Monitor` will be receiving a fraction of the SEDC data with appropriate redundancy for failure mitigation.

IV. Data Sources

This section describes the data sources we are currently collecting information from, how frequently it is being collected, where it is being aggregated, and how it is being stored.

A. *Logs*

Logging when errors or meaningful transistions occur to files is used by many subsystems as a means of providing system administrators and troubleshooters with diagnostic information. On the Cray XC system there are many sources of log information: syslog, console, power_management, smw, event, alps, and more. All of these logs are forwarded from various components to the SMW and placed in appropriate directories. In order to make them available for analysis in conjunction with the rest of the data we are collecting, we forward all log files to our `Monitor` host using `rsyslog`. This data path is depicted in Figure 1.

B. *System Environmental Data Collections (SEDC)*

Cray's System Environment Data Collections (SEDC) [2] provides a rich source of environmental data for many low level system components such as CPUs, memory, power supplies, nodes, blades, and more. The beauty of this information is that it is completely out-of-band with respect to

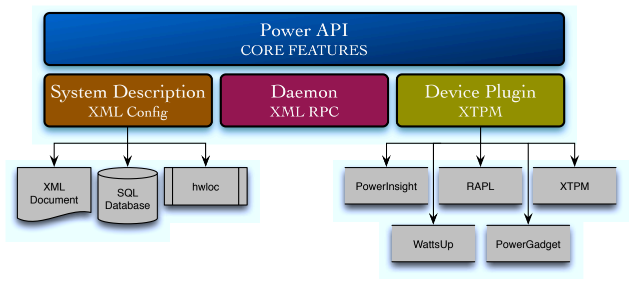


Figure 3. Framework of the Power API Prototype

node level computation and network communication. While the typical (default) configuration is to push the SEDC data to an aggregation point (the SMW) over the Hardware Supervisory System (HSS) network at 60 second intervals, we configured it to be pushed at 1 second intervals and configured `rsyslog` on the SMW to forward it to our `Monitor` host. One of the problems we see with the current configuration is that independent of platform size, all SEDC data is configured to go directly to the SMW. This presents a bottleneck to parallelizing the processing of this data. Ultimately we would like the ability to incorporate other devices, such as our `Monitor` hosts, into the HSS network and have the SEDC data distributed across them. When analyzing this data we discovered that data related to water (e.g., temperatures, pressures, flow rates) were missing from the SEDC stream. Upon inquiry it turns out that this is a known bug with a fix coming in CLE7.2UP03 [3].

C. Power API

We use the Power API prototype, which is a reference implementation of the Power API specification version 1.0[4] released to the HPC community in September of 2014, to collect node level data at 10Hz. The Power API specification describes a comprehensive system software API for interfacing with power measurement and control hardware. The specification defines the system model, theory of operations, and features exposed, covering the facility level down to low level software / hardware interfaces. The prototype supports most core features of the Power API specification. The prototype is a layered architecture that conforms to the specification and provides rich descriptive system configuration semantics, supports runtime plugins for a variety of devices and resources, and enables distributed communication for remote invocations of capabilities (see Figure 3).

For our study, we describe our system using a node level XML configuration file and utilize a plugin specifically created for the Cray platform which gains us access to the power management features of the system. The combination of the configuration file and XTPM plugin allow us to abstract the mechanisms of measurement and control from the specifics of the platform by mapping Power API attributes to the underlying plugin `sysfs` exposed parameters. We gathered data at a sample rate of 10Hz for each node of the system using this facility. In this case data was saved to a shared file system for post processing. In the future we will use LDMS (Section IV-D) to transport this data directly to the `Monitor` for run-time processing.

Note that while Cray provides power monitoring capabilities and a power management database (PMDB) for storing and querying power utilization data, these are inadequate for our purposes. The power monitoring capability collects at lower frequencies than we are interested in investigating, and even when higher frequency data can be obtained, it is limitedin its ability to collect for large number of nodes and long periods [5]. Additionally, the power database is located on the SMW, which has inherent limitations in access and size, while we seek to enable continuous, near-indefinate runtime and historical analysis integrated with other data sources. For these reasons, we do not include power management database as source of data in this work. For convenience, however, we do use RUR output for some general, relative, overall application energy utilization.

D. Lightweight Distributed Metric Service (LDMS)

We use LDMS [6] for node level data collection and transport via the High Speed Network (HSN). This information, collected at 1Hz, includes HSN performance counters, Lustre client activity, application memory utilization, and other counters exposed by the OS. As of this writing LDMS collects node level energy data from the `sysfs` interface at 1Hz. The LDMS energy sampler plugin will be upgraded to collect 10Hz power/energy data via the Power API (see Section IV-C) thus making this full fidelity (10Hz) data available at 1Hz across the whole system. Use of the PowerAPI will enable LDMS to use the same power data collection mechanism independent of platform which will translate into needing to support fewer plugins and easier configuration.

The HSN performance counter data collection relies on Cray's `gpcdr` kernel module to expose this data via the `sysfs` interface. While this interface has been utilized for over a year on NCSA's Blue Waters platform (Cray XE/XK), this is our first use of it on a Cray XC. This exposed a small problem with the default `gpcdr` configuration which exposed all 160 counters via a single `sysfs` file. While this configuration is fine when the counter values are small, it causes the aggregate size to exceed the 4KB limit imposed on `sysfs` entries as the values become large. The author at Cray quickly diagnosed the root cause of our apparent counter corruption and gave us the fix which was to break the set into 4 smaller sets based on information type (traffic, stalls, receive link status, and send link status). This required only a slight change to the `gpcdr` configuration file and a reload of the kernel module. After the change we were able

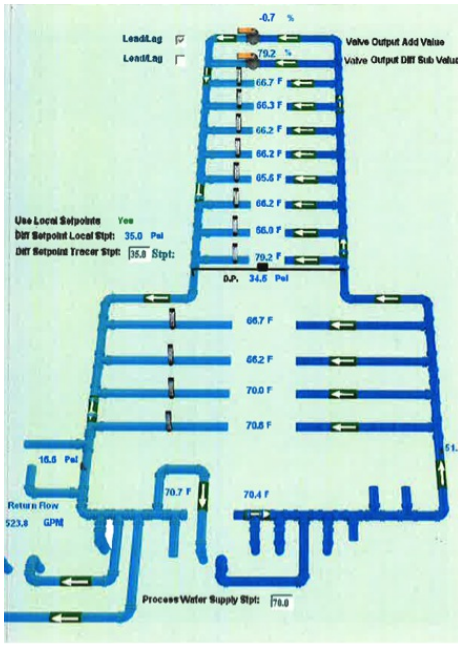


Figure 4. The Underfloor Pipe Diagram for Trinitite.

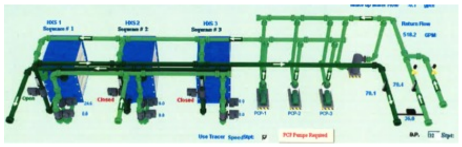


Figure 5. Trinitite Heat Exchangers and Pumps

to gather the following HSN data at 1Hz on a per-node basis: traffic, stalls, receive link status, send link status.

In addition, we collect the following non-HSN data via LDMS:

- Lustre file system counters
- CPU load averages
- Current free memory
- LNet traffic counters
- ipogif counters
- power and energy metrics via `sysfs`

E. Facilities

The facilities infrastructure that provides the cooling to Trinite is composed of two main loops: the primary loop and the secondary loop, these are separated by the heat exchangers. The primary loop consists of four cooling towers and three pumps. The secondary loop consists of the three heat exchangers and three pumps on variable frequency drives (VFD).

The facility water supply temperature is 45 degrees inlet to the heat exchangers The secondary loop water supply temperature to the machine and preconditioner is 75 degrees.

The building automation system that controls and monitors the facility cooling equipment is Trane. The Trinitite system and associated facility infrastructure installation was substantially completed in February and testing began in March. Not all of the building automation systems were operational for this testing time frame.

The facilities data collected for these tests were limited to power. Facilities power data was collected at five second intervals from the compute rack feeder breaker using a fluke meter 1730 Energy Logger.

F. EnvScript

Because the SEDC data stream currently does not contain the majority of the cooling water related data (see Section IV-B we have augmented our data collection using the envdata [7] script. From the SMW this script collects water-related data directly from the cabinet. Currently, We are calling the script at 10 second intervals and storing the output to disk for post processing. Note that this is a stop-gap solution which will be discontinued once this data is included in the SEDC stream.

V. Workload Description

For this work, we developed a work package that would exercise the machine under a variety of conditions. Multiple iterations of single and multiple node runs of a combustion code, hpl, and hpcg were run as a group. This group was first run normally, and then in turbo mode. These repeated cases we refer to as a `Series`. This `Series` was then run 3 times, once under each of the following conditions: 1) no power capping (415 Watts), 2) 50 percent node-level power capping (322 Watts), and 3) 0 percent node level power capping (230 Watts).

HPL is the MPI implementation of the high performance Linpack benchmark [8] A highly regular dense LU factorization, HPL is computationally intensive. High Performance Conjugate Gradient (HPCG) [9] is a very different benchmark, by design. HPCG comprises operations such as sparse matrix-vector products that stress the memory subsystem and network communications, and its performance may be uninhibited by modest reductions in CPU resources or frequency. HPL and HPCG represent extremes in the spectrum of applications, from compute-bound to memory-bound, with orders-of-magnitude differences in Flops. In June 2014, Tianhe-2 reported the top numbers in both benchmarks, with 33.9 Pflops on HPL but only 0.58 Pflops on HPCG [10]. In addition to these benchmarks, we also included in our system evaluation an application code for direct numerical simulations of turbulent combustion. Its uses an explicit Runga-Kutta method with mostly nearest neighbor communications.

An idea of these 3 run `Series` and their time line can be gotten from Figure 7 (top). Each of the three `Series` and their power cap status is marked. A `Series` runs for 2-3 hours. Within a `Series`, it can be seen that there is generally similar behavior repeated in the set. More details of the timeline within a set is marked in Figure 15(top).

VI. Analysis and Results

In this section we first present the analysis methodologies used in this work and planned. We then present both visual and analytic results of interest derived from data taken over a two day period while running the workload described in Section V. The results section first discusses power related understanding obtained in our testing thus far. We then present cooling related information from the machine perspective only as our facilities data collection system is still being installed. Finally we show traffic and congestion related metrics and briefly discuss their utility in understanding performance variation as we move to larger scale systems.

In this work we combine information from log and numeric data sources from both facilities and our ART platforms to accentuate issues facing all large scale HPC host sites as we move to pre- and exascale platforms.

A. Analysis Methodologies

Our ultimate goal is to understand sub-systems and their relationships and characterize system behavior in order to more optimally use the machine and to diagnose issues.

In this work we consider several methodologies for the analysis of data. These are highlighted below, with specific application in the following sections. It is important to note that for the material discussed here, the complete understanding relies on the integration of data from a variety of sources and from a combination of analysis methods.

1) Log Analysis: Baler [11] is a log message processing tool that extracts patterns from message streams, where a pattern is deterministically extracted from each message by marking pre-defined known words (like words in the English dictionary) as static fields, and unknown words as variable fields–representing by Kleen's star (*)–in the pattern. Baler differs from other log clustering tools in its determinism as it does not rely on message population to determine static/variable fields like many other clustering tools.

We use Baler in this work to process all system logs, browse for interesting patterns, count the occurrences of patterns in time-node space, and generate plot files for visualizing where and when events of interest occur. Some patterns discovered by Baler relevant to this work are presented in Figure 6. It is interesting to note that pattern 283 (power budget exceeded), was discovered in the smwmessages log and not in the power_management log.

In this case there were 1.8 million lines of log files that Baler distilled into 251 unique patterns. This reduction enables a system administrator to easily browse or grep for key issues in a managable list and then drill down on a time interval or node set for greater understanding. An example of the use of pattern occurrences in node-time space

```
Example patterns:

280 * * - -  Node * interrupt *=*, *=*, *=* *[*]: * * *
    Processor Hot

283 * * - -  Node * power budget exceeded! Power=*,
    Limit=*, * Correction Time=*


Example messages corresponding to patterns 280
and 283 respectively:

bcsysd 2080 - -  Node 2 interrupt IREQ=0x20000,
    USRA=0x0, USRB=0x80 USRB[7]: C0_PROCHOT CPU 0
    Processor Hot

bcpmd 2140 - -  Node 2 power budget exceeded!
    Power=340, Limit=322, Max Correction Time=6
```

Figure 6. Example of log message patterns and their corresponding messages.

for visualizing when and where particular events occur is presented in Figure 10. The visualization is of the example patterns shown in Figure 6.

2) Numeric and Visual Analysis: In addition to the log analysis, we use visual intergrations of data and visual and simple numerical analyses. We examine data time-series in relation to events in order to get an overview and understanding of system and variable behaviors and to discover and to hone in on situtations of interest. We examine data in the physical machine layout in order to detect physical system relationships Finally we consider numerical analysis in order to determine abnormal behaviors.

B. Perspectives in Power Use and Performance: Facilities vs. Machine

In order to perform effective power management at a HPC platform level it is necessary to understand the relationships between the platform view of power draw and that of the physical plant (`facilities`). Thus we not only collected power and energy data from a variety of sources on Trinitite (our ART platform under test) including in-band on the compute nodes at 10Hz, via SEDC at 1Hz, and energy from Cray's Resource Utilization Reporting (RUR) facility. But we also collected the `facilities` view (what really matters) using the methods described in Section IV-E. Figure 7 (top) shows a combination of three power views. PowerS_Total_avg (blue) is plotted as a time series of 5 second averages of the total for PowerS (this is the complex sum of PowerS over all three power phases and represents what the Utility company sees and bills according to). PowerS_Total_Max (green) represents the maximum power draw over the past 5 second window. The platform view of the average power draw over the previous 1 second window is represented by Rectifier Total PO (red). The discrepency between average power and peak power over the time our applications (Section V) were being run on Trinitite varies as the amount of fluctuation of power draw by the compute nodes (see Figure 8. As can

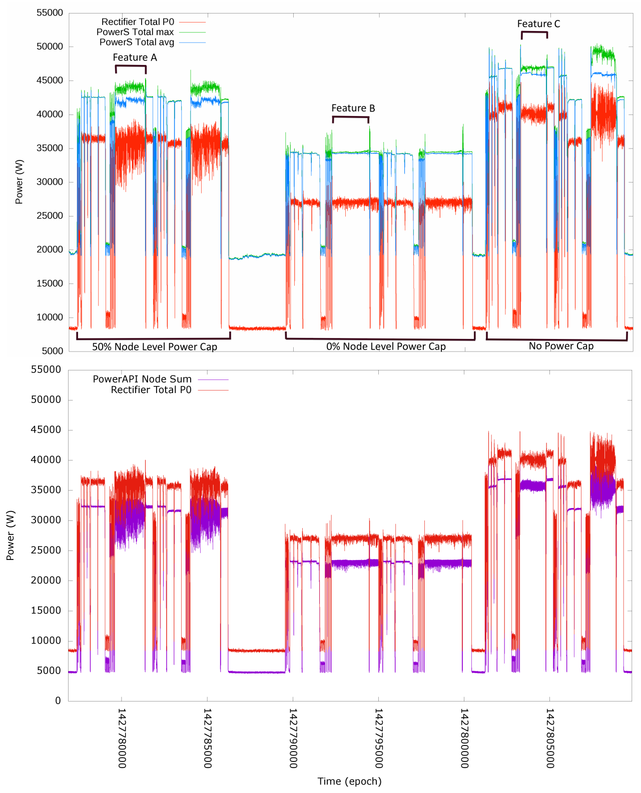


Figure 7. Power information from Facilities (greeen, blue), System (red), and Node Level data (purple)

be seen in the figure the platform's view of power draw is about 20% lower than the facilities view. The rectifiers and powersupplies require power. This power is not recorded in the SEDC data. This must of course be taken into account by the power management software. Additionally the power factor variation can skew this discrepency. Under normal conditions (85 percent utilization) the power is satifactory; however if the utilization runs below these parameters then there impact to platform calculating the appropriate power cap? while the the system is idle it's power factor will decrease which will result in power inefficiencies that must accounted for on facilities infrastructure.

We call out Features A, B, and C in this figure as time windows of interest over which we also plot 10Hz power data taken from the 100 compute nodes during a series of application runs. The reason for our choice of these regions is that they are wide enough an stable enough to be able to discern obvious differences in the behavioral characteristics of facilities and machine based perspectives on power both within a region and between similar regions where power capping is the difference. Additionally the behaviors of the compute nodes with respect to the power caps for each region are clear (See Figures 8 and 9).

Figure 7 (bottom) shows compute node power data collected at 10 Hz and summed over all nodes for each 100ms period (purple) vs. the same Rectifier Total PO (red) plotted on the top trace. As expected the summed power draw as seen by the compute nodes is less than the total for the platform (by about 10%). Note that this system is only 2/3rds populated so that number will change on a fully populated rack.

In this set of application runs there was a node (nid00176) that did not change its power cap correctly and remained uncapped the whole time. Figure 9 shows a plot of nid00176 with another compute node plotted for comparison over region B. The failure to change caps was reported in the logs and would need to be taken into account by the resource manager in order to do appropriate power management on a large scale system where many such failures would be expected. While nid00176 was stuck in the no-cap state in this case, it seems equally probable that a node could get stuck in a lower power cap state. In such a case, if the resource manager naively handed the node out to a job that was running with no-cap the reduced performance of this node would adversly impact the whole job and could even cost more energy overall.

The plots in Figure 8 show an overlay of time-history plots of the 10Hz power data collected, using PowerAPI, over all compute nodes during time intervals labeled as Feature A, B, and C in Figure 7. These plots correspond to: 50 percent node level power cap (top) (Feature A in Figure 7 (top)), 0 percent node level power cap (middle) (Feature B in Figure 7 (top)), no power cap (bottom) (Feature C in Figure 7 (top)). The maximum power defined by the cap level is shown as a horizontal yellow line in each figure. The 10Hz data shows values that exceed the cap. Note that nid00176, which did not respond to the cap command is not included in the figures but is shown seperately in Figure 9. Generally, facilities have to account for the fact that power capping is not an absolute.

It is interesting to note that the noisiest of the three features is A in which there is a 50% power cap and in which compute nodes regularly exceed the cap by up to 25%. This can also be seen in Figure 10 which shows many more "power budget exceeded" log message occurences (red) than in the 0% cap region. In fact in region A the compute nodes regularly spike to the same levels seen in region C which has no cap though Figure 7 (top) clearly shows both the average and peak from a facilities are higher (and less noisey).

Figure 10 is a heat map of Baler [11] patterns representing "power budget exceeded" (pattern 283) messages (red) and "processor hot" (pattern 280) messages (green). The P283 messages are mostly seen during our application runs at a 50% power cap while the P280 messages were all during no-cap application runs. Note that though the P280 messages indicate a hot processor, there were no corresponding throttle events.

We performed single node experiments in order to better understand the node to node performance variability that

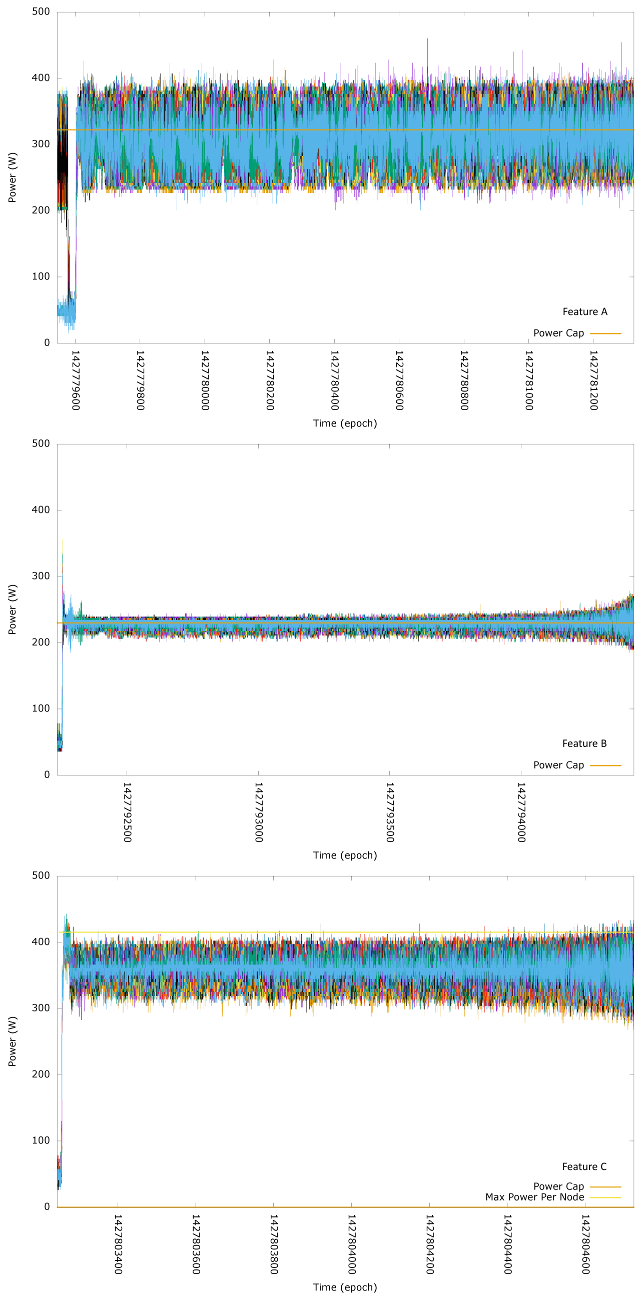


Figure 8. Plots of compute nodes 10Hz power profiles for the cases of 50% (Feature A), 0% (Feature B), and no (Feature C) power cap from top to bottom respectively

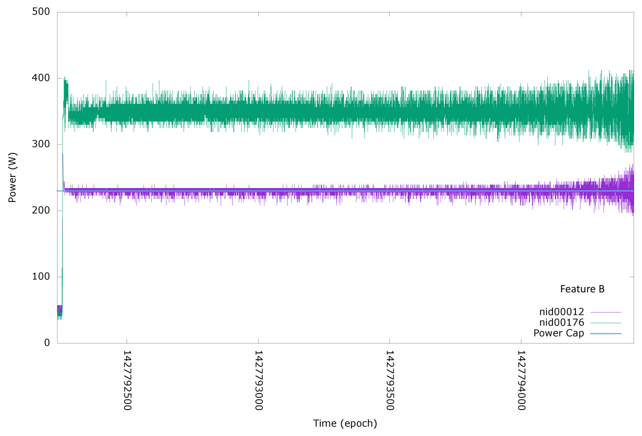


Figure 9. Non-capped behavior of nid00176 vs an arbitary nid with 0% cap (Feature B) (would have similar general behavior in the other two cases as well).

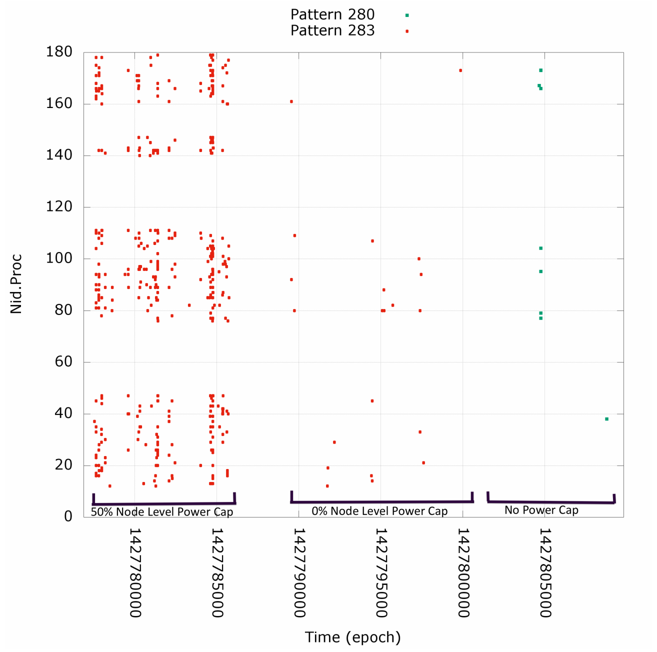


Figure 10. Baler error patterns relating to power (280 - Processor Hot, 283 - power budget exceeded, see Figure 6 for full patterns and example messages).

occurs when operating under a power cap. Due to part to part manufacturing variability, the power required to operate at a given performance level will be different from processor to processor, and hence node to node. Operating under a power cap should fix the maximum power used by a node, at least in theory, while allowing performance to vary. This is in contrast to setting a P-state, which results in a relatively fixed performance level across nodes, but with a variable power usage.

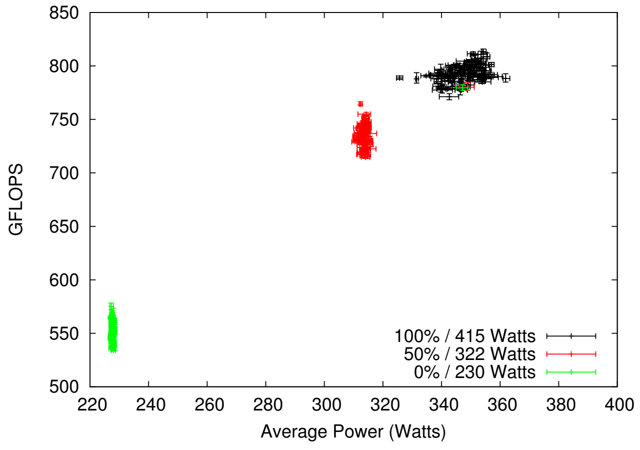
For these experiments, we ran single node instances HPL and HPCG on each of the 100 compute nodes in Trinitite.

For each run we recorded the performance of the benchmark as reported in the benchmark's normal output, as well as the average power used by the run over its entire execution, as reported by Cray's power measurement infrastructure. Each benchmark was run five times on each node for each of the three power cap configurations tested, 0%, 50% and 100%. Additionally, we tested with Intel's turbo boost feature on and off for each power cap configuration. For the Intel Xeon E5-2698 v3 (Haswell) processors used on Trinitite, the base non-turbo frequency is 2.3 GHz. Enabling turbo boost allows the processor's frequency to scale up to 3.6 GHz based on the number of cores being used and thermal headroom.

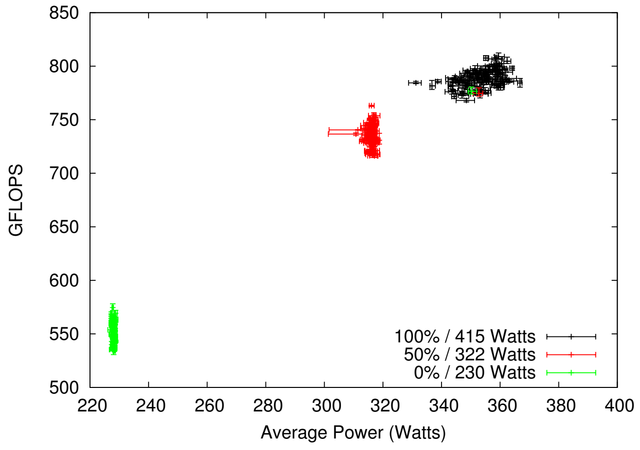
Results for these single node experiments are plotted in Figures 11 and 12, for HPL and HPCG respectively. In these plots each point represents the averagve value for the five trials and error bars (on both x and y axis) represent the minimum and maximum values recorded. HPL in the 100% power cap configuration (no power cap) results in a node-to-node performance spread of 767 to 812 GFLOPS and a power spread of 331 W to 367 W. Since there is no cap in this configuration, each node is operating at the maximum speed it is able to, which depends on the energy efficiency of the processors in the nodes, environmental conditions, and other factors. In contrast, the 50% and 0% power cap levels result in a more vertical profile, indicating that the power cap is being hit. Each node uses as much power as it can, up to the power cap limit, and achieves a performance level based on the energy efficiency of the particular processors used in the node.

HPCG results, shown in Figure 12 show a much narrower band of performance for the different power cap levels. The 100% and 50% configurations result in essentially the same performance levels, indicating that the power cap is not being reached. The 0% configuration leads to a sharply vertical profile due to the power cap being reached, and performance drops by approximately 17% compared to the other configurations. The HPCG 100% turbo on configuration is interesting because the processor is choosing to operate at a higher power level, even though it does not result in improved performance. This indicates that the processor's power management policy could be improved for HPCG, and likely other memory bandwidth bound codes as well (e.g., operate at the lowest frequency needed to saturate the memory subsystem).

Figures 13 and 14 show histogram distributions of energy utilzation over the course of the 0% and no-cap cases shown just discussed in Figure 12. The number of nodes is binned by percent of the average per-node energy for the application run (excluding nid00176 due to problems setting its cap) with the bin width being 1%. Both distributions look relatively normal but given the spread, a power aware resource manager should be able to take advantage of this when assigning nodes to applications. Knowledge of the run-time characteristics of an application could increase this advantage.



(a) No Turbo



(b) Turbo On

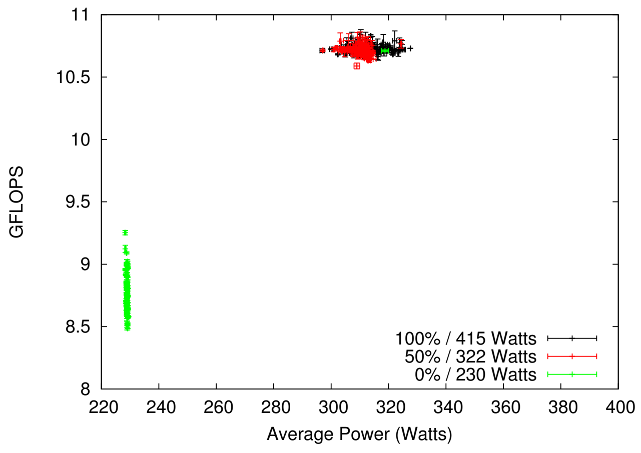
Figure 11. Performance vs. power for HPL with different power caps and turbo off (top) and turbo on (bottom).

C. *Machine Environmental Data and Facilities Interest*

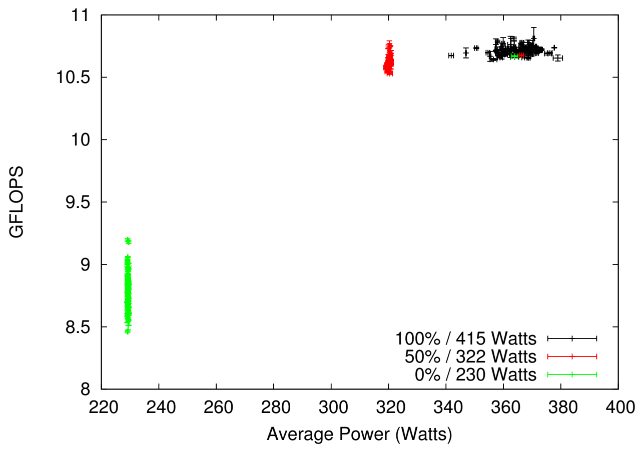
In this section we examine the cooling environmental data associated with the platform. This section delves into the platform facility information and explores application to gain facility infrastructure efficiencies.In this section data has been presented from both Mutrino and Trinitite for comparison. Time ranges associated with applications during the work package in the non-turbo phase are marked.

Figure 15 Valves open in reponse to the the job (top). This set of data can be a useful indicator of facility room conditions. The preconditioner valve position is calculated based on the inlet ambient temperature and dew point and the cabinet valve position maintains room neutral discharge air.

The water line pressure varies during normal operation (middle). When the valve opens this results in lower pressure. This data can be used to determine pressure differential



(a) No Turbo



(b) Turbo On

Figure 12. Performance vs. power for HPGC with different power caps and turbo off (top) and turbo on (bottom).

control and calculate pump horsepower. This type of information could be used as input to facility building automation systems and drive the variable frequency drive (VFD) that operate the pumps.

Water outlet temp is lower during the job because of the valve open (bottom). Facility personnel use these inlet and outlet temperatures to ensure they 75 degree water specification has not been exceeded. Monitoring the delta temperature helps to validate the coil is working effectively.

In a tightly integrated system there is potential for automated control of pumps based on feedback from the platform environmental data. Automated control of the facility infrastructure based on highly integrated platform and facilities data would allow for enormous efficiency gains in the plant.

Figures 16 is a comparison Figures for Mutrino, although, not under the same conditions. This is just going into an HPL job (started at 8:45), but it does show some differences due to the water temperatures and differences in the high speed

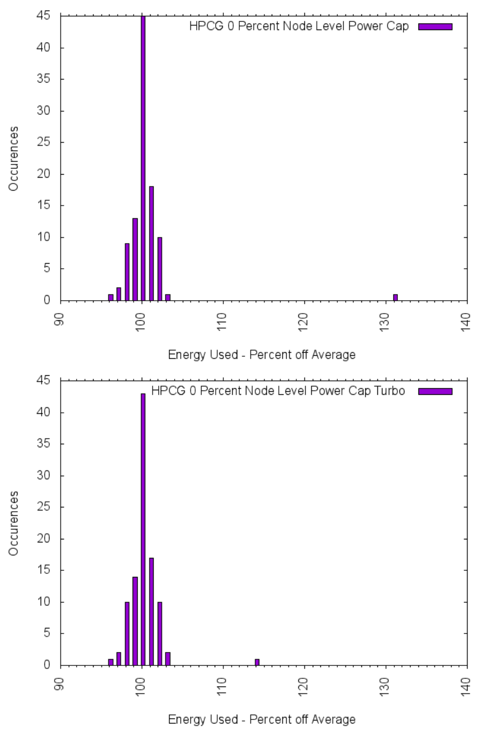


Figure 13. Distribution of number of nids vs. energy binned by % of the average energy (average does not include nid00176) for one set of the single nid runs hpcg with and w/o turbo and 0% Power capping

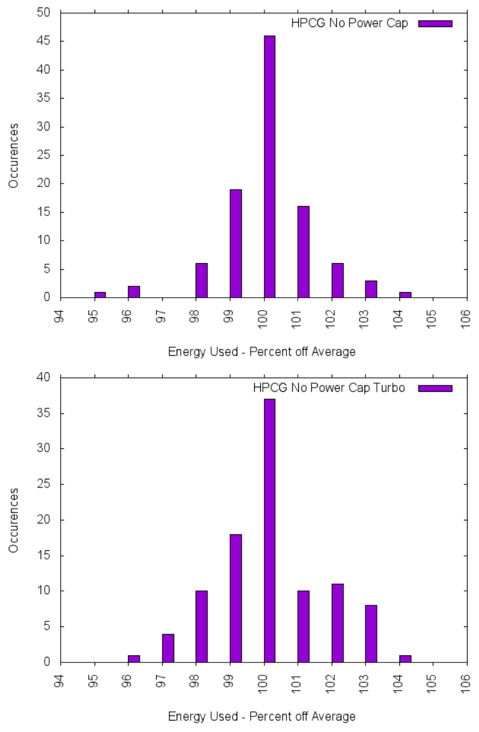


Figure 14. Distribution of number of nids vs. energy binned by % of the average energy (average does not include nid00176) for one set of the single nid runs hpcg with and w/o turbo and no Power capping

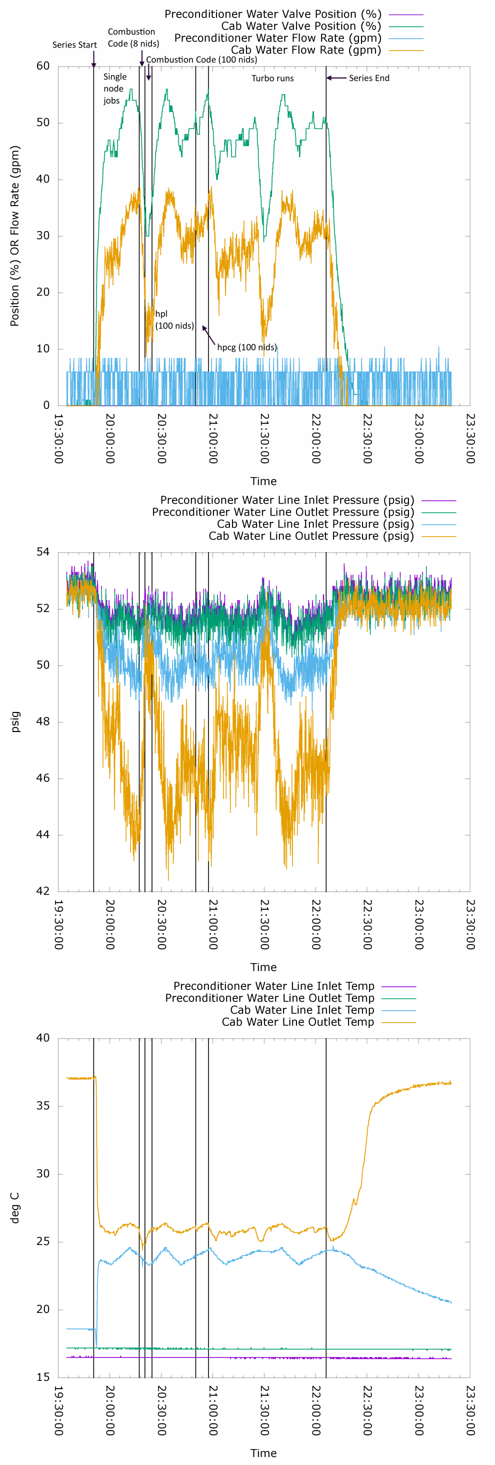


Figure 15. Trinitite data over the Series: Flow Rates and Valves (top), Water Line Pressures (middle), Water Outlet Temp (bottom).

blowers.

In order to compensate for a higher altitude Trinitite was equiped with high speed fans. This data is used to proactively predict fan failures. The RPMs decrease as the fan reaches end of life. As can be seen in Figure 17 the Trinitite fans are set at constant 75 percent of max speed. Facilities data also shows that for both Mutrino and the fan power does not change. The issues seen with Trinitite have been identified as a software problem.

D. *Thermal Irregularities*

Thermal issues are of interest for a number of reasons, including facilities cooling, performance impact, and device degredation and aging. Issues in the machine and the room environment may be causes of such irregularities.

Numerical analysis of the SEDC data shows significant temperature variation across the CPU's of both systems when each was independently supposed to be running a similar workload across the nodes. Visual analysis of the numeric data in a physical layout gives insight into this issue, as well as giving rise to further investigation.

They layout on the board is generally as follows. On a blade, Nodes 2,1,3,0 are in that order, front to back. There are 2 CPU's per node, with CPU's 0/1 alternating left/right with each node. For clarity, then, Node 2 is in front with CPU 0 on the left.

Within the rack, chassis are vertically stacked. Two slots are populated left and right in a chassis.

Note that our ART systems are not fully populated. Figure 18 shows the layout for the two machines (Trinitite (top), Mutrino (bottom)). Non-compute nodes and their associated CPU's are indicated by `XX/XX`, with the exception of the c0-0s12n0, circled in the upper right of each layout.

Maximum temperatures over the workload for each case are shown. The workloads were not the same for each machine. For Trinitite, the workload was the entire set of runs discussed in this work. For Mutrino, the workload was the entire HPL run (approx 3 hours) pertaining to the Mutrino figures in Section VI-C. While it is not expected for the values to be comparable, certain similarity occur in each.

Overall, there is a significant variation in temperature (25-30 degrees) which could lead to different temperature-related aging issues that could affect the lifetime and performance of the nodes. The hotter nids are seen to be those for which the left and right slots are both populated. In addition, c0-0c2s12n0 has issues in *both* systems: In Mutrino the node exhibits temperature related throttling, in Trinitite the SEDC data included error codes for all attempts at collecting temperature related data for this run and in the log data this nid was the only nid reporting an error when attempting to apply the power capping profiles.

As a result we seek further understanding of the common positional dependence of the problem nid, of the overall

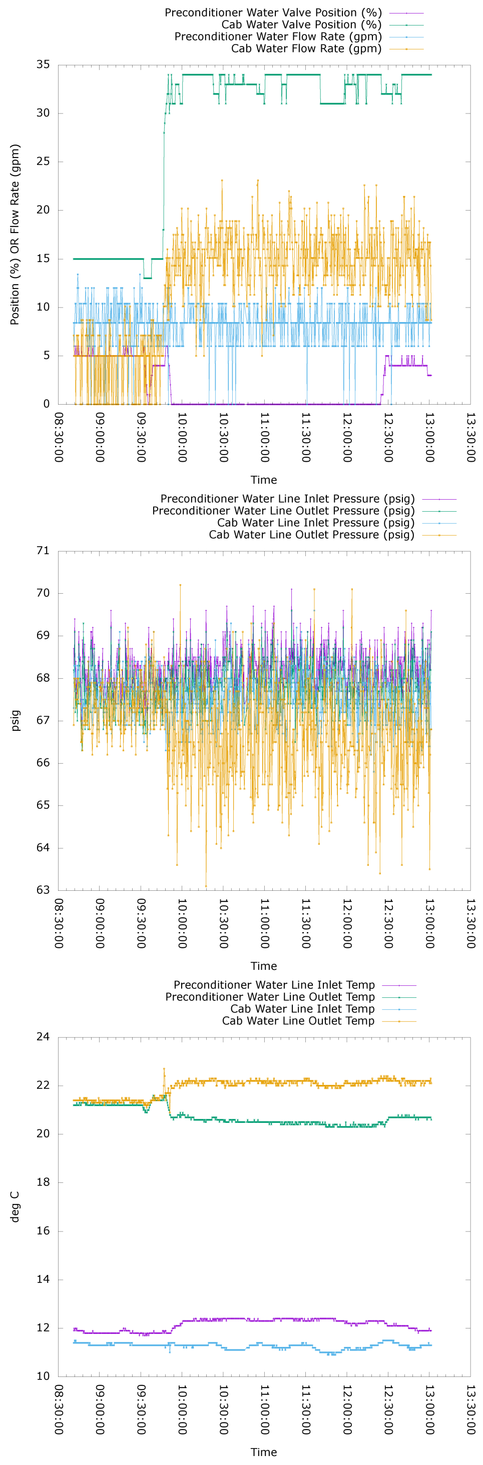


Figure 16. Mutrino data for comparison, going into an HPL job: Flow Rates and Valves. Preconditioner water valve position is always 0.

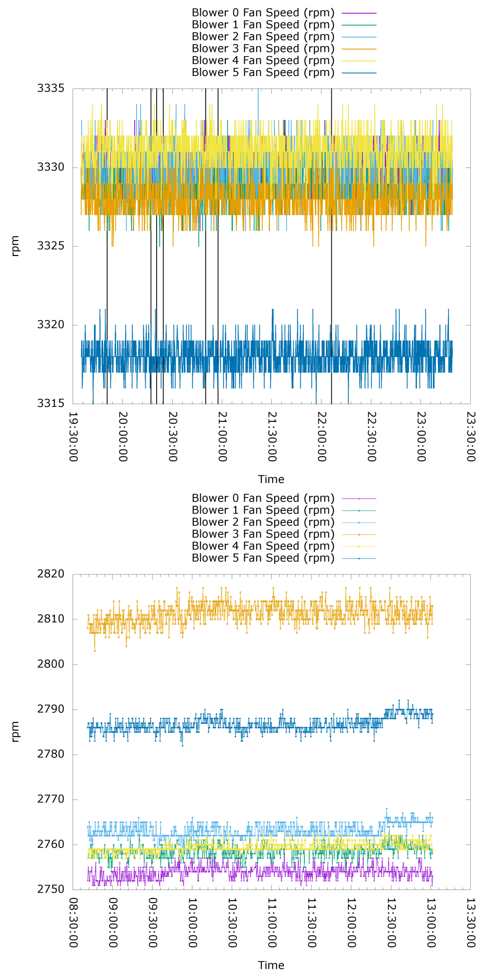


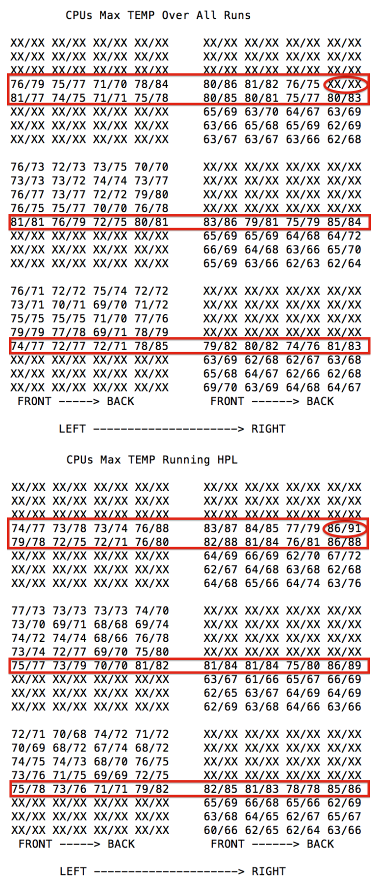
Figure 17. Blower Speed: Trinitite data over the Series (top). Mutrino going into an HPL job(bottom)

expectations of temperature within the partially populated racks, and of how we can expect the results to extrapolate to fully populated systems.

E. Network

While the workload did not target investigation of network performance and Aries enviromental data profiling, understanding when contention for network resources (congestion) is affecting performance can aid in root cause analysis of performance variability and help in optimization of job placement that minimizes congestion. We briefly present a first look at this data in the form of link bandwidth used and associated stalls (a measure of congestion).

Figure 19 plots network traffic (top) and stalls (middle) data for each of the 40 Aries router tiles. This data was gathered from the gpcdr interface via LDMS (Section IV-D) during the course of the same application runs previously presented. As expected there are no traffic or stall values during the single node jobs and highest values occur during



```
         CPUs Max TEMP Over All Runs

XX/XX XX/XX XX/XX XX/XX     XX/XX XX/XX XX/XX XX/XX
XX/XX XX/XX XX/XX XX/XX     XX/XX XX/XX XX/XX XX/XX
XX/XX XX/XX XX/XX XX/XX     XX/XX XX/XX XX/XX XX/XX
76/79 75/77 71/70 78/84     80/86 81/82 76/75 XX/XX
81/77 74/75 71/71 75/78     80/85 80/81 75/77 80/83
XX/XX XX/XX XX/XX XX/XX     65/69 63/70 64/67 63/69
XX/XX XX/XX XX/XX XX/XX     63/66 65/68 65/69 62/69
XX/XX XX/XX XX/XX XX/XX     63/67 63/67 63/66 62/68

76/73 72/73 73/75 70/70     XX/XX XX/XX XX/XX XX/XX
73/73 73/72 74/74 73/77     XX/XX XX/XX XX/XX XX/XX
76/77 73/77 72/72 79/80     XX/XX XX/XX XX/XX XX/XX
76/75 75/77 70/70 76/78     XX/XX XX/XX XX/XX XX/XX
81/81 76/79 72/75 80/81     83/86 79/81 75/79 85/84
XX/XX XX/XX XX/XX XX/XX     65/69 65/69 64/68 64/72
XX/XX XX/XX XX/XX XX/XX     66/69 64/68 63/66 65/70
XX/XX XX/XX XX/XX XX/XX     65/69 63/66 62/63 62/64

76/71 72/72 75/74 72/72     XX/XX XX/XX XX/XX XX/XX
73/71 70/71 69/70 71/72     XX/XX XX/XX XX/XX XX/XX
75/75 75/75 71/70 77/76     XX/XX XX/XX XX/XX XX/XX
79/79 77/78 69/71 78/79     XX/XX XX/XX XX/XX XX/XX
74/77 72/77 72/71 78/85     79/82 80/82 74/76 81/83
XX/XX XX/XX XX/XX XX/XX     63/69 62/68 62/67 63/68
XX/XX XX/XX XX/XX XX/XX     65/68 64/67 62/66 62/68
XX/XX XX/XX XX/XX XX/XX     69/70 63/69 64/68 64/67
 FRONT -----> BACK           FRONT ------> BACK

        LEFT ---------------------> RIGHT

         CPUs Max TEMP Running HPL

XX/XX XX/XX XX/XX XX/XX     XX/XX XX/XX XX/XX XX/XX
XX/XX XX/XX XX/XX XX/XX     XX/XX XX/XX XX/XX XX/XX
XX/XX XX/XX XX/XX XX/XX     XX/XX XX/XX XX/XX XX/XX
74/77 73/78 73/74 76/88     83/87 84/85 77/79 86/91
79/78 72/75 72/71 76/80     82/88 81/84 76/81 86/88
XX/XX XX/XX XX/XX XX/XX     64/69 66/69 62/70 67/72
XX/XX XX/XX XX/XX XX/XX     62/67 64/68 63/68 62/68
XX/XX XX/XX XX/XX XX/XX     64/68 65/66 64/74 63/76

77/73 73/73 73/73 74/70     XX/XX XX/XX XX/XX XX/XX
73/70 69/71 68/68 69/74     XX/XX XX/XX XX/XX XX/XX
74/72 74/74 68/66 76/78     XX/XX XX/XX XX/XX XX/XX
73/74 72/77 69/70 75/80     XX/XX XX/XX XX/XX XX/XX
75/77 73/79 70/70 81/82     81/84 81/84 75/80 86/89
XX/XX XX/XX XX/XX XX/XX     63/67 61/66 65/67 66/69
XX/XX XX/XX XX/XX XX/XX     62/65 63/67 64/69 64/69
XX/XX XX/XX XX/XX XX/XX     62/69 63/68 64/66 63/66

72/71 70/68 74/72 71/72     XX/XX XX/XX XX/XX XX/XX
70/69 68/72 67/74 68/72     XX/XX XX/XX XX/XX XX/XX
74/75 74/73 68/70 76/75     XX/XX XX/XX XX/XX XX/XX
73/76 71/75 69/69 72/75     XX/XX XX/XX XX/XX XX/XX
75/78 73/76 71/71 79/82     82/85 81/83 78/78 85/86
XX/XX XX/XX XX/XX XX/XX     65/69 66/68 65/66 62/69
XX/XX XX/XX XX/XX XX/XX     63/68 64/65 62/67 65/67
XX/XX XX/XX XX/XX XX/XX     60/66 62/65 62/64 63/66
 FRONT -----> BACK           FRONT ------> BACK

        LEFT ---------------------> RIGHT
```

Figure 18. Thermal distributions on Trinitite (top) and Mutrino (bottom) shown in the layout of the rack. Temperatures are markedly higher when the left and right slots are both populated (marked with rectangles) . c0-0c2s12n0 (circled) is a problem node in both systems.

the 100 node combustion code run. (Only the times in the first set of runs in the `Series` (non-turbo) are marked.) The bottom figure shows related SEDC environmental data on the Aries associated with the entire slot for that node. Note all four nodes of a slot share an Aries rounter and the data shown here, though collected by one node, is for all traffic and stalls associated its Aries router. For this workload, there is only a slight, but noticable, effect on the current (bottom) during the runs. Future work involves consideration of more communication heavy workloads.

VII. Conclusion

Integration of data from a variety of sources is necessary for system understanding, improving system performance, and problem diagnosis. This becomes increasingly necessary as we continue to push the edge of the facilities supporting HPC systems.

In this work we have considered information integration and analysis functionalities currently deployed and under development on the ACES ART systems for Trinity. We

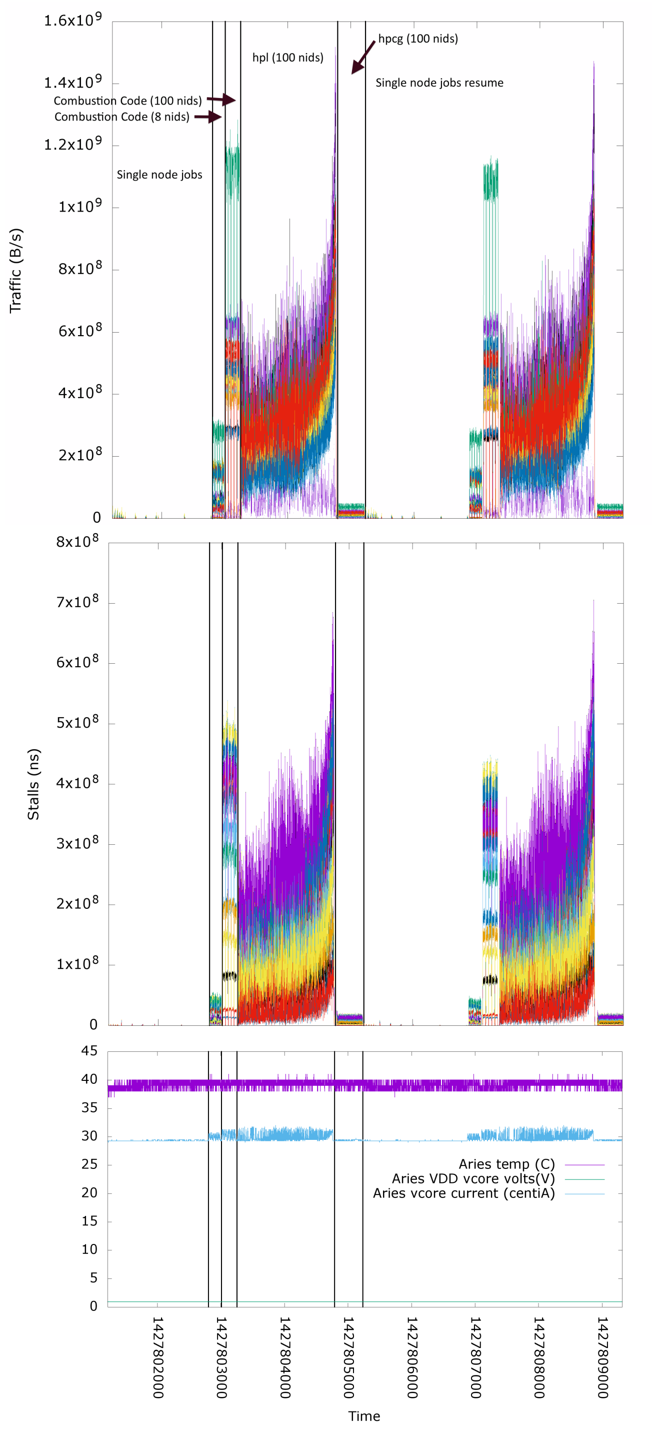


Figure 19. Traffic (top), stalls (middle) nid00012 during the no capping case. Aries SEDC values for the slot for nid 12 (bottom)

presented actual cases of data integration and analysis in regard to power, thermal, and facilities data; and how those cases require consideration as we move to the Trinity system.

Acknowledgements

The authors would like to thank Joshi Fullop (NCSA) and Victor Kuhns (Cray) for useful discussions on the forwarding of SEDC and log data; Eloy Romero (LANL) and Alynna Montoya-Wuiff (LANL) for access to facilities data; and Adam DeConinck (LANL), Kathleen Kelly (LANL), Jason Repik (SNL), and Jim Williams (LANL) who administer the platforms and facilitated the runs used in this work.

References

[1] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-driven, highly-scalable dragonfly topology," *SIGARCH Comput. Archit. News*, vol. 36, no. 3, pp. 77–88, Jun. 2008. [Online]. Available: http://doi.acm.org/10.1145/1394608.1382129

[2] "Using and Configuring System Environment Data Collections (SEDC) Cray Doc S-2491-7001," 2012. [Online]. Available: http://docs.cray.com/books/S-2491-7001/S-2491-7001.pdf

[3] P. Falde, private communication.

[4] J. Laros, D. DeBonis, R. Grant, S. Kelly, M. Levenhagen, S. Olivier, and K. Pedretti, "High Performance Computing - Power Application Programming INterface Specification, Version 1.0," Sandia National Laboratories, Albuquerque, New Mexico 87185 and Livermore, California 94550, Technical report SAND2014-17061, 2014.

[5] "Monitoring and Managing Power Consumption on the Cray XC System Cray Doc S-0043-7202," 2014. [Online]. Available: http://docs.cray.com/books/S-0043-7202/S-0043-7202.pdf

[6] A. Agelastos, B. Allan, J. Brandt, P. Cassella, J. Enos, J. Fullop, A. Gentile, S. Monk, N. Naksinehaboon, J. Ogden, M. Rajan, M. Showerman, J. Stevenson, N. Taerat, and T. Tucker, "Lightweight Distributed Metric Service: A Scalable Infrastructure for Continuous Monitoring of Large Scale Computing Systems and Applications," in *Proc. IEEE/ACM International Conference for High Performance Storage, Networking, and Analysis (SC14)*, 2014.

[7] C. McMurtrie, L. Gilly, and T. Belotti, "Cray Hybrid XC30 Installation - Facilities Level Overview," in *Cray User's Group*, 2014.

[8] "hpl." [Online]. Available: http://www.netlib.org/benchmark/hpl/

[9] "hpcg." [Online]. Available: http://www.hpcg-benchmark.org

[10] "HPCG Performance." [Online]. Available: https://software.sandia.gov/hpcg/2014-06-hpcg-list.pdf

[11] N. Taerat, J. Brandt, A. Gentile, M. Wong, and C. Leangsuksun, "Baler: deterministic, lossless log message clustering tool," *Computer Science - Research and Development*, vol. 26, no. 3-4, pp. 285–295, 2011. [Online]. Available: http://dx.doi.org/10.1007/s00450-011-0155-3