Analyzing System Logs: A New View of What's Important

Sivan Sabato, Elad Yom-Tov, Aviad Tsherniak
IBM Haifa Labs,
Haifa University Campus,
Haifa 31905, Israel

Saharon Rosset
IBM T.J. Watson Research Center
Yorktown Heights,
NY 10598

Abstract

System logs, such as the Windows Event log or the Linux system log, are an important resource for computer system management. We present a method for ranking system log messages by their estimated value to users, and generating a log view that displays the most important messages. The ranking process uses a dataset of system logs from many computer systems to score messages. For better scoring, unsupervised clustering is used to identify sets of systems that behave similarly. We propose a new feature construction scheme that measures the difference in the ranking of messages by frequency, and show that it leads to better clustering results. The expected distribution of messages in a given system is estimated using the resulting clusters, and log messages are scored using this estimation. We show experimental results from tests on xSeries servers. A tool based on the described methods is being used to aid support personnel in the IBM xSeries support center.

1 Introduction

System logs, such as Windows Event Logs or Linux system logs, are an important resource for computer system management. These logs hold textual messages emitted from various sources in the computer system during its day-to-day operation. Emitted messages may be informational, or they can indicate a problem in the system, whether trivial or more serious.

Periodic monitoring of system logs by system administrators allows the identification of anomalies and security breaches in the system. In addition, the information in system logs is vital for problem diagnosis. In reality, however, system logs hold a large number messages, most of which are not interesting to the user. It is time-consuming and sometimes impossible to manually find the valuable messages in this abundance of information.

Previous works on the subject of log analysis present a variety of approaches. One approach is to have a human expert define a set of message patterns to find, along with desired actions to be taken when encountering them ([5], [6], [13]). The effort invested in writing and maintaining these rules is proportional to the number of message types and the rate at which they change. Another approach for log analysis focuses on visualizing the log data in a useful way ([2], [11]). This is achieved, for instance, by showing a succinct representation of the log data, by graphically showing patterns in the data or by presenting time statistics of messages.

Works differ in the type and extent of pattern detection applied to log data. Some of the techniques are analysis of the frequency at which message occur [11], grouping of time correlated messages ([10], [7]), and the use of text analysis algorithms to categorize messages ([10], [7]). Unlike the approach we present here, all these works base their analysis only on the log data of the inspected computer system.

In this paper we present a method for ranking log messages by their estimated value to users, based on information from a large population of computer systems. We generate a new *ranked log view*, in which the messages are shown in order of rank and in a condensed form. We applied our method on a dataset of the combined Windows Event Log (Security, Application and System messages) taken from 3,000 IBM xSeries servers that are used for diverse purposes. A characteristic Event Log holds between 3,000 and 30,000 messages. We show that using a new feature construction scheme, we can find a structure in the logs of computer systems to improve ranking.

The rest of the paper is organized as follows: In Section 2 we describe our method for scoring log messages and its use of clustering as a building block. In Section 3 a new feature construction scheme for sample data is introduced. This scheme achieves better clustering results in the message ranking scenario. In Section 4 we describe the experiments and analyzes the results. We summarize in Section 5.

2 Ranking Messages by Exceptionality

Given a system log of a computer system, we generate a summarized ranked view of this log. This view can help administrators and support personnel to identify and diagnose problems in the computer system more effectively, by displaying a much shorter log view of messages ordered by their importance to the user. A tool based on the described methods is being used to aid support personnel in the IBM xSeries support center.

To generate the ranked log view from the original log of a computer system, we first group the messages in the original log into mutually exclusive sets that correspond to message types. A message type is characterized by a base string that generates all the messages of this type, though possibly with different parameters. Grouping into types is trivial if the original log specifies the source and unique identification of each message, as in the Windows Event Log. Identifying message types without this information is a challenge that we do not address in this paper. (See [10], [7] for some approaches to this problem.) We henceforth refer to messages of the same type as instances of the same message, though the string parameters may differ between instances.

In the ranked log view, a single log line is displayed for each message type that appeared in the original log. This line lists the number of message instances, the largest common string pattern of the message instances, and the time-range in which the message instances appeared. Ranks are assigned to each message type and the lines are sorted in order of rank.

Our ranking method is based on the premise that a message in a system log is more important to the user if it has more instances in the log than is expected for this computer system. This is based on the idea that although it is possible that many computer systems have some problems reported in the system log, it would usually not be the *same* problem in all systems. To formalize this notion, let us represent system log i by a vector $\vec{c_i} = (c_i[1], \ldots c_i[n])$, where n is the number of possible message types, and $c_i[m]$ is the number of instances of message m in system log i.[1] Also, let $P = \{p_1, \ldots, p_n\}$ be a set of probability cumulative distribution functions $p_m : \mathbb{N} \rightarrow [0, 1]$, where $p_m(c)$ is the probability that message m would appear c or less times in a system log. If the probability of getting more than $c_i[m]$ instances of message type m is low, then the number of appearances of message m is more than expected, and therefore message m should be ranked higher. Therefore, the ranking of messages should approximate an ascending ordering of $(p_1(c_i[1]), \ldots p_n(c_i[n]))$.

Given a large enough dataset of system logs from actual computer systems, we can estimate P from the empirical distribution $\hat{P} = \{\hat{p}_1, \ldots, \hat{p}_n\}$ of the number of instances of each message type in each system. We define the *Score* of message type m in a log i to be $\hat{p}_m(c_i[m])$, and use this score to rank the messages within the log.[2] The messages that are top-ranked by this method usually indicate important problems in the system. This is illustrated in the ranked log view in Table 1, which was generated from one of the samples in our dataset.

The estimation of P using the empirical distribution of the entire population is based on the implicit assumption that the population of computer systems in our dataset is homogeneous enough to treat all of them as generated from the same distribution. In actuality, different computer systems are used for very different purposes. Each purpose dictates a use-model that results in a different message distribution. For example, a computer system that serves as a file-server would probably be more likely to issue 'File Not Found' messages than a personal workstation. On the other hand, a personal workstation might issue more system-restart messages.

To improve the accuracy of our estimation of P, we group the computer systems in our dataset into sets of systems with a similar use-model, and estimate P separately for each set. We group the systems using k-means clustering [1] on the system log dataset. To generate the ranked log view for a given system, we first find the cluster it belongs to, and then rank its log messages based on the estimation of P for that cluster. In the following section, we present a new feature construction scheme for the system log dataset. This scheme achieves a significantly better clustering than the original feature-set.

3 Using Rank Correlation for Feature Construction

In the original feature-set of our dataset, system log i is represented as the message count vector $\vec{c_i}$ defined above. There are 15,000 message types in our dataset, hence this results in a $3{,}000 \times 15{,}000$ matrix. This matrix is very sparse; only about 0.6% of the entries are non-zero. The

[1] For the sake of simplicity, we ignore here the timing of messages. Because system logs sometimes span a long time period, it would generally be necessary to address this issue, for example, by processing only message instances that occurred within a certain time frame.

[2] The method of using tf-idf weights [8] to rank words in documents in information retrieval engines, is comparable to our scoring method though it uses a different formula.

Rank	Times	Source	Message
1	10	ViperW2K	**The device, \Device\Tape1, has a bad block.**
2	4	Oracle.cq1	Audit trail: ACTION : 'CONNECT' DATABASE USER: '/' PRIVILEGE : SYSOPER ...
3	1	SAPCQ1_20	SAP Basis System: Run-time error "TIME_OUT" occurred
4	1014	SAPCQ120	SAP Basis System: Transaction Canceled 00 158 ( )
5	1	MRxSmb	**Delayed Write Failed ... may be caused by a failure of your computer hardware ...**
6	8	ql2300	**The device, \Device\Scsi\ql23002, did not respond within the timeout period.**
7	54	DnsApi	The system failed to register pointer (PTR) resource records (RRs) for network adapter ...
8	1	Kerberos	The kerberos subsystem encountered a PAC verification failure. ...
9	1	Windows Update Agent	Installation Failure: Windows failed to install the following update with error ...
10	1	NETLOGON	The Netlogon service could not read a mailslot message from The system ...

Table 1: A Ranked log view of an actual system (with shortened messages). Bold font indicates hardware problems.

high dimensionality of the data and its sparseness make k-means clustering impractical for this representation.

Since our objective is message ranking, we propose a new feature construction scheme of the system-log dataset that measures the difference in the ranking of messages between system logs. Two known rank correlation measures can be used to achieve this: The Spearman rank correlation [12] and Kendall's tau rank correlation [12]. Let $\vec{x}$ and $\vec{y}$ be vectors of dimension N. Let $\vec{r_x}$ and $\vec{r_y}$ be vectors of ranks for $\vec{x}$ and $\vec{y}$, i.e. $r_x[i] = k$ if $x[i]$ is the k'th largest number in $\vec{x}$, and similarly for $\vec{r_y}$.[3] The two correlation measures are defined as follows:

Definition 1 (Spearman Rank Correlation)**.** *Let $\vec{d} \stackrel{def}{=} \vec{r_x} - \vec{r_y}$. The Spearman rank Correlation between $\vec{x}$ and $\vec{y}$ is defined by:*

$$\rho(\vec{x}, \vec{y}) \stackrel{def}{=} 1 - \frac{6\| \vec{d} \|^2}{N(N^2-1)} \tag{1}$$

Definition 2 (Kendall's Tau Rank Correlation)**.** *Let $P(\vec{x}, \vec{y})$ be the number of pairs i, j such that both $r_x[i] > r_x[j]$ and $r_y[i] > r_y[j]$. Kendall's tau rank correlation between $\vec{x}$ and $\vec{y}$ is defined by:*

$$\tau(\vec{x}, \vec{y}) \stackrel{def}{=} \frac{4P(\vec{x}, \vec{y})}{N(N-1)} - 1 \tag{2}$$

We first define a new *Spearman-based* feature-set. In this feature-set, system log i is represented by the vector $(\rho(\vec{c_i}, \vec{c_1}), \ldots, \rho(\vec{c_i}, \vec{c_k}))$, where k is the number of samples in our dataset. A Kendall's-tau-based feature-set can be generated in an analogous way. The resulting matrix is a sample correlation matrix, and the new feature-set has k dimensions instead of the much larger n (the number of different message types in the dataset). It is generally expected that in a typical dataset, n would be much larger than k as it is in our own dataset, because of the diversity of possible messages in a computer system.

It is interesting to note that using either the Spearman-based feature-set or the Kendall's-tau-based feature set generates a kernel similarity matrix for the original dataset. This opens the possibility of using these correlation measures in kernel-based algorithms [9]. We prove that both sample correlation matrices are kernel matrices, using the fact that a kernel matrix is a Positive Semi-Definite (PSD) matrix. A matrix A is PSD if for any non-zero vector x, $x'Ax \geq 0$ [9].

We first prove that the Pearson Sample Correlation matrix [12] is PSD, and then conclude that so are the Spearman rank correlation matrix and the Kendall's-tau sample correlation matrix.

Definition 3 (Pearson Sample Correlation)**.** *Let $\vec{x}$ and $\vec{y}$ be vectors of the same dimension. The Pearson correlation coefficient is defined by:*

$$r(\vec{x}, \vec{y}) = \frac{cov(\vec{x}, \vec{y})}{\sqrt{var(\vec{x}) \cdot var(\vec{y})}} \tag{3}$$

where cov is the sample covariance function and var is the sample variance function.

Theorem 1. *A Pearson sample correlation matrix is PSD.*

Proof. Let X be a matrix in which each row is a sample. Let S be a diagonal matrix such that entry (i, i) is the variance of row i in X. Assume, without loss of generality, that the mean of each sample in X is zero. Then the Pearson correlation matrix can be written in vector form as:

$$R = S^{-\frac{1}{2}} X X' S^{-\frac{1}{2}}$$

[3] In case of ties, the coordinate with the larger index is ranked higher.

For any non-zero vector $\vec{x}$, the expression $\vec{x}' R \vec{x}$ can be written as:

$$\vec{x}' R \vec{x} = \vec{x}' \left(S^{-\frac{1}{2}} X X' S'^{-\frac{1}{2}} \right) \vec{x} = \left(\vec{x}' S^{-\frac{1}{2}} X \right)^2 \quad (4)$$

The rightmost element is a square term, hence it is greater than or equal to zero. Therefore R is PSD. □

Theorem 2. *A Spearman rank correlation matrix is PSD.*

Proof. Spearman correlation is Pearson correlation applied to ranks [4]. Therefore, the Spearman rank correlation matrix is PSD. □

Theorem 3. *A Kendall's-tau correlation matrix is PSD.*

Proof. For a vector $\vec{x}$ of dimension N, let $\vec{x_K}$ of dimension N^2 be defined by:

$$x_K[j + (i-1) \cdot N] = \begin{cases} 1 & r_x[i] > r_x[j] \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

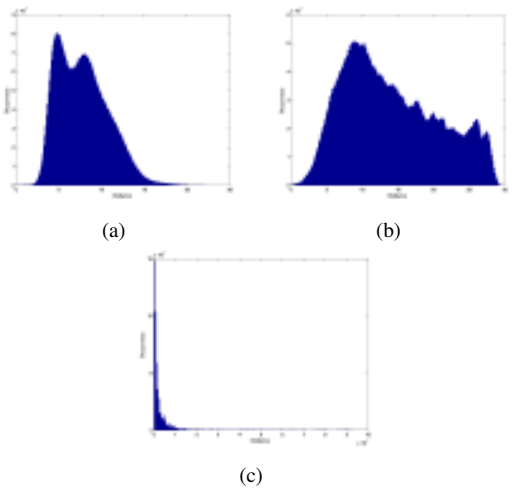
Then $P(\vec{x}, \vec{y}) = \sum_{k=1}^{N^2} x_K[k] \cdot y_K[k]$, and it can be easily verified that Kendall's tau correlation of $\vec{x}$ and $\vec{y}$ is the Pearson correlation of $c \cdot \vec{x_K}$ and $c \cdot \vec{y_K}$, for c a constant that depends on n. Hence, Kendall's tau correlation matrix is also a Pearson correlation matrix, and so it is PSD. □

4 Testing Different Feature Construction Schemes

The outline of the log-ranking process is as follows:

1. Generating a representation of the original dataset of system logs, using a feature construction scheme;
2. Using k-means clustering to divide the computer systems in the dataset into distinct sets;
3. Estimating P, the vector of cumulative distribution functions, for each cluster, using the empirical distribution in this cluster;
4. Given a system log to rank, identifying the cluster it belongs to and ranking its messages by the score calculated from $\hat{P}$ of that cluster.

We implemented this process with the new Spearman-based feature-set, and compared it to two simpler feature-sets with the same number of dimensions. We did not experiment with Kendall's tau because of the computational load associated with it. For each feature-set, we used k-means clustering to generate each of two to five clusters. In total we tested the following three feature-sets:



(a) (b) (c)

Figure 1: Histograms of the pairwise distances between samples: (a) Spearman correlation, (b) Pearson, (c) FM.

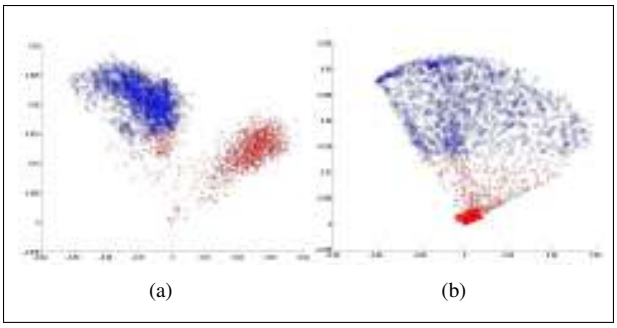
1. The Spearman-based feature-set;
2. A *Pearson-based* feature-set: The dataset is represented using the Pearson Sample Correlation matrix.
3. A *Frequent-Message* (FM) feature-set: Let $m_1, \ldots, m_k$ be the k message types that appear in the largest number of logs in the dataset. System log i is represented by $(c_i[m_1], \ldots, c_i[m_k])$.

Since we do not have an external indication for the 'right' ranking of messages, we use several statistical analysis techniques to analyze the clustering results.

One way of determining whether a given dataset contains clusters is by looking at the pairwise distances between the data samples[3]. If the dataset contains clusters, we expect a bi-modal distribution of the pairwise distances, where one mode represents the inter-cluster distances and the other the intra-cluster distances. Figure 1 shows the histogram of pairwise distances in each of the three representations we tested. It is easy to see that the Spearman-based representation arranges the data in a bi-modal way, much more so than the other two representations do.

Another way to visualize the spatial structure of the samples in the two first feature-sets is to plot the two largest eigenvectors of the resulting correlation matrices on the plane, giving the first two Principal Components of the sample points. In Figure 2, this plot is shown for the Spearman-based and for the Pearson-based feature-sets.[4]

[4]Since the Frequent Message matrix is not a correlation matrix, its eigenvectors include imaginary parts and therefore cannot be plotted.



(a) (b)

Figure 2: The two most significant eigen-values of the correlation matrices, plotted against each other on the plane. (a) The Spearman-based representation (b) The Pearson-based representation. In each plot, the division into two clusters is indicated in color and shape of dots.

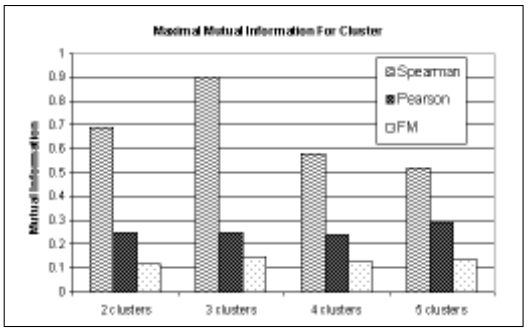


Figure 3: The maximal mutual information between a cluster and a binary feature found in each of the clustering tests.

Much more structure is revealed in the Spearman-based feature-set. The division into two clusters by the k-means algorithm is also depicted in the plot. Note that since this plot is only a two dimensional projection of the high dimensional space, the partition of the data into clusters may appear as if it were sub-optimal in this plot.

To investigate how well the clusters found for each test represent real use-model properties of the computer systems in the sample, we used external information on each computer system in our sample, that included specifications of installed hardware and software components and their configuration parameters. The information was represented as binary features of the computer systems. For each cluster in each test, we searched for the binary feature that had the highest mutual information with the property of belonging to the cluster. In Figure 3 we list the highest mutual information found in each test. The clusters in the Spearman feature-set are consistently more correlated with actual properties of the computer systems. Since there are usually several such features and their interpretation is highly technical, we do not list here the actual features for each cluster.

The cluster with the highest mutual information coefficient in all our tests is one found in the Spearman test conducted with three clusters. It has mutual information of 0.9 with the best-correlated features of the systems; it was therefore easiest to find its 'meaning'. The features that were most highly correlated with this cluster indicated that several services related to IBM Director are installed on the computer systems in this cluster, but not on computer systems that are not in this cluster. IBM Director is a suite of tools for managing computer systems that affects the behavior of many components in the system. It is therefore reasonable that it would significantly affect the log behavior of systems. An interesting addition to our log ranking tool would be the ability to automatically generate and display a meaningful description of the cluster to which the inspected system belongs.

If the clusters truly represent sets of systems that are more homogeneous in terms of their log behavior as compared to the entire set of computer systems, then we expect the average score of messages in a ranked log to be lower when the score is computed using the $\hat{P}$ of the cluster, compared to the average score computed using the $\hat{P}$ of the entire population. Figure 4 compares, for each of the clustering tests, the average difference in the average score of all the system logs in our dataset, between the cluster-based scoring and the non-cluster-based scoring. The Spearman-based clustering achieved the largest lowering of the score. We calculated the statistical significance of the results using a paired T-test, with resulting p-values of 0 in all cases.

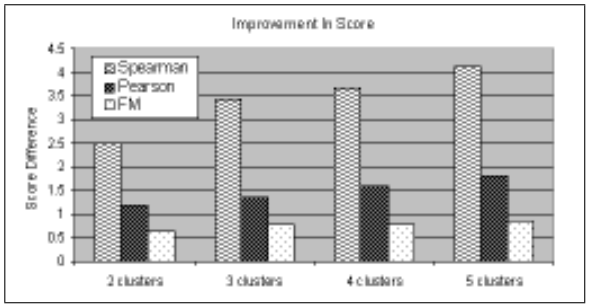


Figure 4: The change in the mean score (in a scale of 0-100) of messages in all machines, when calculated by the cluster the machine belongs to instead of the entire set of machines.

No single message was found to have a high level of mutual information with the clusters that were found in any of the tests. This is expected, as most messages appear in only a small fraction of the logs. Nonetheless, to visualize the way different messages are manifested in different clusters, we plot for each cluster the probability

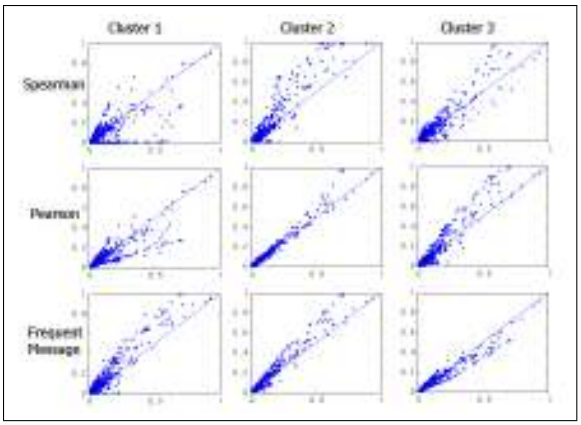


Figure 5: The probability of each message in the cluster versus its probability in the entire sample, with 3 clusters. Each row depicts the three clusters of one feature-set.

of each message appearing in the cluster versus its probability of appearing in the entire dataset. For a cluster that is highly distinctive among messages, a relatively large number of messages would appear far from the diagonal.

Plot 5 shows the case of three clusters. The Spearman-based approach yields the most distinctive clusters. To numerically estimate this difference, Table 2 shows the mean ratio between the probability of a message appearing in a cluster and its probability within the entire sample set, averaged over all the clusters in each test.

5 Summary

We presented a novel approach to ranking log messages based on sampling a population of computer systems and using a new feature construction scheme that proves to be highly appropriate for the ranking objective. The results show that an interesting structure is revealed in populations of systems based on the behavior of their log messages. This structure can also be used for purposes other than ranking log messages, such as identifying similar systems for the sake of assisting in problem diagnosis.

Clusters	Spearman	Pearson	FM
2	**0.65**	0.77	0.82
3	**0.62**	0.77	0.77
4	**0.57**	0.74	0.78
5	**0.57**	0.75	0.76

Table 2: The mean of the ratio between the probability of each message in each cluster and its probability in the general population. For ratios higher than 1 their inverse is taken. A smaller mean ratio implies a more distinctive clustering.

Using the Spearman correlation matrix opens the door for using kernel methods on machine learning tasks that use system logs as input.

Further research is required to answer questions such as how to choose the best number of clusters for the log ranking application, and how to incorporate the ranked log view with other approaches for log analysis that take time-dependency into consideration.

6 Acknowledgement

We thank Chris Dombrowdki and Christopher McCann from the IBM xSeries Support & Tools team for their indispensable help in retrieving and analyzing the data.

References

[1] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern classification*. John Wiley and Sons, Inc, 2001.

[2] Stephen G. Eick, Michael C. Nelson, and Jerry D. Schmidt. Graphical analysis of computer log files. *Commun. ACM*, 37(12):50–56, 1994.

[3] Michael Steinbach Levent Ertoz and Vipin Kumar. Challenges of clustering high dimensional data. In *New Directions in Statistical Physics: Econophysics, Bioinformatics, and Pattern Recognition*. Springer, 2004.

[4] Arthur M. Glenberg. *Learning from Data: An Introduction to Statistical Reasoning*. Lawrence Erlbaum Associates, 1996.

[5] S. E. Hansen and E. T. Atkins. Automated system monitoring and notification with swatch. *Proceedings of the Seventh Systems Administration Conference (LISA VII) (USENIX Association: Berkeley, CA)*, page 145, 1993.

[6] Wolfgang Ley and Uwe Ellerman. Logsurfer. http://www.cert.dfn.de/eng/logsurf/.

[7] Tao Li, Feng Liang, Sheng Ma, and Wei Peng. An integrated framework on mining logs files for computing system management. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 776–781. ACM Press, 2005.

[8] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., 1986.

[9] Bernhard Schölkopf and Alexander J. Smola. *Leaning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT Press, 2002.

[10] J. Stearley. Towards informatic analysis of syslogs. In *CLUSTER '04: Proceedings of the 2004 IEEE International Conference on Cluster Computing*, pages 309–318. IEEE Computer Society, 2004.

[11] Tetsuji Takada and Hideki Koike. Mielog: A highly interactive visual log browser using information visualization and statistical analysis. In *LISA '02: Proceedings of the 16th USENIX conference on System administration*, pages 133–144. USENIX Association, 2002.

[12] G. Upton and I. Cook. *Oxford dictionary of statistics*. Oxford university press, 2002.

[13] R. Vaarandi. Sec - a lightweight event correlation tool. In *IEEE IPOM'02 Proceedings*, 2002.