Context-aware Root Cause Localization in Distributed Traces Using Social Network Analysis (Work In Progress paper)

Mahsa Panahandeh
Electrical and Computer Engineering department,
University of Alberta
Edmonton, Alberta, Canada
panahand@ualberta.ca

Naser Ezzati-Jivan
Department of Computer Science, Brock University
St. Catharines, Ontario, Canada
nezzatijivan@brocku.ca

Abdelwahab Hamou-Lhadj
Department of Electrical and Computer Engineering,
Concordia University
Montreal, Quebec, Canada
wahab.hamou-lhadj@concordia.ca

James Miller
Department of Electrical and Computer Engineering,
University of Alberta, Canada
Edmonton, Alberta, Canada
jimm@ualberta.ca

ABSTRACT

The complexity of microservices and their distributed nature necessitates constant monitoring and tracing of their execution to identify performance problems and underlying root causes. However, the large volume of collected data and the complexity of distributed communications pose challenges in identifying and locating abnormal services. In this paper, we propose a novel approach that takes into consideration the importance of execution contexts in propagating and localizing performance root causes. We achieve this by integrating social network analysis techniques with spectrum analysis. To evaluate our proposed approach, we conducted an experiment using a real-world benchmark, and we observed promising preliminary results, with a success rate of 91.3% in correctly identifying the primary root cause (top-1), and a perfect 100% success rate in finding the root cause within the top three candidates (top-3).

CCS CONCEPTS

• **Software and its engineering → Software reliability**; **Software performance**.

KEYWORDS

Root-cause Localization, Social Network Analysis, Spectrum Analysis, Distributed Traces, Contextual Analysis

ACM Reference Format:

Mahsa Panahandeh, Naser Ezzati-Jivan, Abdelwahab Hamou-Lhadj, and James Miller. 2024. Context-aware Root Cause Localization in Distributed Traces Using Social Network Analysis (Work In Progress paper) . In *Companion of the 15th ACM/SPEC International Conference on Performance Engineering (ICPE '24 Companion), May 7–11, 2024, London, United Kingdom.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3629527.3651426

ICPE '24, May 7–11, 2024, London, United Kingdom

ACM ISBN 979-8-4007-0445-1/24/05…$15.00
https://doi.org/10.1145/3629527.3651426

1 INTRODUCTION

Despite the widespread adoption of microservices for their scalability, modularity, and rapid deployment capabilities, their distributed architecture introduces significant challenges in diagnosing performance issues and localizing their root causes. Consider a complex e-commerce platform built upon this architecture. Services frequently depend on each other to accomplish tasks. When a performance issue such as a slowdown occurs, it rarely remains isolated but instead propagates through dependent services. A performance issue, say a slowdown, in a single service can have a cascading effect on all services dependent on it. Alternatively, it might also be the case that poor performance in a particular service is actually rooted in another service it depends upon. Without understanding these structural interconnected dependencies, diagnosing issues will become a complicated process. In the worst cases, the actual root cause may be entirely overlooked.

Therefore, accurate diagnosis requires understanding services interactions and dependencies. This necessitates an in-depth study of structural dependencies, particularly in cases of forward or backward anomaly propagation where the starting point of an issue might be several services away from where it eventually manifests.

Understanding complex inter-dependencies is essential for comprehending anomaly propagation and troubleshooting distributed systems, a topic explored in various research works [5, 8, 13, 15]. Some researchers advocate for prioritized diagnostic processes based on the likelihood of anomaly propagation in different components [4, 18]. However, the specific challenges faced by existing studies vary. Certain studies [13] face challenges in precisely pinpointing the direction of dependencies and analyzing the propagation path of anomalies. Others are restricted to insights derived solely from abnormal system executions, overlooking the information provided by normal request propagation [8]. Some studies [4, 5, 15, 18] remain constrained, often relying on isolated aspects of individual services in anomaly propagation, lacking the holistic view needed for accurate root cause identification. This limitation stems from the inherent structural complexity of distributed systems, where services are interconnected and interdependent, adding multiple layers of complexity to anomaly resolution.

To address existing limitations, we introduce a strategy using social network algorithms to enhance spectrum analysis, a method for estimating system component faultiness based on successful and failed executions. Our approach incorporates three key contexts: individual services, service communities, and execution paths. Then, it explains how anomalies propagate through these contexts and identifies key nodes where performance issues likely originate. Utilizing graph theory concepts like PageRank and community detection adds a new layer of depth to our spectrum analysis. The result is a context-aware, finely-grained method for accurately pinpointing the root causes of performance issues. Our approach utilizes distributed traces [11], to represent these contexts.

Upon detecting anomalies, we generate graphs and use social network techniques to calculate the structural importance of services, including an assessment of execution path importance. The resulting impact scores are used as weights in our spectrum analysis, generating a ranked list of probable root cause services, thus reducing the debugging effort required by programmers [17].

Our preliminary tests on an instance of a real-world production microservice system in China Mobile Zhejiang, a known complex and large-scale system, indicate a 91.3% success rate of our approach in identifying the primary root cause (top-1) and a 100% success rate for locating the root cause among the top three candidates (top-3) across various scenarios. Our preliminary tests employed rigorous sampling and evaluation metrics, ensuring the robustness of our findings. We have designed our method for generalizability, so it can be applied in many different situations and we have concrete plans for future empirical validation to strengthen our findings.

Our main contributions are as follows: I) Employing social network analysis to construct and analyse the structural interdependence of services, communities, and execution paths for forward and backward anomaly propagation and root cause identification, II) Proposing an enhanced weighted spectrum analysis. While existing methods often rely on individual services and isolated executions, our approach breaks new ground by introducing contextual layers, encompassing individual services, service communities, and execution paths, into root cause analysis. This context-aware methodology refines spectrum analysis, providing a more accurate identification of root causes, and thus advances the state-of-the-art.

2 FOUNDATIONAL EXPERIMENTS

Spectrum-based techniques are commonly used for debugging and fault localization in software applications. These techniques gather various types of test coverage data to identify likely root causes of failures. Specifically, they collect metrics such as O_{ef}, O_{nf}, O_{ep}, and O_{np}, which count the presence or absence of a given component in both failed and successful test cases [18]. A risk factor, like the Ochiai risk factor, is then used to quantify the suspicion level for each component being the root cause [3, 10]. In microservices, spectrum-based methods utilize distributed traces for both normal and abnormal system states to perform similar analyses [17].

Applying spectrum-based methods to distributed traces has limitations in root cause localization. For instance, our experiments, shown in Figure 1, based on a scenario from dataset C published by Li et al. [6], found that the original spectrum analysis ranked the true root cause (docker_003) only fifth in suspicion, while it placed os_021 at the top, identified as the most suspicious due to having the highest Ochiai score of 0.319. Yu et al.'s approach [18], which integrates a personalized PageRank algorithm into spectrum analysis, also falls short. It places emphasis on service frequency in determining the significance of an execution path for root cause localization, but it lacks granularity in understanding anomaly propagation, leading to suboptimal root cause identification. Figure 1, middle table, shows our experiments when we integrate a PageRank algorithm with spectrum analysis. It identifies the true root cause in the third place, while os_021 remains at the top.

In our experiments, we noticed two issues with integrating spectrum analysis with PageRank for root cause identification. First, about 30% of the cases presented multiple services with identical suspicion scores, complicating the ranking. Second, both original and integrated methods struggle when the root cause does not frequently appear in abnormal traces, e.g., the presence of a frequent loop between non-true root cause services in abnormal traces.

To address these issues, we study the significance of services in interconnected groups (communities) rather than prioritizing them based on their frequency across all traces. Here, by 'communities,' we refer to clusters of services that frequently interact with each other, thereby forming a closely-knit functional group within the larger system. Finding communities assists in distinguishing observed contexts and differentiating between similar connectivity patterns, which highly reduces (up to 98%, according to our preliminary experimental results) the likelihood of encountering multiple services with the same suspicious score all occupying the same position in the rank list of candidates. Additionally, studying the significance of services according to the significance of their interconnected services prioritizes less frequently observed services when they are significant within their own context. Therefore, the root cause service can be detected even if it is not frequently invoked in the collected abnormal traces. In addition to studying the significance of services in communities, we introduce a novel aspect to the root cause identification process by evaluating contextual details in requests or traces. While existing research by Yu et al. [18] assigns more weight to shorter traces, our approach innovatively refines this by equalizing the importance of services and trace diversity, regardless of the trace length. This offers a more comprehensive and effective method for root cause localization. Please refer to the right table in Figure 1 for a comparison between our method (context-aware root cause localization) and traditional approaches, which clearly illustrates the efficacy of our strategy. With our approach, the true root cause is ranked at the top with an Ochiai score of 0.286.

3 SYSTEM DESIGN

Figure. 2 represents our context-aware root cause localization approach including several steps of data collection, Service Call Graph (SCG) construction, social network analysis, and spectrum analysis.

3.1 Data Collection

Our process is initiated in response to a detected anomaly. Once an anomaly is detected, we collect normal and abnormal distributed traces within a specified time window (5 minutes in this paper). Any

Services	Original Spectrum Analysis [15]					Spectrum Analysis Integrated with PageRank					Context-aware Root Cause Localization				
	O_{ef}	O_{nf}	O_{ep}	O_{np}	Ochiai score	Weighted O_{ef}	Weighted O_{nf}	Weighted O_{ep}	Weighted O_{np}	Ochiai score	Weighted O_{ef}	Weighted O_{nf}	Weighted O_{ep}	Weighted O_{np}	Ochiai score
docker_007	0	115	208	240	0	0	0	10.85	12.52	0	0	0	106.51	0	0
os_023	0	115	1	447	0	0	0	0.28	12.79	0	0	0	0.075	33.52	0
os_021	59	56	237	211	0.319	6.11	5.80	14.87	13.24	0.386	13.67	12.98	237.0	211.0	0.167
db_003	0	115	448	0	0	0	0	59.99	0	0	0	0	184.70	0	0
docker_006	0	115	214	234	0	0	0	10.87	11.88	0	0	0	31.45	34.39	0
os_022	56	59	229	219	0.309	3.57	3.76	25.82	24.69	0.243	18.32	19.30	211.82	202.57	0.196
docker_003	29	86	113	335	0.226	6.95	20.61	7.55	22.39	0.347	29.0	86.0	60.38	179.02	0.286
docker_004	30	85	110	338	0.236	7.29	20.65	7.30	22.47	0.361	23.04	65.29	110.0	338.0	0.212
docker_001	33	82	114	334	0.253	4.11	10.22	9.40	27.55	0.295	18.56	46.13	86.89	254.59	0.224
db_007	0	115	448	0	0	0	0	18.62	0	0	0	0	30.63	35.34	0
docker_005	3	112	320	128	0.015	0.08	3.11	15.28	6.11	0.011	0.33	12.34	46.99	18.79	0.013
docker_008	0	115	216	232	0	0	0	11.41	12.25	0	0	0	31.66	34.0	0
db_009	0	115	448	0	0	0	0	53.47	0	0	0	0	208.58	0	0
docker_002	25	90	111	337	0.199	4.93	17.75	9.05	27.50	0.276	25.0	90.0	11.0	337.0	0.199

Figure 1: Comparison of Original, PageRank-Integrated, and Context-aware Weighted Spectrum Analyses in a Real-world Scenario (True root cause: docker_003, Top-1 identified root cause by each method has been highlighted.)

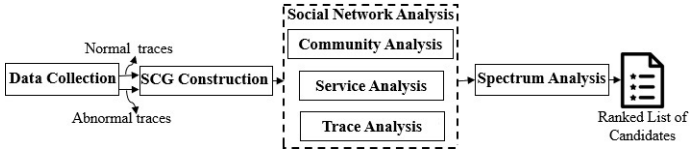

Figure 2: Context_aware Root Cause Localization

state-of-the-art anomaly detection techniques [12] can be employed to detect performance anomalies and label distributed traces as normal or abnormal. In this paper, we utilize the anomaly detection approach proposed by Li et al. [4]. However, as recommended by the literature [18], subsequent occurrences of the same anomalous state within the time window are not treated as separate anomalies.

3.2 SCG Construction

Following the data collection phase, two SCGs, built from each group of normal and abnormal traces, serve as weighted graphs. In these graphs, nodes symbolize services, edges represent service calls, and edge weights quantify the frequency of these calls within the respective set of traces. Distributed traces provide the necessary context information, capturing parent-child relations between services, which helps in constructing SCGs.

3.3 Social Network Analysis

In this phase, social network methods are employed to assess the structural influence of services communities, individual services, and traces during anomalies within SCGs. The output comprises importance scores for services and traces in both normal and abnormal conditions.

3.3.1 Community Analysis. Upon constructing normal and abnormal SCGs, we apply the Louvain graph community algorithm [1] to each SCG to partition them into smaller, closely related contextual communities. This aids in identifying cohesive groups and strong inter-node communication. For example, in abnormal SCG, these communities highlight partitions susceptible to anomalies should they contain an anomaly-affected node.

The Louvain method has two phases. First, nodes within an SCG are iteratively assessed and assigned to neighbouring nodes based on modularity cost function gains[1], continuing until no more modularity gains are achievable. Second, communities identified in the first phase are amalgamated into supervertices, converting nodes within each community into a single node. Supervertices' connectivity depends on at least one edge between nodes from corresponding communities, with the edge weight determined by the sum of all edges weight between their respective lower-level partitions. The algorithm iteratively applies these phases to supergraphs until communities stabilize, typically after a few rounds.

3.3.2 Service Analysis. Next, after identifying the community contexts, we quantify services importance within their community. To this end, we adopt the suggested approach in trace abstraction by Wang et al. [14]. First, we use an iterative PageRank algorithm [16] for each community to determine the significance of services based on their interactions within the community. The process begins by initializing the PageRank values for each service within the community. These initial values are set to $\frac{1}{n}$, where n represents the total number of services in the community. Subsequently, the PageRank values are iteratively calculated until they converge to a stable value. The PageRank for a service n in a community of C is determined based on the PageRanks of its neighbouring services over t iterations, and it can be defined as follows:

$$PR(n)_t = \alpha \sum_{(n \to n') \in \text{ edges of } C} \frac{PR_{t-1}(n')}{out\_degree(n')} \quad (1)$$

where α is a normalization factor for the total rank of all services and out_degree is the number of outgoing edges from n'.

This step is performed for services in identified communities of both normal and abnormal SCGs.

3.3.3 Trace Analysis. In this step, we focus on prioritizing different request types considering their effectiveness in uncovering a root

cause. According to Yu et al. [18], less diverse traces expedite root-cause localization. This is because more similar traces indicate a narrower scope of difference, simplifying the pinpointing of the root cause. However, Yu et al. measure trace diversity using the count of operations covered in traces, which makes it dependent on trace size. Consequently, shorter traces with less important services may overshadow longer ones with more critical services. In our approach, we prioritize traces by considering both their diversity and the significance of the services they cover, regardless of the number of services involved.

We first, cluster collected traces based on their request type separately for normal and abnormal distributed traces. This approach ensures that we study all observed request types, regardless of how frequently they have occurred. Then, for each request-type cluster, we calculate a score based on the diversity and importance of covered services.

To measure the importance score of clusters based on the importance of the services they cover, we adjust the formula recommended by Chen et al. [2] as follows. This refinement allows us to measure the rank score of a cluster cl_i based on the PageRank score (PR) of services they cover. In this context, PR is the computed scores using equation 1, The function $L(X)$ denotes the position of score X within the ordered list of ascending PR scores within the SCG, and |SCG| represents the number of SCG's nodes (services). For abnormal request-type clusters, SCG refers to abnormal SCG and PRs are scores of all services computed from abnormal SCG. The same is applied to normal request-type clusters.

$$Rank(cl_i) = \frac{L(Max(PRs \in cl_i) - 1}{|SCG|} + \frac{Mean(PRs \in cl_i)}{|SCG|sum(PRs \in SCG)} \quad (2)$$

Next, we measure diversity between request-type clusters using Jaccard distance [7], favouring less diverse clusters. Finally, the adapted heuristic search algorithm [2] searches for the next cluster based on the prior one, aiming to maximise the sum of Rank(cl_i) score while minimizing the cluster diversity.

3.4 Spectrum Analysis

Considering the importance of services in the community context and request type (trace), we redefine spectrums. For instance, O_{ef} is modified as follows where, T is a set of abnormal traces of $T_1, T_2, ..T_k$ including service s_i, cl_{T_j} is the abnormal request-type cluster for trace T_j, and $PR(s_i)$ is the Pagerank for s_i in the abnormal SCG.

$$O_{ef}(s_i) = PR(s_i) \times \sum_{\forall T_j \ including \ s_i, T_j \in T} rank(cl_{T_j}) \quad (3)$$

Similarly, other notation definitions are updated by being influenced by the rank scores, while O_{ep} and O_{np} are computed based on the normal SCG, normal set of traces, and normal request-type clusters identified in the previous step.

To estimate the suspicious score using the notations, we use the Ochiai factor [10], measured for each service s_i as:

$$Ochia(s_i) = \frac{O_{ef}}{\sqrt{(O_{ef} + O_{ep})(O_{ef} + O_{nf})}} \quad (4)$$

4 PRELIMINARY RESULT AND DISCUSSION

We evaluate the efficiency of our approach by conducting experiments on a real-world microservice benchmark (dataset C), provided during the 2020 AIOps Challenge Event[1] [6]. Given that this dataset extends beyond microservice applications, in alignment with literature recommendations [4, 18], we exclusively focus on faults related to microservice. Our evaluation involves the random selection of 46 time windows, each containing labelled root causes, as well as corresponding normal and abnormal distributed traces. Our experiment dataset includes 15 instances of CPU stress, 15 cases of network delays, and 16 occurrences of network loss. We define "Top-1" to "Top-3" as the probability of locating the true root causes within the top 1 to 3 service instances among all services, descendingly sorted based on their computed Ochiai score. This sorted list of ranked services is referred to as the ranked list of candidates [8, 18]. Figure. 3 shows the result of root cause identification by our context-aware root cause localization approach for all 46 time windows compared to the original spectrum analysis [17] and spectrum analysis integrated with PageRank, inspired by [18]. The context-aware root cause localization outperforms the spectrum analysis integrated with PageRank by 13.5% in detecting the true root cause at the top position of the ranked list of candidates, and it performs 35% better than the original spectrum analysis.

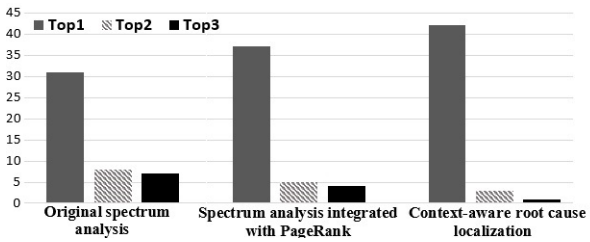

Figure 3: Performance Comparison Across 46 Scenarios: Original Spectrum Analysis [17] vs. Enhanced Methods

Our approach also exhibits a higher success rate in detecting root causes of CPU exhaustion scenarios, followed by network loss scenarios, and finally network delay cases. This performance variation can be attributed to the ability of our approach to effectively capture abnormal behavioural patterns within the collected traces. As anomalies propagate through more traces and spans, they become more likely to be successfully identified. Our initial investigation revealed that CPU exhaustion affected a larger number of services compared to the delay. This discrepancy can be attributed to the different approaches used to inject these anomalies into the system.

To evaluate the importance of studying services in contexts of communities and traces for root cause localization, we examine a scenario containing more than 530 traces collected after injecting the CPU stress into one of the services of the benchmark. After labelling normal and abnormal traces identified by the anomaly detection stage, we perform an original spectrum analysis to find the root cause. We then incorporate our approach components into the original spectrum analysis one by one to highlight how each contributes to enhancing the result of root cause localization.

[1] https://github.com/NetManAIOps/AIOps-Challenge-2020-Data

Table 1 demonstrates the result of each improvement applied to the same scenario, indicating the position within the ranked list of candidates where the root cause was identified by that specific improvement. The original spectrum analysis [17] locates the underlying root cause of this scenario in the sixth position of the ranked list of candidates. As shown, using our context-aware root cause localization, the true root cause is identified at the top position while each component also enhances the accuracy of root cause identification.

Table 1: Performance Ranking of Localization Components

Method	Pos.
Orig. Spectrum [17]	6th
Orig. + Service PageRank	4th
Orig. + Community_based Service PageRank	2nd
Context-aware: Full Method	1st

5 RELATED WORK

Root cause localization has become prominent in recent research due to its significant role in assuring the quality of complex systems [9, 13, 15, 17, 18]. A common approach to finding root causes involves studying the dependencies between services or traces to understand anomaly propagation, ultimately pinpointing the underlying root cause [5, 8, 15]. However, certain research works [13], face limitations in determining the propagation direction between dependent components. Furthermore, a significant portion of root-cause localization methods concentrates solely on abnormal executions [8, 13, 15].

Ye et al. [17] emphasize the significance of using both normal and abnormal traces, proposing a root cause localization approach based on an original spectrum analysis that leverages all traces to identify root causes. Addressing the requirements of spectrum analysis for distributed traces, Yu et al. [18] suggest integrating a personalized PageRank algorithm with the original spectrum analysis. The proposed personalized PageRank prioritizes services and traces based on their importance in uncovering root causes. However, as mentioned in Section 2, we found that studying the importance of individual services in Yu et al. study [18], is not always effective, especially when the true root cause is not frequently observed in abnormal traces. Moreover, Yu et al. [18] study the importance of traces in revealing the root cause based on their frequency and length, which is not always applicable, especially when the true root cause occurs in longer traces calling only a few services. These limitations are also discussed in Sections 2 and 3.

To address these limitations, we incorporate spectrum analysis with social network concepts to assess structural importance in forward and backward anomaly propagation across interconnected services. Studying the importance of services in interconnected communities helps analyze the significance of services within their respective communities. This approach overcomes biased rankings of services based solely on their high outgoing connections, neglecting the density of connections in overall SCGs. Moreover, to examine the importance of trace scope in uncovering root causes, we introduce a heuristic search algorithm to simultaneously evaluate trace importance based on both the significance and diversity of covered services within the trace scope. This makes trace scope analysis independent of the trace length and differentiates traces based on what they cover rather than their length. As our preliminary results show, this improves upon simply counting called services in each trace context, as done in Yu et al.'s work [18].

While there are research works employing social network analysis in root cause localization, they are often limited in utilizing techniques for studying the prominence of individual nodes within a network [9, 18, 19]. To the best of our knowledge, our work stands as the only research investigating the impact of different levels of contextual structures, such as individual services, service communities, and traces, in uncovering root cause localization using social network techniques.

6 CONCLUSIONS AND FUTURE PLAN

This study introduces a novel context-aware approach for root cause localization in distributed systems. Our methodology underscores the pivotal role of service communities, individual services, and trace scope in pinpointing the root causes of system anomalies. Through our work, we have effectively mitigated the issues delineated in Section 2. Preliminary outcomes showcase a high success rate, ranging from 91.36% to 100%, in accurately identifying root causes across diverse settings.

Moving forward, there are several directions for further improvement and extension. Firstly, we plan to explore modifications to obtain more detailed models beyond SCGs. These models can incorporate additional modalities, such as profiling metrics, enabling us to add performance insights to our social network analysis and analyze execution states instead of solid services. This will also help provide explanations about the issues associated with the ranked candidates, aiding in debugging or further investigations.

Furthermore, we aim to investigate how our approach impacts the detection of multi-root causes within service communities. By focusing on the interplay of anomaly propagation within communities, we anticipate that our methodology may excel in identifying complex scenarios where multiple root causes manifest within or across service communities.

Additionally, we aim to explore and adapt network analysis concepts that align with the unique characteristics of distributed traces. By leveraging these concepts, we can further investigate their correlation with different system types or collected data, potentially uncovering new insights instrumental in customizing network analysis concepts, such as community detection for distributed traces, thereby enhancing the precision of our methodology.

In conjunction with these efforts, we plan to evaluate our approach across a diverse set of case studies varying in size, complexity, and nature of their design. This assessment will help us gauge the scalability and adaptability of our approach for real-world systems.

Lastly, we acknowledge the importance of performing comprehensive comparative assessments against established methods. Such evaluations are crucial for validating the efficacy of our approach and identifying its advantages, shortcomings, and avenues for refinement across different use cases.

REFERENCES

[1] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[2] Lizhe Chen, Ji Wu, Haiyan Yang, and Kui Zhang. 2022. Does PageRank apply to service ranking in microservice regression testing? *Software Quality Journal* 30, 3 (2022), 757–779.

[3] James A. Jones, Mary Jean Harrold, and John Stasko. 2002. Visualization of Test Information to Assist Fault Localization. In *Proceedings of the 24th International Conference on Software Engineering* (Orlando, Florida) *(ICSE '02)*. Association for Computing Machinery, New York, NY, USA, 467–477. https://doi.org/10.1145/581339.581397

[4] Zeyan Li, Junjie Chen, Rui Jiao, Nengwen Zhao, Zhijun Wang, Shuwei Zhang, Yanjun Wu, Long Jiang, Leiqin Yan, Zikai Wang, Zhekang Chen, Wenchi Zhang, Xiaohui Nie, Kaixin Sui, and Dan Pei. 2021. Practical Root Cause Localization for Microservice Systems via Trace Analysis. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*. 1–10. https://doi.org/10.1109/IWQOS52092.2021.9521340

[5] Zeyan Li, Nengwen Zhao, Mingjie Li, Xianglin Lu, Lixin Wang, Dongdong Chang, Xiaohui Nie, Li Cao, Wenchi Zhang, Kaixin Sui, et al. 2022. Actionable and interpretable fault localization for recurring failures in online service systems. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 996–1008.

[6] Zeyan Li, Nengwen Zhao, Shenglin Zhang, Yongqian Sun, Pengfei Chen, Xidao Wen, Minghua Ma, and Dan Pei. 2022. Constructing large-scale real-world benchmark datasets for AIOps. *arXiv preprint arXiv:2208.03938* (2022).

[7] Jackson A Prado Lima and Silvia R Vergilio. 2020. Test Case Prioritization in Continuous Integration environments: A systematic mapping study. *Information and Software Technology* 121 (2020), 106268.

[8] JinJin Lin, Pengfei Chen, and Zibin Zheng. 2018. Microscope: Pinpoint performance issues with causal graphs in micro-service environments. In *Service-Oriented Computing: 16th International Conference, ICSOC 2018, Hangzhou, China, November 12-15, 2018, Proceedings 16*. Springer, 3–20.

[9] Leonardo Mariani, Cristina Monni, Mauro Pezzé, Oliviero Riganelli, and Rui Xin. 2018. Localizing faults in cloud systems. In *2018 IEEE 11th International Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 262–273.

[10] Lee Naish, Hua Jie Lee, and Kotagiri Ramamohanarao. 2011. A model for spectra-based software diagnosis. *ACM Transactions on software engineering and methodology (TOSEM)* 20, 3 (2011), 1–32.

[11] Austin Parker, Daniel Spoonhower, Jonathan Mace, Ben Sigelman, and Rebecca Isaacs. 2020. *Distributed tracing in practice: Instrumenting, analyzing, and debugging microservices*. O'Reilly Media.

[12] Jacopo Soldani and Antonio Brogi. 2022. Anomaly detection and failure root cause analysis in (micro) service-based cloud applications: A survey. *ACM Computing Surveys (CSUR)* 55, 3 (2022), 1–39.

[13] Jörg Thalheim, Antonio Rodrigues, Istemi Ekin Akkus, Pramod Bhatotia, Ruichuan Chen, Bimal Viswanath, Lei Jiao, and Christof Fetzer. 2017. Sieve: Actionable insights from monitored metrics in distributed systems. In *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference*. 14–27.

[14] Ji Wang and Naser Ezzati-Jivan. 2020. Enhanced execution trace abstraction approach using social network analysis methods. *Softwaretechnik-Trends* 40, 3 (2020), 58–60.

[15] Li Wu, Johan Tordsson, Jasmin Bogatinovski, Erik Elmroth, and Odej Kao. 2021. MicroDiag: Fine-grained Performance Diagnosis for Microservice Systems. In *2021 IEEE/ACM International Workshop on Cloud Intelligence (CloudIntelligence)*. 31–36. https://doi.org/10.1109/CloudIntelligence52565.2021.00015

[16] W. Xing and A. Ghorbani. 2004. Weighted PageRank algorithm. In *Proceedings. Second Annual Conference on Communication Networks and Services Research, 2004.* 305–314. https://doi.org/10.1109/DNSR.2004.1344743

[17] Zihao Ye, Pengfei Chen, and Guangba Yu. 2021. T-Rank:A Lightweight Spectrum based Fault Localization Approach for Microservice Systems. In *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. 416–425. https://doi.org/10.1109/CCGrid51090.2021.00051

[18] Guangba Yu, Pengfei Chen, Hongyang Chen, Zijie Guan, Zicheng Huang, Linxiao Jing, Tianjun Weng, Xinmeng Sun, and Xiaoyun Li. 2021. MicroRank: End-to-End Latency Issue Localization with Extended Spectrum Analysis in Microservice Environments. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 3087–3098. https://doi.org/10.1145/3442381.3449905

[19] Guangba Yu, Zicheng Huang, and Pengfei Chen. 2021. TraceRank: Abnormal service localization with dis-aggregated end-to-end tracing data in cloud native systems. *Journal of Software: Evolution and Process* (2021), e2413.