Lessons learned from spatial and temporal correlation of node failures in high performance computers

Siavash Ghiasvand[†], Florina M. Ciorba[*], Ronny Tschüter[†], and Wolfgang E. Nagel[†]
[†]Technische Universität Dresden, Dresden, Germany. Email: {firstname.lastname}@tu-dresden.de
[*]University of Basel, Basel, Switzerland. Email: florina.ciorba@unibas.ch

***Abstract*—In this paper we study the correlation of node failures in time and space. Our study is based on measurements of a production high performance computer over an 8-month time period. We draw possible types of correlations between node failures and show that, in many cases, there are direct correlations between observed node failures. The significance of such a study is twofold: achieving a clearer understanding of correlations between node failures and enabling failure detection as early as possible. The results of this study are aimed at helping the system administrators minimize (or even prevent) the destructive effects of correlated node failures.**

I. Introduction

The failure rate of high performance computers rapidly increases due to their growth in size and complexity. Failures, thus, become the norm rather than the exception. There are several de-facto failure recovery mechanisms for high performance computers (e.g., checkpoint-restart, duplication, and re-execution). The efficiency of recovery mechanisms depends on the mean time between failures (MTBF). It is expected that in the near future, the MTBF of high performance computers becomes too short, such that current de-facto failure recovery mechanisms will no longer be able to recover the system from failures [1]. Early failure detection is a new class of failure recovery methods which can be beneficial for high performance computers with low MTBF. Detecting failures in their early stage can reduce their negative effects by preventing propagation of their side effects to other parts of the system [2].

One way to detect failures in their early stage is via monitoring the nodes' behavior and seeking behavioral anomalies. Performance is a key property of high performance computers. To prevent any performance penalty due to actively probing of nodes, we employ a passive monitoring approach. The native Linux message logging (also referred to as syslog) is the source of our monitoring information. The syslog daemon, records ongoing hardware- and software-related events of the system. In this paper, we refer to such hardware/software events in the syslog as syslog entries.

When studying failures, the granularity of components plays an important role in interpreting the system behavior. As long as a node behaves as expected, it can withstand failures. When a node is unable to carry out its expected load, it is viewed as a "node outage". Certain failures will lead to node outages (e.g., a failed switch). A node outage is an observable indication of a failure occurrence. A *node outage* is defined as the case when no syslog entries from a particular node can be observed. We consider three reasons for node outages: (1) site-wide power outages, (2) planned maintenance, and (3) other reasons. In this study we focus on the failures observed at the node level and derive correlations along three dimensions. (1) *Temporal*: denotes cases when the time gap between consecutive failures falls below a certain threshold; (2) *Spatial*: denotes cases when the failed nodes share a physical resource (e.g., chassis); and (3) *Logical*: denotes cases when the failed nodes share a logical resource (e.g., batch job).

The main contribution of this paper is a detailed study that aims to help the system administrators minimize the destructive effects of correlated node failures.

Our methodology is based on a cyclic workflow, as follows: System monitoring $\rightarrow$ Analysis of monitoring data $\rightarrow$ Derivation of correlations $\rightarrow$ Early failure detection $\rightarrow$ Timely failure prediction. In this paper we concentrate on "derivation of correlations". The subsequent steps are part of future work.

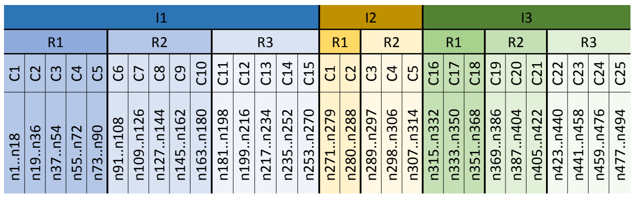
II. Related Work

Distributed computer systems are hierarchically structured, e.g., system, cluster, rack, and node. Furthermore, such systems share various resources, such as power supply units, network, or (distributed) file systems. As a result, failures in one location may trigger further failures and, as such, propagate between different system components. Failure correlations can be classified into temporal, spatial, logical, and combinations thereof.

Yigitbasi et al. [3] investigated the temporal correlation of failures in large-scale distributed systems. The examined event logs featured strong daily patterns and high auto-correlation. Sahoo et al. [4] analyzed event logs of heterogeneous servers and also found different forms of strong correlation structures including significant periodic behavior. A proactive failure prediction system considering the temporal order of the events was described by Sahoo et al. [5].

Liang et al. [6] analyzed logs from an IBM BlueGene/L system and found skewness in the distribution of network failures. Gallet et al. [7] used a moving window to generate groups of spatially correlated failures from empirical data. They showed that spatial correlation of failures cannot be neglected for an accurate analysis of system downtimes. Fu et al. [8] processed event logs to identify event dependencies in order to improve failure prediction and root cause diagnosis.

Fu and Xu [9] enhanced this approach by taking into account both temporal and spatial correlations. Their model used event clustering to quantify the temporal correlation and developed another model to characterize spatial correlation. They showed that failure events exhibit strong correlations in



I1															I2					I3									
R1					R2					R3					R1		R2			R1			R2			R3			
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C1	C2	C3	C4	C5	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25
n1..n18	n19..n36	n37..n54	n55..n72	n73..n90	n91..n108	n109..n126	n127..n144	n145..n162	n163..n180	n181..n198	n199..n216	n217..n234	n235..n252	n253..n270	n271..n279	n280..n288	n289..n297	n298..n306	n307..n314	n315..n332	n333..n350	n351..n368	n369..n386	n387..n404	n405..n422	n423..n440	n441..n458	n459..n476	n477..n494

Figure 1. Taurus topology - *I*, *R*,*C*, and *n* stand for *island*, *rack*, *chassis*, and *node*, respectively. Nodes, chassis, and racks shown in different shades of the same color are located in same island and they are in close proximity to each other.

the time and space domains. Tang and Iyer [10] proposed c- and p-dependent models to estimate reliability for systems with two- and multi-way correlations, respectively.

In this work, we examine event logs and focus on identifying both temporal and spatial correlations of failures. We want to share our knowledge gained via these observations with the community and provide the foundation for early failure detection techniques. As future work, we will extend our analysis with batch job allocation information gathered from the batch system to correlate the jobs affected by the same failure.

III. System Monitoring

Taurus[1] is a general purpose production high performance computing (HPC) system at Technische Universität Dresden, Germany. It consists of three partitions called *islands* and has a total of 494 compute nodes. The first island has 4320 Intel E5-2690 (Sandy Bridge) 2.90GHz cores (270 nodes), the second island in addition to the 704 Intel E5-2450 (Sandy Bridge) 2.10GHz cores, is also equipped with 88 NVidia Tesla K20x GPUs (44 nodes), while the third island has 2160 Intel X5660 (Westmere) 2.80GHz cores (180 nodes). The total peak performance is 137 TFlop/s (without GPUs). Taurus is powered by Bullx Linux and uses Slurm as batch system.

The Taurus structure is shown in Fig. 1 and will be used as the reference for numbering the nodes. The native Linux syslog-ng daemon runs on all compute nodes. The login node aggregates the syslog files from all compute nodes. The syslog daemon on the login node is configured to push all syslog entries to a storage space outside of the cluster. Regardless of the content of syslog entries, we consider each syslog entry as a *heartbeat* of its respective generating node. Using these heartbeats one can monitor the liveliness of the nodes and, as such, determine whether a node is dead (no heartbeat) or alive (has heartbeat). Fig. 1 shows that nodes with consecutive node IDs are, in fact, located physically next to each other. Based on this information, in Fig. 2 we can see that the number of node outages for those nodes which are physically next to each other is more or less equal. These observations motivate the present study and the search for temporal and spatial correlations of node failures on Taurus.

The system was monitored over an 8-month period from 01-09-2014 to 30-04-2015. Within this period a total of 1669 node outages were observed. Considering all outages of all nodes during the 8-month period, leads to an average meantime between node outages (MTBO) of 3.65 hours. The formula used to calculate the MTBO is MTBO=$\frac{\sum T_{BO}}{N}$, in which, T_{BO} is the interval between two consecutive node outages and N is the total number of outages on Taurus in the 8-month period of study.

[1]https://doc.zih.tu-dresden.de/hpc-wiki/bin/view/Compendium/SystemTaurus#Phase_1

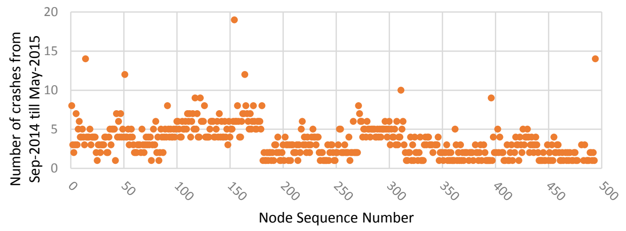


Figure 2. Total number of outages for each node, over an 8-month period from 01-09-2014 to 30-04-2015. The total number of node outages is 1669.

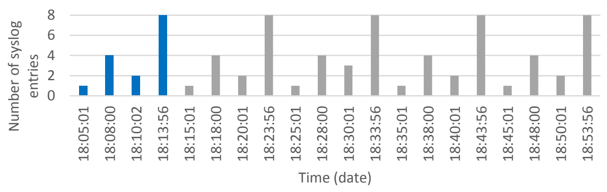


Figure 3. Syslog entries of a healthy node (n482) over one hour on 28-04-2015. The main pattern is shown in blue and pattern repetition is shown in gray.

Each *island* of Taurus is intended for a different use. The percentage of node outages within each island is as follows, I1: 65.12%, I2: 13%, and I3: 21.86%. By having a closer look at the outage percentage of each island we realized that the second island (containing GPUs) is responsible only for a very small share of outages and, therefore, we decided to focus our correlation study on the first and third islands.

IV. Workflow for Node Failure Detection

Each individual compute node generates its own syslog entries and transmits them to the master (usually a login) node. Therefore, upon outage of a certain node, no further syslog entry is received from that node. This results in a common problem shared by all remote-access systems: it is not possible to truly indicate whether a node is dead, too busy to respond, or simply lost its connectivity. However, the effect is the same, regardless of the cause: the particular node can no longer be used. Therefore, we count all the above situations as a node outage. Common services and daemons run on each node and some of them regularly generate log entries. If the time between two log entries exceeds an expected amount of time (i.e., a timeout threshold), we can assume a node outage occurs. As an example, Fig. 3 shows the number of syslog entries generated by node 482 over one hour. A repetitive pattern of syslog entries is visible.

Syslog-based monitoring is not sufficiently accurate and results in many false positive node outages. The time precision of our monitoring data is 1 second. The minimum time between syslog entries is, however, less than 1 second. The maximum time, even for a single node, is not constant and might change over time as a result of modifications in hardware or software. In general, the time between syslog entries on Taurus nodes during the 8-month time period of this study was always below 10 minutes. When a problematic node instantly reboots after an outage, the time between two consecutive heartbeats will be less than the *expected mean time between* *syslog entries* (MTBSLE), in our case below 10 minutes, thus, contributing to a false negative by hiding the node outage.

The alternative method used in this study is tracking back the system reboots to avoid false positives and false negatives as much as possible. Using this method we first seek syslog entries which indicate a node reboot; then we track back in the syslog entries to find the last available syslog entry before the reboot line. We consider the last recorded syslog entry before a system reboot as the outage point. Although we know that the node might be still alive after its last syslog entry, since it is no longer responsive, we consider it to be out. On Taurus, the first observable entry in the syslog output after a successful boot of the operating system is `syslog-ng starting up; version *`. After finding all node outages using this alternative method we use a partial heartbeat tracking method (described earlier) to extract all remaining node outages from the syslog. In the partial heartbeat tracking we compare the timestamp of the last available syslog entry of each node against 30-04-2015 23:59:59. If the difference between these two timestamps is more than 10 minutes (i.e., MTBSLE), we assume a node outage.

V. Analysis and Correlation of Node Failures

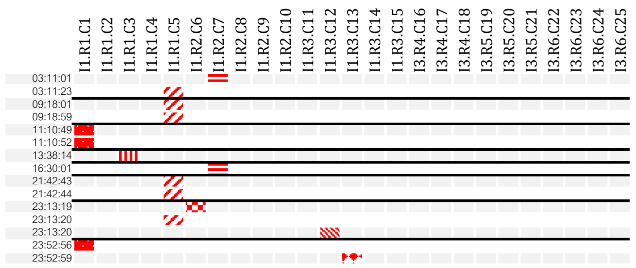
Among the three dimensions of failure correlation (time, space, and logic), the most relevant one is the logical dimension as it can help prevent re-occurrence of the same failure in the future, or even enable the prevention of failure propagation. However, this correlation is the hardest to infer. Oftentimes the logic behind a group of failures is so complex that we can easily miss the correlation, and therefore we might assume that the failures were uncorrelated (false negative correlation). In this study we attempt to clarify that whenever the logical correlation between node failures is not immediately visible, we can analyze the other two dimensions (time and space) of failure correlation and infer the existence or absence of a logical correlation between failures. Logical correlations have the highest potential for early node failure detection.

First insight: By definition, logical correlations are derived from strong correlations in space and time [11].

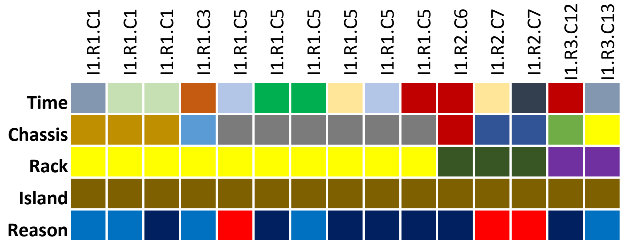
For the use cases included in this work, we selected only those node outages which we believe were not caused by general power outages or scheduled system maintenances. Therefore, these outages are considered as *node failures*. In the remainder of this section we refer to node failure simply as *failure*. We study several use cases which had the same type of correlations and select the most illustrative one as the representative of those types of correlations. In the following subsections we discuss four such use cases as the representative of their respective group of correlation.

1) First use case: Fig. 4a shows 15 failures during a period of 24 hours on December 6, 2014 on Taurus. The failures are divided into eight different temporal sections. Since two sections have only one failure, no temporal correlation applies. The remaining 6 sections indicate temporal correlations between failures. The color pattern also indicates the spatial failure correlation in all of the 8 sections. In this use case the temporal correlation is very weak, and with the exception of one section, the sections have only 2 failures each.

In Fig. 4b, color coding is used to show the correlation between failures in time and space. The last row indicates the similarity of reasons for failures (logical correlation) on each chassis, which we extracted from the syslog entries.



(a) Node failures over 24 hours on 06-12-2014. Temporally correlated failures are shown in several different horizontally divided sections. All failures with same color pattern are also spatially correlated, regardless of their section.



(b) Correlation of node failures over 24 hours on 06-12-2014. Temporal correlation can be a sign of identical reason for a group of failures only when it appears between 3 or more nodes. Otherwise, most likely the simultaneous occurrence of two failures even at the exact same moment is simply a coincidence.

Figure 4. First use case: Observed failures and their correlations.

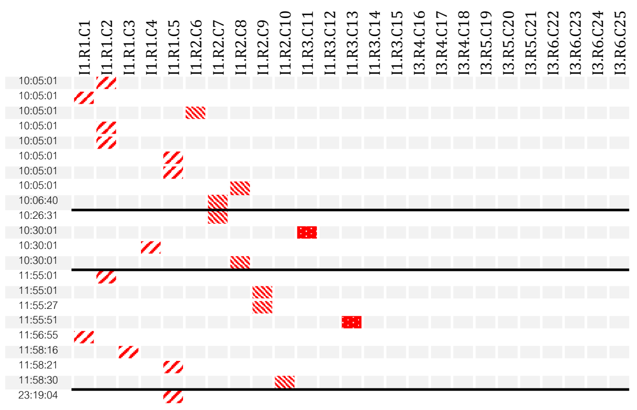
In all other similar figures in this section, the following patterns were found. (1) Failures with identical color in the *time* row occurred in less than a certain amount of time after a previous failure. (2) Failures with identical color in the *chassis*, *rack*, or *island* rows, occurred in the same chassis, rack, or island, respectively, as the failures preceding them in these locations. (3) Failures with identical color in the *reason* row, occurred due to the same reason as other failures on this day.

By looking at Fig. 4b, we can see that only the section with 3 failures has identical colors in the *reason* row. The failures occurring on 4 out of the 6 nodes located on I1.R1.C5 are spatially correlated and have the same reason for failure.

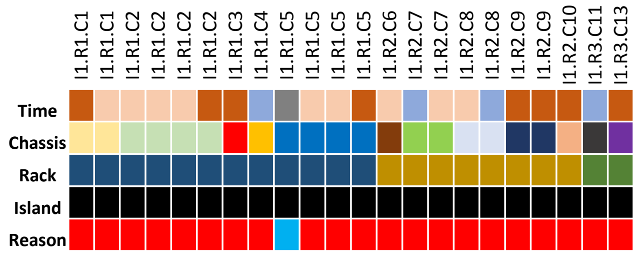
Lesson 1: On Taurus, if a temporal correlation is being detected only between two compute nodes, even if both failures occurred in the exact same moment, we cannot correlate the reason(s) behind those failures, and most likely their simultaneous occurrence is just a coincidence. A strong temporal correlation between failures occurring on three or more compute nodes, in many cases, implies an identical failure reason.

2) Second use case: In Fig. 5a we observe a group of 22 failures over 24 hours on April 21, 2015 on Taurus. Failures are divided into four temporal sections, out of which one of them has only a single failure and is, therefore, excluded from further correlation analysis. The first observation regarding the spatial correlation of failures in Fig. 5a is the fact that all failures occurred on the first island.

Fig. 5b shows that except for a single failure, the remaining 21 failures are caused by the same reason. Via backtracking the system logs, we realized that the 21 failures were raised by a problem in the distributed file system, and the failure with a different reason was due to an *out of memory* problem.



(a) Node failures over 24 hours on 21-04-2015. Temporally correlated failures are shown in three different horizontally divided sections. All failures with same color pattern are also spatially correlated, regardless of their section.



(b) Correlation of node failures over 24 hours on 21-04-2015. The number of correlated failures plays an important role. In each group of correlated failures, the dimension which has more correlations is the dominant one.

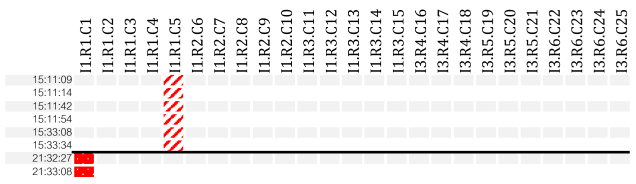
Figure 5. Second use case: Observed failures and their correlations.

Lesson 2: Complementary to **Lessons 1**, we learned that on Taurus, when the number of correlated failures is relatively high, the spatial correlation dominates. When inferring correlations and analyzing reasons of failures, the highest priority should be given to spatial correlations followed by temporal correlations. The chances of finding the same reason for spatially correlated failures are higher than in the case of temporally correlated failures.

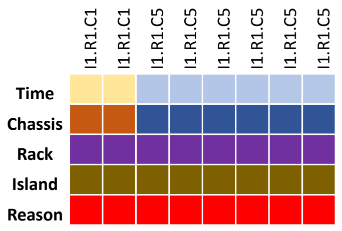
3) Third use case: Fig. 6a shows 8 failures over 24 hours on December 14, 2014 on Taurus. Based on their time of occurrence, failures are divided into a group of six and a group of two failures. By backtracking the system logs, we realized that all failures were due to an *out of memory* problem, which implies a problem related to a shared logical resource (e.g., a job). The temporal correlation in the second group is strong. A stronger correlation is observable in Fig. 6b along the spatial dimension.

Lesson 3: On Taurus, the combination of temporal and spatial correlations is highly revealing. In situations in which both strong temporal and spatial correlations are observable, the reason behind the failure is identical. This lesson can reveal the logical correlation of failures in situations which the logical correlation is not independently detectable.

4) Fourth use case: Fig. 7 shows all outages on Taurus during the 8-month period of 01-09-2014 to 30-04-2015. Since this use case covers all outages on Taurus, the small share of second island is also demonstrated. A similarity in the number of outages on many chassis of the same rack and same island is observable.



(a) Node failures over 24 hours on 14-12-2014. Temporally correlated failures are shown in two different horizontally divided sections. All failures with same color pattern are also spatially correlated, regardless of their section.



(b) Correlation of node failures over 24 hours on 14-12-2014. A strong temporal and spatial correlation in a group of failures, strongly implies an identical reason for all observed failures.

Figure 6. Third use case: Observed failures and their correlations.

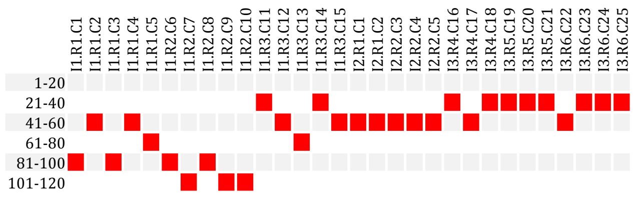


Figure 7. Fourth use case: Node failures over an 8-month period from 01-09-2014 to 30-04-2015. Chassis in the same rack and racks in the same island exhibit spatial correlation even over a longer period of time (no temporal correlation is made here).

Each red block indicates the total number of outages in a single chassis over the 8-month period of this study. General power outages and general planned maintenances affect the entire cluster. The number of planned partial maintenances is very limited. In Fig. 7 we group the outages in blocks of 20. Therefore, the correlation observable between outages illustrated in Fig. 7, is also a valid correlation between failures.

Lesson 4: On Taurus, if failures are not distinguishable from general power outages and planned maintenances, we can simply use node outages for analyzing the long term correlations between failures of the entire cluster, rather than using actual failure syslog entries.

Revisiting the above use cases, one can see that regardless of the correlation being temporal or spatial, in all cases the *reason* row in Fig. 4b, 5b, and 6b, follows almost the same color pattern as at least one of the other columns, implying logical correlation.

Early node failure detection can be achieved via two methods: (1) identifying the reason behind the failures (logical correlation) which is not always easy or (2) identifying the temporal and/or spatial correlation of failures, which indirectly reveals the logical correlations. Finally, we can state that all nodes which have the probability to have the same logical correlation will eventually encounter failures.

VI. Conclusion and future work

In this paper we report on the lessons learned while correlating node failures observed via the system logs obtained on a production HPC system over eight months. We learned that the logical correlation in general is derivable from the temporal and spatial correlation of failures. When a shared physical resource is responsible for a group of failures, we should observe failures with strong spatial correlation which are also temporally correlated. We recall that the temporal correlation is only meaningful when it is observed on more than two failures. When we observe a large group of failures, usually the spatial correlation plays the key role in identifying the main cause of failure.

This work is a first step in detecting and correlating node failures. Only based on a deeper understanding of the failure patterns and their correlations, effective failure recovery and prediction mechanisms can be devised. In general on Taurus many failures are temporally correlated, spatially correlated, or both, depending on the time interval within our examined 8-month time period. This variation is mainly caused by the different ways of using the system over time; this naturally leads to the need for logical correlation. The logical correlation is not always easy to infer. In this study we learned that logical correlation can more easily be revealed by further examination of the strong spatially-temporally correlated failures. This study is, however, not yet completed. To be able to detect early, diagnose, and correlate generic failures, further investigation and analysis is needed for a generic failure diagnosis and correlation methodology, which is part of our future work.

At the conclusion of the study, such a generic failure diagnosis and correlation methodology could be used to detect and prevent failures in a shorter time and more efficiently than the techniques used nowadays. In the next phase, we will study the logical correlation of failures. With knowledge of all three types of correlations (temporal, spatial, and logical) we will begin to identify correlation patterns that can help detect failures in their early stage or even predict them.

As future work further investigation of the logical correlation is planned. Also, the automation of the analysis workflow will be very beneficial. More accurate extraction of actual failures from all node outages is also necessary. In this work, we began the analysis using the available information, out of which, node outages were the most prominent types of failures. Therefore, for the analysis of future system logs we know how to interpret node reboots and we can proceed to analyze other types of node outages. We agree that the current results would be more accurate if other types of node outages would be taken into account. This is planned as part of our future work. Based on the current study we cannot claim that the discussed failures and correlations are representatives of other HPC systems. The focus of this paper is not to find similarities in the failure patterns we observe on Taurus with other HPC systems, but to develop a methodology that can be applied to other systems as well. During the time of writing of this study, a new computing system (Taurus2) has been installed at Technische Universität Dresden, as a second part of the system which we reported about (Taurus). Our hope is that lessons learned from Taurus will facilitate the failure monitoring-detection-prevention of Taurus2. Using the proposed analysis workflow on the system logs of other HPC systems and comparing their results with the lessons we learned during this study can provide a more general insight about HPC systems behavior and will help in early detection of failures.

Acknowledgement

This work is in part supported by the German Research Foundation (DFG) in the Cluster of Excellence "Center for Advancing Electronics Dresden" (cfaed) and in the Collaborative Research Center 912 "Highly Adaptive Energy-Efficient Computing" (HAEC). The authors also thank Holger Mickler of Technische Universität Dresden for his support in collecting the monitoring information on the high performance computing system.

References

[1] F. Cappello, A. Geist, and W. Gropp, "Toward Exascale Resilience: 2014 update," *Supercomputing Frontiers and Innovations*, vol. 1, no. 1, pp. 5–28, 2014.

[2] A. Gainaru, F. Cappello, M. Snir, and W. Kramer, "Failure prediction for HPC systems and applications: Current situation and open issues," *International Journal of High Performance Computing Applications*, vol. 27, no. 3, pp. 273–282, Jul. 2013.

[3] N. Yigitbasi, M. Gallet, D. Kondo, A. Iosup, and D. Epema, "Analysis and modeling of time-correlated failures in large-scale distributed systems," in *Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on*, Oct 2010, pp. 65–72.

[4] R. K. Sahoo, M. Squillante, A. Sivasubramaniam, and Y. Zhang, "Failure data analysis of a large-scale heterogeneous server environment," in *Dependable Systems and Networks, 2004 International Conference on*, June 2004, pp. 772–781.

[5] R. K. Sahoo, A. J. Oliner, I. Rish, M. Gupta, J. E. Moreira, S. Ma, R. Vilalta, and A. Sivasubramaniam, "Critical event prediction for proactive management in large-scale computer clusters," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '03. New York, NY, USA: ACM, 2003, pp. 426–435.

[6] Y. Liang, Y. Zhang, M. Jette, A. Sivasubramaniam, and R. K. Sahoo, "Bluegene/l failure analysis and prediction models," in *Dependable Systems and Networks, 2006. DSN 2006. International Conference on*, June 2006, pp. 425–434.

[7] M. Gallet, N. Yigitbasi, B. Javadi, D. Kondo, A. Iosup, and D. Epema, "A model for space-correlated failures in large-scale distributed systems," in *Euro-Par 2010 - Parallel Processing*, ser. Lecture Notes in Computer Science, P. DAmbra, M. Guarracino, and D. Talia, Eds. Springer Berlin Heidelberg, 2010, vol. 6271, pp. 88–100.

[8] X. Fu, R. Ren, S. Mckee, J. Zhan, and N. Sun, "Digging deeper into cluster system logs for failure prediction and root cause diagnosis," in *Cluster Computing (CLUSTER), 2014 IEEE International Conference on*, Sept 2014, pp. 103–112.

[9] S. Fu and C.-Z. Xu, "Quantifying event correlations for proactive failure management in networked computing systems," *J. Parallel Distrib. Comput.*, vol. 70, no. 11, pp. 1100–1109, Nov. 2010.

[10] D. Tang and R. Iyer, "Analysis and modeling of correlated failures in multicomputer systems," *Computers, IEEE Transactions on*, vol. 41, no. 5, pp. 567–577, May 1992.

[11] S. Ghiasvand, F. M. Ciorba, R. Tschüter, and W. E. Nagel, "Analysis of Node Failures in High Performance Computers Based on System Logs," Poster Supercomputing, 2015.