Functional Topology Inference from Network Events

Antoine Messager*, George Parisis*, István Z. Kiss†, Robert Harper‡, Phil Tee§ and Luc Berthouze*
*Department of Informatics, University of Sussex, Brighton, UK, Email: {A.Messager, G.Parisis, L.Berthouze}@sussex.ac.uk
†Department of Mathematics, University of Sussex, Brighton, UK, Email: I.Z.Kiss@sussex.ac.uk
‡Moogsoft Ltd., Kingston upon Thames, UK, Email: rob@moogsoft.com
§Moogsoft Inc., San Francisco, USA, Email: phil@moogsoft.com

***Abstract*—In this paper we present a novel approach for inferring functional connectivity within a large-scale network from time series of emitted node events. We do so under the following constraints: (a) non-stationarity of the underlying connectivity, (b) sparsity of the time-series of events, and (c) absence of an explicit model describing how events propagate through the network. We develop an inference method whose output is an undirected weighted network, where the weight of an edge between two nodes denotes the probability of these nodes being functionally connected. Two nodes are assumed to be functionally connected if they show significantly more coincident or short-lagged events than randomly picked pairs of nodes with similar levels of activity. We develop a model of time-varying connectivity whose parameters are determined by maximising the model's predictive power from one time window to the next. We assess the accuracy, efficiency and scalability of our method on a real dataset of network events spanning multiple months.**

***Index Terms*—network management; network events; topology inference;**

I. Introduction

Swiftly identifying network and service outages to ensure network and service availability in modern, large-scale networks is crucial [1]. Network operators continuously collect log data from all devices and running processes that are deemed to be important. Ensuring continuous network and service availability relies on the efficient and effective analysis of collected data so that outages can be quickly identified or predicted before user experience gets disrupted. This is a very challenging task [2], [3]. Networks are large, complex, heterogeneous and evolving. They support diverse services that are widely distributed, and also evolving. The aggregate rate of collected events is commonly high due to the very large number of monitored devices and services; a typical rate for a large-scale network deployment would be 10^6 events per second [4]. However, although the aggregate event rate is large, the rate at which individual devices emit events is extremely low such that correlating emitted events is inherently challenging; this becomes even more cumbersome in the presence of periodical informational events [5]. In addition, the vast majority of collected event data is noise and only a few of them may correlate with actionable incidents. Concurrency across network and services results in collected events whose timestamps may be unreliable in terms of absolute values and ordering, due to misconfiguration or loose synchronisation. This makes workflow-based anomaly detection [6] and concurrent log analysis approaches [7] difficult to apply.

978-3-903176-15-7 © 2019 IFIP

Root Cause Analysis [8] has therefore been a prominent research area [9]. Network operators commonly employ rule-based analysis where a pre-defined and manually updated list of rules is used to exclude uninteresting log data and make analysis of remaining events practical. This is a time consuming and error-prone process. Misconfiguration may result in fatal outages which could have been otherwise easily detected or predicted [10]. Kobayashi et al. [5] recently proposed an inference algorithm for mining causality of network events that has been shown to have good performance. However, the algorithm's complexity is cubic in the number of events.

In this paper, we take a radically new approach to identifying network and service outages by inferring *functional* topologies within a large-scale evolving network from time series of emitted events. This functional approach revealed to be very valuable in the context of biological networks [11], [12]. With the term functional connectivity, we refer to the integrated involvement of a set of network nodes in the provision of a particular service. Our approach operates under the assumption that the probability of two nodes being functionally connected will be reflected in the distribution of delays between their respective events. We develop an inference method whose output is an undirected weighted network where the weight of an edge between two nodes denotes the probability of these nodes being functionally connected. Our method outputs functional topologies consisting of underlying network nodes; a node may belong to multiple functional topologies. With our method a network operator is informed at all times about ever-changing service deployments (and the underlying network topology which can be seen as a functional one at the physical/link or IP layers). This provides a powerful tool for swiftly pinning down root causes of recent or imminent failures, by only examining log data emitted by devices in the same topology.

II. Functional Topology Inference

Low event rates at node level make it difficult to determine the statistical properties necessary to estimate ro-

bust confidence intervals on pairwise correlations. Instead, a fundamental empirical assumption of this work is that the propensity of two nodes being functionally interacting will be reflected in the distribution of delays between their respective events. Specifically, if events are rare, then the distribution of delays between events of two nodes participating in the same functional topology can be expected to differ from those between a randomly picked pair of nodes.

A. Score: estimating pairwise functional couplings

The cross-correlation between the time series of events of two nodes, say f and g, provides useful information regarding the presence of recurring temporal interaction between these two nodes. In the absence of a priori knowledge about how 'situations' unfold in the network and without any knowledge as to how precisely the timing of events is being recorded, we use an adaptation of the cross-correlation that does not distinguish between positive and negative delays:

$$(f \star g)(\delta t) = \sum_{t=\delta t}^{T-\delta t} f(t)\Big(g(t+\delta t) + g(t-\delta t)\Big), \qquad (1)$$

where δt denotes a lag (or delay) in units of sample time, and T is the duration of the record. Put simply, for each delay δt, this function quantifies the number of events from both time series separated by less than this delay (irrespective of which comes first) – this particular point addresses the aforementioned challenge of concurrency.

To quantify the presence of a functional link between two nodes, we construct a score that characterises the shape of the cross-correlation in terms of the number of its peaks and the distribution of these peaks within a range of delays. Specifically, we operate under two main assumptions motivated by the application domain. *Assumption 1*: Two nodes are likely to be functionally connected if there are multiple peaks in their cross-correlation. Indeed, since precise timing is unavailable and different computer network situations can entail different propagation times (see Introduction), there can be no expectation of a well defined single peak in the cross-correlation. Instead, coincident events could occur over various ranges of delays. This, however, could lead to a lack of sensitivity in the measure, namely, spurious correlations being treated as evidence of functional relationship. This leads to the second assumption. *Assumption 2*: There should be more peaks for small delays than for large ones.

To translate the first assumption into a quantitative measure, we calculate $R_{f,g}(\tau)$, the cumulated number of peaks in the cross-correlation between f and g over all lags in the interval $0, \tau$ where τ is a delay in units of sample time. It is given by:

$$R_{f,g}(\tau) = \sum_{\delta t=0}^{\tau} (f \star g)(\delta t). \qquad (2)$$

It is important to note that this quantity is sensitive to the number of events emitted by f and g and therefore does not support valid comparisons between all different pairs. In what follows, we consider that two pairs of nodes with event time series (f_1, g_1) and (f_2, g_2) belong to the same grouping if the products of their number of events are approximately equal: $n_{f_1} \times n_{g_1} \approx n_{f_2} \times n_{g_2}$[1]. This quantity is binned in order to obtain a computationally tractable number of groupings with each grouping featuring a sufficiently high number of pairs.

In order to support the quantification of *Assumption 2*, we first calculate the mean $\mu(\tau)$ and standard deviation $\sigma(\tau)$ of $R_{i,j}(\tau)$ for lag τ for all pairs (i, j) of nodes within a grouping. These quantities provide an estimate of the expected number of peaks at lag τ for a randomly picked pair of nodes within that grouping. We then score the propensity of an edge e (between nodes f and g) having more peaks at lower delays by calculating the normalised deviation between its cumulated cross-correlation $R_{f,g}$ and that expected at random from the grouping. Namely, we define score s_e as:

$$s_e = \frac{1}{\tau_{max}+1} \sum_{\tau=0}^{\tau_{max}} \frac{R_{f,g}(\tau) - \mu(\tau)}{\sigma(\tau)}, \qquad (3)$$

where τ_{max} is a free parameter discussed below. With the average approximating the integral of the area under the normalised deviation curve, the more peaks there are in the cumulated cross-correlation, the larger the score is. Further, since it can be shown (analytical proof will be provided elsewhere) that for sufficiently small delays τ both mean $\mu(\tau)$ and standard deviation $\sigma(\tau)$ are increasing functions of τ, unexpectedly large numbers of peaks at small lags will have a greater contribution to the score than those at larger lags, as needed to quantify *Assumption 2*. To mitigate the eventuality of a particularly large peak at a larger delay dominating over the contributions of smaller peaks at smaller delays, we limit the range of delays over which the score is calculated to τ_{max} (see Section III for its value in our experiments). It is worth noting that since in a large-scale network, calculating cross-correlations over all possible pairwise interactions is computationally extremely costly, limiting ourselves to τ_{max} has great computational benefit as well.

B. Model of time-varying connectivity

In this section, we describe our approach to translating the scores introduced in Section II-A into time-varying probabilities of the existence of functional edges. Since scores require estimates of cross-correlations, a fundamental assumption of the method is that of separation of timescales, namely, that changes in functional connectivity should occur much slower than the rate at which processes generate events; this is a realistic assumption in the context of computer network management. Changes in the functional connectivity occur when hardware is commissioned / de-commissioned and services are deployed / un-deployed. Even in very dynamic network deployments that support elastic cloud services, changes in the functional connectivity can be safely assumed to take place at timescales that are significantly smaller than the respective

[1]Intuition for this condition comes from noting that for two independent Poisson processes, the expected cumulated number of peaks is proportional to the product of their number of events

event generation rates. Another source of changes are failing devices (e.g., servers, routers). Such failures do happen frequently, especially in large-scale deployments, however, they result in a stream of events (by neighbouring or monitoring devices) and therefore provide information to our method about functional connectivity around the failing node.

A key principle of the proposed methodology is that the score $s_e(t_w)$ for a pair of nodes within a time window t_w provides the information required to update the estimate of the value of the probability $p_e(t_w - 1)$ of a functional edge existing between these nodes at the previous time window More precisely, we consider that information is gained about the probability of an edge existing only when both nodes emit events during the time window considered. This is a natural implication of the sparsity constraint. The fact that only one node in a pair emits an event does not necessarily imply that an edge does not exist (or no longer exists). For each pair of nodes and each time window t_w, there are therefore three cases to consider:

1) The score is positive, $s_e(t_w) > 0$, i.e., there were more coincident or short-lagged events between these two nodes than between randomly picked pairs of nodes with similar levels of activity. This increases confidence about the existence of an edge and therefore the probability $p_e(t_w)$ should increase as some function h_1 of the score.
2) The score is negative, $s_e(t_w) \leqslant 0$, i.e., there were fewer coincident or short-lagged events than expected at random. This lowers confidence about the existence of an edge and therefore the probability $p_e(t_w)$ should decrease as some function h_2 of the score.
3) At least one of the node does not emit events: This scenario does not provide any information and the probability should remain unchanged.

This leads to the following model formulation:

$$p_e(t_w{+}1) = \begin{cases} \Big(1 - (1 - p_e(t_w)) \\ \quad \times (1 - h_1(s_e(t_w)))\Big) & \text{if } s_e(t_w) > 0, \\ p_e(t_w) \times (1 - h_2(s_e(t_w))) & \text{if } s_e(t_w) \leqslant 0, \\ p_e(t_w) & \text{if no information,} \end{cases} \tag{4}$$

If h_1 and h_2 are continuous, monotonically increasing and decreasing, respectively, functions of the score with output in $[0; 1]$, this formulation ensures that $p_e(t_w)$ remains in $[0; 1]$. In our implementation, h_1 and h_2 are simple sigmoid functions, each involving a single free parameter (thereafter referred to as α and β respectively). Other formulations are possible but do not affect the principle of the method, provided they are differentiable in their parameter(s). Since changes in functional connectivity from one window to the other are assumed to be small, we formulate the problem of determining the two free parameters as one of minimising the error of a binary classifier predicting the sign of the score at time t_w given the edge probability at time $t_w - 1$. In other words, if the edge probability at time $t_w - 1$ is greater than a threshold th (0.5 throughout) and both nodes emit events in time window t_w, we expect the score at time t_w to be positive. Conversely, if the edge probability at time $t_w - 1$ is less than the threshold and both nodes emit events in time window t_w, we expect the score at time t_w to be negative. Our error criterion is formally defined as:

$$E = \sum_{t_w=1}^{N_w} \Bigg(\sum_{s_e(t_w)\leqslant 0 \text{ and } p_e(t_w-1)\geqslant th} (p_e(t_w - 1) - th) \quad + \sum_{s_e(t_w)>0 \text{ and } p_e(t_w-1)<th} (th - p_e(t_w - 1)) \Bigg). \tag{5}$$

It penalises misclassifications, namely $p_e(t_w) \geqslant th$ and $s_e(t_w) \leqslant 0$, or $p_e(t_w) > th$ and $s_e(t_w < 0)$), with a cost proportional to the difference between edge probability and threshold. As a sum of edge probabilities that are differentiable functions of the parameters, a simple gradient descent can be used to determine the values of the free parameters. Time-varying edge probabilities can then be calculated for all pairs using the update equation (4).

III. Experimental results

A. Description of the dataset

This paper is based on a dataset of network events from a large retail bank[2]. The infrastructure supports both central and distributed operations in remote sites. The network consists of a core of meshed routers, distribution switches and the supported application and infrastructure servers. The network supports both hub, hub to spoke and intra-spoke operations for financial transactions, and the supporting back office systems. Network events span a duration of 54 days (in period 5/2018 to 06/2018). Overall 13,428 different nodes emitted 3,000,418 events leading to a mean value of 4.14 events per node per day. Just 1% of the nodes emitted 65% of all events whilst only 260 nodes (or 2%) emitted more than one event per hour.

B. Validation of the proposed method

To assess the performance of our method, we investigate the predictive power of the method when systematically varying the length of the data over which the method is trained (from 2 to 50 days). The testing period (unseen data) consisted of the first 2 days of data after the training period. The window (the unit of adaptation time, i.e., when probabilities are updated) was set to 1 day. The threshold (determining whether an edge existed) was set to 0.5. The maximum delay τ_{max} over which cross-correlations were calculated was set to 120s. To quantify predictive power in the absence of ground truth, we adopted the following definitions. An edge is a true positive if the method predicted an edge and there was short-lagged activity across this edge in the testing period such that the score would predict the presence of an edge. An edge is a false

[2]This dataset is currently not publicly available due to its commercially sensitive nature.

positive if the method predicted an edge and there was some short-lagged activity across this edge in the testing period but the score for this activity would not predict the presence of an edge. True and false negatives are defined as the logical counterparts of true and false positives. It is essential to note that these definitions are contingent to short-lagged interaction happening in the testing period. This is because lack of activity across an edge over a period does not provide any information as to the existence of an edge. The edge might exist but not be active over the period. Such property was explicitly included in the construction of the model (see Section II-B).

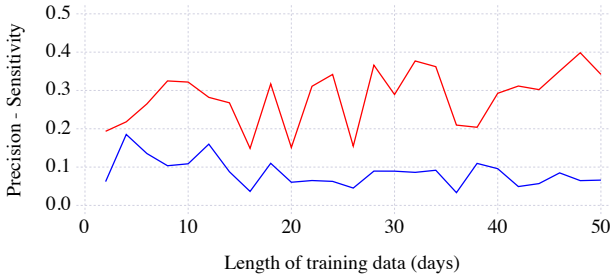

Fig. 1. Precision (red) and sensitivity (blue) as a function of the length of the training data (single run, full dataset).

Figure 1 shows the evolution of precision and sensitivity as the length of the training set was varied between 2 and 50 days. Both precision and sensitivity are low and relatively stable across the length of the record. This stability suggests that performance of the method is relatively insensitive to the length of the training data, however, such observation is contingent on knowing the extent to which the underlying functional topology changed during the record (this will be investigated below). To provide more confidence into the result, we repeated the experiment but averaged performance over 50 subnetworks of 1000 nodes picked at random.

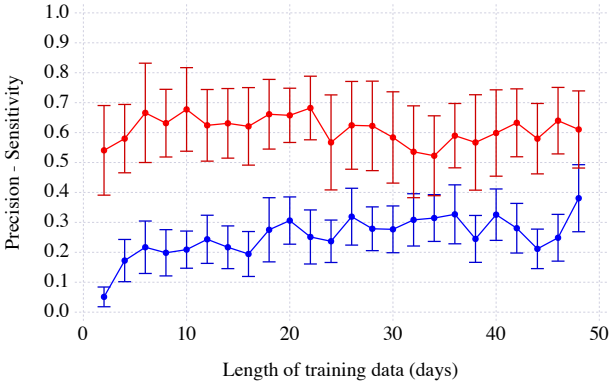

Fig. 2. Precision (red) and sensitivity (blue) as a function of the length of the training data (averaged over 50 subnetworks of 1000 nodes each).

Figure 2 confirms that predictive power is mostly insensitive to the length of the training data with only a marginal increase in sensitivity as the length of the training dataset increases. The values of both precision and sensitivity are significantly higher than in Figure 1. This is a somewhat counter-intuitive observation at first but can be explained in terms of the ability of a single (small) set of hyper-parameters to model heterogeneity

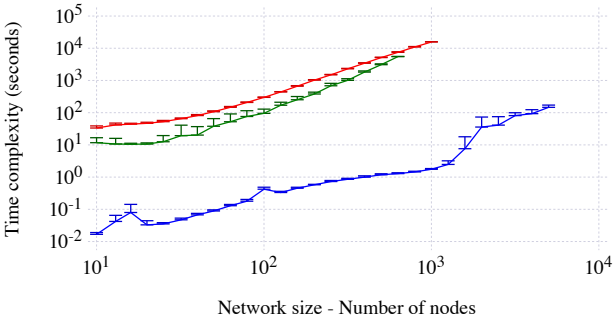

Fig. 3. Time complexity (in seconds) for our method (blue), Hallac et al. [13] (green) and Kobayashi et al. [5] (red) when network size is varied between 10 and up to 10,000 nodes (depending on methods).

in different components of the underlying functional topology. By considering subnetworks, this heterogeneity is reduced and therefore performance increases across both precision and sensitivity. An interesting operational implication could be that in a highly heterogeneous environment, it might be beneficial to deploy multiple instances of the method (each dealing with specific types of events) rather than one.

C. Scalability Analysis

Using time complexity as measure, we analysed how our method scales with network size and compared its performance with that of two state of the art approaches [5], [13]. Figure 3 demonstrates the clear superiority of our method when the number of nodes in the network is systematically varied, with roughly a factor 10^4 for network sizes of 10^3 nodes.

IV. Conclusion

In this paper, we have described a new method to infer the functional connectivity of a large-scale computer network from sparse time series of its node events. Our method operates under three strong constraints: (a) non-stationarity of the functional connectivity due to unknown temporal changes in the network, (b) sparsity of the time-series of events that limited the effectiveness of classical correlation-based analysis, and (c) lack of an explicit model describing how events propagate through the network. This is a hard problem. Both precision and sensitivity remained quite low. However, this should not detract from the fact that the method was able to recover a substantial amount of the connectivity, including its changes over time, from an extremely limited amount of information. Importantly, unlike existing network inference methods , it remains computationally tractable even with large networks (here, 10,000 nodes) over very long records (here, 10^7 observations).

In principle, this method could be applied to any system in which functional relationships between nodes translate into short delays between their respective activities. To fulfil its applicative potential, however, a more complete understanding of the various assumptions and parameters underpinning the method must be obtained.

REFERENCES

[1] R. Govindan, I. Minei, M. Kallahalla, B. Koley, and A. Vahdat, "Evolve or die: high-availability design principles drawn from Google's network infrastructure," in *Proc. of ACM SIGCOMM*, 2016.

[2] K. Zhang, J. Xu, M. R. Min, G. Jiang, K. Pelechrinis, and H. Zhang, "Automated IT system failure prediction: A deep learning approach," in *Proc. of Big Data*, 2016.

[3] A. Oprea, Z. Li, T. F. Yen, S. H. Chin, and S. Alrwais, "Detection of early-stage enterprise infection by mining large-scale log data," in *Proc. of DSN*, 2015.

[4] A. Messager, G. Parisis, R. Harper, P. Tee, I. Z. Kiss, and L. Berthouze, "Network events in a large commercial network: What can we learn?" in *Proc. of IEEE/IFIP NOMS AnNet*, 2018.

[5] S. Kobayashi, K. Otomo, K. Fukuda, and H. Esaki, "Mining causality of network events in log data," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 53–67, 2018.

[6] X. Yu, P. Joshi, J. Xu, G. Jin, H. Zhang, and G. Jiang, "Cloudseer: Workflow monitoring of cloud infrastructures via interleaved logs," in *Proc. of ASPLOS*, 2016.

[7] I. Beschastnikh, Y. Brun, M. D. Ernst, and A. Krishnamurthy, "Inferring models of concurrent systems from logs of their behavior with CSight," in *Proc. of ICSE*, 2014.

[8] M. Steinder and A. S. Sethi, "A survey of fault localization techniques in computer networks," *Science of Computer Programming*, vol. 53, no. 2, pp. 165–194, 2004.

[9] R. Harper and P. Tee, "The application of neural networks to predicting the root cause of service failures," in *Proc. of IFIP/IEEE IM AnNet*, 2017.

[10] P. Tee, G. Parisis, and I. Wakeman, "Vertex entropy as a critical node measure in network monitoring," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 646–660, 2017.

[11] E. N. Brown, R. E. Kass, and P. P. Mitra, "Multiple neural spike train data analysis: state-of-the-art and future challenges," *Nature Neuroscience*, vol. 7, no. 5, pp. 456–461, 2004.

[12] M. Hecker, S. Lambeck, S. Toepfer, E. Van Someren, and R. Guthke, "Gene regulatory network inference: data integration in dynamic models - A review," *Biosystems*, vol. 96, no. 1, pp. 86–103, 2009.

[13] D. Hallac, Y. Park, S. Boyd, and J. Leskovec, "Network inference via the time-varying graphical lasso," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 205–213.