PREDICTIVE ANALYTICS FOR HIGH PERFORMANCE COMPUTING

Forecasting system failures with predictive analytic tools

Challenge: When large-scale High Performance Computing (HPC) systems go offline, the impact to scientific users can be catastrophic. Research data is lost or delayed, and costs mount the longer the system is offline awaiting diagnosis and repair.

Currently large-scale HPC centers routinely monitor system logs and sensors embedded throughout the HPC environment to review system status and stability. However, job checkpointing at the frequency of single node failures is not feasible (imagine checking every bulb on a whole string of Christmas tree lights on a regular basis).

In addition to hardware failures, system misconfigurations are a large problem. Operating system and network changes deployed to add features and fix security vulnerabilities can have unintended consequences. System performance may appear unaffected after the change, but the right combination of user jobs will expose a new system bottleneck. Detecting and tracking down these problems is time-consuming.

To combat time and budget losses associated with offline systems, we applied machine learning, data mining techniques and relationship analysis to the data collected in and around the HPC system to gain potential insights into system failures. We sought to discover if judicious use of these techniques would allow us to determine some level of causality that could provide a basis for predictive failure analysis, fully keeping in mind that correlation does not imply causality?

Optimizing and maintaining large-scale computing infrastructure is challenging and time-consuming. Regardless of HPC-center size, a team of system engineers is required to keep the computing environment stable. Methods to quickly detect and identify potential fault-state precursors or predict failures in any system would result in enormous benefits. Significant efficiencies could be gained by "smart" checkpointing that saves the software state on a triggered checkpoint before predicted failures.

Analyzing system log files is one of the main tools for diagnosing system failures. Nearly every process/daemon that runs on a system outputs detailed log messages stored on the individual system. This high volume of information is vital to problem investigations, but the logs often contain informational and unrelated data that make the search a proverbial "finding a needle in a haystack."

We present a scalable process to parse log files and calculate the likelihood of the system slipping into an error state. The process has three general steps:

1. Generate a message dictionary from log data.
2. Use unsupervised classification techniques to determine if a given time interval is "abnormal."
3. Identify the abnormal states as known error conditions, tag fault states as known positives and classify the full body of data to train neural networks for machine learning.

Potential Benefits: An automated monitoring system for deep diagnostics, continually running prediction algorithms in real-time and generating alerts, offers an opportunity to proactively combat system failures. Initial results promise potential efficiency increases. Such a solution would enable system engineers to fix problems before they escalate into a system-wide cascade of errors. Additionally, these alerts can be consumed directly via software to trigger software mitigations if available. For example, if a network storage error is predicted, software can refrain from accessing it or increase I/O timeout values.

PREDICTIVE ANALYTICS FOR HIGH PERFORMANCE COMPUTING

Improving overall system operational efficiency through intelligent job scheduling using science methods

Challenge: Currently HPC systems monitor and collect large amounts of data related to both the hardware of the system and users' jobs. Job-related data can provide detailed information about how a user's application is utilizing various system components. This may include information about network traffic, file system traffic (I/O), efficiency of CPU utilization, etc. Analyzing users' job data in the context of a shared HPC system can provide insight into the overall dynamic and static loads on the network, file system and processors.

How might such data be analyzed? What patterns can be observed? And how can the data and patterns improve the overall efficiency of the system? For example, if the I/O usage patterns of different user applications are collected and analyzed, it might provide insight as to what kind of jobs should or should not be scheduled at the same time. This can be combined with other I/O activities being done over the same I/O network on the system by system administrators for functions such as backing up data.

The Comet HPC

The Comet HPC system housed at San Diego Supercomputer Center (SDSC) is currently serving hundreds of academic users from various universities in the U.S., and the user jobs are related to simulations from a wide variety of science, engineering and social science disciplines.

Users may choose from three separate queues:

1. The compute queue allows users to run exclusively on Comet's standard compute nodes,
2. The GPU queue allows users to run on Comet GPU nodes, and
3. The shared queue runs jobs in shared mode on Comet's shared nodes. The scheduling for the three queues is implemented by SLURM (Simple Linux Utility for Resource Management).

Comet's Specs

- A Dell cluster delivering ~2.0 petaflops
- Intel Haswell processors with AVX2 and Mellanox FDR Infiniband networks
- Standard compute nodes built with:
  - Intel Xeon E5-2680v3 processors
  - 128 GB DDR4 Dram (64 GB per socket)
  - 320 GB of SSD local scratch memory
- GPU nodes that have four NVIDIA GPUs each
- Large memory nodes each contain 1.5 TB of DRAM and four Haswell processors
- Network topology is 56 Gbps FDR InfiniBand with rack-level full bisection bandwidth and 4:1 oversubscription cross-rack bandwidth
- Seven petabytes of 200 GB/second performance storage and six petabytes of 100 GB/second durable storage
- NFS is used primarily for home directory of Comet's users

Using aggregate I/O data from jobs that used the SSD and Lustre file systems on Comet, we analyzed distinct patterns in the dataset caused by different applications. We have seen that processes unrelated to NFS-based gateway jobs created a distinct linear pattern on the SSD read vs SSD write plot. We also identified distinct patterns generated by astrophysics, molecular dynamics and Hadoop-based applications.

Potential Benefits: In only analyzing the aggregate I/O for each job, we identified I/O behaviors of different job applications. In the future, we can further analyze the time series data. For example, we can attempt to break up the time series of the jobs based on I/O usage at the beginning of the job, at the middle and at the end. We can also analyze jobs separately based on parameters like run time of the job, known applications and even-edge cases. Therefore, we can extract different, meaningful information based on further analysis. These analyses will allow us to better understand the I/O usage patterns of different applications with the potential to apply the information in a scheduler to improve overall system performance.

For more information, please contact:

Dr. Rajiv Bendale | Engility Corporation | Tel: 858.212.5865 | rajiv.bendale@engilitycorp.com
Kimberly Robertson | Engility Corporation | Tel: 703.707.2736 | kimberly.robertson@engilitycorp.com

ENGILITY
Engineered to Make a Difference