Investigating Event Log Analysis with Minimum Apriori Information

Adetokunbo Makanju, A. Nur Zincir-Heywood, Evangelos E. Milios
Faculty of Computer Science
Dalhousie University
Halifax, Nova Scotia, Canada. B3H 1W5.
+1-902-494-2093
{makanju, zincir, eem}@cs.dal.ca

***Abstract*—This thesis proposes a hybrid log alert detection scheme, which incorporates anomaly detection and signature generation to accomplish its goal. Unlike previous work, minimum apriori knowledge of the system being analyzed is assumed. This assumption enhances the platform portability of the framework. The anomaly detection component works in a bottom-up manner on the contents of historical system log data to detect regions of the log, which contain anomalous (alert) behaviour. The identified anomalous regions (after inspection by a human administrator through a visualization system) are then passed to the signature generation component, which mines them for patterns. Consequently, future occurrences of the underlying *alert* in the anomalous log region, can be detected on a production system using the discovered patterns. The combination of anomaly detection and signature generation, which is novel when compared to previous work, ensures that a framework which is accurate while still being able to detect new and unknown alerts is attained.**

***Index Terms*—Algorithms; Networked Systems; System Management; Modeling and Assessment**

I. Introduction

Autonomic computing can be described as the goal of building self-managing computer systems [1]. The need for self-managing computer systems has become more glaring with the ever increasing size and complexity of today's computer systems. At the heart of autonomic computing are four objectives which are sometimes referred to as self-* properties: i.e. self-configuration, self-optimization, self-healing and self-protection. To enable the autonomic system to meet these objectives it must also posses at least one of the following attributes: self-awareness, self-situation, self-monitoring and self-adjustment [2].

A system which is capable of self-healing is one that is capable of detecting, diagnosing and recovering from its fault conditions with minimal human intervention. The first step in this process is alert detection. Alert detection involves the discovery of the symptoms of the fault condition. Several sources of information are available for the detection of fault symptoms on computer systems and networks e.g. system log files, system activity reports, trouble tickets, co-workers, off-site sources and system activity paths. However, system logs stand out due to the fact that they play a crucial role in manual fault resolution and are also the largest onsite information source (by volume) available to system administrators.

This thesis[1] aims to develop a hybrid interactive learning framework for alert detection in system logs. This proposed framework attempts to automatically detect computer system error conditions (alerts) in system logs. The computer systems can either be on wired or wireless infrastructures, as the assumption here is that logs have been reported by the application layer or generally components running above Layer-2 in the protocol stack. We differentiate here between errors as symptoms of a fault and the actual faults. Faults usually leave traces on systems before and after they occur. These traces manifest themselves in the form of errors (alerts) in the system. The goal here is the automatic identification of these error conditions, thereby reducing troubleshooting time and preventing downtime events altogether.

System log files pose a lot of challenges for the task at hand. One is their semi-structured nature and another is their diversity. Another possible challenge is that log files may not capture fully information about failure in all cases. Moreover, application developers cannot anticipate all fault conditions which can occur. Therefore, this framework will employ a mix of data mining and information-theoretic techniques to overcome some of these challenges. While the focus of this research is fault management, the intention is to produce a framework which is general enough to apply to other network management functions. The system can analyze its own logs automatically and provide hints to the administrator about possible error conditions. Once error conditions are confirmed, the system can develop signatures for such errors and flag them when they occur in the future. Using the objectives [1], attributes [2], design approaches [3] and framework types [4] associated with autonomic computing, Table I places the proposed framework within the context of autonomic computing. We also state that the system has an autonimicity level of Level-3 based on the five level hierarchy proposed by IBM [5].

The overall goal of the research carried out in this thesis would be to design and evaluate a framework such as specified in Table I. The analysis will take various forms but will utilize a mix of unsupervised and supervised techniques while

[1]Downloadable from http://web.cs.dal.ca/∼makanju/files/Makanju-Adetokunbo-PhD-CSCI-April-2012.pdf

978-3-901882-50-0

TABLE I
PLACING THE FRAMEWORK WITHIN THE CONTEXT OF AUTONOMIC COMPUTING

Property	Category
Objective	Self-Healing, Self-Protection
Attribute	Self-Awareness, Self-Monitoring
Design Approach	External
Framework Type	Technique-Based, Injection of Autonomicity into Non-Autonomous Systems
Autonomicity Level	3-Predictive

keeping human involvement to a minimum. The components will be self-contained but could be interrelated as well, i.e. the output from one could serve as input to the other. To this end, the following research objectives are set.

- **Minimum Apriori Information.** Platform portability is important for any effective framework for alert detection in system logs. For a system to be platform portable, it must not rely on system specific characteristics and must assume very little knowledge of the architecture of the system it would be monitoring, i.e. minimum apriori information.
- **Unstructured Data.** Log files contain unstructured data in the form of the natural language descriptions of events, i.e. messages. This unstructured content can serve as a stumbling block to automatic analysis. As stated in [6], abstracting these natural language descriptions is fundamental and contributes greatly to accuracy of analysis. However, abstraction is not a straightforward process; this thesis will develop techniques to deal with the unstructured content of system logs.
- **Interactive Learning.** Due to the difficulty and uncertainty associated with automatic analysis performed by frameworks such as proposed by this thesis, human input is necessary [7]. Thus our framework will include a visualization component which allows human administrators to view the results of the analysis of the system and provide feedback to the system.
- **Hybrid Detection.** Systems which carry out automatic alert detection in system logs can be classified typically as either signature-based or anomaly-based detectors, each approach has its pros and cons. The framework we plan to develop combine both approaches so as to take advantage of the strengths of both approaches.
- **Self-Awareness.** We hope to have our framework contribute to self-awareness in computing systems by developing techniques which can be used to detect not just alert states but all system states which are discernible in a system log.
- **Computational Cost.** One of the major hindrances to the automation of system logs is their size, which makes the application of certain techniques impracticable in this domain. Hence it is one of the goals of this thesis to ensure that most of the techniques used are simple and have low computational complexity.

II. BACKGROUND AND METHODOLOGY

System logs seem to have received the most attention as a source of information for automatic alert detection and system monitoring [8]. Hence, it is not surprising that there are several commercial and open source tools which aid system administrators in the monitoring of their event logs. Examples of such tools include Splunk [9] and Sisyphus [10].

Nevertheless, these tools are still incapable of of automating the process of log monitoring and alert detection fully, are most times non-interactive and assume a great degree of knowledge about the system being monitored. Therefore, for the most part, these tasks are still carried out manually by system administrators. A lot of research is still required to develop tools and techniques which will bring the level of automation to the required level. These research efforts will have to focus on one or more of these problems i.e. unstructured message analysis, indexing/feature creation, event correlation and anomaly detection. The research performed in this thesis intends to contribute to approaches for dealing with each of these problems.

With these limitations in mind, the framework which this thesis aims to design and develop can be described as a hybrid interactive learning framework for alert detection in system logs. An overview of what the architecture of such a system may be is given in Fig. 1. At the core of the proposed system is its anomaly detection mechanism. The anomaly detection mechanism analyzes the contents of event logs, in the process, generating event log clusters which define the different system states or behaviors that are discernible in the log. It also determines whether these clusters are likely to contain normal or anomalous states. Then it presents these clusters to an administrator using a visualization system after which the administrator can then confirm the anomalous clusters detected by the system and provide labels. Subsequently, the system can then send the events in these labelled anomalous clusters to a signature generation system which generates a detection signature for the alert state(s) represented in the clusters. These signatures will be used to detect future occurrences of the alert state. Meanwhile, the anomaly detection component continues to use the feedback it gets from the administrator and the signature generation system to improve its anomaly detection capability and the cycle then continues. In the next few sections we will discuss our proposed methodology for each of the components of the system.

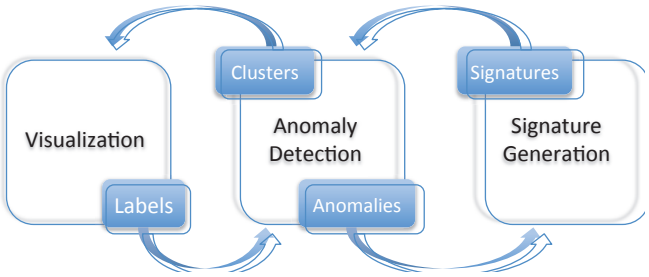

Fig. 1. Overview of the hybrid interactive learning framework.

III. Visualization with Treemaps

While log files are invaluable to a network administrator, the vast amount of data they contain can overwhelm a human and can hinder rather than facilitate the tasks of an administrator. Visualization is an effective means of aiding humans to make sense of large amounts of data and could prove useful with event logs. Visualization can also prove useful for analysis system when data is dynamic and interaction is required between the system and a human operator, in our case the system administrator. For these reasons a visualization component is included in our framework.

In our thesis we propose a dynamic an interactive tool called LogView, which can play the part of the visualization component in the framework proposed in this thesis [11]. The LogView prototype was built using the prefuse visualization toolkit [12]. The prefuse visualization toolkit is a software framework written using the Java2D graphics library. It provides reusable building blocks, for the building of custom visualization tools.

One of the biggest decisions that had to be made with the design of LogView was the visualization technique to utilize. After careful consideration of the fact that we needed to visualize the content and clusters of the event log as a hierarchy we settled on Treemaps as the visualization technique of choice [13]. A treemap is a methodology for the visualization of hierarchical data which uses a 2-dimensional space filling approach. Treemaps provide an alternative to the node-link structure diagrams traditionally used for visualizing hierarchical data.

With the inclusion of a visualization system, a human administrator can now interact with system by providing labels, i.e. textual description and categorization. The categorization labels determine if the clusters are *normal* or *anomalous* and can affirm or override the categorization determined by the system. The system can use these labels to generate signatures for clusters that have been confirmed as *anomalous* and all other labels to improve its anomaly detection capability. We are of the opinion that the level of user interaction would be minimal, as the system does not require a complete labeling of all clusters to learn their signatures. A label on a single cluster is enough for the system to learn a signature if need be.

IV. The Proposed Anomaly Detection Framework

The anomaly detection mechanism is at the core of the alert detection framework. The novel framework proposed by this thesis for anomaly detection in system logs is called STAD i.e Spatio-Temporal Alert Detection. It works on the assumption that *"System logs events which are produced by similar spatial sources or produced during periods of similar system activity are likely to be similar,"*. The main contributions of the proposed system are: (i) the extension of entropy-based approaches to alert detection is system logs that allow the detection of alerts without resorting to ranking schemes; and (ii) the enabling the analysis of a group of dissimilar nodes, which means that it can be applied to distributed systems.

An overview of the approach is provided in Fig. 2. Overall, each step of the approach is general enough to allow flexibility in the choice of methods used. The phases in the proposed framework are described in the following sections.

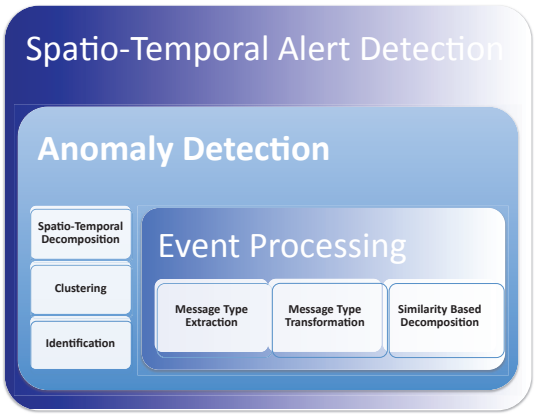

Fig. 2. Overview of the STAD framework.

A. Event Processing

The goal of this phase of the framework is the extraction of structure from the unstructured component of the logs. Event logs contain semi-structured information from heterogenous sources. This fact coupled with the ambiguity introduced by the semantics of terms used by application developers makes this step important. The event processing phase of the framework is sub-divided into three components which are Message Type Extraction, Message Type Transformation and Similarity-based Decomposition.

1) **MessageType Extraction.** The goal of message type extraction is the discovery of textual templates that define the semantic clusters in the unstructured component of the logs. We developed an algorithm called IPLoM (Iterative Partitioning Log Mining) to carry out this task in our implementation [14].
2) **Message Type Transformation.**
Using the message type templates discovered during the message type extraction step, the messages in the event data are transformed so that a more concise representation

is achieved. We utilized Full Message Type Transformation in our implementation.

3) **Similarity-based Decomposition.** This step of the process attempts to decompose the log into more homogenous units to allow for useful analysis in line with the basic assumption of the approach. In our implementation we utilized several approaches to carry out similarity-based decomposition. This include decomposition based on node function and decomposition based on reporting application.

B. Anomaly Detection

The goal of this phase of the framework is to identify the portions of the log which contain anomalous events. This phase has three components which are Spatio-Temporal Decomposition, Clustering and Identification.

1) **Spatio-Temporal Decomposition.** This component decomposes the content of any log based on source and time information. After decomposition each resultant unit will contain log information from a single source over a unit of time. The basic unit of spatio-temporal decomposition used throughout the thesis is the nodehour [15], which is one hour of log information from a single node on the networ. This thesis also argues that spatio-temporal decomposition can be used in the discovery of correlated message types [16].
2) **Clustering.** The goal here is to group the spatio-temporal units such that members of each group are very similar to each other while being very dissimilar from the members of other groups. We utilized information content clustering (ICC) for this step in our implementation [16]. ICC is a contribution of this thesis, it utilizes the entropy-based information content score assigned to nodehours using the *Nodeinfo-Uniq* equation [17] as means of clustering the nodehours.
3) **Identification.** This step aims to identify those clusters containing anomalous spatio-temporal units. Identification was carried out in our implementation using a rule-base that tests a cluster to determine to what degree it exhibits properties of an anomalous clusters. The properties which we identified include the bursty property, the epidemic property, the endemic property and the near-periodic property [18].

V. Signature Generation

The input to the signature generation mechanism, see Fig. 3, consists of the nodehours that belong to the cluster(s) that have been identified by the human administrator as containing traces of activity that relate to a particular alert type. This action would only require the administrator going over the clusters identified by the system as being anomalous. A cluster pruning step is then performed. During this step, the set of message types reported in each nodehour is pruned by iteratively computing the difference between the set of message types in the nodehour and the cluster centroids for each of the clusters identified as *normal* by STAD. Our method for choosing cluster centroids can be found in [16].

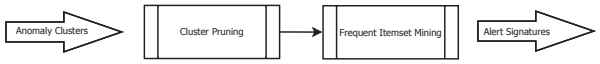

Fig. 3. Signature Creation Mechanism: This figure shows the steps in the signature creation phase of the framework.

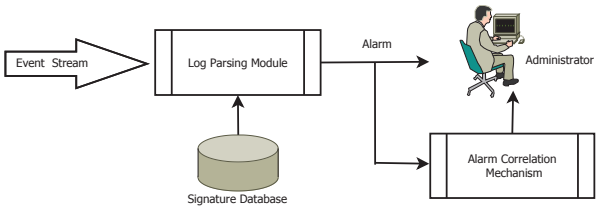

Fig. 4. Online Alert Detection: This shows how the alert signatures produced by our system can be used for online alert detection on a production system.

After pruning, frequent itemset mining is performed on the nodehours. Given a set of objects or items (called an item base), S, we define $\mathbf{T}$ as a set of transactions defined over S such that for all $T \in \mathbf{T}$, $T \subseteq S$. A subset (also called an itemset) S' of S is said to be *frequent* if the number of transactions in $\mathbf{T}$ that are supersets of S' exceed a user-specified support threshold and is referred to as a *frequent itemset*. The goal of frequent itemset mining is to find all itemsets that occur in $\mathbf{T}$ with a minimum support threshold. The apriori algorithm is a classical algorithm for frequent itemset and association rule mining [19] and was be used in our implementation.

We apply the frequent itemset mining paradigm to the problem of alert signature generation by supposing that message types are items and a nodehour is a transaction. Hence the set of message types found in a system log is the item base and the transaction database is the set of nodehours found in the log. We assume that for any set of related transactions (nodehour cluster) in the transaction database (system log), the set of frequent itemsets that occur in the transaction cluster would be an effective signature for identifying future occurrences of that transaction type. If a transaction cluster is related to an alert type, then the frequent itemsets mined from such a cluster would form a signature for that alert.

Once the signatures are generated, they can be stored in a database and used to detect future alerts on a production system, Fig. 4. This database will of course be updated as more up-to-date signatures are discovered. The log parsing module would parse the log event stream, searching for any events (or log partitions) that match any of the signatures in the database. If a match is found, an alarm is raised. The alarms can then be passed directly to the administrator for action. Thus, we have a system that can automatically search its own logs for symptoms of failure and notify the administrator. Unlike other systems that do this using signatures that are produced and managed manually, our proposed system produces the signatures itself, while only relying on the administrator for class labels.

VI. Results

We performed a number of independent experiments to test the various components of our framework. Each experiment involved an implementation and the testing of the implementation against log data collected from actual systems. We did not use simulated log data in any of experiments. We then evaluate the results of test either by comparison against ground truth or directly through the use of appropriate metrics. In all we performed experiments on combined log data containing in excess of 0.75×10^6 log events collected on a variety of system infrastructures including High Performance Clusters (HPC), cloud-based systems and distributed systems. The ground truth used differed for each evaluation, however most of the ground truth was derived from labelling carried out by domain experts i.e. system administrators of the systems from which the data was collected [20], [21], [22]. We did not set the ground truth ourselves except in a few cases where it was not provided but this accounts for less than 5% of the log data that was analyzed. We provide summary results of these evaluations below.

- **Message Type Extraction.** Our proposed message type extraction algorithm i.e. IPLoM not showed that it could be more accurate than previous approaches [14], [20], it was able to achieve an F-Measure of 91% based on micro-average classification accuracy when tested against log data from one of the fastest supercomputers in the world [23].
- **Message Type Transformation.** We demonstrated the importance of message type transformation to the analysis process by showing that utilizing message type transformation in conjunction with a previous log data analysis method i.e Nodeinfo [15] that we could achieve up to 100 fold reduction in computational effort without a drop in detection performance [24]. We also demonstrated the utility of message type transformation to the effective storage and retrieval of system log data [25].
- **Clustering.** We developed the Nodeinfo_Uniq method for information content assignment [17] which is used in our proposed information content clustering scheme. The evaluations show that ICC was not only able to produce well formed clusters (cluster separation and cluster cohesiveness), the clusters produced could also be mapped to different alert states with a high degree of confidence (conceptualization) [16].
- **Anomaly Detection.** Our anomaly detection mechanism was able to achieve an average detection rate of 78% of all administrator identified alerts at a false positive rate of 5.4% [18]. This was achieved without the use of user feedback. Further tests still need to be carried out to determine how the system will perform when user feedback is involved. The time span covered in most of the log files was significant i.e. 6-18 months, this would imply that the anomaly signatures discovered have the potential to be relevant over long time periods. However, further tests need to be carried out to validate this.
- **Signature Generation.** The signature generation evaluations demonstrated that once an alert cluster is identified an effective signature that is able to detect about 83% of all future occurrences of the alert with negligible false positives can be automatically deduced by the system [22].

VII. Conclusion and Future Work

The objectives of the work carried out in this thesis are highlighted in Section I. We conclude by detailing the contributions of the thesis toward meeting each of these objectives. The framework is interactive as it includes a visualization component through which feedback can be given. It is hybrid in that it combines anomaly detection and signature generation. These components of the system work through a bottom-up approach to detect alerts in the log. The approach involves several techniques which are used for message type extraction, message abstraction, entropy-based analysis, clustering, identification of anomalous clusters and signature generation. Most of the techniques are novel contributions of this thesis. Further details of the contributions are given below.

- **Minimum Apriori Information.** The alert detection framework proposed by this thesis assumes very little about the infrastructure on which it will run, it can detect alerts successfully without semantic analysis. Indeed, the only significant piece of apriori information which the system needs to be aware of are the similarity categories used for anomaly detection. Automating this step through an analysis of the message types produced by different *sources* can be an interesting direction for future work.
- **Unstructured Data.** The IPLoM message type extraction algorithm is an important technical contribution of this thesis. IPLoM does not require apriori knowledge of the domain from which the log data emanates, as is required by some previous approaches which discover message types by parsing source code or searching the log data for patterns of well known variable tokens such as IP addresses. Our experiments indicate that IPLoM may have linear complexity with respect to the number of messages in the logs, thus reducing the computational expense associated with message type extraction. Furthermore, it is more accurate than previous approaches.
- **Interactive Learning.** This thesis proposes a scheme which allows log data to be visualized as a hierarchy of clusters using treemaps [13]. This scheme is implemented in a prototype interactive visualization tool called LogView. The interactive nature of the LogView prototype ensures that administrators can provide feedback to the anomaly detection mechanism. This allows the framework developed in this thesis to be implemented as an interactive learning framework, which is an improvement on previous work.
- **Hybrid Detection.** The proposed framework combines STAD (anomaly detection framework) with a signature-based approach based on frequent itemset mining. The signature generation system generates signatures from

identified anomalies. The proposed framework is thus a hybrid approach and takes advantage of both approaches.

- **Self-Awareness.** This thesis makes a contribution to enhancing self-awareness through the mining of system logs by correlating events which can be found by decomposing the log spatio-temporally. Correlated events in the log define different states of a system, however discovery can be computationally expensive. This thesis proposes a method based on information content clustering for the the timely discovery of an accurate initial set of system states which can form the basis for further analysis.
- **Computational Cost.** Most the techniques introduced in this thesis are either linear or pseudo-linear in time and memory cost, thus we make a significant contribution to creating a log analysis framework that scales gracefully in the face of large data. Also, most of the analysis is carried out on historical log data (hence it can be carried out offline), while the domain knowledge derived from the analysis is used for real-time monitoring of the system. The time spent analyzing historical data should reduce as the system becomes stable, as only novel data i.e. data that cannot be classified using existing domain knowledge will need to be analyzed.

Future work includes automating the process of identifying similar event sources, automating the process of parameter setting which will enhance its flexibility and automating the discovery of the alert detection rules used by STAD. Other directions include adapting the framework into an ensemble that uses information from other sources such as system metrics and system activity paths [26], [27] and modifying the framework to be capable of detecting activity which relate to security incidences and not just faults as is the case at the moment. The results of this thesis appear in the following papers [11], [14], [23], [24], [17], [25], [21], [16], [20], [18], [22].

ACKNOWLEDGEMENTS

The authors would like to thank Markus Latzel and Steve Stergiopoulos of Palomino Systems Inc. and Raffael Marty of PixlCloud for their support in completing this thesis. This research is supported by a Natural Science and Engineering Research Council of Canada (NSERC) Strategic Project Grant. This work is conducted as part of the Dalhousie NIMS Lab at http://www.cs.dal.ca/projectx/.

REFERENCES

[1] J. Kephart and D. Chess, "The Vision of Autonomic Computing," *Computer, Monthly publication of the IEEE Computer Society*, vol. 36, pp. 41– 50, June 2003.

[2] S. Dobson, R. Sterritt, P. Nixon, and M. Hinchey, "Fulfilling the Vision of Autonomic Computing," *Computer, Monthly publication of the IEEE Computer Society*, vol. 43, no. 1, pp. 35–41, January 2010.

[3] D. Garlan, S. Cheng, A. Huang, B. Schmerl, and P. Steenkiste, "Rainbow: Architecture-based self adaptation with reusable infrastructure," *IEEE Computer*, vol. 37, no. 10, pp. 46–54, October 2004.

[4] A. Khalid, M. Haye, M. Khan, and S. Shamail, "Survey of frameworks, architectures and techniques in autonomic computing," in *Autonomic and Autonomous Systems, 5th International Conference on*, vol. 0. Los Alamitos, CA, USA: IEEE Computer Society, 2009, pp. 220–225.

[5] M. Huebscher and J. McCann, "A survey of autonomic computing—degrees, models, and applications," *ACM Comput. Surv.*, vol. 40, no. 3, pp. 1–28, 2008.

[6] L. Huang, X. Ke, K. Wong, and S. Mankovskii, "Symptom-based problem determination using log data abstraction," in *Proceedings of the 2010 Conference of the Center for Advanced Studies on Collaborative Research*, ser. CASCON '10. New York, NY, USA: ACM, 2010, pp. 313–326. [Online]. Available: http://doi.acm.org/10.1145/1923947.1923979

[7] S. Amershi, A. Lee, B.and Kapoor, R. Mahajan, and C. Blaine, "Human-Guided Machine Learning for Fast and Accurate Network Alarm Triage," in *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI 2011, Barcleona, Spain)*, July 2011, pp. 2564–2569.

[8] J. Stearley, "Towards Informatic Analysis of Syslogs," in *Proceedings of the 2004 IEEE International Conference on Cluster Computing*, 2004, pp. 309–318.

[9] Splunk Inc., "Splunk: Operational Intelligence Software," Published online at http://www.splunk.com/product. Accessed, April 2012. [Online]. Available: http://www.splunk.com/product.

[10] J. Stearley, "Sisyphus Log Data Mining Toolkit," Published online at http://www.cs.sandia.gov/sisyphus. Accessed, April 2012. [Online]. Available: http://www.cs.sandia.gov/sisyphus

[11] A. Makanju, S. Brooks, A. Zincir-Heywood, and E. Milios, "Logview: Visualizing Event Log Clusters," in *Proceedings of Sixth Annual Conference on Privacy, Security and Trust (PST)*, October 2008, pp. 99 – 108.

[12] Sourceforge.net, "The prefuse Visualization Toolkit," Published online at http://www.prefuse.org. Last Accessed,, April 2012. [Online]. Available: http://www.prefuse.org

[13] B. Shneiderman, "Tree Visualization with Tree-Maps: A 2-D space filling approach," in *ACM Transactions on Graphics.*, vol. 2, 1992, pp. 92–99.

[14] A. Makanju, A. Zincir-Heywood, and E. Milios, "Clustering Event Logs Using Iterative Partitioning," in *Proceedings of the 15th ACM Conference on Knowledge Discovery in Data*, July 2009, pp. 1255–1264.

[15] A. Oliner, A. Aiken, and J. Stearley, "Alert Detection in System Logs," in *Proceedings of the International Conference on Data Mining (ICDM). Pisa, Italy.* Los Alamitos, CA, USA: IEEE Computer Society, 2008, pp. 959–964.

[16] A. Makanju, A. Zincir-Heywood, and E. Milios, "System State Discovery via Information Content Clustering of System Logs," in *Proceedings of the 2011 International Conference on Availability, Reliability and Security, ARES 2011. Vienna, Austria.*, August 2011, pp. 301–306.

[17] ——, "An Evaluation of Entropy Based Approaches to Alert Detection in High Performance Cluster Logs," in *Proceedings of the 7th International Conference on Quantitative Evaluation of SysTems (QEST)*, September 2010, pp. 69–78.

[18] ——, "Spatio-Temporal Decomposition, Clustering and Identification for Alert Detection in System Logs," in *Proceedings of the 27th ACM Symposium on Applied Computing*, ser. SAC '12. New York, NY, USA: ACM, March 2012.

[19] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," in *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, J. B. Bocca, M. Jarke, and C. Zaniolo, Eds. Morgan Kaufmann, 12–15 1994, pp. 487–499.

[20] A. Makanju, A. N. Zincir-Heywood, and E. Milios, "A Lightweight Algorithm for Message Type Extraction in System Application Logs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 11, November 2012.

[21] A. Makanju, A. Zincir-Heywood, and E. Milios, "A Next Generation Entropy-based framework for Alert Detection in System Logs," in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, May 2011, pp. 626–629.

[22] ——, "Spatio-Temporal Decomposition, Clustering and Identification for Alert Detection in System Logs," in *Proceedings of the 13th IEEE/IFIP Network Operations and Management Symposium*, ser. NOMS '12. IEEE Computer Society, April 2012.

[23] ——, "Extracting Message Types from BlueGene/L's Logs," in *22nd ACM Symposium on Operating Systems Principles, Workshop on the Analysis of System Logs (WASL 2009).*, 2009.

[24] A. Makanju, A. N. Zincir-Heywood, and E. E. Milios, "Fast Entropy Based Alert Detection in Super Computer Logs," in

International Conference on Dependable Systems and Networks Workshops (DSN-W), ser. DSNW '10. Washington, DC, USA: IEEE Computer Society, June 2010, pp. 52–58. [Online]. Available: http://dx.doi.org/10.1109/DSNW.2010.5542621

[25] A. Makanju, A. Zincir-Heywood, and E. Milios, "Storage and Retrieval of System Log Events using a Structured Schema based on Message Type Transformation," in *Proceedings of the 26th ACM Symposium on Applied Computing (SAC)*, March 2011, pp. 528 – 533.

[26] I. Cohen, S. Zhang, M. Goldszmidt, J. Symons, T. Kelly, and A. Fox, "Capturing, indexing, clustering, and retrieving system history," in *Proceedings of the twentieth ACM symposium on Operating systems principles*, ser. SOSP '05. New York, NY, USA: ACM, 2005, pp. 105–118. [Online]. Available: http://doi.acm.org/10.1145/1095810.1095821

[27] M. Y. Chen, A. Accardi, E. Kiciman, J. Lloyd, D. Patterson, A. Fox, and E. Brewer, "Path-based Failure and Evolution Management," in *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation - Volume 1*. Berkeley, CA, USA: USENIX Association, 2004, pp. 23–23. [Online]. Available: http://portal.acm.org/citation.cfm?id=1251175.1251198