Beyond Traditional Problem-Solving: A Causal Pathway Approach for Complex Operational Environments

Ahmed Dawoud* Shravan Talupula†

February 9, 2025

Abstract

Root Cause Analysis (RCA) is becoming ever more critical as modern systems grow in complexity, volume of data, and interdependencies. While traditional RCA methods frequently rely on correlation-based or static rule-based techniques, these approaches can prove inadequate in highly dynamic, multi-layered environments such as microservices, cloud-native architectures, and real-time operational workflows. In this paper, we present an extended pathway-tracing framework built on top of the *DoWhy* causal inference library. Our method integrates *conditional anomaly scoring*, *noise-based attribution*, and *depth-first path exploration* to reveal multi-hop causal chains. By systematically prioritizing anomalous pathways rather than isolated nodes, we provide a practical, end-to-end RCA solution that pinpoints initial triggers with minimal false leads. Experimental evaluations with synthetic anomaly injections demonstrate the framework's ability to accurately isolate triggers, ranking root causes in order of their impact on the observed disruptions. These results underscore the importance of shifting from correlation-based anomaly detection to *causality-aware* approaches, with immediate relevance for fields spanning software reliability, healthcare, retail, finance, manufacturing, supply-chain logistics, and more.

*adawoud@profitops.ai
†stalupula@profitops.ai

Contents

1 Introduction and Motivation

Modern operational landscapes—spanning retail, healthcare, finance, and software systems—frequently confront anomalies that are difficult to diagnose. Although a variety of data-driven tools exist (e.g., correlation-based dashboards or anomaly detection algorithms), they often stop at revealing *what* went wrong without clarifying *why*. In practice, multiple factors typically converge to generate system disruptions, leading to subtle but far-reaching ripple effects that cannot be explained by simple pairwise correlations.

Our overarching goal in this paper is to advance from point-level anomaly detection toward a more holistic *root cause pathway* perspective. Rather than focusing on a single culprit node (e.g., `CPU_OVERLOAD`), we propose systematically tracing entire multi-hop pathways that manifest in real-world events (e.g., excessive discounting driving down margins, distribution system failures propagating to missed shipments, etc.). By identifying each link in the causal chain, decision-makers can intervene precisely where it matters most, minimizing both false leads and repeated disruptions.

Why Do Existing Tools Fall Short?

Table 1 contrasts three commonly referenced approaches—*DoWhy*, *Power BI Anomaly Detection & Explanation*, and *SHAP*—according to their methodology, scope, and capacity for multi-level causal insights. While these tools excel at certain subtasks (e.g., effect estimation or quick anomaly detection), none natively supports *complete, end-to-end pathway reconstruction* that traverses an entire directed acyclic graph from an observed anomaly back to its root triggers. Moreover, partial correlation analyses cannot distinguish *correlation* from *true causation*, and localized feature importance methods (e.g., SHAP) do not capture how anomalies propagate across multiple nodes.

Goal of this Work

We aim to build a unified pipeline that:

1. *Identifies Anomalies:* Leverages conditional distributions in causal models to detect instances where system behavior deviates significantly from expected patterns.

2. *Traces Multi-Hop Paths:* Employs a depth-first path enumeration, guiding the search from any observed anomaly backward through its parents until reaching terminal or low-scoring nodes.

3. *Ranks Potential Causes:* Uses custom metrics to rank entire *pathways* rather than single points, helping operators focus on the most likely and impactful disruptions.

In doing so, we extend DoWhy's core causal inference capabilities, transforming it into a more comprehensive *root cause discovery* framework. Our approach maintains an accessible interface (e.g., Python code that can be integrated into operational analytics systems) while providing deeper, end-to-end causal insights. Through synthetic and real-world-inspired scenarios, we illustrate how this extension reliably pinpoints the "earliest breaks" in a causal chain, enabling decision-makers to intervene effectively and minimize repeated failures or wasted remediation efforts.

Table 1: Comparison of Established Tools Versus Our Proposed Pathway-Based Approach

	DoWhy	Power BI Anomaly Detection	SHAP
Focus	Causal effect estimation, identification, and refutation	Time-series outlier detection & correlation-based factor analysis	Feature contribution to black-box model predictions
Core Method	Structural causal models (DAG-based)	Statistical outlier detection (e.g., SR-CNN) + dimension-based explanations	Game-theoretic local explanations (Shapley values)
Causality vs. Correlation	Explicit cause-and-effect relationships	Primarily correlation-based (factors "associated" with anomalies)	Feature importance for correlation-based models
Multi-Hop Path Tracking	Limited (anomaly-scoring is not natively included)	No (explanations are per anomaly, lacking chain-level detail)	No (explanations focus on single data points, not multi-node paths)
Use Cases	Policy evaluation, average treatment effects, classical econometrics	Business intelligence dashboards, quick time-series anomaly detection	Model interpretability (e.g., gradient boosting, neural nets)
Proposed Approach (This Paper)	**Fully integrates causal DAGs and anomaly attribution to discover multi-hop, end-to-end pathways linking top-level metrics back to their earliest triggers. Admits threshold-driven path pruning and combined anomaly/noise scores for robust root cause prioritization.**		

2 Literature Review

Root Cause Analysis (RCA) is a cornerstone methodology across disciplines—ranging from **engineering** and **manufacturing** to **healthcare**, **finance**, and increasingly **complex software/IT systems**. Its central objective is to move beyond superficial detection of anomalies and failures, instead *pinpointing the underlying chain of causes* that trigger them. In today's *cloud-native* and *microservice*-driven environments, RCA plays a pivotal role in ensuring reliability, performance, and security. However, modern systems' *massive data volumes*, *intricate interdependencies*, and *highly dynamic architectures* have exposed the limitations of traditional RCA methods, fueling the need for more robust, *causality-aware* techniques [1, 2].

2.1 Challenges in Root Cause Analysis for Complex Systems

Data Volume and Variety. Modern observability pipelines ingest logs, metrics, traces, and alerts in staggering volumes, leading to information overload for human operators. Manual approaches are consequently prone to oversight or bias in identifying suspicious patterns [1, 2]. The complexity is further compounded in large-scale microservice platforms, where each service generates its own performance metrics (e.g., CPU/memory usage, response times), trace data, and logs [3, 4].

Complex Interdependencies and Cascading Effects. Failures can propagate across multiple services and infrastructure layers, making it challenging to isolate an *initial trigger* from downstream symptoms [5, 6, 7]. Interdependent network structures—such as microservices running on top of a container orchestration layer on top of cloud compute—intensify these causal chains [6, 7].

Dynamic and Evolving Systems. Frequent updates (e.g., CI/CD pipelines), load fluctuations, and scaling events mean system behavior is never static. As a result, *fixed* or *rule-based* approaches fail to adapt adequately over time, rendering them less effective [5, 8]. Online or continuous RCA frameworks must therefore accommodate rapidly evolving architectures [9, 10].

Heterogeneity and Noise. Distributed systems often incorporate diverse technologies (databases, message queues, caches), each producing heterogeneous metrics and logs [6]. This mixture of signals can lead to high noise levels, complicating efforts to identify genuine causal relationships [7].

Partial Observability and Unknown Ground Truth. In practice, not all relevant variables or states are monitored; hidden confounders often distort naive correlation analysis [7, 16]. Furthermore, the true root cause of a system failure is rarely *labeled* or verified in production scenarios, hindering purely supervised learning approaches [7, 17, 18].

2.2 Categorization of RCA Approaches

Over the years, the literature on RCA has branched into various methodologies, each addressing specific needs. Below, we synthesize the main categories—**statistical/data-driven**, **knowledge-based**, **causal inference**, and **hybrid**—while highlighting recent works emphasizing multi-level causal explanations.

2.2.1 Statistical and Data-Driven Methods

Anomaly Detection. Anomaly (or outlier) detection is often a *precursor* to RCA, highlighting when and where a system deviates from expected behavior [9, 11, 8, 10]. Approaches range from simple thresholding to advanced machine learning and deep learning techniques. Although anomaly detection is crucial for discovering *when* the system is off track, it seldom clarifies *why* a deviation occurred. As noted by Akoglu et al. [12], future directions include anomaly *explanation*.

Correlation and Regression-Based Methods. Traditional statistical correlation (e.g., Pearson, Granger causality) and linear regression are widely used to pinpoint potential culprits [13, 14]. However, the axiom *correlation* $\neq$ *causation* remains a persistent limitation, particularly in large-scale, high-dimensional microservices with confounding factors [15, 7].

Performance or Resource-Based Analysis. Many data-driven RCA systems link performance symptoms (e.g., high latency) to resource utilization anomalies (e.g., CPU saturation) [18, 19]. For instance, Wu et al. [20] introduced *MicroRCA*, employing graph-based propagation of performance anomalies to localize offending microservices without deep instrumentation. While useful in operational contexts, purely data-driven methods still do not guarantee *causal* explanations [6].

2.2.2 Knowledge-Based and Expert Systems

Troubleshooting Guides (TSGs) and Playbooks. Organizations often maintain static documentation or step-by-step guides to tackle known failure modes [21, 22]. Although invaluable, their static nature and reliance on explicit domain knowledge make them less agile in fast-evolving cloud environments [22].

Rule-Based and Expert Systems. Early RCA solutions often encoded domain knowledge as sets of if-then rules [14]. However, the *combinatorial complexity* of modern systems, plus frequent deployments, has rendered purely rule-based methods difficult to maintain at scale.

Domain Knowledge + Data. Some organizations blend domain expertise with anomaly detection or regression-based signals to form semi-automated RCA pipelines [23, 24]. Jeyakumar et al. [25] propose *ExplainIt!*, offering a *declarative interface* for time-series RCA that integrates user queries and domain knowledge in real-time. These hybrid approaches, however, mostly rely on correlations rather than formal causal reasoning.

2.2.3 Causal Inference and Causal Discovery Methods

To transcend the pitfalls of correlation-based analyses, an increasing number of studies integrate *causal inference* into RCA [26, 27]. Causal approaches promise robust, explainable root cause detection by explicitly modeling the causal structure of the system.

Causal Bayesian Networks (CBNs). CBNs represent causal dependencies, allowing *probabilistic inference* to identify the factor that most likely caused an anomaly [26]. Li et al. [28] extend this with *intervention recognition*, systematically determining *which node's distribution shift* caused the observed fault.

Structural Causal Models (SCMs) and Mechanism-Based Explanations. SCMs decompose a system's joint distribution into autonomous causal mechanisms. Budhathoki et al. [29] apply SCMs to outlier root cause analysis, using Shapley values to quantify each ancestor's contribution. Janzing et al. [30] emphasize structure-preserving interventions and *intrinsic causal contributions* to disentangle direct from inherited influences. Similarly, the question of *"Why did the distribution change?"* [31] uses KL divergence to pinpoint mechanism-level shifts.

Causal Discovery Algorithms. Algorithms like PC learn causal graphs from observational data by leveraging conditional independence tests [27]. Okati et al. [32] address *missing structural knowledge*, introducing methods that rely on partial DAGs or topological ordering to localize a single root cause. While these accelerate building a system-wide causal model, partial observability and hidden confounders remain major hurdles [32, 33].

Multi-Level or Interdependent Causal Networks. Large systems often span multiple layers (e.g., containers, hosts, services). Wang et al. [34] develop *interdependent causal networks* and hierarchical GNNs to capture cross-layer fault propagation. This aligns with microservices research that calls for hierarchical RCA frameworks at various levels of abstraction [21].

2.3 Relevance to a DoWhy Extension for Pathway Tracing

Recent expansions to the *DoWhy* library [36] and specifically *DoWhy-GCM* by Blöbaum et al. [37] extend beyond classic effect estimation to include anomaly scoring, attribution, and partial support for causal discovery. Indeed, the built-in `conditional_anomaly_scores`, and `attribute_anomalies` methods demonstrate how anomalies can be flagged relative to conditional distributions in a causal graph. However, *end-to-end root cause pathway tracking*—i.e., systematically reconstructing the multi-hop causal chain from an ultimate anomaly back to its earliest trigger—remains underexplored.

Discussions with core DoWhy contributors confirm that the library's current functionality allows one to *score* each node for anomalous behavior in a DAG, but does not provide an *out-of-the-box* mechanism for path enumeration and ranking. As the literature on microservices and complex operational systems suggests [1, 21], robust RCA for real-world anomalies demands a more structured approach—one that integrates:

- **Enumerates Possible Paths**: Traverses from an anomalous node up to potential roots without leaving out plausible intermediates.
- **Prioritizes Pathways**: Scores each candidate chain according to the severity or likelihood of its contributing anomalies.
- **Supports Multi-Level DAGs**: Accommodates cases where nodes may have multiple parents spanning different architectural or organizational layers.

Such functionality would make DoWhy a comprehensive toolkit for both *model-level* causal analysis and *operational-level* root cause discovery. Hence, the next logical step is to design a DoWhy-based RCA extension that systematically returns a "root cause pathway" rather than isolating single anomalous variables in isolation.

3 Methodology

Our proposed framework for root cause pathway analysis leverages core functionalities from the DoWhy framework—such as Structural Causal Model (SCM) construction, noise attribution, and conditional anomaly scoring—and extends them with novel contributions designed to extract, trace, and rank multi-hop causal pathways. In the following, we first briefly describe the inherited components and then focus on detailing our original contributions.

3.1 Inherited Components from DoWhy

The baseline components used in our framework are as follows:

- **SCM Construction:** We build a directed acyclic graph (DAG) $G = (V, E)$, where V is the set of variables (nodes) and E is the set of causal edges. Each node $v \in V$ is assigned a causal mechanism $M(v)$ that maps parent inputs to its output. These steps—including automatic

node inference, mechanism assignment, and model fitting—are implemented using DoWhy's standard routines.

- **Noise Attribution:** For each node v, a noise contribution vector $C(v) \in \mathbb{R}^{n_v}$ is computed using DoWhy's `attribute_anomalies` function with a `MedianCDFQuantileScorer`. This vector quantifies the deviation of observed values from their expected distribution.

- **Conditional Anomaly Scoring:** For nodes with parents, conditional anomaly scores $S(v)$ are derived by comparing the observed value x_v against the prediction of the mechanism $M(v)$ given the parent values $\boldsymbol{x}_{P(v)}$. These scores are computed using DoWhy's `anomaly.conditional_anomaly_scores` function.

3.2 Our Contributions

Our work extends the DoWhy framework with the following novel components:

3.2.1 Combined Node Score

We introduce a *combined node score* that fuses the structural anomaly score and the noise contribution into a single metric for each node v. Specifically, let:

(a) The average structural anomaly score over the anomaly subset $S_A(v)$ be

$$\bar{S}(v) = \frac{1}{|S_A(v)|} \sum_{s \in S_A(v)} s. \tag{1}$$

(b) The maximum absolute noise contribution be

$$\tilde{C}(v) = \max_{i=1,\ldots,n_v} \Big\{ \big| C(v)_i \big| \Big\}. \tag{2}$$

Our combined score is then defined as a weighted sum:

$$\text{CombinedScore}(v) = \alpha\, \bar{S}(v) + (1 - \alpha)\, \tilde{C}(v), \tag{3}$$

where α is a weight parameter (set to $\alpha = 0.7$ in our experiments). Equation (3) represents our primary novel contribution, effectively integrating both model-driven and empirical anomaly information.

3.2.2 Root-Cause Path Discovery

We propose a depth-first search (DFS) algorithm to trace multi-hop causal pathways from a target node (e.g., `PROFIT_MARGIN`) back to potential root causes. Let the target node be denoted by v_0 and consider a path:

$$P = \{v_0, v_1, \ldots, v_k\},$$

where each v_{i+1} is a parent of v_i. Our algorithm extends a path based on the following threshold criteria:

(i) For each intermediate node v_i ($i = 1, \ldots, k-1$), require:

$$\text{CombinedScore}(v_i) \geq \beta\theta, \tag{4}$$

where θ is a predefined threshold (e.g., $\theta = 0.8$) and β is set to 0.7.

(ii) For the terminal (root) node v_k, require:

$$\text{CombinedScore}(v_k) \geq \theta. \tag{5}$$

Thus, the set of accepted paths is defined as:

$$\mathcal{P} = \left\{ P = \{v_0, v_1, \ldots, v_k\} \,\middle|\, \begin{array}{l} \forall\, i = 1, \ldots, k-1 :\ \text{CombinedScore}(v_i) \geq \beta\theta, \\ \text{and} \quad \text{CombinedScore}(v_k) \geq \theta \end{array} \right\}. \tag{6}$$

This DFS-based path discovery is our original extension, enabling the exploration of multi-hop causal chains based on a composite importance measure.

3.2.3 Path Significance Evaluation

To rank the candidate causal pathways, we introduce a novel metric that evaluates each path based on a combination of the root node's noise anomaly and the causal consistency along the path.

(a) Noise Component: Let the average noise contribution at the terminal node v_k be

$$\mu_{v_k} = \frac{1}{n_{v_k}} \sum_{i=1}^{n_{v_k}} C(v_k)_i. \tag{7}$$

(b) Causal Consistency Component: For each adjacent pair (v_i, v_{i+1}) along the path, compute the absolute Pearson correlation coefficient between their noise vectors:

$$\rho_{i,i+1} = \left| \text{corr}\Big(C(v_i),\, C(v_{i+1}) \Big) \right|. \tag{8}$$

Then, the overall consistency of the path is given by:

$$\text{Consistency}(P) = \frac{1}{k} \sum_{i=0}^{k-1} \rho_{i,i+1}. \tag{9}$$

(c) Overall Path Significance: We compute the significance of the path as a weighted sum of the noise and consistency components:

$$\text{Significance}(P) = \gamma\, \mu_{v_k} + (1 - \gamma)\ \text{Consistency}(P), \tag{10}$$

with γ set to 0.7. Equations (7)–(10) together constitute our novel metric for ranking multi-hop causal pathways.

3.3 Summary of the Methodology

Our complete framework can be summarized in the following steps:

1. **SCM Construction (Inherited from DoWhy):** Build the causal DAG $G = (V, E)$ and assign causal mechanisms $M(v)$ to each node.

2. **Noise Attribution and Conditional Anomaly Scoring (Inherited from DoWhy):** For each node v, compute the noise contribution $C(v)$ and the conditional anomaly scores $S(v)$.

3. **Combined Node Score (Our Contribution):** Compute the combined score using

$$\text{CombinedScore}(v) = 0.7\,\bar{S}(v) + 0.3\,\tilde{C}(v) \quad \text{(see Eq. (3))}.$$

4. **Root-Cause Path Discovery (Our Contribution):** Starting from the target node v_0, perform a DFS-based search to find paths $P = \{v_0, v_1, \ldots, v_k\}$ satisfying the thresholds in Equations (4) and (5), yielding the accepted set of paths $\mathcal{P}$ as in Equation (6).

5. **Path Significance Evaluation (Our Contribution):** For each path P, compute the overall significance as

$$\text{Significance}(P) = 0.7\,\mu_{v_k} + 0.3\ \text{Consistency}(P) \quad \text{(see Eq. (10))},$$

   where μ_{v_k} and $\text{Consistency}(P)$ are defined in Equations (7)–(9).

In summary, by integrating these novel contributions with the foundational techniques of DoWhy, our methodology offers a comprehensive, multi-level approach to root cause analysis that is well-suited for complex, real-world operational environments.

4 Controlled Anomaly Injection

To evaluate the performance of our approach, we tested it on a synthetic dataset with anomalies injected at known times and severities. The complete data generation code is publicly available.[1] By introducing disruptions whose root causes we knew *a priori*, we created a benchmark that reveals how precisely the framework can isolate actual triggers rather than correlational artifacts.

4.1 Anomaly Types and Schedule.

We defined three anomalies to emulate real-world issues:

- **ExcessiveDiscount**: Artificially raises `DISCOUNT` to a high fraction of `SALES`.
- **COGSOverstatement**: Inflates `UNIT_COST`, driving up `COST_OF_GOODS_SOLD`.
- **ReturnSurge**: Spikes `RETURN_COST`, lowering `PROFIT_MARGIN`.

Each anomaly is injected on a specific date, affecting only a subset of transactions. All dependent metrics (e.g., `NET_SALES`, `PROFIT`) are recalculated to maintain consistency.

```
# Generate base data:
df = generate_fashion_data_with_brand('2023-01-01', '2023-12-31')

# Define anomalies:
anomaly_schedule = {
    '2023-06-10': ('ExcessiveDiscount', 0.4),
    '2023-07-15': ('COGSOverstatement', 0.4),
    '2023-10-30': ('ReturnSurge', -10),
}

# Inject anomalies:
df_anomalous = inject_anomalies_by_date(df, anomaly_schedule)
```

Listing 1: Example of applying anomalies with specified date and severity.

4.2 Ground-Truth Validation.

Because each disturbance's timing and magnitude are known, we can verify whether the framework traces anomalous outcomes directly back to the original cause. This controlled setting thus provides a clear measure of multi-hop RCA accuracy.

[1] https://github.com/Economist-Ahmed-Dawoud/ProRca/blob/main/src/create_synthetic_data.py

5 Implementation, Results, and Evaluation

5.1 Implementation Pipeline

Our framework proceeds in two stages: (1) *anomaly detection* identifies dates with suspicious metrics (without yet knowing the cause), and (2) *causal root cause analysis* pinpoints the exact variables and pathways responsible. Listing 2 shows the entire workflow in only a few lines of code.

```
# 1) Detect anomalies in daily profit margin
detector = AnomalyDetector(df_agg, date_col="ORDERDATE", value_col="PROFIT_MARGIN"
    )
anomalies = detector.detect()
anomaly_dates = detector.get_anomaly_dates()

# 2) Build a Structural Causal Model from the known DAG
scm = ScmBuilder(edges=causal_edges).build(df_agg)

# 3) Perform multi-hop Root Cause Analysis for flagged dates
analyzer = CausalRootCauseAnalyzer(scm, min_score_threshold=0.3)
analysis_results = analyzer.analyze(df_agg, anomaly_dates)

# 4) Visualize discovered causal pathways
visualizer = CausalResultsVisualizer(analysis_results)
visualizer.plot_root_cause_paths()
```

Listing 2: Minimal workflow: detect anomalies, then identify root causes via a causal model.

5.2 Anomaly Detection (Table 1)

In the first step, our `AnomalyDetector` simply notes the *dates* where `PROFIT_MARGIN` deviates from normal patterns. At this stage, the system does not label *why* the anomalies happened; it only flags *when* they appear. Table 2 exemplifies these detection results.

Table 2: Initial Anomaly Detection in Daily Profit Margin (Unknown Causes)

Date	**Suspicious?**
2023-06-10	Yes
2023-07-15	Yes
2023-10-30	Yes

As shown, June 10, July 15, and October 30 are anomalous dates. However, the detector alone does not inform us *why* these anomalies occurred—only that `PROFIT_MARGIN` is unexpectedly off.

5.3 Root Cause Explanations (Table 2)

To discover *how* these anomalies arose, we apply the `CausalRootCauseAnalyzer`, which assigns both *structural* and *noise-based* scores to each node in the DAG, then uncovers multi-hop pathways that explain the observed disruptions. In our experiments, we introduced three distinct anomalies—although at this point the system only knows them as anomalies, not their specific types. Table 3 displays the final explanation paths:

Table 3: Primary Root Cause Paths for Each Injected Anomaly

Original Anomaly	**Detected Pathway**	**Score**
ExcessiveDiscount	DISCOUNT → NET_SALES → PROFIT_MARGIN	0.86
COGSOverstatement	UNIT_COST → COST_OF_GOODS_SOLD → PROFIT → PROFIT_MARGIN	0.91
ReturnSurge	RETURN_COST → PROFIT → PROFIT_MARGIN	0.83

Table 3 reveals that each flagged date correlates with a different underlying cause in the causal DAG. In reality, these three anomalies were *ExcessiveDiscount*, *COGSOverstatement*, and *ReturnSurge*—yet the system inferred precisely the correct source variable (e.g., `DISCOUNT`, `UNIT_COST`, `RETURN_COST`) purely from data-driven analysis.

5.3.1 Per-Date vs. Aggregate Analysis

Crucially, the framework can analyze all anomaly dates together or examine each date separately. In both scenarios, these same causal links emerged, underscoring the method's robustness even in multi-anomaly contexts.

6 Discussion and Future Research

Our results highlight the merits of a *causality-aware* approach to anomaly detection: rather than merely flagging suspicious time points, the method uncovers *why* deviations happen. Potential next steps include:

- **Online/Streaming RCA**: Adapting the pipeline for real-time data ingestion to deliver immediate alerts and causal insights.
- **Hidden Confounders**: Incorporating advanced strategies for partially observed or unmonitored variables that may skew naive analyses.
- **Contextual Knowledge**: Allowing domain experts to refine the DAG or override certain edge assumptions for improved interpretability.

7 Conclusion

We introduced a unified pipeline for **anomaly detection and causal explanation**, tested on a synthetic retail-like dataset where anomalies were deliberately injected. The `AnomalyDetector` swiftly identified suspicious dates, while the `CausalRootCauseAnalyzer` traced disruptions back to their true sources. Notably, *all* injected anomalies were precisely uncovered, reflecting the framework's power in revealing multi-hop causal chains. Future work will expand this system to real-time streaming contexts and domains with partial observability, strengthening its applicability to modern operational challenges.

References

[1] Mahida, A. et al. Automated Root Cause Analysis with Observability Data - A Comprehensive Review. *Journal of Engineering and Applied Sciences Technology*, 2023.

[2] Gan, Y. et al. Sage: practical and scalable ml-driven performance debugging in microservices. In *ASPlOS*, 2021.

[3] Barve, Y.D. et al. Fecbench: A holistic interference-aware approach for application performance modeling. In *IEEE International Conference on Cloud Engineering (IC2E)*, 2019.

[4] Liu, H. et al. Jcallgraph: Tracing microservices in very large scale container cloud platforms. In *International Conference on Cloud Computing*, 2019.

[5] Emmons, S. et al. A microscope on microservices. *Netflix Technology Blog*, 2022.

[6] Budig, M.J. Gender, self-employment, and earnings: The interlocking structures of family and professional status. *Gender & Society*, 20(6):725–753, 2006.

[7] Okati, N. et al. Root Cause Analysis of Outliers with Missing Structural Knowledge. In *UAI*, 2021.

[8] Wibisono, S. et al. Multivariate weather anomaly detection using dbscan clustering algorithm. *Journal of Physics: Conference Series*, 1869, 2021.

[9] Hawkins, D.M. Identification of outliers. *Springer*, 1980.

[10] Pang, G. et al. Deep learning for anomaly detection. *ACM Computing Surveys*, 54(2):1–38, 2021.

[11] Vanhoeyveld, J., Martens, D. and Peeters, B. Value-added tax fraud detection with scalable anomaly detection techniques. *Applied Soft Computing*, 86:105895, 2020.

[12] Akoglu, L. et al. Graph based anomaly detection and description: a survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.

[13] Ay, N. and Polani, D. Information flows in causal networks. *Advances in Complex Systems*, 11(1):17–41, 2008.

[14] Noble, C.C. and Cook, D.J. Graph-based anomaly detection. In *KDD*, 2003.

[15] Kriege, N. et al. A survey on graph kernels. *Applied Network Science*, 5(1):1–42, 2020.

[16] Kennedy, E. On forward and backward causal questions for SHAP. *arXiv preprint*, arXiv:2207.08157, 2022.

[17] Hooi, B. et al. Fraudar: Bounding graph fraud in the face of camouflage. In *KDD*, 2016.

[18] Li, Y. et al. AutoOD: Automated outlier detection via curiosity-guided search and self-imitation learning. *arXiv*, 2006.11321, 2020.

[19] Rayana, S. and Akoglu, L. Collective opinion spam detection: Bridging review networks and metadata. In *KDD*, 2015.

[20] Wu, L. et al. MicroRCA: Root cause localization of performance issues in microservices. In *IEEE/IFIP NOMS*, 2020.

[21] Liu, D. et al. Microhecl: high-efficient root cause localization in large-scale microservice systems. In *ICSE-SEIP*, 2021.

[22] Jaber, A. et al. Causal discovery from soft interventions with unknown targets: Characterization and learning. In *AIES*, 2021.

[23] Gao, J. et al. On community outliers and their efficient detection in information networks. In *KDD*, 2010.

[24] Beutel, A. et al. Copycatch: stopping group attacks by spotting lockstep behavior in social networks. In *WWW*, 2013.

[25] Jeyakumar, V. et al. ExplainIt!: A declarative root-cause analysis engine for time series data (extended version). *arXiv preprint*, arXiv:1903.08132, 2019.

[26] Pearl, J. Causality: Models, Reasoning, and Inference. *Cambridge University Press*, 2nd ed., 2009.

[27] Spirtes, P. et al. Causation, prediction, and search. *MIT Press*, 2nd ed., 2000.

[28] Li, M. et al. Causal inference-based root cause analysis for online service systems with intervention recognition. In *KDD*, 2022.

[29] Budhathoki, K. et al. Causal structure-based root cause analysis of outliers. In *ICML*, 2022.

[30] Janzing, D. et al. Quantifying intrinsic causal contributions via structure preserving interventions. In *AISTATS*, 2024.

[31] Budhathoki, K. et al. Why did the distribution change? In *AISTATS*, 2021.

[32] Okati, N. et al. Root Cause Analysis of Outliers with Missing Structural Knowledge. *arXiv preprint*, arXiv:2406.05014, 2024.

[33] Gong, C. et al. PORCA: Root Cause Analysis with Partially Observed Data. *arXiv preprint*, arXiv:2407.05869, 2024.

[34] Wang, D. et al. Interdependent causal networks for root cause localization. In *KDD*, 2023.

[35] Deng, A. and Zheng, B. Explainable Root Cause Analysis for Multivariate Time Series via Dynamic Graph Neural Networks. *arXiv preprint*, arXiv:2103.14260, 2021.

[36] DoWhy GitHub. DoWhy: an end-to-end library for causal inference. https://github.com/py-why/dowhy, Year: 2025.

[37] Blöbaum, P. et al. DoWhy-GCM: An Extension of DoWhy for Causal Inference in Graphical Causal Models. 2024.