Analyzing Burstiness and Causality of System logs

Kazuki Otomo
Univ. Tokyo
otomo@hongo.wide.ad.jp

Satoru Kobayashi
Univ. Tokyo
sat@hongo.wide.ad.jp

Kensuke Fukuda
NII
kensuke@nii.ac.jp

Hiroshi Esaki
Univ. Tokyo
hiroshi@wide.ad.jp

1 INTRODUCTION

Analyzing network logs is one of the most useful methods for network operators to understand network problems. With these logs, we investigate detailed status and events for each devices in network system. However, there are two issues to analyze system logs. First, a large scale network generates a huge amount and kind of log messages and it makes the log analysis to be a hard task. Second, the important logs related to system fault are usually hidden in the majority of ones that report daily routine processes. Because of these two issues, manual inspection is an unrealistic method and automatic method is highly required. To overcome these issues, there have been a lot of literature on automated system log analysis [1],[3],[5]. One of efficient log analysis methods is to extract time series relations, especially causal inference. Causal inference is a statistical technique to identify a causal relationship between events. A popular causal inference algorithm is called PC algorithm [8], [4] that is based on directed acyclic graphs (DAG). Chen et al. [2] apply the PC algorithm to a set of time series (e.g., RTT, TCP window size) for identifying the source of network traffic delay. Kobayashi et al. [7] apply the PC algorithm to log time series and extract causal relationships.

In this work, we focus on log burstiness, which is one of the most popular time series characteristics, and its causality, for the problem. A simple question arose in this work is how the burstiness is meaningful and how pairs of burstiness represent a causal relationship. To detect burstiness, we rely on a Kleinberg's burst detection method [6] and conduct burstiness analysis to single time series and burst cooccurrence analysis to pairwise time series. To combine burst detection and causal inference, we can find some meaningful pairs of bursts for troubleshooting and these cases have relatively high frequency of burst cooccurrence.

CoNEXT'17, December 2017, Seoul/Incheon, South Korea

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

2 DATASET AND PREPROCESSING

We use a set of network logs collected at SINET [9], which is a Japanese research and education network. This network connects over 800 academic organizations in Japan and consists of eight core routers, 50 edge routers, and 100 layer-2 switches. This dataset is same with [7]. In this work, we first generate log templates from dataset and classify them into each templates. We call this group as an event. We use the pre-processed dataset for removing frequent logs, by applying Fourier analysis and linear regression analysis as preprocessing, which is the same preprocessing with [7].

3 BURSTINESS AND CAUSALITY

3.1 Methodology

To detect burstiness of log messages, we employ the burst detection algorithm by Kleinberg[6]. We convert raw log messages to time series data and divide the dataset to one-day long per event per device and detect burstiness in seconds, for each events.

Next, we intend to extract meaningful burst event pairs for troubleshooting from results of single burst detection using the cooccurrence of bursts and causality between events. First, we analyze event cooccurrence related to burstiness using single burst detection results. Then, to analyze causality of burstiness, we compare cooccurrence of bursts and causal inference results.

Here, we define Co-burst and burst cooccurrence ratio to evaluate relevance between two bursts. Co-burst is a pair of two co-occurred burst in two event time series. Co-burst event pair is a pair of two events which include at least one co-burst. We define the cooccurrence of two bursts if they start in the same 1-min bin. We calculate the cooccurrence ratio of events A and B as Jaccard similarity coefficient $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$. The higher $J(A, B)$, the more frequent bursts occur at the same time between event A and B. To calculate $J(A, B)$, we define $|A \cap B|$ and $|A \cup B|$. First, A and B are events that are sets of time series data classified by each log templates. $|A|$ or $|B|$ means the total number of bursts detected in 456 days. $|A \cap B|$ is the total number of co-bursts between event A and B in 456 days. $|A \cup B|$ is a disjunction of A and B.

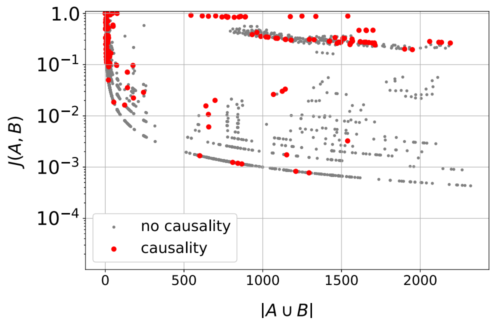

Figure 1: Jaccard coefficient and causal co-burst event pair distribution

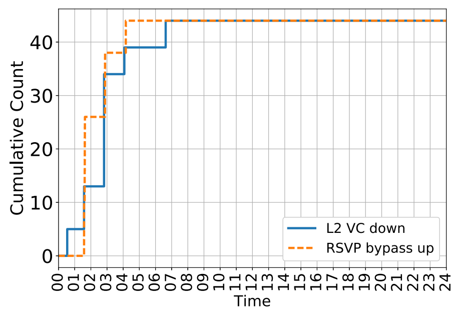

Figure 2: Causal co-burst example

	Co-burst	Causal Co-burst
all	22,756	241 (1.06 %)
$J(A,B) \geq 0.1$	17,541	215 (1.23 %)

Table 1: The number of co-burst events and causal ones

Use these definitions, we can calculate and compare burst cooccurrence ratio.

3.2 Results

We calculate the burst cooccurrence ratio using single burst detection result and then compare all the co-burst event pairs to all the causal event pairs in causality DB obtained in [7] (see also Table 1). The number of all the co-burst event pairs is 22,756 and 2,776 causal event pairs are recorded in the causality DB. If the same event pair of co-burst events is recorded in causality DB, we consider that the co-burst is not a coincident but a causal cooccurrence. We call such event pair as a causal co-burst. With this comparison, we find 241 causal co-burst pairs from all 22,756 co-burst pairs. The distribution of the cooccurrence ratio is shown in Figure 1. The vertical line shows the cooccurrence ratio in log-scale and the horizontal one the size of the union set of each event pair, representing how often these bursts happen. Each point shows a co-burst event pair. Red dots in Figure 1 illustrate such causal co-burst events and gray dots illustrate not causal but co-burst events.

Our results highlight two findings. First, we find a relationship between causal co-burst events and burst coefficient ratio. In Figure 1, most part of red dots located at an upper part of the graphs and from Table 1; over 89% of causal co-burst events have over 0.1 coefficient ratio. Focus on horizontal distribution of causal co-burst (red points), we can see there are two clusters; a small burst disjunction cluster ($A \cup B < 500$) and large one ($A \cup B > 500$). To investigate the detail of the large one, it includes a lot of daily process such as logs related to "show interface"" command. On the contrary, the small cluster has many meaningful pairs of burst. Thus, we conclude that causal co-burst event pairs have relatively high coefficient ratio. Second, we confirm the effectiveness to combine causal inference results and burst detection. There are about 22K co-burst event pairs by applying the burst detection. However, combining the causal inference, only 1% of them remain. Therefore, 99% of co-burst event pairs are coincident and we have to focus 1% of them for extracting meaningful bursts. Combining causal inference results is very effective to reduce trivial bursts.

Checking the details of log events, we find some useful cases for network operation. A case is that outage of a L2 switch introduces enabling a bypass event, which means change of network routes in order to avoid outage devices and keep network availability. Figure 2 shows the cumulative time series of two events. The horizontal line shows time and the vertical one shows one-day cumulative time-series of an event. Generally, system down and enabling bypass event are not always appear at the same time because there are many causes of the burst of the bypass enabling event. However, to combine burstiness and causality, we can automatically pinpoint when and what event causally burst with the bypass enabling event.

4 CONCLUSION

We focused on burstiness and causality appeared in network log messages to extract meaningful information from log data for troubleshooting. First, we conduct single burst detection. After that, we analyze pairwise time series burstiness with single burst detection results. To combine burst detection results and causal inference results, and we find some useful cases for troubleshooting from remained burst results.

We are analyzing causality from a non-burst event to a burst event (and vice versa) to highlight the root cause of the burst (and to predict further events after the burst). We plan to extend our results in this study to an automated log analysis system.

REFERENCES

[1] E. Baseman, S. Blanchard, and E. Zongzelimyuntedu. Relational Synthesis of Text and Numeric Data for Anomaly Detection on Computing System Logs. *in Proc. IEEE ICMLA'16*, 1:2–5, 2016.

[2] P. Chen, Y. Qi, P. Zheng, and D. Hou. CauseInfer: Automatic and distributed performance diagnosis with hierarchical causality graph in large distributed systems. *in Proc. IEEE INFOCOM'14*, pages 1887–1895, 2014.

[3] E. Chuah, S. H. Kuo, P. Hiew, W. C. Tjhi, G. Lee, J. Hammond, M. T. Michalewicz, T. Hung, and J. C. Browne. Diagnosing the root-causes of failures from cluster log files. *in Proc. HiPC'10*, pages 1–10, 2010.

[4] M. Kalisch and P. Buehlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8:613–636, 2005.

[5] T. Kimura, A. Watanabe, T. Toyono, and K. Ishibashi. Proactive Failure Detection Learning Generation Patterns of Large-scale Network Logs. pages 8–14, 2015.

[6] J. Kleinberg. Bursty and Hierarchichal structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397, 2003.

[7] S. Kobayashi. Mining causes of network events in log data with causal inference. *in Proc. IEEE IM'17*, pages 45–53, 2017.

[8] P. Spirtes and C. Glymour. An Algorithm for Fast Recovery of Sparse Causal Graphs. *Social Science Computer Review*, 9(1):62–72, 1991.

[9] S. Urushidani, M. Aoki, K. Fukuda, S. Abe, M. Nakamura, M. Koibuchi, Y. Ji, and S. Yamada. Highly available network design and resource management of SINET4. *Telecommunication Systems*, 56(1):33–47, 2014.