PREDICTING PERFORMANCE DEGRADATION AND FAILURE OF APPLICATIONS THROUGH SYSTEM ACTIVITY MONITORING

Allocation: Exploratory/50 Knh
PI: Ravishankar K. Iyer[1]
Co-PI: Zbigniew Kalbarczyk[1]
Collaborator: Saurabh Jha[1], Benjamin Lim Wen Shih[1]

[1]University of Illinois at Urbana-Champaign

EXECUTIVE SUMMARY

Our overarching investigation addresses complex data-driven problems associated with online system monitoring for understanding causes of application failures and performance degradations. Our contributions include the design of machine learning-based deep analytics framework to distinguish the cause of application and system performance degradation due to failure- and nonfailure-related issues. The tool leverages probabilistic graphical models to conduct machine learning at scale for runtime detection of congestion parameters and its effects on running applications. Our approach correctly identifies the cause of congestion in 74% of the cases (of 302 cases found in our dataset) and attributes it either to resource contention issues or to failures in the system. In addition, working with Sandia National Laboratories and the National Energy Research Scientific Computing Center, we are addressing the issue of congestion in the Cray Aries interconnect to help advance scheduling decisions of applications by porting our algorithms for online system monitoring and mitigation.

RESEARCH CHALLENGE

Extreme-scale high-performance computing (HPC) systems require a holistic approach to monitoring and coordinating many disparate subsystems (both hardware and software) to enable continued scaling and efficient execution of applications. HPC systems are typically used for executing tightly coupled simulation applications across hundreds of thousands to millions of processor threads. Resource contention due to failures or design issues (of applications/systems) impacts applications in two ways: (1) by degrading application performance, which causes application runtime unpredictability and limits scaling to full system; and (2) through propagation of errors and failures in application logic and code-flow, which causes application failures or invalid/error-prone outputs. As a result, effective failure/degradation mitigation response(s) in complex systems require analysis of the propagation of faults/errors and of performance issues due to interference among applications or resource exhaustion.

Our analysis approach addresses more general scenarios that result in performance degradation in which timely and appropriate response can significantly improve both application runtimes and system throughput. The eventual product of this work will be an interoperable set of capabilities for extreme-scale systems that provides monitoring, analysis, and appropriate response to both resilience and performance issues. These capabilities will support both automated and exploratory analysis, both at runtime and in post-processing. Specifically, the Blue Waters grant allocation was used to characterize performance logs and machine-generated error logs to holistically understand the propagation of performance degradation and faults affecting applications. We use Bayesian network-based machine learning methods to distinguish between failure-based resource contention issues and nonfailure-based contention issues.

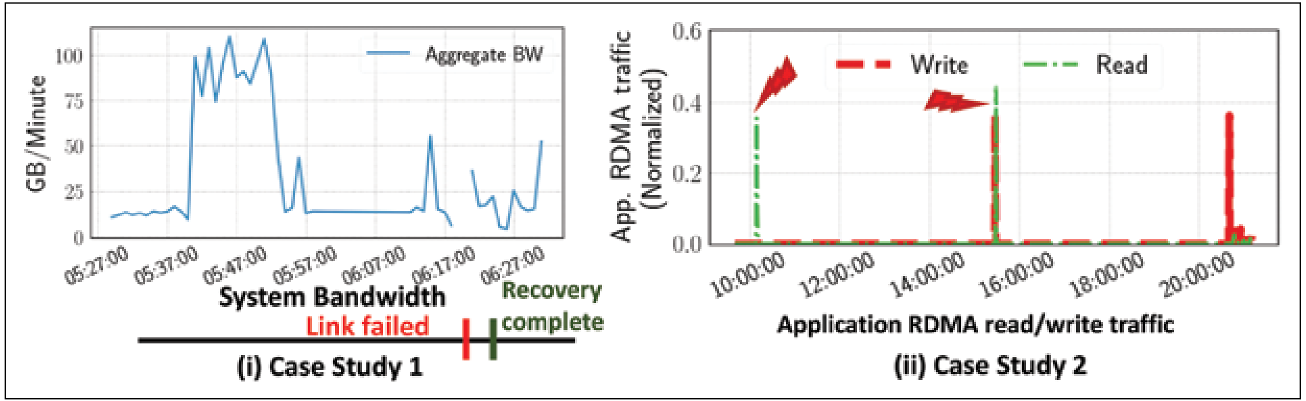

Figure 1: Case study 1 shows link-failure-related performance degradation. Case study 2 shows design issue-related performance degradation.

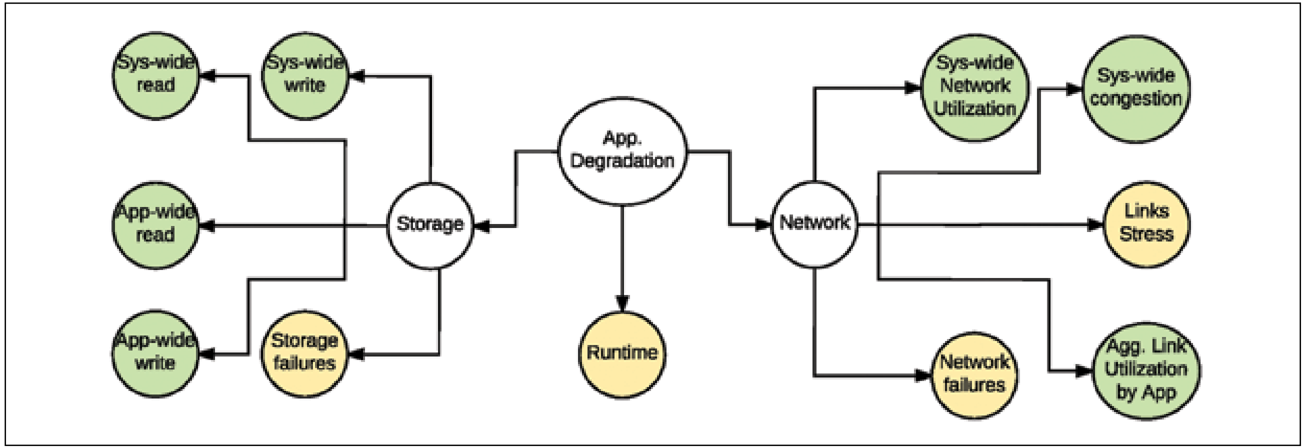

Figure 2: Bayesian network for diagnosing the cause of application degradation. Green-filled circle values are obtained using unsupervised clustering algorithms from raw data. Yellow-filled circle values are obtained directly from measure-derived metrics.

METHODS & CODES

Regardless of whether performance problems such as contention for shared resources are failure-related (e.g., caused by link failures) or nonfailure-related (e.g., caused by design issues in an application or system), they can quickly propagate in systems and result in severe performance degradation, rather than application failure as discussed below.

Case Study 1: Fig. 1 (i) shows the impact of network link failure and its corresponding recovery on the Gemini 3D torus interconnection network of Blue Waters. This impact occurs because a single-link failure can cause the traffic pattern of the application to change in the underlying network. Event logs corresponding to this failure are shown in the figure. It also shows the aggregate data passing through all the Gemini routers. Basic statistics such as these indicate bandwidth utilization of the system. To understand whether there is actual performance degradation and, if so, which Gemini router may be the source of the problem, we calculate a derived metric, "average packet latency," which captures the average time taken by a router to deliver packets.

Case Study 2: Fig. 1 (ii) shows the impact on performance of the system and application due to interplay of design issues between the application (write pattern) and system (scheduling strategy). Specifically, a 32-node job caused high congestion in the system interconnect, triggering two congestion protection events, first at 10:00 a.m. (within 10 seconds of job launch) and then at 15:20 (red lightning bolts). Congestion within the torus can adversely impact the performance of the application generating the messages and of other running applications, and is a major cause of inconsistent application runtimes. This job was assigned a linear shape in the "Z" direction by the topology-aware scheduler. Hence, when it was reading large amounts of data (over RDMA, or remote direct memory access), most of the I/O calls were funneled through specific links, causing the congestion.

RESULTS & IMPACT

Our tools take a holistic approach for differentiating design-related performance issues from failure-related issues. This is done by building a Bayesian network-based diagnostic model using features obtained from systems and applications. These features are either obtained directly from measured raw metrics (yellow-filled circles in Fig. 2) or by running unsupervised clustering algorithms on raw metrics (green-filled circles). We tested our design and tools by manually verifying the cause of congestion for 302 reported cases and achieved 74% accuracy.

WHY BLUE WATERS

Blue Waters is one of the few open-science capacity systems that provides a test bed for scaling computations to tens or hundreds of thousands of cores on central processing units (CPUs) and graphics processing units (GPUs). It also enables the study of failures and degradation of applications in production petascale systems because of its unique mix of XE6 and XK7 nodes. This allows us to understand the performance–fault-tolerance continuum in HPC systems by enabling the investigation of application-level designs for mixed CPU and GPU node systems, and fault isolation in system components to mitigate failures at the application level.

PUBLICATIONS AND DATA SETS

Jha, S., et al., Resiliency of HPC Interconnects: A Case Study of Interconnect Failures and Recovery in Blue Waters. *Proceedings of the IEEE Transactions on Dependable and Secure Computing*, PP:99 (2017), DOI: 10.1109/TDSC.2017.2737537.

Formicola, V., et al., Understanding Fault Scenarios and Impacts through Fault Injection Experiments in Cielo. *CUG 2017*, Redmond, Wash., May 7–11, 2017.

Jha, S., et al., Holistic Measurement Driven System Assessment. *Workshop on Monitoring and Analysis for High Performance Computing Systems Plus Applications*, Honolulu, Hawaii, September 5, 2017.