The Malthusian Catastrophe Is Upon Us! Are the Largest HPC Machines Ever Up?

Patricia Kovatch, Matthew Ezell, and Ryan Braby

National Institute for Computational Sciences,
The University of Tennessee
{pkovatch,rbraby}@utk.edu, ezell@nics.utk.edu

Abstract. Thomas Malthus, an English political economist who lived from 1766 to 1834, predicted that the earth's population would be limited by starvation since population growth increases geometrically and the food supply only grows linearly. He said, "the power of population is indefinitely greater than the power in the earth to provide subsistence for man," thus defining the Malthusian Catastrophe. There is a parallel between this prediction and the conventional wisdom regarding super-large machines: application problem size and machine complexity is growing geometrically, yet mitigation techniques are only improving linearly.

To examine whether the largest machines are usable, the authors collected and examined component failure rates and Mean Time Between System Failure data from the world's largest production machines, including Oak Ridge National Laboratory's Jaguar and the University of Tennessee's Kraken. The authors also collected MTBF data for a variety of Cray XT series machines from around the world, representing over 6 Petaflops of compute power. An analysis of the data is provided as well as plans for future work. High performance computing's Malthusian Catastrophe hasn't happened yet, and advances in system resiliency should keep this problem at bay for many years to come.

Keywords: high performance computing, resiliency, MTBF, failures, scalability.

1 Introduction

Conventional wisdom dictates that as supercomputers become larger, they also become more complex with an increased number of parts. Each individual part might have a long Mean Time Between Failure, but when many parts are combined together, the chance for any one part to fail is great. The failure of specific parts might cause an entire machine to fail, and even more likely it will cause one or more running applications to fail. This seemingly makes it impossible for large-scale applications to run to completion without interruption. This situation reminded us of the Malthusian Catastrophe: the idea that populations grow geometrically, but the food supply only grows linearly, with population size being limited by starvation. We decided to explore the parallels between supercomputing and food supply.

M. Alexander et al. (Eds.): Euro-Par 2011 Workshops, Part II, LNCS 7156, pp. 211–220, 2012.

There are four main reasons why population growth hasn't been limited by food supply:

1. economies of scale
2. national and world-wide markets mitigate shortages/problems
3. more efficient technologies, and
4. best practices.

Over time, agricultural resources became more concentrated, with companies making larger infrastructure investments. In this way, more food could be produced more efficiently, taking advantage of economies of scale. This has also happened with supercomputing, with the fewer larger machines becoming more efficient at delivering cycles. This has been done with more shared physical infrastructure, for instance, at Oak Ridge National Laboratory (ORNL), where three supercomputers (one for the Department of Energy, one for the National Science Foundation, and one for the National Oceanic and Atmospheric Administration) share the same computer room. These centers are more efficient are delivering power and cooling, and concentrate and take advantage of the intellectual expertise.

To cope with local shortages, agriculture developed wider markets and better distribution networks. A drought in one area did not cause local people to starve because they could get food from another location. This is also true within supercomputing, where users have access to different machines–for instance, the TeraGrid [1] offers multiple supercomputers, and users have allocations at different sites, to help mitigate the situation when one site is down.

In agriculture, more efficient technologies were developed that took advantage of new equipment and techniques. For instance, farmers started using tractors instead of horses and plows. In supercomputing, vendors have used one large industrial strength fan per cabinet, instead of many PC-quality fans. This shared physical infrastructure for the cabinet reduces the overall total number of components, and makes the individual nodes and overall machine more reliable.

And lastly, farmers developed better practices, like canning, and storing food in case of hard times. Supercomputing has taken similar action: implementing application checkpointing in case of failure.

The University of Tennessee's National Institute for Computational Sciences (NICS) has employed methods to keep their machine available as much as possible. They have redundant power in the facilities, put the machines through a rigorous acceptance test to ferret out as many bad parts as possible, perform maintenance regularly to fix all the down nodes, and run regression tests to verify the system after planned and unplanned outages. Cray, the vendor for their XT4 and XT5 machines, has incorporated power redundancy, shared cooling with multiple liquid cooling cabinets, error correcting memory, reduced components per cabinet (fan, for instance), and support for Berkeley checkpoint/restart. Cray is also working on an MPI implementation to survive link failures. To see if these strategies are working well for production supercomputing, this paper examines if the largest machines "stay up" for reasonable amounts of time–long enough for full machine jobs to be run routinely.

This paper explains some of the difficulties in defining and collecting resiliency statistics, and presents initial findings. Failure rates for individual components are examined over the lifetime of a machine, and conclusions are made based on the data. Failure rate and mean time between failure data from several large systems with similar architectures are examined and patterns and trends are extracted from the data. Is the HPC industry on the cusp of the intersection of system complexity and job length (that is, the inability to get through a single day without an interrupt)?

2 Defining a Resilient System

Different people have different definitions of the qualities that make a system resilient. Do you examine resiliency at the full system level, cabinet level, node level, or processor level? Do you have to examine all of the above? Modern high performance computing systems are often complex and hierarchal in nature, meaning that failure of a single component may or may not affect the availability of additional components.

Although it would be nice to have HPC systems that are 100% reliable, "unbreakable" systems are not practical. System design is often based on a number of factors, balanced according to the the "project triangle" shown in Figure 1. The wisdom of the project triangle states that you can build something fast (rapid engineering design), you can build something good (high quality), and you can build something cheap (low cost), but you only get to pick two. In HPC, a goal is often to place well on the Top500 [4] list. Every dollar spent on high-reliability parts takes away from the system's peak performance. Designing a high-performance reliable system becomes a complex balancing act.

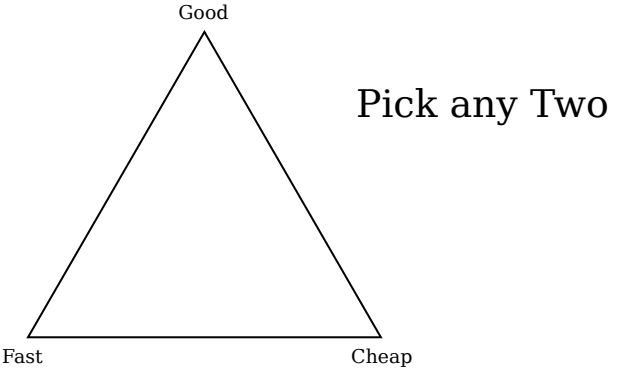

Fig. 1. The Project Triangle [2]

In [3], Stearley provides definitions and equations for reliability, availability, serviceability, and utilization based largely on the semiconductor industry's SEMI-E10 specification. Of particular interest, mean time between failure (MTBF) for an entire system or node is defined as:

$$MTBF_{System} = \frac{\text{production time}}{\text{number of system failures}} \tag{1}$$

$$MTBF_{Node} = \frac{\text{production time}}{\text{number of node failures}} \tag{2}$$

Stearley notes that reliability is often defined as:

$$R(t) = e^{-\lambda t} \tag{3}$$

where

$$\lambda = \frac{1}{MTBF} \tag{4}$$

is a *constant* failure rate, but Stearley and others [6] suggest a time-varying failure rate may be more appropriate.

3 Difficulties in Obtaining and Comparing Data

It seems that most supercomputing centers develop their own policies and methods for collecting, analyzing, and reporting resiliency data. The equations in Section 2 from [3] provide high-level direction for reporting the data, but there is no advice for a common data collection format. The USENIX Computer Failure Data Repository [5] contains a wealth of failure data, but no common data format exists between the entries.

It is common to see a difference in vendor-specified failure in time (FIT) rate compared to those observed in a large-scale HPC system [7]. These large computer systems provide a unique source of reliability data that a vendor does not get to see with the rest of their business. An integrated circuit vendor may produce 10 million units of a certain part that gets shipped to 100 customers who then integrate it into a product that gets sold to thousands of their customers. This quickly becomes a fragmented field data base where data can be lost or left unreported. HPC vendors may purchase 250,000 of that part and place them in a dozen systems that are closely tracked for failure types and trends.

Another factor that makes comparing field data to vendor data difficult is the ever-present percentage of "no trouble found" failure cases. HPC centers will count these as failures, but the vendor will likely ignore them. There are a number of possible reasons for this:

1. Site staff misdiagnosed the failure.
2. Vendor component retest is unable to duplicate the specific operating conditions.
3. Failure is of an intermittent nature and does not fail during retest.
4. The component was proactively replaced because of indications a failure was imminent.

4 Reliability of Kraken at the National Institute for Computational Sciences

4.1 Cray XT Architecture Description

Cray and Sandia National Laboratories designed a distributed memory supercomputer codenamed *Red Storm* that utilized commodity AMD Opteron processors and a proprietary *SeaStar* network. In 2004, this technology was turned

into a commercial offering called the Cray XT3. Each Cray XT cabinet contains 24 blades, and the machine contains a mixture of compute and service blades. Each compute blade holds 4 compute nodes, while each SIO blade has two service nodes. SeaStar network chips (one for each node) live on a mezzanine card and provide access to the three dimensional torus network.

4.2 NICS Kraken System Description

The National Institute for Computational Sciences (NICS) was granted a $65M award from the National Science Foundation in September 2007, and began to acquire, install, and place in production a series of Cray HPC systems. Each of these systems has been called Kraken after the mythical sea monster of Norse legend. The first dedicated NICS supercomputer was a Cray XT3, comprised of 40 cabinets and with a peak computational rate of 38 teraflops using AMD dual-core processors that went into production in June 2008. This system was rapidly upgraded over the following month to a Cray XT4 with the addition of 8 cabinets using the new AMD Budapest 2.3 GHz quad-core processors, and the replacement of the existing dual-core processors by the new quad-core chips, taking the peak performance to 166 teraflops from 18,048 cores.

The next incarnation of Kraken was a completely new Cray XT5 comprised of 88 cabinets and 608 Teraflops peak in February 2009. The "Barcelona" processors at 2.3 GHz were used in the XT5, in preparation for an upgrade to one petaflop of "Istanbul" chips. This new machine was actually installed in the ground-floor machine room at ORNL, while the XT4 remained as a completely separate system in the upper machine room. Both the old and new Kraken systems ran as parallel production systems for two months, through February and March of 2009, while users migrated to the newer machine. In September of 2009 Kraken was upgraded to 2.6 GHz six-core Istanbul processors, and entered production ahead of schedule on Oct 5, 2009. In February of 2011, an additional 12 cabinets were added to Kraken, for a total of 1.17 Petaflops of peak computing power and 147 Terabytes of memory spread through 9408 compute nodes. Kraken has a 3.3 PB parallel file system directly connected with 30 GB/sec bandwidth.

Kraken has a varied workload that supports scientists from numerous fields of study. NICS has optimized scheduling policies to efficiently handle both capacity and capability computing [8] while maintaining very high utilization numbers (see Figure 2). Utilization that is nearly uniform over time is helpful when analyzing resiliency data over time. Kraken has achieved over 95% uptime during its entire lifetime.

4.3 Data Collection

At NICS, system outage data is manually entered into a database after every downtime event. The database supports high level categories as well as component-level failure reasons.

NICS uses the Simple Event Correlator with Cray-specific rules [9] to monitor the hardware event logs for failures. Failures and anomalies are logged to a

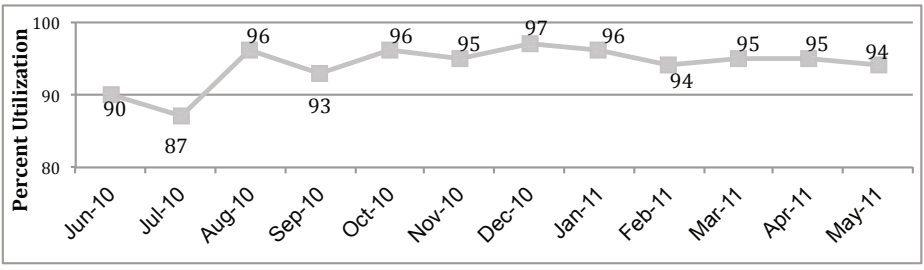

Fig. 2. Kraken Utilization

central file, and events that require manual intervention from the Cray hardware engineers are automatically entered into Cray's case tracking system. Additional records are added as necessary for any maintenance performed on the machine. Cray uses this tool to record every hardware failure at every site with Cray machines. They keep an extensive database of each failure, and are thus able to calculate a variety of metrics.

4.4 Kraken Node Failure Data

Figure 3 shows Kraken's node failure data over time. Events tagged as memory errors correspond to bank 4 machine check exceptions. Opteron errors correspond to all other machine check exceptions. While it is known that not all bank 4 machine check exceptions are due to DIMM failures, they do relate to the memory subsystem and determining which component failed is not always easy. Therefore, this method was chosen for the first pass to review the data. Power events are due to voltage faults in various components. SeaStar errors, in this context, were errors that caused nodes to fail. Due to the nature of the XT's high speed network, it is highly likely that a single failing SeaStar or mezzanine caused many nodes to fail (in fact, it may cause the whole system to fail). Uncategorized events correspond to ambiguous events where no root cause was identified or the failure type is so rare that a new category was not created.

The graph shows expected "infant mortality" with the memory at the beginning of the machine's lifetime, but there is an interesting long term drop in memory errors starting in September and October of 2009. Reviewing the data, it became clear that this coincides with the September 2009 processor upgrade of Kraken to AMD hex-core Istanbul. These Istanbul processors have more sophisticated "chipkill" ECC. This explains the dramatic decrease in uncorrectable memory errors. There is also a spike in memory failures in July 2009, which was contributed to by a cooling failure and emergency power off (EPO). Early analysis of failure data for JaguarPF (which shares the machine room with Kraken, and had the same processor upgrade performed) shows a similar spike in July and a similar drop in failures in late 2009.

After the initial burn-in period and processor upgrade, Kraken's data fairly consistently shows more Opteron failures than memory failures. Although CPUs

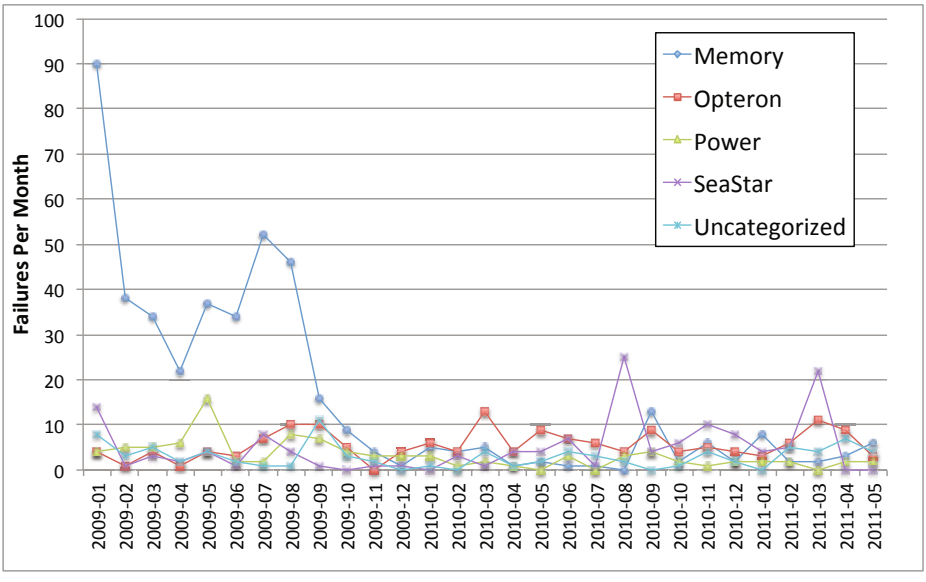

Fig. 3. Kraken XT5 Node Failure Causes by Month

are a more complicated part, the authors expected to see more failures due to memory than attributed to CPUs, as was observed in most systems in [10]. One possible explanation is more recent ECC memory technologies (Chipkill, SDDC) and improved memory controllers may have significantly reduced the memory failure rates. Upon analyzing the Opteron failure data, it was discovered that most of the errors were attributed to ECC errors in the on-processor cache hierarchy. Investigating this with Cray and AMD, it was discovered that a recently released bios update should significantly reduce the number of these failures.

Similar to the improvements seen in DRAM over recent years, it is expected that processor reliability will also improve. Improved error correction algorithms are being developed and will be built into the processors at various levels, including the cache hierarchies. We expect that the processor failure rates will see drops similar to those observed with the memory. This should more than make up for the potential increase in failures as processor cache sizes grow.

5 Comparison of Large Cray Systems Resiliency

5.1 Data Collection

We asked several sites with Cray hardware to provide failure statistics about their machines. To simplify the data collection and avoid categorization ambiguity, the authors asked for a month-by-month count of unscheduled full-system outages over the lifespan on the machine, as shown in Figure 4 for Kraken.

This data does not include scheduled maintenance periods or “environmental” events that caused outages. With this data, an average monthly failure rate can be

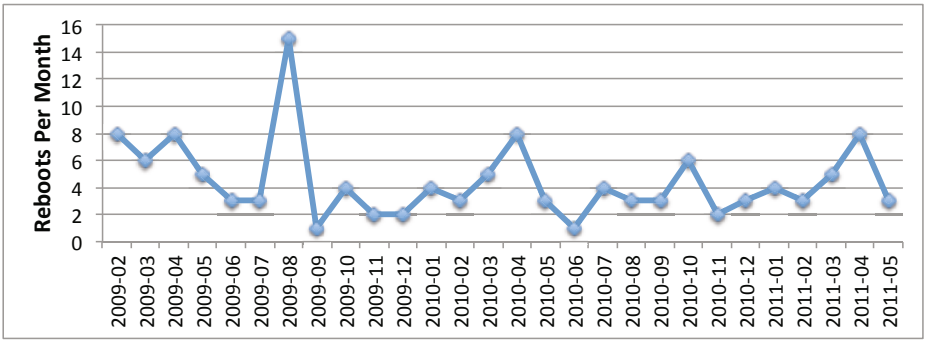

Fig. 4. Kraken Unscheduled Reboots per Month

computed by simply averaging all the data points provided. Assuming 30 days per month, the mean time between failure can be computed by Equation (5).

$$MTBF_{days} = \frac{30 \times \text{months of data}}{\sum \text{all failures}} \tag{5}$$

5.2 The Machines

The collaborating sites were very helpful, and the authors were more successful than expected in receiving responses. In total, data exists from ten systems spanning several generations of Cray hardware. The results are shown in Table 1.

Table 1. Full System Reboot Data

Machine	Site	# Cabinets	TFLOPS	Reboots/Month	MTBF (days)
JaguarPF XT5	ORNL	200	2331	11.4	2.6
Frankin XT4	NERSC	102	356	5.2	5.7
Kraken XT5	NICS	88	1029	4.5	6.6
Jaguar XT4	ORNL	84	260	6.8	4.4
Hopper XE6	NERSC	68	1289	3.8	8.0
Athena XT4	NICS	48	165	2.8	10.7
Kraken XT4	NICS	48	165	3.0	10.0
Raptor XE6	AFRL	30	410	3.0	10.0
Hexagon XT4	NOTUR	15	51	0.6	52.1
Gaea XT6	NCRC	14	260	2.6	11.7

Figure 5 shows the relationship between system size and failure rate. As one might expect, increasing the size and complexity of a machine linearly increases the likelihood of failure. The data also suggests that failure rate is more highly correlated to number of components than peak performance rating; as individual components get more powerful, their failure rate does not significantly increase.

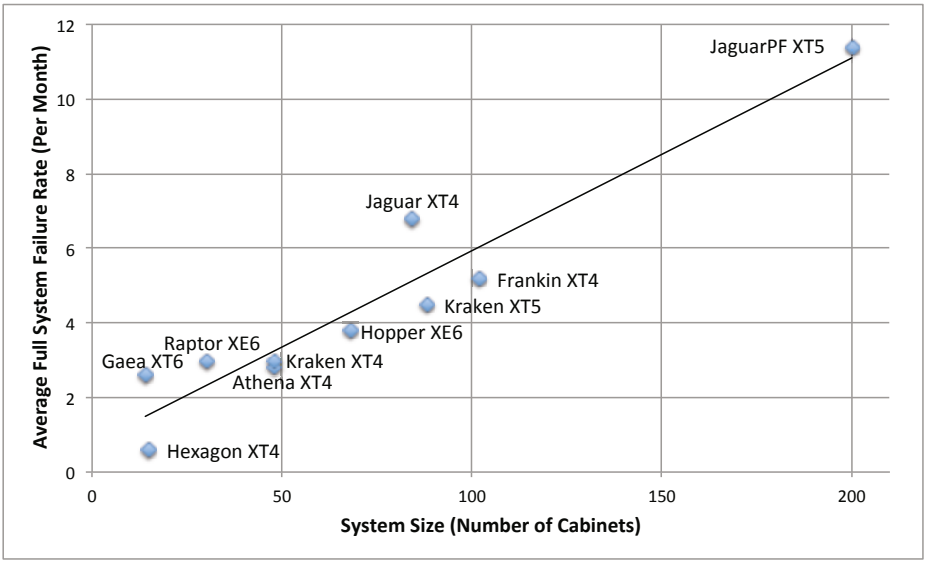

Fig. 5. Cray System Failure Rates

6 Future Work

There are several areas that warrant further investigation. The authors have obtained the component-level "Failures per Month" data for Jaguar, and it appears to match the rough shape of the Kraken data. Comparing it more closely to the Kraken data would be interesting. A study comparing the individual component (CPU, memory) MTBF numbers with what we've seen in the field at scale would be interesting, but it requires obtaining data from vendors. Unfortunately, vendors do not like to share this. Future work will hopefully include both node and system failure data between Cray and other machine types, including IBM BlueGenes, IBM Power series and conventional clusters. This would show if a particular design or machine is more reliable, at scale, than others. Future work might examine in more depth why some machines of the same kind, that are similar in size exhibit almost a 30% difference in failure rates (see Figure 5 for Jaguar XT4 and Franklin XT4 failure rates).

7 Conclusions

In this paper, the authors have shared node and system-wide failure data from the largest systems in the world and shown that although the number of components in the systems is quite large, applications can regularly run full machine jobs. Vendors and centers have developed techniques, technologies and approaches to help mitigate the geometrically growing number of parts in the largest machines.

Acknowledgements. The authors would like to thank XTreme, the Cray system administrators user group who facilitated the sharing of site specific machine uptime data, and Jim Craw, its President during the time of data collection. Thanks go out to the representatives from each site along with their management: Tina Butler and Nick Cardo (NERSC), Joni Viranen (CSC), Hank Kuehn, Don Maxwell and Buddy Bland (ORNL), Steve Andrews (HECToR/STFC Daresbury), Lloyd Slonaker (AFRL/RCMT), and Alexander Oltu (Uni). The authors would also like to thank Steve Johnson and Pete Ungaro from Cray for their data, support and assistance. Several folks at ORNL helped analyze and explain the data, including Stephen L. Scott and Robert Harrison, and Rick Mohr and Troy Baer from The University of Tennessee. Lastly, the authors would like to thank Phil Andrews for recognizing and suggesting the comparison with the ideas of Thomas Malthus.

References

1. TeraGrid, `http://www.teragrid.org/`
2. Piazzalunga, D.: Project Triangle. Figure in public domain, downloaded from, `http://en.wikipedia.org/wiki/File:Project_Triangle.svg`
3. Stearley, J.: Defining and Measuring Supercomputer Reliability, Availability, and Serviceability (RAS). In: 6th LCI Conference on Linux Clusters (April 2005)
4. Top500 Supercomputer Sites, `http://top500.org/`
5. The Computer Failure Data Repository, `http://cfdr.usenix.org/`
6. Gottumukkala, N., Nassar, R., Paun, M., Leangsuksun, C., Scott, S.: Reliability of a System of k Nodes for High Performance Computing Applications. IEEE Transactions on Reliability 59(1), 162–169 (2010)
7. Johnson, S.: Cray Inc. Personal Communication
8. Andrews, P., Kovatch, P., Hazlewood, V., Baer, T.: Scheduling a 100,000 core Supercomputer for Maximum Utilization and Capability. In: 39th International Conference on Parallel Processing Workshops (2010)
9. Becklehimer, J., Willis, C., Lothian, J., Maxwell, D., Vasil, D.: Real Time Health Monitoring of the Cray XT3/XT4 Using the Simple Event Correlator (SEC). Cray Users Group (2007)
10. Schroeder, B., Gibson, G.: A Large-Scale Study of Failures in High-Performance Computing Systems