G-Cause: Parameter-free Global Diagnosis for Hyperscale Web Service Infrastructures

Xinrui Jiang[1], Yang Zhang[4], Tingzhu Bi[1], Xiangzhuang Shen[4], Yu Zhang[4],
Yicheng Pan[2], Meng Ma[3,5], Linlin Han[4], Feng Wang[4], Xian Liu[4], Ping Wang[3,5]
[1] School of Software and Microelectronics, Peking University, Beijing, China
[2] School of Computer Science, Peking University, Beijing, China
[3] National Engineering Research Center for Software Engineering, Peking University, Beijing, China
[4] ByteDance Inc., Beijing, China
[5] Shuanghu Laboratory, Beijing, China
{jxrjxrjxr, aqpyc, mameng, pwang}@pku.edu.cn, {bitingzhu}@stu.pku.edu.cn,
{zhangyang.329, shenxiangzhuang, felix.zhang, hanlinlin.intern, wangfeng.ai, liuxian.1}@bytedance.com

***Abstract*—Hyperscale web service infrastructures are becoming increasingly complex and facing a variety of threats, raising the demand for more sophisticated automated operations and diagnosis solutions. Existing anomaly root cause localization approaches often focus on service-level components without drilling down to the lower-level resources where services are deployed, hindering the implementation of fine-grained failure fix measures. This paper introduces a challenging task called global diagnosis and addresses it by proposing a technique called G-Cause, which is applicable to both service-level and host-level root cause analysis scenarios. G-Cause builds a highly adaptive diagnostic framework based on the frequency-domain and time-domain characteristics of monitoring metrics, allowing it to handle global diagnosis requirements from app to host with minimal parameter adjustments. We deploy and validate our approach in two typical scenarios: homogeneous metric diagnosis from app to microservice, and heterogeneous metric diagnosis for various host resources. The results demonstrate that G-Cause outperforms state-of-the-art diagnosis algorithms while providing strong interpretability. Our approach helps operators understand the core mechanism of anomaly propagation and adjust their management strategies more effectively. With these strengths, G-Cause successfully services our global product operations and also makes an impressive contribution in many other workflows.**

***Index Terms*—Anomaly diagnosis, Root cause analysis, Frequency analysis, Granger Causality, Microservice architecture**

I. Introduction

Hyperscale web services infrastructures are business-critical facilities that support modern computing, data storage, and network services. They are constantly under threat from a variety of sources, including traffic spikes, hardware failures, misconfigurations and malicious attacks [1], [2], [3]. Frustratingly, large-scale web systems break down frequently and cause untold economic losses [4]. Although the problem has received considerable attention in academia and industry [5], [6], [7], [8], [9], [10], [11], most previous approaches have focused on using microservice performance metrics or logs to characterize anomaly's propagation between *service-level* components. In fact, web operations and maintenance staff (denoted as O&M staff hereinafter) need further clues in lower-level environments to understand the heart of the fault mechanism, such as CPU usage, memory usage, IO, sockets, inbound bandwidth, and hardware sensors. In light of these, we introduce a novel and challenging task named global diagnosis.

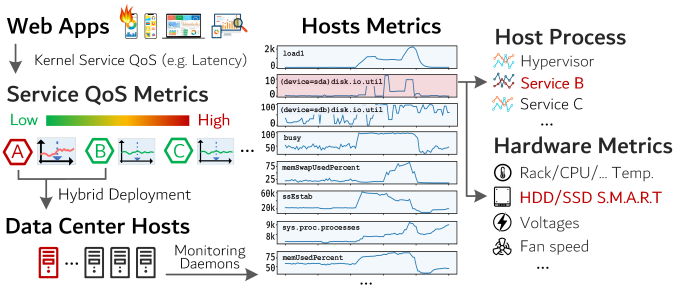


Fig. 1: Global diagnosis for web service infrastructures.

Global Diagnosis. Figure 1 illustrates a demo procedure when we *drill down* from application layer QoS down to underlying host resource anomalies (i.e., follow the arrows in Figure 1). In this case, a business app is built with multiple microservices collaboratively to satisfy client requests. Among them, our engineers observe a substantial rise in client request latency in `Service-A` and they quickly identify a host where `Service-A` has been deployed and report to O&M team. After careful investigations, the team confirms that *frequent disk I/O* is the cause of a series of abnormal metrics, and finds using a `top` command that the frequent disk I/O is caused by another I/O intensive `Service-B` process on the host. The final solution is to migrate `Service-B` to a separate host with no other services, and `Service-A` is back to normal shortly after. Through the case, we shed light on key challenges in achieving global diagnosis in hyperscale web systems:

(i) *Heterogeneity of lower level diagnostic metrics.* Diagnostic metrics show much higher heterogeneity in data formats and implications with the drilling down of fault analysis. The faulting mechanism inside the host environment connects these heterogeneous metrics through direct and indirect correlation. Therefore, finding the propagation path through these intertwined indicators becomes significantly more complex than homogeneous service-level metrics.

(ii) *Frequency-domain diversity of diagnostic metrics*. Our extensive observation verifies that metrics show varying characteristics in different frequency bands. Moreover, the frequency-domain feature of the same metric will also shift significantly before and after the abnormality occurs. As far as we know, frequency-domain heterogeneity has not been effectively valued and utilized in fault diagnosis.

(iii) *Inhomogeneity of normal/abnormal samples*. Anomalies are generally random and often cause systemic failures quickly. Ensuring the timeliness of O&M team's intervention for low-priority is difficult given restricted labor costs. As a result, the time-domain distribution of abnormal metrics is highly imbalanced and random. On the contrary, our system constantly accumulates massive normal data but has not effectively guided the troubleshooting.

This paper presents G-Cause, a lightweight and parameter-free tool to provide interpretable and accurate diagnosis in global tasks. Our contribution is manifold:

(i) *Multi-band causal graph construction*. We extend the traditional Granger causality test [12] to construct multi-band causal graphs and demonstrate that the increase of high-frequency causality component prevails in the abnormal time interval. The multi-band causal analysis enables G-Cause to autonomously adapt to fault characteristics distributed in different frequency bands.

(ii) *Stable causality discovery by the stochastic metric sampling*. Our work reveals that stable causality can be extracted from a particular host's historical and post-anomaly data. Such causalities are inherent characteristics strongly associated with a specific host's operational tasks. Based on these insights, our approach actively adapts to the uneven distribution of abnormal metrics by offline extracting inherent causalities between dozens of metrics.

(iii) *A lightweight, parameter-free, and global solution*. We deconstruct the global diagnosis problem into two fundamental targets, namely service-level and host-level diagnosis tasks, covering diagnostic activities from web apps to backend host environments. G-Cause can work adaptively, accurately and interpretably under time and frequency-domain heterogeneity challenges. G-Casue is a lightweight, purely data-driven tool that operates solely on metrics without any system modification or prior knowledge.

We deploy G-Cause in our hyperscale production environment and verify that it works faster than all baselines while does not require any GPU for model training. In addition, to ensure its generality, we simplify the workflow so that G-Cause does not contain any parameters and validate G-Cause's effectiveness on different types of service/host failures. This demonstrates our solution can be utilized out of the box without the hassle of tuning parameters. In addition, G-Cause's results show strong interpretability, successfully assisting our O&M staff in understanding the core mechanism of anomaly propagation and guiding fix strategies more effectively.

II. Related Work

Root Cause Analysis. Root cause Analysis of cloud apps and distributed systems has garnered significant attention from academia and industry [13], [11], [10], [14], [15], [16], [17], [18], [19]. They generally utilize resources like logs [13], [11], [10], distributed traces [14], [15], [16], and monitoring metrics [17], [18], [19]. Traditional root cause location identifies the system's root cause based on the dynamic changes in monitoring metrics and the service dependencies of the application [6], [20], [21], [22], [8], [10], [17]. Based on the granularity of the returned results, root cause localization is divided into coarse-grained *application-level* (returning the service where the root cause is located) and fine-grained *service-level* (returning the KPI of the root cause service) [23]. Our approach utilizes single-host heterogeneous monitoring metrics instead of multi-service homogeneous KPIs and returns more granular *host-level* root causes.

Causal Inference. When the underlying topology of an application is not accessible, identifying the root cause becomes challenging. In such scenarios, causal inference-based time series analysis is widely employed to discover potential relationships among entities (services or their KPIs) [9], [24], [5], [25], [26], [27], [28]. Representative causal inference methods include the PC algorithm[29], Granger causality test[12]. We select Granger Causality in our approach because of its pairwise manner of computing causality without making any assumptions about the underlying structure of metrics.

Frequency Analysis. As a kind of technique with strong mathematical foundation, frequency analysis [30], [31] often provides a unique perspective for studying time series data, making it a valuable tool for analyzing KPIs of services. In recent years, two main methods of frequency analysis have emerged: Discrete Fourier Transform (DFT) [32] and Discrete Wavelet Transform (DWT) [33]. In recent years, DFT and DWT are commonly used as a component within a larger framework to provide multi-frequency information. For example, they have been applied to various tasks, such as time series forecasting [34], [35] and representation learning [36], [37]. We believe that frequency analysis can effectively combine with causal inference methods in root cause analysis tasks to reveal more underlying information about abnormal events.

III. Background

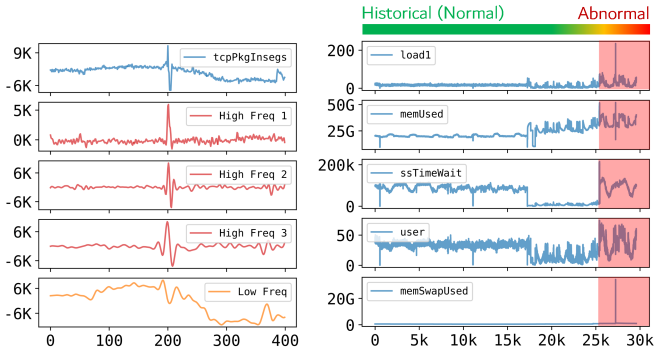
A. Problem Statement

Our research motivation originates from the real operational challenges of a leading Internet company. Our web services infrastructure supports many different mobile apps, web services, computing, and data storage businesses with over 1 billion worldwide MAU (Monthly Active Users). Maintaining its availability and performance is crucial to the company's stable operation and economic benefits. When any fault occurs, typical O&M requirements can be categorized into service-level diagnostic tasks (SDT) and host-level diagnostic tasks (HDT). SDT focuses on the root cause analysis (RCA) of app anomalies in microservice systems, abstracting them as

TABLE I: Sample metrics from a production environment.

Category: Number	Representative Metric: [Description] / Representation Service: [Metric][3]	Typical Range
Disk[1]: 26	temperature: [storage thermal metrics] fanspeed: [disk array fan speed]	[30°C, 50°C] [5400RPM, 7200RPM]
GPU[1]: 18	gpu_utilization: [utilization of GPU] gpu_fan_speed_perc: [GPU fan speed percentage]	[0%, 80%] [0%, 60%]
Runtime[2]: 15	user: [user space CPU time percentage] switches: [process context switch number]	[0%, 80%] $[0, 1 * 10^6]$
Memory[2]: 18	memSwapUsed: [swap memory usage] memFreePercent: [free memory percentage]	[0GB, 20GB] [20%, 90%]
Network[2]: 96	ssEstab: [sockets number in connection] inBytes: [bytes number received by eth0]	$[0, 5 * 10^4]$ $[0B, 3 * 10^8B]$
I/O[2]: 38	await: [average time required per I/O] write_sectors: [written sectors number]	[0ms, 15ms] $[0, +\infty)$
Service APP[3]: 1723	com/module/service/api; POST: [latency] com/module/service/api; GET: [throughput]	[0.00s, 1.00s] [0, 100]

* 1. Hardware-level Metrics; 2. Host-level Metrics; 3. Service-level Metrics



(a) Frequency domain (b) Time domain

Fig. 2: Motivations from our empirical observations.

service dependency graphs. In contrast, HDT aims to diagnose host machine-level anomalies for more precise fault control measures. Both tasks involve multidimensional time series data with unknown causal structures. Formally, given N services or monitoring metrics related to an anomaly period $[l_{ab}, r_{ab}]$ in a wide period $[1, M]$ ($[l_{ab}, r_{ab}] \subset [1, M]$), SDT and HDT collect Quality of Service (QoS) metrics or heterogeneous monitoring metrics, respectively, recorded as $T \in \mathbb{R}^{N*M}$. The goal is to locate the root cause service or specific metric v_{rc} responsible for the anomaly reflected by the frontend service or metric v_{entry}. In the subsequent discussion, we use the term *metric* to refer uniformly to the same metric in different services in SDT (such as latency) and different metrics in the same host in HDT (such as CPU and memory metrics).

B. SDT and HDT Metrics

In SDT scenario, we consider several representative metrics such as latency, throughput, power [38], and availability (the percentage of service available time). They are easy to obtain from typical APM (Application Performance Monitoring) tools or API logs [39]. For HDT, we deploy monitoring tools in the host environment to collect working statuses such as CPU load, memory usage, network traffic, and GPU usage. These tools store a certain length of historical metrics (2 or 3 months) on disk to provide information to O&M staff for reference. Due to the large variety of metrics to be recorded, we collect metrics at a low sampling rate so as not to compete with current processes for CPU resources. The typical sampling interval is 30 seconds, significantly sparser than the 1-5 second interval for service-level metrics. Therefore, although the root cause location algorithms for microservices systems and host-level ones have similar data inputs (both are sets of input monitoring metrics sequences and corresponding abnormal intervals), they fail to identify causal relationships among metrics in host-level scenarios and are therefore ineffective.

Table I provides an overview of categories and typical metrics, covering spatial units (B), temporal units (s), percentages (%), counters, etc., and the range of metrics varies. Correlations between these heterogeneous records are rather cryptic through manual observation alone. In addition, given some host-level metrics need to be split according to different eths/disks/mount points, the total number of heterogeneous metrics we need to analyze will be in the hundreds. Fortunately, many measures convey comparable meanings and have the same statistical features after normalization, which can be combined at the preprocessing stage.

The monitoring systems of mature commercial enterprises typically provide alerting services for metrics. The monitoring of these alerts is usually based on rules defined manually by O&M staff, such as recognizing that an exception exists when the CPU load is greater than 50. An aberrant event (e.g. software failure, container misconfiguration) can cause most host-level metrics to be abnormal. Since metric anomalies are currently determined by simple threshold criteria, they can easily be triggered at the same time, resulting in what is known as an *alert storm*. When alarm storms occur, O&M staff receive thousands of overload messages, leading to fatigue and important message omissions. A fast and effective RCA algorithm that lightly quantifies host-level metrics will help O&M staff quickly locate the real factor to focus on from a wide variety of alert messages.

C. Observations

Table I lists typical metric examples for SDT and HDT. SDT involves homogeneous service-level metrics (e.g., latency and throughput for each service), while HDT comprises heterogeneous host-level and hardware-level metrics (e.g., temperature and rotation speed of the disk, user CPU time, and process context switch number). We enumerate the typical fault-free ranges for each representative metric to illustrate the heterogeneity of the metrics used in HDT. In SDT, following the convention of previous works, we perform root cause localization of microservice systems using single metrics. Note that there is no valid prior work on HDT, and thus remains an open challenge. We propose a parameter-free and global diagnosis approach, G-Cause, to achieve reasonable accuracy in SDT and HDT scenarios, enabling an integrated diagnostic flow for web app developers and system engineers. G-Cause is motivated by key observations from both homogeneous and heterogeneous metrics:

(i) *Frequency Domain*. We observed that the monitoring metrics show different characteristics across frequency bands. Specifically, low-frequency components typically represent inherent sequence features, while high-frequency compo-

nents reveal sequence details or differences. As shown in Figure 2a, high-frequency components capture anomalies in received TCP packets, while the low-frequency component illustrates the metric's large-scale variation. Therefore, our method aims to capture different characteristics of metrics by distinguishing frequency bands.

(ii) *Time Domain*. Historical monitoring or QoS metrics can aid root cause analysis during abnormal intervals. As demonstrated in Figure 2b, examining the historical interval reveals metrics co-varying with `load1`, while `memSwapUsed` shows almost no correlation with `load1` in the historical interval, making its spike in the abnormal interval suspicious. Thus, our goal is to automate such empirical findings to facilitate troubleshooting.

IV. Our Approach

A. Framework

G-Cause models causal relationships among all sequences of monitoring metrics (from apps, microservices, hosts, hardware, etc.). In particular, we separately consider the variation of causality in the frequency and time domains.

Frequency domain. To capture various characteristics on different frequency bands, we perform a multi-band decomposition [40] of the metrics set T such that each metric in T is decomposed into a few high-frequency components and one low-frequency component. The causal relationships of the metrics set on each frequency band can be revealed by pairwise causality test [41]. As the causal relationships among metrics change over time, the length of the given time interval is relatively short in comparison to M. Thus, it is a causal graph localized in the time domain.

Time domain. We first generate multi-band causal graphs on the abnormal time interval, which include the set of causality among metrics reflected by abnormal interval data, called the *abnormal interval causal graphs*. Inspired by our observations, we build multi-band causal graphs reflected by historical interval data, called the *historical interval causal graphs*. We sample L intervals in the historical interval and generate localized multi-band causal graphs. We continuously update the historical interval causal graph on each band incrementally. We then remove the historical interval graph from the abnormal interval causal graph on each frequency band using a pre-defined graph subtraction operator to obtain the *anomaly-related causal graphs*. Finally, we perform the Personalized PageRank [42] on the graph to obtain a steady probability distribution and make root cause metric recommendations.

As shown in Figure 3, G-Cause is deployed in the production environment as an independent root cause location service. It uses metric collection tools to periodically collect heterogeneous or homogeneous metrics from the application, service, and host layers and provide automated diagnostics during failures. Operators can use any single or group of service/host-level metrics to run G-Cause. That is, G-Cause can run flexibly at any fault analysis level.

B. Multi-band Decomposition

G-Cause first decomposes the metrics set T into multi-band time series by the discrete wavelet transform (DWT) [40].

Discrete wavelet transform. The discrete wavelet transform starts by defining a dilated and translated mother wavelet

$$\psi_{u,s}(t) = \frac{1}{\sqrt{s}}\psi(\frac{t-u}{s}) \tag{1}$$

where $\psi(t)$ is a function with integral zero and unit energy in $[-\infty, +\infty]$, u and s denote a specific frequency band and time point, respectively. For a time series $x(t)$, we quantify its variation by

$$W(u,s) = \int_{-\infty}^{\infty} x(t)\psi_{u,s}(t) \tag{2}$$

In DWT, we summarize the information contained in the signal by setting $s = 2^{-j}$ and $u = k2^{-j}, j, k \in \mathbb{F}$ in the minimum wavelet coefficients. The upper limit of j is denoted as J. By defining a series of parent functions $h_j = (h_0, ..., h_{J-1})$ and parent functions $g_j = (g_0, ..., g_{J-1})$ that satisfy orthogonality, we can recursively decompose the sequence into a series of wavelet coefficients $x(t) = [w_1(t), w_2(t), ..., w_J(t), w_J(t)]$.

To reconstruct the time series of each frequency band from the wavelet coefficients series, we apply the inverse DWT on v_J and w_j. We first reconstruct the father wavelet $\phi(t)$ and the father wavelet $\psi(t)$ from $[w_1(t), w_2(t), ..., w_J(t), w_J(t)]$ and then project $x(t)$ onto each wavelet basis:

$$s_{J,k} = \int \phi_{J,k}(t)x(t)dt \qquad d_{j,k} = \int \psi_{j,k}(t)x(t)dt \tag{3}$$

for $j = 1, 2, ..., J$. We define

$$cA_J(t) = \sum_k s_{J,k}\phi_{J,k}(t) \qquad cD_j(t) = \sum_k d_{j,k}\psi_{J,k}(t) \tag{4}$$

for $j = 1, 2, ..., J$, where cA, cD denote the approximated component (low frequency) and detailed component (high frequency), respectively. In this way, we can reconstruct the original time series as

$$x(t) = cA_J(t) + cD_J(t) + cD_{J-1}(t) + ... + cD_1(t) \tag{5}$$

Thus, we decompose an arbitrary metric sequence $T_i(t)$ in a specific interval $[l,\ r]$ into multiple high-frequency components and one low-frequency component by discrete wavelet decomposition. Considering a typical sampling interval of 30s, cD_1 corresponds to the high-frequency component on a time scale of 1 to 2 minutes, and the subsequent time scales are multiplied sequentially. The cA_J component corresponds to the low-frequency component on a time scale larger than 2^J minutes. The high-frequency component contains detailed information about the metric and captures its frequent fluctuation patterns. The low-frequency component is the overall character of the original sequence after removing the local high-frequency information, characterizing the metric variation on a larger time span.

We refer to the frequency band as $f_i,\ i = 0, 1, ..., J+1$, where $f_1, f_2, ..., f_J$ represents the J high-frequency component $cD_1 \sim cD_J$, and f_{J+1} represents the low-frequency

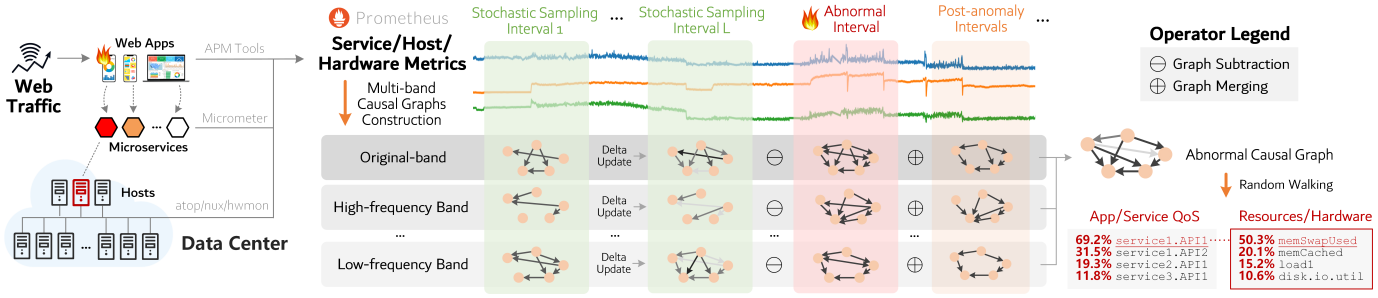


Fig. 3: Diagnosis framework and the deployment details of G-Cause.

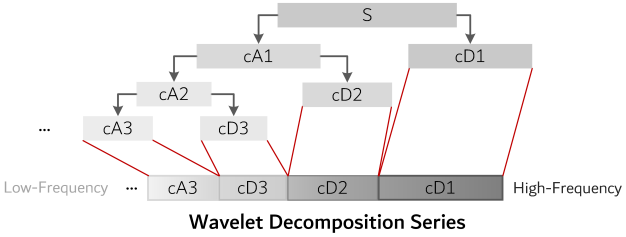


Fig. 4: Discrete wavelet decomposition for metric series.

component cA_J remaining after removing $1 \sim J$ high-frequency component. Specifically, we consider the original sequence T as a frequency band f_0 as well. Each band contains a decomposition subset $T^{(i)} \in \mathbb{R}^{N*L_i}$ of T, where L_i denotes the sequence length on band f_i satisfying

$$L_0 = r - l + 1 \qquad L_i \approx \lfloor L_{i-1}/2 \rfloor \qquad L_{J+1} = L_J \tag{6}$$

where $i = 1, 2, ..., J$. The maximum frequency band J is limited by L_J, guaranteeing that the subsequent Granger causality test [12] can be conducted. An example of the discrete wavelet transform can be seen in Figure 4, where we select three high-frequency components and one low-frequency component.

C. Multi-band Causal Graphs Construction

For the metrics set $T^{(i)}$ on each frequency band f_i, the causal strength among them must be determined. Since we make no assumptions about the structure of the metrics set, structured causality tests like PC [29], LiNGAM [43] are not applicable in host-level root cause localization scenarios. Consequently, we create causal graphs by pairwise Granger causality test [12], which is based on an autoregressive (AR) model that performs statistical analysis of the predicted results of the model. This method has low computational complexity and good interpretability.

Granger causality test. For $X(t), Y(t),\ t \in [l, r]$ in a given interval $[l, r]$, we build two regression models $(\mathcal{H}_p, \mathcal{H}_f)$, respectively:

$$\mathcal{H}_p \quad \hat{Y}_p(t) = \sum_{i=1}^{K} \alpha_i^p y(t-i) + b^p + \epsilon(t) \tag{7}$$

$$\mathcal{H}_f \quad \hat{Y}_f(t) = \sum_{i=1}^{K} (\alpha_i^f y(t-i) + \beta_i^f x(t-i)) + b^f + \epsilon(t) \tag{8}$$

where $\alpha_i^p, \alpha_i^f, \beta_i^f, b^p, b^f$ represents the regression parameters, $\epsilon(t)$ reprensents the noise term, and K represents the maximum lag of the regression model. We estimate these parameters by mature algorithms such as OLS [44] and calculate the prediction errors of the two regression models separately, denoted as

$$SSE_p = \sum_t \|\hat{Y}_p(t) - Y(t)\|, SSE_f = \sum_t \|\hat{Y}_f(t) - Y(t)\| \tag{9}$$

where $t \in [l, r]$. The partial model $\mathcal{H}_p$ only utilizes the historical data of Y, whereas the full model $\mathcal{H}_f$ uses the history of both metrics X, Y. Granger causality exists from $X(t)$ to $Y(t)$ on $[l, r]$ if the SSE of $\mathcal{H}_p$ is to be statistically significantly less than that of $\mathcal{H}_f$. We use F-test [45] to determine the causality existence. In a specified interval $[l, r]$, for the metrics set $T^{(i)}[l : r]$ throughout all frequency bands $f_i(\forall i \in [0, J+1])$, we pairwise conduct the Granger causality test and create the final causal graph

$$G^{(i)} = (V, E),\ E(u, v) \in \{0, 1\},\ \forall u, v \in V = 1, ..., N \tag{10}$$

In $G^{(i)}$, $E(u, v) = 1$ signifies that the metric pair u, v passes the Granger causality test under the constraints of interval $[l, r]$ and frequency band f_i, and thus Granger causality exists. Finally, we create multi-band graphs $\left\{G^{(i)}\right\}, i = 0, 1, ..., J+1$ in interval $[l, r]$.

D. Stochastic Sampling

Given the abnormal interval $[l_{ab}, r_{ab}]$ and the corresponding metrics set $T[l_{ab}, r_{ab}]$, we first build multi-band causal graphs $\left\{G_{ab}^{(i)}\right\}, i = 0, 1, ..., J+1$ according to the algorithm in subsection IV-C. Such causal graphs show the causal relationship reflected by the metrics of the anomaly interval and are therefore called the *anomaly interval causal graphs*. Notably, the causal relationship in the anomaly interval consists of two parts: the stable causal relationship and the abnormal part. Therefore, we propose eliminating the stable causality from anomaly intervals to refine the real-time impact of anomalies. Clearly, the stable causal graph can be derived from historical data. However, abnormal and normal statuses coexist in the historical data, making it difficult to evaluate which of the obtained causal associations are stable. Therefore, we employ multiple stochastic samplings to tackle this problem.

We maintain the causal graphs $\left\{G_{his}^{(i)}\right\}, i = 0, 1, ..., J+1$, called the *historical interval causal graphs*, which depict

stable causalities captured from historical data. Initially, the graph is empty, i.e.,

$$E_{his}^{(i)}(u,v) = 0, \forall u,v \in V \tag{11}$$

The time period of a particular host or microservice system preceding the abnormal interval is referred to as the historical time period, i.e., $[1, l_{ab}]$. G-Cause stochastically selects L historical intervals $[l_l, r_l], l \in [1, L]$ within $[1, l_{ab}]$. Following the method in subsection IV-C, for the interval $[l_l, r_l]$, we extract the metrics set $T[l_l : r_l]$ and build multi-band causal graphs $\left\{G_l^{(i)}\right\}, i = 0, 1, ..., J+1$. We then incrementally update the historical interval causal graphs by

$$E_{his}^{(i)}(u,v) = \frac{1}{l}(E_{his}^{(i)}(u,v) * (l-1) + E_l^{(i)}), \ \forall u,v \in V \tag{12}$$

on each frequency band f_i. As the number of samples l increases, the historical interval causal graph tends to stabilize, as verified in Figure 9. In the historical interval causal graphs, larger edge values suggest a greater likelihood of a stable causal relationship. Relatively, the lower the probability that such a relationship is associated with the anomaly.

It is worth noting that this incremental update enables *offline computation* of the historical interval causal graphs. That is, instead of waiting until anomalies occur to build the historical interval causal graphs, we can continuously keep stochastically sampling the time interval as host-level metrics are collected, and then incrementally update the historical interval causal graphs to keep them stable. The historical time period causal graphs are stored until they are retrieved when anomalies occur and can be used directly without having to be recalculated. This expedites the acquisition of the root cause metric recommendations.

If the system records metrics after an anomaly has occurred (that is $T[r_{ab}, M]$), we can still establish causal relationships using these data. Similarly, we maintain the *post-anomaly interval causal graphs* $\{G_{pos}^{(i)}\}, i = 0, 1, ..., J+1$, which enhances causalities related to the anomaly. The process of constructing the post-anomaly interval causal graphs is identical to that of constructing $\{G_{his}^{(i)}\}$, with the exception of using metrics data that occurred after the anomaly. We may optionally incorporate the post-anomaly interval causal graphs to provide an information gain.

E. Refining the Causal Graph

To facilitate causal graphs refining, we define two operators, the graph subtraction operator $\ominus$ and merging operator $\oplus$.

Graph extraction operator $\ominus$. Given two graphs with same nodes $G_1 = (V, E_1), G_2 = (V, E_2)$, we define $G_2 \ominus G_1 = G_3$ where

$$\begin{aligned} &G_3 = (V, E_3), \\ &E_3(u,v) = max\left\{E_2(u,v) - E_1(u,v), 0\right\}, \ \forall u,v \in V \end{aligned} \tag{13}$$

This operation depicts the reduction of the edge weight occupied by G_1 in G_2 while guaranteeing they are non-negative.

Graph merging operator $\oplus$. Given K graphs with same nodes $G_i = (V, E_i), i = 1, 2, ..., K$, we define $G_1 \oplus G_2 \oplus ... \oplus G_K = G$ where

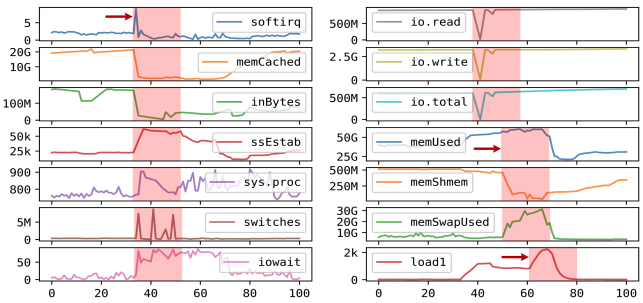


Fig. 5: A demo fault propagation.

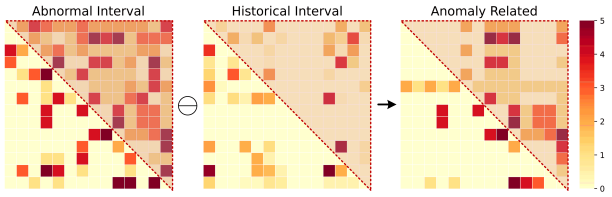


Fig. 6: Causal graphs built by selected host-level metrics in Figure 5

$$G = (V, E), \ E(u,v) = \sum_{i=1}^{K} E_i(u,v), \ \forall u,v \in V \tag{14}$$

This operation merges multiple graphs' edge weights.

Given the abnormal interval causal graphs $\{G_{ab}^{(i)}\}$, the historical interval causal graphs $\{G_{his}^{(i)}\}$, and the post-anomaly interval causal graphs $\{G_{pos}^{(i)}\}$, where $i = 0, 1, ..., J+1$, we compute

$$\left\{G^{(i)} = (G_{ab}^{(i)} \ominus G_{his}^{(i)}) \oplus G_{pos}^{(i)}\right\} \tag{15}$$

and define them as the *anomaly-related causal graphs*. We then merge the graphs of multiple frequency bands to one graph

$$G = G^{(0)} \oplus G^{(1)} \oplus ... \oplus G^{(J+1)} \tag{16}$$

To control the sparsity, we establish the threshold θ and retain only the edges with weights greater than θ in G to obtain the final refined anomaly-related causal graph G_θ. Subsequent experiments show that the threshold θ has little effect on the precision of root cause analysis and is therefore not considered a critical parameter.

We acquire the abnormal interval causal graph $G_{ab} = G_{ab}^{(0)} \oplus ... \oplus G_{ab}^{(J+1)}$ and the historical interval causal graph $G_{his} = G_{his}^{(0)} \oplus ... \oplus G_{his}^{(J+1)}$. Using the demo case in section I, we illustrate the distinctions among the three causal graphs. Figure 5 shows the sequential order of abnormal fluctuations of key metrics in HDT, while Figure 6 presents the corresponding causal graphs. To improve clarity, we filter edge weights by setting $\theta = 1$. The abnormal interval causal graph is dense, making pattern identification difficult. In contrast, the historical interval causal graph reveals stable intrinsic causal relationships, such as `memCached`, `memUsed`, `memSwapUsed` and `softirq`, `switches`, `load1`. These stable causalities are not strongly correlated with anomalies, so they appear less in

TABLE II: Statistics of SDT and HDT Datasets

Dataset Name	Statistics Type	Value
Microservice System Anomaly Metrics (SDT)	Number of alarmed microservice APIs	33 out of 1723
	Number of samplings per API	7200
	Sampling interval(s)	5 seconds
	Post-anomaly Data Availability	No
Host Machine Anomaly Metrics (HDT)	Number of abnormal host-level Metrics	38 out of 247
	Samplings number per metric	30402
	Sampling interval(s)	30 seconds
	Post-anomaly Data Availability	Yes

the anomaly-related causality graph. As a result, the anomaly-related causal graph exhibits a clearer upper triangular structure, showing the characteristic asymmetry of causality.

F. Root Cause Analysis

On the inverse graph of the anomaly-related causal graph $G_\theta^T = (V, E_\theta^T)$, we use the algorithmic framework of PageRank [46] to rank the importance of the metrics. As a classical, low-complexity graph recommendation technique, PageRank is frequently employed in root cause location of microservices systems [19], [17], [47], [48]. In the host-level root cause location scenario, entry metrics are selected as the abnormal metric most easily observed by O&M staff, such as CPU load or network traffic anomalies. We start each PageRank procedure with such a v_{entry} metric, and our graph random walk algorithm uses the Personalized PageRank [42] algorithm, a variant of PageRank. We define the probability distribution vector

$$v = \{v_{entry} = 1; v_i = 0, \forall i \in [1, N], i \neq entry\} \tag{17}$$

which reflects the initial node distribution at the restart of each random walk. Each time PageRank restarts a wander, it chooses the starting point of the wander depending on v. With the probability distribution vector v, we specify the entry node on the graph at the time of the random walk and lower the likelihood that PageRank would wander to a metric that has little association with the anomaly. We rank each metric in descending order of its stable transfer probability and provide k metrics and the corresponding probability values as our final recommendations.

In addition, we may skip computing the historical and post-anomaly interval causal graph and alternatively aggregate the abnormal interval causal graphs for each frequency band $\left\{G_{ab}^{(i)}\right\}, i = 0, 1, ..., J+1$ to obtain

$$G_{ab} = G_{ab}^{(0)} \oplus G_{ab}^{(1)} \oplus ... \oplus G_{ab}^{(J+1)} \tag{18}$$

and then make Personalized PageRank on G_{ab}. Subsequent experiments show that we can exchange faster computational efficiency with a certain loss of accuracy in this way.

V. Empirical Study

This section presents the training datasets, baselines, experimental result analysis, and implementation details in our production environment. We explore several important research questions (RQ) to demonstrate that G-Cause effectively discovers and exploits system-stable causal relationships when enabling multi-band root cause localization.

A. Experiments Setup

Datasets. We collect two datasets for both SDT and HDT. Our SDT dataset collects API logs of abnormal services for two hours, forming more than 10 million metrics covering over 1,500 different APIs on average. Root causes of incidents are labeled based on reports from the SRE team. The HDT dataset comprises dozens of anomalous cases that on-call operators encountered and handled in our production environment. O&M staff labeled each case with the most beneficial root cause metrics for analysis, which we use as the algorithm's ground truth. Dataset details are described in Table II.

Baselines. We select several state-of-the-art root cause analysis methods for microservice system as our baselines. To the best of our knowledge, our approach is the only algorithm currently using host-level monitoring metrics for root cause analysis, thus we modify selected baselines for microservice systems to enable processing of the host-level metrics. Baselines include:

- *MonitorRank* [16] is a classical algorithm based on a random walk on the dependency graph to get root cause recommendations. Since there is no potential ground-truth for the dependency graph, we use the causal graph generated by the PC algorithm as the dependency graph according to the algorithm in the previous paper [7]. We choose it because it is a classic and well-known algorithm for root cause location of microservices systems.
- *CloudRanger* [19] creates the dependency graph by PC algorithm [29], then makes a second-order random walk on the graph and generates the ranking of metrics based on the probability of the walk. We select it to compare the difference in effectiveness due to different cause-effect discovery methods.
- *DyCause* [7] creates dynamic causal graphs based on temporal Granger Causality test and then generates recommendations by inverse BFS on the graph. G-Cause, on the other hand, creates causal graphs on different frequency bands instead of different time periods and also differs in the search algorithm on the graph.

Evaluation metrics. We use two commonly used metrics in root cause analysis [16], [7], [19], PR@k and RankScore. PR@k measures the proportion of metrics given by the algorithm in the set of root cause metrics given the first k metrics in order. Another metric PR@avg measures the average accuracy of recommending 1 to 5 metric(s). RankScore(**RS**) measures the ranking of all recommended metrics, which is linearly correlated with the recommendation quality. RankScore reaches 1 when all the root cause metrics are listed at the top of the recommendation, and it is 0 when there is no correct candidate in the recommendation list.

Experiments are conducted on a workstation equipped with Intel(R) Xeon(R) CPU E5-2699 v3 @ 2.30GHz and 256 GB RAM. Our code runs on Anaconda 3 Python 3.8.16. For acceleration, we run the pairwise Granger causality test in subsection IV-C with 10 parallel works. The tool is released anonymously at https://ufile.io/vwg7r2pe.

TABLE III: Performance on our SDT and HDT datasets.

Dataset	Methods	PR@5	PR@Avg	RS	Time(s)
SDT	MonitorRank	0.823	0.829	0.982	47.18
	CloudRanger	0.889	0.845	0.979	42.09
	DyCause	0.980	0.864	0.995	**8.33**
	G-Cause (Abnormal)	**1.000**	0.975	0.975	17.66
	G-Cause (Fusion)	**1.000**	**1.000**	**1.000**	18.42
HDT	MonitorRank	0.167	0.275	0.417	39.14
	CloudRanger	0.208	0.287	0.325	37.12
	DyCause	0.312	0.322	0.350	**7.78**
	G-Cause (Abnormal)	0.521	0.535	0.565	15.92
	G-Cause (Fusion)	**0.636**	**0.661**	**0.693**	16.51

TABLE IV: Performance on HDT under different θ Values.

	θ	PR@∗				RS
		1	5	10	Avg	
G-Cause (Abnormal)	0	0.250	0.521	0.819	0.535	0.565
	1	0.167	0.549	0.903	0.554	0.604
	2	0.250	**0.667**	**0.910**	**0.609**	**0.668**
	3	**0.417**	0.479	0.792	0.585	0.613
G-Cause (Fusion)	0	0.508	0.636	**0.882**	0.661	0.693
	0.8	0.458	0.641	0.834	0.639	0.670
	1.6	0.483	0.657	0.851	0.664	0.695
	2.4	**0.600**	**0.714**	0.849	**0.705**	**0.738**

B. Result Analysis

Table III shows our average root cause location accuracy on both SDT and HDT. *G-Cause (Fusion)* denotes the algorithm that fuses historical and post-anomaly interval causal graphs to construct the anomaly-related causal graphs $\{G^{(i)} = G^{(i)}_{ab} \ominus G^{(i)}_{his}\}$ (or $\{G^{(i)} = (G^{(i)}_{ab} \ominus G^{(i)}_{his}) \oplus G^{(i)}_{pos}\}$ if G_{pos} is available), whereas *G-Cause (Abnormal)* utilizes only the abnormal interval causal graphs $\{G^{(i)}_{ab}\}$ for random walk, $i = 0, 1, \ldots, J+1$. Due to the stochastic nature of the sampling process of historical intervals, we randomize 5 experiments and average the accuracies. Our algorithm substantially outperforms the other baselines in all metrics, while baselines' low accuracies make them provide limited advice to O&M staff in real-world scenarios. The algorithm takes about 15∼18s to run if only data from the abnormal interval are used, which is even faster than the sampling interval of host-level monitoring metrics (30s), showing its superior performance. Sampling stochastically in historical time periods and creating stable causal graphs can bring PR@1 and PR@5 in 25.8% and 11.5% on HDT, but with a significant time overhead. However, generating stable causal graphs can be done offline, as mentioned in subsection IV-E. Thus the generation of stable causal graphs does not affect the time overhead when an alarm storm comes. Considering the high accuracy of both G-Cause and baselines on SDT, we focus on the more challenging HDT in the following experiments.

Figure 7 shows the results of each experiment on HDT. The stochasticity of sampling has a relatively large effect on PR@1, but a small effect on PR@5, which indicates that stochasticity will have an effect on the order of the top five, but has a limited effect on the validity of the recommendation results. For the stability analysis of the intermediate results

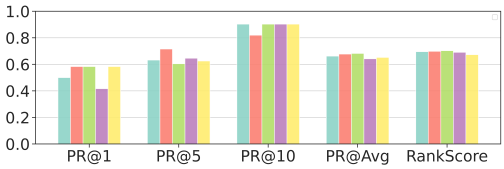


Fig. 7: Accuracies in a five-round experiment on HDT. Each bar represents an independent experiment.

TABLE V: Average edge weight over each frequency band on HDT. *Abnormal* and *Historical* represent the causal graph created in the corresponding time interval.

	Origin	cD1	cD2	cD3	cA3
Abnormal	0.312	0.232	0.254	0.266	0.296
Historical	0.226	0.196	0.225	0.272	0.293

of each experiment (i.e., the historical interval causal graphs generated in each experiment) see subsection V-D.

C. RQ1: Is it better to calculate Granger Causality by frequency bands than to calculate it only by original sequence?

The benefit of calculating Granger by frequency bands is reflected in two aspects: first, it provides a way to distinguish the causality strength and control the sparsity of the graph; second, it reveals the characteristics of the metrics on different frequency bands and demonstrates better interpretability. To justify its ability to distinguish the causality strength, we select the abnormal interval causal graph of case 4 in HDT as a demonstration. Its root causes are memory metrics. In Figure 8, *Origin* represents the Granger causal graph using only the original series in the abnormal interval, and *Cumulated* indicates the aggregated graph after sub-band calculation, as denoted in subsection IV-E. *cD1* and *cA3* represent the highest-frequency band and the low-frequency band, respectively. Compared to other graphs where each causal edge is indistinguishable, *Cumulated* sub-graph clearly enhances the causality strength of all memory metrics to others. In addition, the causality strength of all metrics to TCP metrics also increases. Therefore, $Memory \rightarrow TCP$ receives higher attention and memory metrics rank higher in the results.

To explain what it indicates about the properties of the metrics at different frequency bands, the mean of the edge weights of high and low-frequency causal graphs for the anomalous and historical intervals is calculated independently for each case in HDT, as shown in Table V. We find that the original series in the abnormal interval contains 9% more metric pairs passing the Granger test than the historical interval, indicating that the host system has a stronger causal linkage during the anomalous interval. If we examine each frequency band in greater detail, the average weights of the causal graph in the cA3 band do not grow much, however, they increase by 4% in the cD1 band. This suggests that the majority of the enhanced causal linkage is due to the high-frequency component, whereas the low-frequency component contributes little to the total increase in causal relationships. This is consistent with our experience: the degree of metric

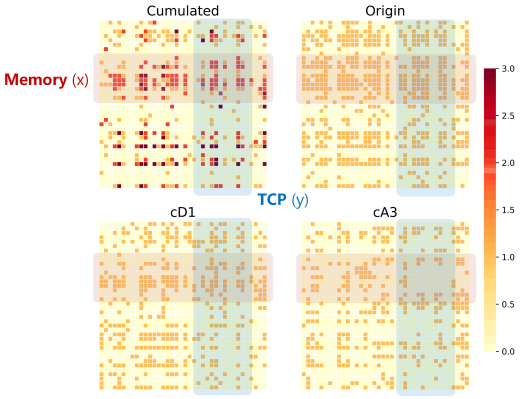


Fig. 8: Comparison of different frequency bands in the abnormal interval causal graph of Case 4 in HDT.

TABLE VI: Performance on HDT over different bands.

	Band	PR@∗				RS
		1	5	10	Avg	
Ours (Abnormal)	Origin	0.000	0.412	0.729	0.404	0.445
	cD1	0.000	0.250	0.486	0.261	0.301
	cD2	0.167	0.396	0.625	0.419	0.440
	cD3	0.167	0.188	0.486	0.232	0.251
	cA3	0.000	0.104	0.535	0.180	0.227
	Cumulated	**0.250**	**0.521**	**0.819**	**0.535**	**0.565**
Ours (Fusion)	Origin	0.500	0.471	0.675	0.522	0.535
	cD1	0.108	0.388	0.563	0.363	0.386
	cD2	0.367	0.513	0.706	0.521	0.549
	cD3	0.242	0.454	0.623	0.450	0.461
	cA3	0.192	0.469	0.737	0.463	0.510
	Cumulated	**0.508**	**0.636**	**0.882**	**0.661**	**0.693**

volatility increases dramatically during the abnormal interval, which corresponds to an active causality at high frequencies. Calculating Granger causality in frequency bands can therefore provide a more comprehensive understanding.

We also compare the results relying on the cumulated causal graph and each band's causal graph, as shown in Table VI. Here $J = 3$. *Cumulated* denotes the algorithm that uses the cumulated graph denoted in subsection IV-E, and *Origin*, *cD1*, *cD2*, *cD3*, *cA3* denotes the one that is using the sub-graph on the corresponding band. The experimental results show that the cumulated causal graph over all bands outperforms the causal graph of the individual band (especially the graph obtained by computing Granger causality using only the original sequence), as we expected. It is worth noting that PR@1 with only the original sequence and the anomaly interval data drops to 0, indicating that it is not available. It shows that the pairwise Granger test made directly on the original sequence is inaccurate since it does not guarantee any graph structure and sparsity.

D. RQ2: How to quantify the stability of historical causality among metrics?

If historical interval causality is not stable, then the process of removing historical interval causality from the abnormal interval is meaningless. We compute G_{his} for each case in HDT after 150 stochastic historical time periods sampled for

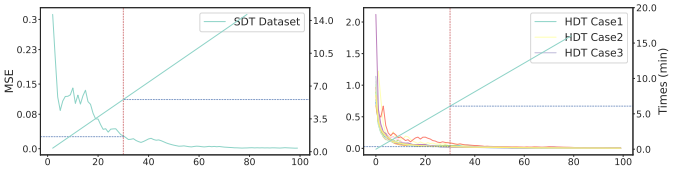


Fig. 9: The convergence and computation cost of historical interval causal graphs on post-anomaly interval causal graphs on SDT dataset (left) and HDT dataset (right).

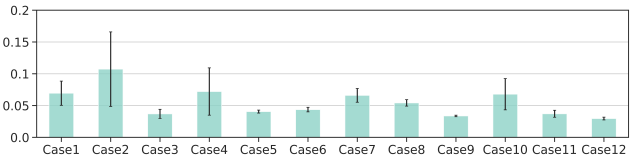


Fig. 10: The mean and variance of MSE of historical interval causal graphs on several typical cases in HDT.

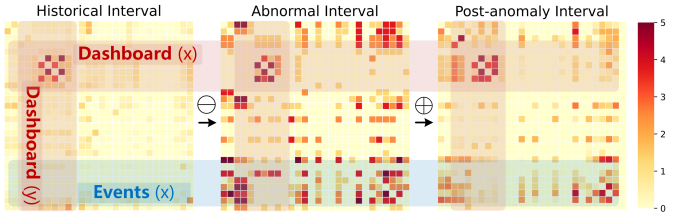
updating and compute the result $[G_{his}]_t$ after $t, t < 150$ times sampling and updating. We plot the mean square error (MSE) of the graph adjacency matrix $MSE(G_{his}, [G_{his}]_t)$ and the corresponding computation time separately, as shown in Figure 9 (legends in the right subgraph are truncated). Similarly, We also compute post-anomaly interval graphs $[G_{pos}]_t$ in SDT and show $MSE(G_{pos}, [G_{pos}]_t)$ with computation time. We see that the historical and post-anomaly interval causal graphs converge very quickly. For the sake of accuracy and cost, we set the samples for historical and post-anomaly intervals to 30.

Similarly, we would like to know how much the difference between the stable history causal graphs generated in different experiments on HDT. We average all stable historical interval causal graphs and then calculate the MSE of all historical interval causal graphs and the average graphs. Figure 10 shows the mean and variance of MSE in several typical cases. From the figure, we can find that all the mean MSE except for case No.2 is less than 0.1. Considering that the range of side weights is $[0, 1]$, such a fluctuation is acceptable.

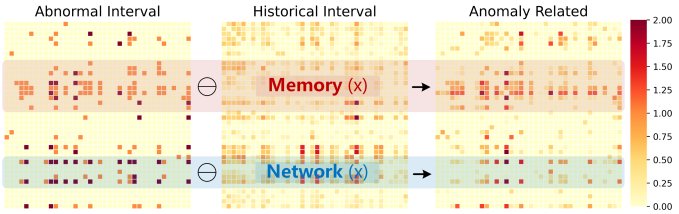
E. RQ3: How much influence does historical stable causality have on the generation of the final anomalous causal graph?

Following our previous case, we illustrate the effect of stable causal graphs on the abnormal interval graph. Figure 11b shows the abnormal interval, historical interval and anomaly-related causal graph on Case 6 in HDT, respectively. We find many causal pairs in the abnormal interval causal graph, among which memory and network metrics are predominant but not strongly differentiated. In contrast, the causality of memory metrics on others significantly differs from network metrics in the refined causal graph. This is because network metrics are more causal to other metrics in the historically stable causal graph, weakening their correlation with anomalies. In fact, most memory metrics are similarly weakened to varying degrees, with the actual root cause, `memSwapUsed`, topping the list with a recommended probability of 40.5%.

In Figure 11a, we visualize the historical, abnormal, and post-anomaly interval causal graph in SDT and observe dy-



(a) Historical, abnormal and post-anomaly interval causal graph in SDT dataset.



(b) Abnormal, historical and anomaly-related causal graph in HDT Case 6.

Fig. 11: Causal Graph Visualizations in SDT and HDT.

namic changes in causal relationships between metrics. In the historical interval, there is a strong causal relationship among the latency metrics of only the dashboard service. However, in the abnormal interval, the impact of the event service's latency on the dashboard significantly increases and this effect continues into the post-anomaly interval. If we only consider the causal graph of the abnormal period, it is difficult to identify which causal relationships play a dominant role in anomalous propagation. However, suppose we fuse historical and post-anomaly information. In that case, we can find that the causality between the latency metrics of the dashboard is inherent and unrelated to anomalies. In contrast, the event's impact on the dashboard has never appeared in the historical period but is significant in the abnormal period and continues into the post-anomaly period. Combining the graph fusion algorithm in subsection IV-E, the causal graph of the historical and post-anomaly interval provides information gain for root cause analysis of the abnormal interval.

We further illustrate RQ3 with the details of the demo case as a complement to subsection IV-E. The cause of the incident is that `Service-B` (appeared in Figure 1) needs to pull images from a remote repository during deployment, so the host employs a P2P client to speed up the process. This phase generates a great deal of network connections building (`ssEstab`) and bursts of network traffic (`inBytes`), and the process of receiving network packets also causes `softirq` and `memCached` to fluctuate dramatically. In the subsequent phase, the number of processes (`sys.proc`) in the system starts to increase, causing switching processes (`switches`) frequently, and the time spent waiting for disk I/O (`iowait`) also rises. At some point, the system disk crashes, rapidly decreasing I/O (`io.read/write/total`). Thereafter, as memory usage (`memUsed`) slowly climbs to its peak, the system starts to schedule the use of swap memory (`memSwapUsed`) until all available resources are exhausted. At this time, the system can no longer afford to keep `Service-A` running, and the CPU load spikes to 2K, resulting in an observable exception that is reported to O&M staff. However, this clear timeline is not noticeable in the abnormal interval causal graph. It is because the graph contains too many stable causal relationships, thus preventing the metric pairs associated with the anomaly from coming to the fore. By removing the set of stable causality (mentioned in subsection IV-E), we finally observe the strong influence of I/O metrics on other metrics. After confirming with the O&M staff, we are sure that the root cause of this incident is the *frequent disk I/O* that leads to a system crash, which affects memory usage and eventually causes the `load1` spike. Therefore, the truth of the anomaly is highlighted by removing the stable causality, and the root cause is subsequently determined through a random walk on the anomaly-related causal graph.

VI. Conclusion

This paper addresses the challenges of leveraging high-heterogeneity monitoring metrics for root cause location in global diagnosis tasks in hyperscale web service infrastructures. Inspired by our extensive operation and maintenance experiences, we design a lightweight parameter-free tool, G-Cause, to extract multiple frequency-bands causal relationships of metrics. G-Cause utilizes historical and post-anomaly data to quantify stable causalities using stochastic time domain sampling. We then create a causal graph highly correlated with anomalies and perform a random walk on it to identify the host-level root cause of failures. The results of G-Cause on both SDT (service-level diagnostic task) and HDT (host-level diagnostic task) scenarios in a top-tier Internet corporation show significant improvements in accuracy compared to other state-of-the-art methods. In addition, our approach provides outstanding interpretability and is parameter-free through process simplification. With these advantages, G-Cause successfully serves our worldwide product operations and impressively contributes to many other workflows. Our future work will extend the algorithm framework to other domains with dynamic causal structures, such as urban transportation prediction, social network analysis, etc.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (62072006, 92167104), Qiyuan Lab Innovation Fund (S20210201079), Foundation of Shuanghu Laboratory (2024JK15) and ByteDance University Research Project. Meng Ma and Ping Wang are the academic corresponding authors. Yu Zhang is the industry corresponding author.

References

[1] K. Ye, "Anomaly detection in clouds: Challenges and practice," in *Proceedings of the Workshop on Emerging Technologies for Software-Defined and Reconfigurable Hardware-Accelerated Cloud Datacenters*, ser. ETCD'17, 2017. [Online]. Available: https://doi.org/10.1145/3129457.3129497

[2] J. Ding, R. Cao, I. Saravanan, N. Morris, and C. Stewart, "Characterizing service level objectives for cloud services: Realities and myths," in *Proceedings of the IEEE International Conference on Autonomic Computing (ICAC'19)*, 2019, pp. 200–206.

[3] B. Beyer, C. Jones, J. Petoff, and N. Murphy, "Site reliability engineering: How google runs production systems," 2016.

[4] H. S. Gunawi, M. Hao, R. O. Suminto, A. Laksono, A. D. Satria, J. Adityatama, and K. J. Eliazar, "Why does the cloud stop computing? lessons from hundreds of service outages," in *Proceedings of the ACM Symposium on Cloud Computing*, ser. SoCC'16, 2016, p. 1–16. [Online]. Available: https://doi.org/10.1145/2987550.2987583

[5] M. Ma, J. Xu, Y. Wang, P. Chen, Z. Zhang, and P. Wang, "Automap: Diagnose your microservice-based web applications automatically," in *Proceedings of the Web Conference 2020*, ser. WWW'20, 2020, p. 246–258. [Online]. Available: https://doi.org/10.1145/3366423.3380111

[6] J. Thalheim, A. Rodrigues, I. E. Akkus, P. Bhatotia, R. Chen, B. Viswanath, L. Jiao, and C. Fetzer, "Sieve: Actionable insights from monitored metrics in distributed systems," in *Proceedings of the ACM/IFIP/USENIX Middleware Conference*, ser. Middleware'17, 2017, p. 14–27. [Online]. Available: https://doi.org/10.1145/3135974.3135977

[7] Y. Pan, M. Ma, X. Jiang, and P. Wang, "Faster, deeper, easier: Crowdsourcing diagnosis of microservice kernel failure from user space," in *Proceedings of the ACM SIGSOFT International Symposium on Software Testing and Analysis*, ser. ISSTA'2021, 2021, p. 646–657. [Online]. Available: https://doi.org/10.1145/3460319.3464805

[8] H. Shan, Y. Chen, H. Liu, Y. Zhang, X. Xiao, X. He, M. Li, and W. Ding, "ϵ-diagnosis: Unsupervised and real-time diagnosis of small-window long-tail latency in large-scale microservice platforms," in *Proceedings of the Web Conference*, ser. WWW'19, 2019, p. 3215–3222. [Online]. Available: https://doi.org/10.1145/3308558.3313653

[9] J. Qiu, Q. Du, K. Yin, S.-L. Zhang, and C. Qian, "A causality mining and knowledge graph based method of root cause diagnosis for performance anomaly in cloud applications," *Applied Sciences*, vol. 10, no. 6, p. 2166, 2020. [Online]. Available: https://doi.org/10.3390/app10062166

[10] L. Wang, N. Zhao, J. Chen, P. Li, W. Zhang, and K. Sui, "Root-cause metric location for microservice systems via log anomaly detection," in *Proceedings of the IEEE International Conference on Web Services (ICWS'20)*, 2020, pp. 142–150. [Online]. Available: https://doi.org/10.1109/icws49710.2020.00026

[11] P. Aggarwal, A. Gupta, P. Mohapatra, S. Nagar, A. Mandal, Q. Wang, and A. Paradkar, "Localization of operational faults in cloud applications by mining causal dependencies in logs using golden signals," in *Proceedings of the International Conference on Service Oriented Computing (ICSOC'21)*, 2021, pp. 137–149.

[12] S. L. Bressler and A. K. Seth, "Wiener–granger causality: A well established methodology," *NeuroImage*, vol. 58, no. 2, pp. 323–329, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1053811910002272

[13] X. Zhou, X. Peng, T. Xie, J. Sun, C. Ji, D. Liu, Q. Xiang, and C. He, "Latent error prediction and fault localization for microservice applications by learning from system trace logs," in *Proceedings of the ACM SIGSOFT Symposium on the Foundation of Software Engineering / European Software Engineering Conference*, ser. ESEC/FSE'2019, 2019, p. 683–694. [Online]. Available: https://doi.org/10.1145/3338906.3338961

[14] D. Liu, C. He, X. Peng, F. Lin, C. Zhang, S. Gong, Z. Li, J. Ou, and Z. Wu, "Microhecl: High-efficient root cause localization in large-scale microservice systems," in *Proceedings of the IEEE/ACM International Conference on Software Engineering (ICSE'21)*, 2021, pp. 338–347. [Online]. Available: https://doi.org/10.1109/icse-seip52600.2021.00043

[15] X. Zhou, X. Peng, T. Xie, J. Sun, C. Ji, W. Li, and D. Ding, "Fault analysis and debugging of microservice systems: Industrial survey, benchmark system, and empirical study," *IEEE Transactions on Software Engineering*, vol. 47, no. 2, pp. 243–260, 2021. [Online]. Available: https://doi.org/10.1109/tse.2018.2887384

[16] M. Kim, R. Sumbaly, and S. Shah, "Root cause detection in a service-oriented architecture," in *Proceedings of the ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS'13, 2013, p. 93–104. [Online]. Available: https://doi.org/10.1145/2465529.2465753

[17] L. Wu, J. Tordsson, E. Elmroth, and O. Kao, "Microrca: Root cause localization of performance issues in microservices," in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS'20)*, 2020, pp. 1–9.

[18] Y. Meng, S. Zhang, Y. Sun, R. Zhang, Z. Hu, Y. Zhang, C. Jia, Z. Wang, and D. Pei, "Localizing failure root causes in a microservice through causality inference," in *Proceedings of the IEEE/ACM International Symposium on Quality of Service (IWQoS'20)*, 2020, pp. 1–10. [Online]. Available: https://doi.org/10.1109/iwqos49365.2020.9213058

[19] P. Wang, J. Xu, M. Ma, W. Lin, D. Pan, Y. Wang, and P. Chen, "Cloudranger: Root cause identification for cloud native systems," in *Proceedings of the IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID'18)*, 2018, pp. 492–502.

[20] A. Samir and C. Pahl, "Dla: Detecting and localizing anomalies in containerized microservice architectures using markov models," in *Proceedings of the International Conference on Future Internet of Things and Cloud (FiCloud'19)*, 2019, pp. 205–213. [Online]. Available: https://doi.org/10.1109/ficloud.2019.00036

[21] H. Nguyen, Y. Tan, and X. Gu, "Pal: Propagation-aware anomaly localization for cloud hosted distributed applications," in *Managing Large-scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques*, ser. SLAML'11, 2011. [Online]. Available: https://doi.org/10.1145/2038633.2038634

[22] H. Nguyen, Z. Shen, Y. Tan, and X. Gu, "Fchain: Toward black-box online fault localization for cloud systems," in *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS'13)*, 2013, pp. 21–30. [Online]. Available: https://doi.org/10.1109/ICDCS.2013.26

[23] J. Soldani and A. Brogi, "Anomaly detection and failure root cause analysis in (micro) service-based cloud applications: A survey," *ACM Computing Surveys*, vol. 55, no. 3, 2022. [Online]. Available: https://doi.org/10.1145/3501297

[24] M. Ma, W. Lin, D. Pan, and P. Wang, "Self-adaptive root cause diagnosis for large-scale microservice architecture," *IEEE Transactions on Services Computing*, vol. 15, no. 3, pp. 1399–1410, 2022. [Online]. Available: https://doi.org/10.1109/tsc.2020.2993251

[25] W. Lin, M. Ma, D. Pan, and P. Wang, "Facgraph: Frequent anomaly correlation graph mining for root cause diagnose in micro-service architecture," in *Proceedings of the IEEE International Performance Computing and Communications Conference (IPCCC'18)*, 2018, pp. 1–8. [Online]. Available: https://doi.org/10.1109/pccc.2018.8711092

[26] L. Mariani, C. Monni, M. Pezzé, O. Riganelli, and R. Xin, "Localizing faults in cloud systems," in *Proceedings of the IEEE International Conference on Software Testing, Verification and Validation (ICST'18)*, 2018, pp. 262–273. [Online]. Available: https://doi.org/10.1109/icst.2018.00034

[27] J. Lin, P. Chen, and Z. Zheng, "Microscope: Pinpoint performance issues with causal graphs in micro-service environments," in *Proceedings of the International Conference on Service Oriented Computing (ICSOC'18)*, C. Pahl, M. Vukovic, J. Yin, and Q. Yu, Eds., 2018, pp. 3–20. [Online]. Available: https://doi.org/10.1007/978-3-030-03596-9_1

[28] P. Chen, Y. Qi, and D. Hou, "Causeinfer: Automated end-to-end performance diagnosis with hierarchical causality graph in cloud environment," *IEEE Transactions on Services Computing*, vol. 12, no. 2, pp. 214–230, 2019. [Online]. Available: https://doi.org/10.1109/tsc.2016.2607739

[29] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search, Second Edition*, ser. Adaptive computation and machine learning, 2000.

[30] M. Rhif, A. Ben Abbes, I. R. Farah, B. Martínez, and Y. Sang, "Wavelet transform application for/in non-stationary time-series analysis: A review," *Applied Sciences*, vol. 9, no. 7, p. 1345, 2019.

[31] P. Duhamel and M. Vetterli, "Fast fourier transforms: a tutorial review and a state of the art," *Signal processing*, vol. 19, no. 4, pp. 259–299, 1990.

[32] S. Winograd, "On computing the discrete fourier transform," *Mathematics of computation*, vol. 32, no. 141, pp. 175–199, 1978.

[33] C. E. Heil and D. F. Walnut, "Continuous and discrete wavelet transforms," *SIAM review*, vol. 31, no. 4, pp. 628–666, 1989.

[34] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *International Conference on Machine Learning*. PMLR, 2022, pp. 27 268–27 286.

[35] P. K. d. M. M. Freire, C. A. G. Santos, and G. B. L. da Silva, "Analysis of the use of discrete wavelet transforms coupled with ann for short-term streamflow forecasting," *Applied Soft Computing*, vol. 80, pp. 494–505, 2019.

[36] G. Woo, C. Liu, D. Sahoo, A. Kumar, and S. Hoi, "Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting," *arXiv preprint arXiv:2202.01575*, 2022.

[37] Z. Wang, X. Xu, W. Zhang, G. Trajcevski, T. Zhong, and F. Zhou, "Learning latent seasonal-trend representations for time series forecasting," *Advances in Neural Information Processing Systems*, vol. 35, pp. 38 775–38 787, 2022.

[38] K. K. Ramakrishnan and R. Jain, "A binary feedback scheme for congestion avoidance in computer networks," *ACM Transactions on Computer Systems (TOCS)*, vol. 8, no. 2, pp. 158–181, 1990.

[39] X. Jiang, Y. Pan, M. Ma, and P. Wang, "Look deep into the microservice system anomaly through very sparse logs," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 2970–2978.

[40] D. Sundararajan, *Discrete wavelet transform*, 2015.

[41] F. Benhmad, "Modeling nonlinear granger causality between the oil price and u.s. dollar: A wavelet based approach," *Economic Modelling*, vol. 29, no. 4, pp. 1505–1514, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0264999312000089

[42] B. Bahmani, A. Chowdhury, and A. Goel, "Fast incremental and personalized pagerank," *Proc. VLDB Endow.*, vol. 4, no. 3, p. 173–184, 2010. [Online]. Available: https://doi.org/10.14778/1929861.1929864

[43] S. Shimizu, P. O. Hoyer, A. Hyvarinen, and A. Kerminen, "A linear non-gaussian acyclic model for causal discovery," *Journal of Machine Learning Research*, vol. 7, no. 72, pp. 2003–2030, 2006. [Online]. Available: http://jmlr.org/papers/v7/shimizu06a.html

[44] S. Seabold and J. Perktold, "Statsmodels: Econometric and statistical modeling with python," in *Proceedings of the Python in Science Conference*, vol. 57, no. 61, 2010, pp. 10–25 080.

[45] A. Shojaie and E. B. Fox, "Granger causality: A review and recent advances," *Annual Review of Statistics and Its Application*, vol. 9, no. 1, pp. 289–319, 2022. [Online]. Available: https://doi.org/10.1146/annurev-statistics-040120-010930

[46] D. F. Gleich, "PageRank beyond the web," *SIAM Review*, vol. 57, no. 3, pp. 321–363, 2015. [Online]. Available: https://doi.org/10.1137/140976649

[47] L. Wu, J. Tordsson, J. Bogatinovski, E. Elmroth, and O. Kao, "Microdiag: Fine-grained performance diagnosis for microservice systems," in *Proceedings of the IEEE/ACM International Workshop on Cloud Intelligence (CloudIntelligence'21)*, 2021, pp. 31–36.

[48] G. Yu, P. Chen, H. Chen, Z. Guan, Z. Huang, L. Jing, T. Weng, X. Sun, and X. Li, "Microrank: End-to-end latency issue localization with extended spectrum analysis in microservice environments," in *Proceedings of the Web Conference 2021*, ser. WWW'21, 2021, p. 3087–3098. [Online]. Available: https://doi.org/10.1145/3442381.3449905