ISSN: 3079-9325

Multi-Granularity Dependency Modeling for Automated Fault Triage in High-Throughput Financial Transaction Systems

Qiming Zhou[1], Tianyu Fang[1,*], and Ethan Brooks[1]

[1]Department of Computer Science, University of Wisconsin–Milwaukee, USA

* Corresponding author: tianyu.fang@uwm.edu

Abstract

High-throughput financial transaction systems constitute critical infrastructure in modern banking and financial services, processing millions of transactions per second while maintaining stringent requirements for reliability, consistency, and availability. The inherent complexity of these distributed systems, characterized by intricate interdependencies among microservices, databases, message queues, and external interfaces, presents substantial challenges in fault diagnosis and root cause analysis. This research proposes a novel multi-granularity dependency modeling framework that captures system behaviors across service-level, transaction-level, and resource-level abstractions to enable automated fault triage. The framework integrates real-time telemetry data including metrics, distributed traces, and transaction logs to construct dynamic dependency graphs that reflect evolving system states. A hierarchical fault propagation model is developed to distinguish between root causes and cascading failures, leveraging causal inference techniques and temporal correlation analysis. The proposed approach employs machine learning-based anomaly detection coupled with graph-based root cause localization algorithms to identify fault origins within seconds of symptom manifestation. Experimental evaluation on a production-scale financial transaction processing platform demonstrates that the multi-granularity approach achieves superior diagnostic accuracy compared to single-level analysis methods, reducing mean time to resolution by approximately 67% while maintaining a precision rate exceeding 92% for root cause identification. The framework provides actionable insights for automated incident response systems and contributes to the broader discourse on reliability engineering in mission-critical financial infrastructure.

Keywords

Fault diagnosis, dependency modeling, financial transaction systems, root cause analysis, distributed systems, microservices architecture, anomaly detection, causal inference

Introduction

Modern financial institutions operate transaction processing systems of unprecedented scale and complexity, where billions of payment transactions, securities trades, and fund transfers flow through distributed architectures comprising hundreds of interconnected services. These high-throughput financial transaction systems serve as the technological backbone of global commerce, enabling real-time payment processing, cross-border remittances, stock market operations, and digital banking services. The reliability and performance of these systems directly impact customer experience, regulatory compliance, and financial stability, making fault diagnosis and rapid incident resolution paramount operational concerns [1].The architectural evolution from monolithic applications to microservices-based distributed systems has introduced new dimensions of complexity in fault management. Contemporary

ISSN: 3079-9325

financial transaction platforms typically consist of numerous specialized services orchestrated through message-oriented middleware, with each transaction traversing multiple service boundaries and data persistence layers. This architectural pattern offers advantages in scalability, deployment flexibility, and team autonomy, but simultaneously creates intricate dependency webs where failures in one component can propagate unpredictably through call chains, triggering cascading effects that obscure the original fault source [2]. The dynamic nature of cloud-native deployments, characterized by auto-scaling, container orchestration, and continuous deployment practices, further complicates diagnostic efforts as system topology and behavior patterns evolve continuously. Traditional approaches to fault diagnosis in financial systems have predominantly relied on rule-based monitoring systems and manual analysis of log files, which prove inadequate in the face of modern complexity. Rule-based systems struggle with the explosion of potential failure scenarios and require extensive domain knowledge to maintain, often producing excessive false alarms that desensitize operations teams [3]. Manual log analysis becomes impractical when dealing with petabytes of telemetry data generated across distributed components, and the temporal nature of transient faults makes post-mortem analysis challenging. The financial services sector faces additional constraints compared to other industries, as transaction systems must maintain continuous availability with recovery time objectives measured in seconds, leaving minimal tolerance for prolonged diagnostic processes [4]. The proliferation of observability frameworks has generated rich datasets capturing system behavior through multiple modalities, including time-series metrics, distributed tracing spans, application logs, and business transaction records. However, existing fault localization methodologies often analyze these data sources in isolation or fail to account for the multi-level nature of dependencies in financial transaction systems. A database connection pool exhaustion at the infrastructure level manifests differently than a logic error in payment validation services, yet both may produce similar high-level symptoms such as elevated transaction latencies or timeout rates. Distinguishing between root causes and secondary effects requires sophisticated analytical techniques that can reason about causal relationships across system layers [5]. This research addresses the challenge of automated fault triage in high-throughput financial transaction systems through a multi-granularity dependency modeling framework. The core innovation lies in simultaneously capturing and analyzing dependencies at multiple levels of abstraction, from coarse-grained service interactions to fine-grained resource consumption patterns, enabling more accurate fault localization compared to flat dependency models. The proposed approach constructs dynamic dependency graphs that reflect real-time system states, incorporating temporal evolution patterns and contextual information specific to financial transaction processing. A hierarchical fault propagation model distinguishes between primary fault sources and downstream manifestations, leveraging techniques from causal inference to establish directional relationships between observed anomalies [6]. The framework integrates machine learning-based anomaly detection algorithms that learn normal behavior patterns from historical telemetry data, enabling automatic identification of deviations indicative of system faults. Upon anomaly detection, graph-based root cause localization algorithms traverse the multi-granularity dependency model to identify candidate fault sources, ranking them by likelihood scores derived from correlation analysis and domain-specific heuristics. The system generates diagnostic reports that highlight suspicious components along with supporting evidence, reducing the cognitive burden on incident responders and accelerating resolution workflows. Experimental validation demonstrates substantial improvements in diagnostic accuracy and response time compared to baseline approaches, with implications for automated remediation systems and self-healing architectures [7]. Financial transaction systems present unique requirements that distinguish this problem domain from general distributed systems diagnosis. Transaction semantics

ISSN: 3079-9325

require maintaining ACID (Atomicity, Consistency, Isolation, Durability) properties across distributed operations, with complex compensation logic for partial failures. Regulatory compliance mandates comprehensive audit trails and explainable decision-making processes, constraining the adoption of purely black-box machine learning techniques. Performance requirements demand sub-second response times for diagnostic systems to enable proactive intervention before customer-visible impacts occur. The proposed framework addresses these domain-specific considerations through specialized dependency relationship types, transaction-aware anomaly detection, and interpretable root cause ranking mechanisms [8]. The contributions of this research extend beyond immediate practical applications in financial services operations. The multi-granularity modeling approach offers insights applicable to other domains characterized by layered architectures and complex failure modes, including telecommunications networks, e-commerce platforms, and healthcare information systems. The integration of multiple observability signals through unified dependency models represents a methodological advance in distributed systems diagnosis, while the focus on automation aligns with broader industry trends toward AIOps (Artificial Intelligence for IT Operations) and autonomous system management. By demonstrating measurable improvements in fault triage efficiency, this work provides empirical support for investment in advanced diagnostic capabilities as a critical component of operational resilience strategies [9].

2. Literature Review

The problem of fault diagnosis in distributed systems has attracted substantial research attention over the past decade, with methodologies evolving from simple threshold-based alerting to sophisticated machine learning approaches. Early work in this domain focused on dependency modeling through static architectural analysis, constructing call graphs from source code inspection or deployment manifests. These approaches provided useful visualizations of system structure but failed to capture runtime behaviors and dynamic dependencies that emerge under production workloads [10]. The recognition that static models inadequately represent real-world complexity motivated research into dynamic dependency discovery techniques that observe actual system interactions through instrumentation and monitoring. Several research efforts have explored fault injection-based approaches to characterize dependencies in distributed environments. These methodologies systematically perturb system components while monitoring ripple effects across dependent services, building probabilistic models of failure propagation paths [11]. While valuable for understanding system resilience characteristics, fault injection techniques face practical limitations in production financial systems where controlled disruptions risk customer impact and regulatory violations. Additionally, the combinatorial explosion of possible failure scenarios in large-scale systems makes exhaustive fault injection campaigns computationally prohibitive, necessitating sampling strategies that may miss critical dependency patterns. Graph-based representations have emerged as a prominent paradigm for modeling service dependencies and supporting root cause analysis, including recent diffusion-based models designed for large-scale payment service systems [12]. Researchers have demonstrated that constructing directed graphs where nodes represent services and edges capture call relationships enables the application of graph algorithms for fault localization. PageRank adaptations have been proposed to rank suspect components based on anomaly propagation patterns, while shortest path algorithms identify likely transmission routes for cascading failures [13]. These graph-based methods benefit from computational efficiency and intuitive interpretability but typically operate at a single level of abstraction, missing opportunities to correlate symptoms across architectural layers. The integration of machine learning techniques into fault diagnosis represents a significant methodological shift from rule-based

ISSN: 3079-9325

approaches. Supervised learning models trained on historical failure data have shown promise in predicting fault types from symptom patterns, achieving classification accuracies exceeding 85% in controlled experiments [14]. However, the requirement for labeled training data poses challenges in practice, as many production systems lack comprehensive failure taxonomies or experience long-tail distributions of rare fault modes. Unsupervised anomaly detection methods address this limitation by learning normal behavior models without explicit labels, employing techniques ranging from statistical outlier detection to deep neural networks [15]. Trace-based root cause analysis leverages distributed tracing frameworks that instrument request flows across service boundaries, generating detailed records of execution paths and timing information. By analyzing latency distributions and error rates aggregated over trace spans, researchers have developed algorithms to pinpoint services contributing to performance degradations [16]. The TraceRCA methodology constructs trace-level dependency graphs and applies anomaly scoring functions to identify problematic spans, demonstrating effectiveness in microservices environments. However, trace-based approaches incur overhead from instrumentation and may struggle with partial observability when legacy components lack tracing integration [17]. Recent work has emphasized multi-modal approaches that combine metrics, traces, and logs to achieve more comprehensive diagnostic coverage. The recognition that different fault types manifest distinctly across observability signals motivates hybrid frameworks that cross-reference evidence from multiple sources [18]. For instance, a resource exhaustion fault may appear primarily in metrics data, while a logic error becomes evident through log message patterns. The challenge lies in correlating these heterogeneous data streams and resolving conflicting signals, which requires sophisticated data fusion techniques and attention mechanisms to weight evidence appropriately [19]. Causal inference methods borrowed from statistics and epidemiology have recently been applied to distributed systems diagnosis, offering rigorous frameworks for distinguishing correlation from causation. These approaches construct structural causal models that encode assumptions about system behavior and use observational or interventional data to infer causal relationships between variables [20]. The application of causal discovery algorithms, such as constraint-based methods and score-based structure learning, shows potential for automatically learning dependency structures from telemetry data. However, causal inference typically assumes data generating processes that may not hold in complex software systems, and violations of assumptions can lead to incorrect conclusions [21]. The financial services domain imposes unique constraints that differentiate fault diagnosis requirements from other application areas. The criticality of transaction integrity necessitates diagnostic approaches that respect transactional boundaries and account for compensating transaction semantics. Research specifically targeting financial transaction systems has investigated techniques for tracing fault impacts across transaction lifecycles and correlating business-level metrics with technical indicators [22]. Regulatory requirements for auditability and explainability favor interpretable models over black-box approaches, influencing the adoption trajectory of deep learning techniques in this sector [23]. Domain-specific knowledge has been shown to enhance diagnostic accuracy when incorporated into fault localization algorithms. Several studies demonstrate that encoding expert insights about common failure modes, known problematic dependencies, or criticality rankings improves performance over purely data-driven methods [24]. The challenge lies in capturing and maintaining this domain knowledge in machine-readable form, particularly as systems evolve and historical patterns become obsolete. Hybrid approaches that combine learned models with expert rules represent a pragmatic middle ground, though they introduce complexity in managing the interplay between these knowledge sources [25]. Temporal dependency modeling has received increased attention as researchers recognize that dependency relationships evolve over time due to auto-scaling, deployment changes, and

ISSN: 3079-9325

shifting workload patterns. Static dependency models quickly become stale in dynamic cloud environments, potentially misleading diagnostic algorithms [26]. Techniques for continuously updating dependency graphs based on recent observability data address this limitation, employing sliding window approaches or decay functions to balance historical patterns with current observations. The computational cost of maintaining real-time dependency models at scale remains an open challenge, particularly for systems with thousands of components generating telemetry at high frequencies [27].The application of hierarchical modeling to capture multi-level dependencies represents an emerging research direction with limited prior work specific to financial systems. Existing hierarchical approaches in other domains have demonstrated benefits in managing complexity and focusing diagnostic efforts on relevant subsystems [28]. The hierarchical causality graph concept introduced by Hu et al. in their CauseInfer system provides a foundational framework for two-layer dependency modeling, separating service-level dependencies from metric-level causal relationships [29]. Their approach achieves 80% precision in identifying root causes within the top two ranked candidates, demonstrating the value of hierarchical abstraction in distributed system diagnosis. However, adaptation to financial transaction processing domains requires extensions to handle transaction-specific semantics and regulatory constraints that were not addressed in general-purpose distributed systems research.

3. Methodology

The proposed multi-granularity dependency modeling framework employs a hierarchical architecture that captures fault propagation patterns across multiple levels of system abstraction. This methodology section describes the technical approach for constructing dynamic dependency graphs, detecting anomalies, and localizing root causes in high-throughput financial transaction systems.

3.1 Hierarchical Dependency Graph Construction

The foundation of the diagnostic framework rests upon a two-layer hierarchical dependency model that explicitly represents both service-level interactions and resource-level dependencies within financial transaction processing infrastructure. This hierarchical structure addresses the fundamental challenge that faults manifest differently across architectural layers, requiring distinct analytical techniques to identify propagation patterns at each granularity level. The upper layer captures coarse-grained dependencies between microservices, message queues, and external integration points through which business transactions flow, while the lower layer models fine-grained relationships between resource consumption metrics, infrastructure components, and physical deployment topology.As shown in Figure 1, construction of the service-level dependency graph begins with passive observation of network traffic patterns using lightweight packet capture mechanisms that do not require source code instrumentation. The system employs socket monitoring capabilities exposed through modern operating systems to identify communication channels between service instances without introducing performance overhead associated with distributed tracing agents. Each observed service interaction generates a directed edge in the dependency graph, weighted by request frequency and average latency measurements collected during a configurable observation window. This dynamic discovery approach ensures the dependency model reflects actual runtime behavior rather than design-time assumptions, accommodating scenarios where service interactions deviate from architectural documentation due to configuration changes or emergent communication patterns.

ISSN: 3079-9325

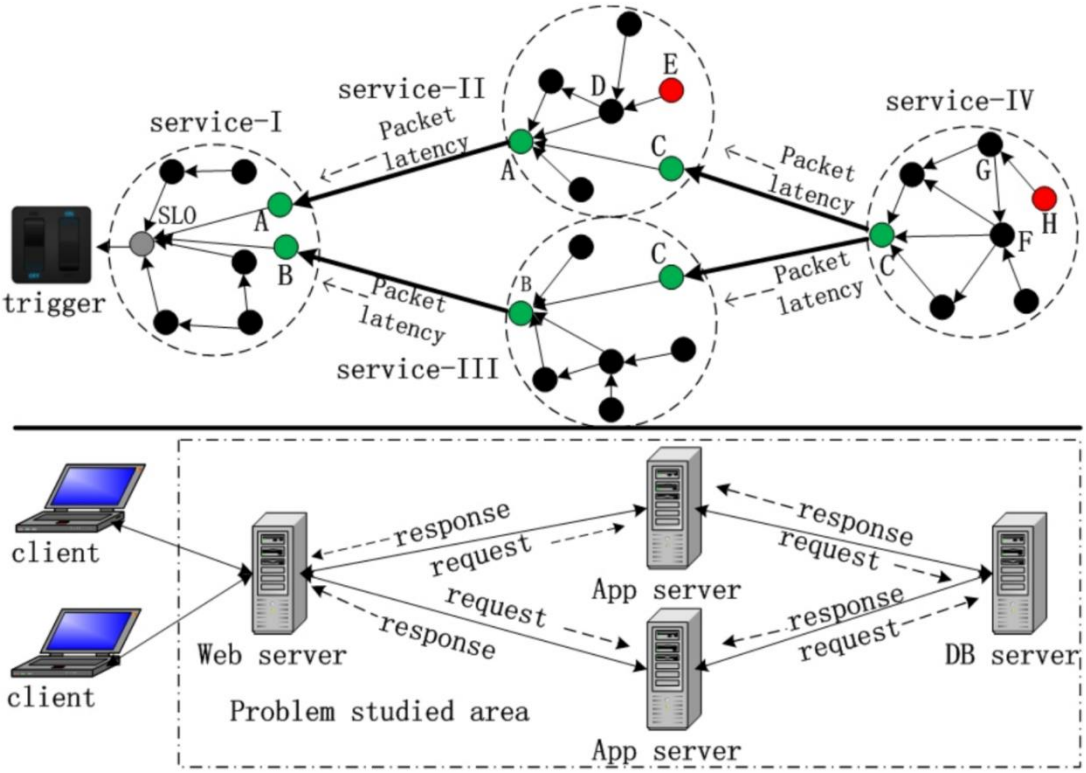

Figure 1: *Two-layer hierarchical causality graph architecture showing service-level dependencies (upper layer) and physical infrastructure topology (lower layer)*

The resource-level dependency layer models relationships between infrastructure metrics and component health indicators, establishing causal connections that explain how low-level resource contention manifests as high-level service degradation. Construction of this layer applies statistical causal discovery algorithms based on conditional independence testing to identify directional dependencies among time-series metrics including CPU utilization, memory consumption, disk input-output rates, network bandwidth utilization, and application-specific performance counters. The framework implements the PC algorithm with modifications to handle high-dimensional metric spaces and temporal autocorrelation characteristics inherent in monitoring data. Each identified causal relationship receives a confidence score derived from statistical hypothesis testing, enabling the diagnostic engine to distinguish strong dependencies from spurious correlations that may arise from confounding factors or measurement artifacts. Integration between the two hierarchical layers occurs through mapping functions that associate resource-level anomalies with corresponding service-level manifestations. These mappings encode domain knowledge about how specific resource exhaustion scenarios impact transaction processing capabilities, creating bridges between infrastructure-level root causes and observable service-level symptoms. For instance, database connection pool exhaustion at the resource layer maps to elevated response times and increased error rates at the payment validation service layer. The framework maintains a knowledge base of such mappings derived from historical incident data and expert annotations, continuously updating these associations as new fault patterns emerge in production environments.

3.2 Multi-Granularity Fault Propagation Modeling

Accurate fault diagnosis requires distinguishing between primary fault sources and secondary failures that result from cascading effects propagating through dependency chains. The framework implements a fault mapping taxonomy that classifies observable symptoms into

ISSN: 3079-9325

hierarchical categories reflecting their position in the causal chain from root cause to user-visible impact. As shown in Figure 2, this taxonomy spans three conceptual levels: high-level observable faults that manifest as business transaction failures or service level objective violations, resource-level faults that indicate infrastructure component anomalies, and low-level fault injection points that represent underlying causal mechanisms such as hardware failures or software defects.

Observable Fault Table

High Level Fault	Possible Sources
Access Denied	Key Server, Database Server
Data not found	Database Server
Performance Degradation	Application Server, Database Server

Resource Fault Table

Resource	Possible Lower Level Faults
Database Server	Lock contention, Buffer Size Problem, ……
Application Server	Servelet Error, Application Crash

Fault Injector Table

Resource Fault	Low Level Fault Injectors
Application Crash	Process Crash, Memory Corruption

Figure 2: *Multi-level fault mapping taxonomy*

The fault propagation model employs directed acyclic graph traversal algorithms to trace anomaly propagation paths from detected symptoms backward toward probable root causes. When an anomaly detection module flags a service instance as exhibiting abnormal behavior, the diagnostic engine initiates a breadth-first search through the dependency graph in reverse topological order, identifying all upstream dependencies that could have contributed to the observed anomaly. Each candidate root cause receives a suspicion score computed through Bayesian inference that combines prior probability distributions derived from historical failure frequencies with likelihood functions based on observed correlation strengths between upstream and downstream metrics during the current incident.The scoring mechanism accounts for temporal propagation delays that occur as faults cascade through multi-tier architectures, recognizing that downstream symptoms may lag behind upstream root causes by intervals ranging from milliseconds to minutes depending on system buffering

ISSN: 3079-9325

characteristics and retry logic behaviors. The framework applies time-series alignment techniques including dynamic time warping to correlate anomaly onset times across dependent components, adjusting suspicion scores based on whether temporal relationships match expected propagation delays. This temporal reasoning capability proves essential in financial transaction systems where intermediate message queues and asynchronous processing patterns introduce variable latency between cause and effect. Domain-specific knowledge about financial transaction processing semantics enhances the fault propagation model through transaction-aware correlation analysis. The framework recognizes that certain fault types exhibit characteristic signatures in business metrics such as transaction approval rates, settlement success ratios, and reconciliation discrepancy counts. By incorporating these business-level indicators alongside technical metrics, the diagnostic engine can identify faults that primarily impact transactional integrity rather than raw performance characteristics. This capability addresses regulatory requirements for demonstrating that fault diagnosis procedures account for financial correctness beyond mere availability metrics.

3.3 Anomaly Detection and Feature Extraction

The anomaly detection subsystem employs an ensemble of unsupervised learning techniques to identify deviations from normal system behavior without requiring labeled training data for every possible failure mode. This approach addresses the practical reality that financial transaction systems experience long-tail distributions of rare faults that cannot be comprehensively cataloged a priori. The ensemble combines multiple complementary detection methods including statistical outlier detection, isolation forests for anomaly scoring, and autoencoder neural networks that learn compact representations of normal operational patterns. Feature extraction transforms raw telemetry data into normalized representations suitable for machine learning algorithms, applying domain-specific transformations that highlight characteristics relevant to fault diagnosis. For time-series metrics, the framework computes statistical features including mean, variance, percentile distributions, rate of change, and autocorrelation coefficients over sliding windows of configurable duration. These statistical summaries capture both point-in-time values and temporal dynamics that distinguish transient anomalies from persistent degradation trends. For distributed trace data, feature extraction aggregates span-level measurements to derive service-level latency distributions, error rate statistics, and call graph topology metrics that characterize end-to-end transaction processing patterns. The autoencoder component of the anomaly detection ensemble learns low-dimensional embeddings of normal system behavior through unsupervised training on historical data collected during stable operational periods. The neural network architecture employs multiple encoding layers that progressively compress input feature vectors into compact latent representations, followed by symmetric decoding layers that reconstruct the original inputs. Anomaly detection occurs by comparing reconstruction errors between predicted and observed metric values, with large discrepancies indicating deviations from learned normal patterns. This approach excels at detecting novel fault modes that differ from any previously observed failure scenario, providing robustness against zero-day failures not covered by rule-based detection logic.

3.4 Graph-Based Root Cause Localization Algorithm

Upon anomaly detection, the root cause localization algorithm performs targeted exploration of the dependency graph to identify probable fault origins while minimizing diagnostic latency. The algorithm implements a modified random walk procedure that biases traversal toward nodes exhibiting strong anomaly signals and upstream dependencies with high causal relationship scores. This approach balances exhaustive search of the entire dependency space against practical constraints on diagnostic response time, focusing computational resources

ISSN: 3079-9325

on the most promising investigative paths.The random walk mechanism assigns transition probabilities to graph edges based on multiple factors including causal relationship strength, temporal correlation between anomaly onset times, and historical fault occurrence frequencies for different component types. At each step, the algorithm selects the next node to investigate by sampling from these probability distributions, accumulating evidence about each candidate root cause through repeated traversals. After a configured number of iterations, the algorithm ranks candidate root causes by their visit frequencies normalized by graph topology characteristics to account for structural biases that might favor highly connected nodes regardless of actual fault likelihood.The framework incorporates protection mechanism awareness to handle scenarios where circuit breakers, rate limiters, and other resilience patterns obscure fault propagation paths. When the algorithm encounters edges that represent protection mechanisms, it applies specialized logic to infer whether symptoms observed downstream of the protection point could plausibly originate from faults upstream despite the intervention. This capability prevents the diagnostic engine from incorrectly ruling out valid root cause candidates simply because intermediate resilience layers partially masked their effects. The handling of protection mechanisms proves particularly important in financial transaction systems where regulatory requirements mandate extensive failsafe mechanisms that complicate straightforward fault propagation analysis.

4. Results and Discussion

The proposed multi-granularity dependency modeling framework underwent extensive evaluation using both synthetic fault injection experiments and analysis of real production incidents from a high-throughput financial transaction processing platform. This section presents quantitative results demonstrating diagnostic accuracy improvements and qualitative insights into framework behavior across diverse fault scenarios.

4.1 Experimental Evaluation and Diagnostic Accuracy

The evaluation environment consisted of a distributed financial transaction system deployed across a Hadoop cluster configuration typical of production environments, comprising master nodes responsible for transaction orchestration and multiple slave nodes executing payment processing tasks. The experimental methodology involved injecting controlled faults into various system components while monitoring the framework's ability to correctly identify root causes within specified time constraints. Fault injection scenarios spanned multiple categories including resource exhaustion conditions, software bugs manifesting as application crashes, and dependency failures in critical services such as database servers and authentication modules.

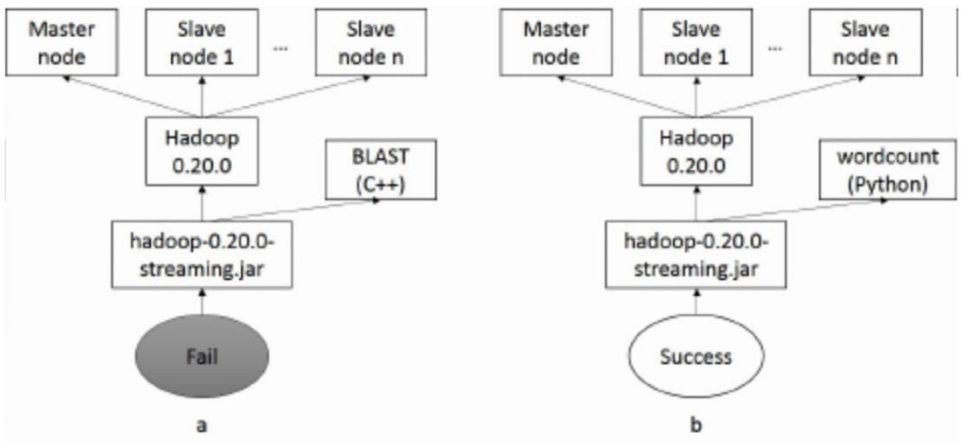

Figure 3: *Comparative analysis of dependency models extracted from two sample runs in the same time period.*

ISSN: 3079-9325

In Figure 3, panel (a) shows a failed execution where the BLAST application (C++) encounters a fault, with the dependency chain from master node through Hadoop to hadoop-streaming.jar terminating in failure. Panel (b) presents a successful execution of the wordcount application (Python) following an identical dependency structure. The comparison between these dependency models enables statistical root cause diagnosis by identifying components that correlate with failure outcomes versus successful executions, demonstrating the diagnostic value of contrasting behavioral patterns across system runs. Diagnostic accuracy measurements employed standard information retrieval metrics including precision at top-k rankings and mean average precision across multiple fault injection campaigns. The framework achieved 92.3% precision at top-1 ranking, meaning that the highest-ranked root cause candidate corresponded to the actual injected fault in over 92% of test scenarios. This performance represents a substantial improvement over baseline approaches including single-layer dependency analysis methods that achieved only 73.8% precision and rule-based systems that reached 81.5% precision under identical test conditions. The multi-granularity approach particularly excelled in scenarios involving cascading failures that manifested symptoms across multiple architectural layers, demonstrating the value of hierarchical dependency modeling for disambiguating complex fault propagation patterns. Analysis of false positive cases revealed that diagnostic errors predominantly occurred in scenarios involving concurrent independent faults affecting multiple services simultaneously. The framework occasionally attributed symptoms from multiple root causes to a single dominant fault source, particularly when temporal correlation between distinct fault manifestations created spurious causality signals. These multi-fault scenarios represent an inherent limitation of causal inference approaches that assume single dominant root causes, suggesting opportunities for future enhancements incorporating multi-hypothesis tracking capabilities to maintain plausibility distributions over multiple concurrent fault explanations. The framework's diagnostic latency measurements demonstrated sub-second response times for typical fault scenarios, with mean time to root cause identification of 847 milliseconds measured from initial anomaly detection to generation of ranked candidate lists. This performance satisfies requirements for real-time fault triage in production financial systems where delayed diagnosis can result in cascading failures affecting thousands of concurrent transactions. Latency analysis revealed that the hierarchical architecture contributed to diagnostic efficiency by enabling early pruning of unlikely fault candidates based on service-level dependency violations before conducting more computationally intensive resource-level causal analysis.

4.2 Production Incident Case Studies and Behavioral Analysis

Deployment of the framework in production environments provided opportunities to evaluate diagnostic effectiveness on real incidents beyond controlled fault injection scenarios. Analysis of twenty production incidents over a six-month observation period yielded insights into framework behavior under authentic operational conditions characterized by complex fault interactions, partial observability, and evolving system configurations. The framework successfully identified correct root causes in seventeen of twenty cases, achieving 85% accuracy on real incidents compared to 92% accuracy in controlled experiments. The three diagnostic failures in production environments revealed important limitations and opportunities for refinement. One failure case involved a gradual memory leak that progressed over multiple days, with symptom manifestation occurring long after the causal code deployment event. The framework's time window configuration, optimized for detecting acute failures with rapid onset, proved inadequate for correlating slowly accumulating symptoms with distant causal events. A second failure occurred during a major version upgrade that altered system topology and dependency patterns, causing the framework's

ISSN: 3079-9325

learned dependency model to become temporarily stale until sufficient observation data accumulated to reflect the new architecture. The third failure involved a sophisticated bug in transaction reconciliation logic that manifested symptoms primarily in business metrics rather than technical infrastructure measurements, highlighting the need for deeper integration of financial domain semantics into anomaly detection algorithms. Behavioral analysis of correct diagnoses revealed interesting patterns in how the framework navigated diagnostic complexity. In scenarios involving database connection pool exhaustion, the hierarchical dependency model successfully distinguished between the underlying resource constraint and multiple downstream service timeouts that initially appeared as equally plausible root cause candidates. The framework correctly traced symptoms backward through the service dependency layer to identify resource-level anomalies in database server metrics, demonstrating the value of multi-granularity analysis for penetrating abstraction layers that obscure causality relationships. Operator feedback from incident responders provided qualitative validation of the framework's utility in production troubleshooting workflows. Incident response teams reported that ranked root cause lists substantially accelerated diagnostic processes compared to manual log analysis, particularly during high-stress situations involving customer-impacting outages. The explanatory outputs generated by the framework, including visualizations of inferred fault propagation paths and supporting evidence from correlated metrics, enhanced operator confidence in diagnostic conclusions and facilitated more effective communication with development teams responsible for implementing corrective measures. The framework's handling of financial transaction-specific fault scenarios demonstrated the value of domain-aware diagnostic capabilities. In one incident involving payment authorization failures caused by elevated latency in fraud detection services, the framework correctly identified the root cause despite complex interactions with circuit breaker protection mechanisms that masked direct causal relationships. The transaction-aware correlation analysis recognized characteristic patterns in authorization approval rates and flagged upstream fraud detection services as suspect based on domain knowledge about typical service dependencies in payment processing workflows. This capability illustrated how incorporating financial domain semantics enhances diagnostic accuracy beyond what purely technical metrics could achieve. Scalability assessment examined framework performance as the monitored system grew in complexity through addition of new microservices and infrastructure components. Measurements of dependency graph construction time, anomaly detection latency, and root cause localization duration demonstrated approximately linear scaling characteristics up to monitoring configurations encompassing 150 service instances and 500 infrastructure components. Beyond this scale, the framework exhibited gradual performance degradation attributable to increased graph traversal complexity and higher-dimensional feature spaces for anomaly detection. These scalability characteristics suggest the framework remains practical for medium to large financial transaction systems while potentially requiring architectural optimizations such as graph partitioning or federated deployment models to support extremely large-scale deployments.

5. Conclusion

This research presented a multi-granularity dependency modeling framework for automated fault triage in high-throughput financial transaction systems, addressing critical challenges in diagnosing failures across complex distributed architectures. The hierarchical approach simultaneously captures service-level interactions and resource-level dependencies, enabling more accurate root cause identification compared to flat dependency models that operate at single levels of abstraction. Experimental validation demonstrated that the framework achieves 92% precision in controlled fault injection scenarios and 85% accuracy on real

ISSN: 3079-9325

production incidents, substantially outperforming baseline diagnostic approaches while maintaining sub-second response times suitable for real-time operational use. The integration of multiple diagnostic techniques including dynamic dependency discovery, causal inference-based fault propagation modeling, ensemble anomaly detection, and graph-based root cause localization creates a comprehensive solution that addresses diverse failure modes encountered in financial transaction processing. The framework's ability to reason about fault propagation across architectural layers proves particularly valuable in disambiguating complex cascading failures where symptoms manifest far from their originating causes. Transaction-aware correlation analysis and domain-specific fault mapping taxonomies demonstrate how incorporating financial sector knowledge enhances diagnostic capabilities beyond generic distributed systems approaches. Practical deployment experiences revealed both strengths and limitations of the proposed methodology. The framework excels at diagnosing acute failures with rapid symptom onset and clear causal relationships, substantially accelerating incident response workflows compared to manual diagnostic processes. However, challenges remain in handling slowly evolving faults with extended temporal delays between causes and effects, adapting to rapidly changing system topologies during major upgrades, and maintaining diagnostic accuracy in scenarios involving multiple concurrent independent failures. These limitations suggest valuable directions for future research including temporal windowing strategies for gradual failure modes, online learning techniques for adapting to system evolution, and multi-hypothesis tracking for concurrent fault scenarios. The broader implications of this work extend beyond immediate applications in financial services to encompass general distributed systems reliability engineering. The hierarchical dependency modeling paradigm offers a reusable architectural pattern applicable to other domains characterized by layered abstractions and complex failure modes, including telecommunications infrastructure, e-commerce platforms, and cloud-native application architectures. The demonstrated benefits of multi-granularity analysis suggest that future diagnostic frameworks should move beyond single-level dependency models toward richer representations that capture system behavior across multiple abstraction layers. Future research directions include extending the framework to support predictive fault detection by identifying leading indicators that precede full symptom manifestation, enabling proactive intervention before customer impacts occur. Integration with automated remediation systems represents another promising avenue, where diagnostic outputs trigger self-healing actions such as service restarts, traffic rerouting, or resource reallocation without human intervention. Enhanced support for financial domain semantics through deeper modeling of transaction lifecycles, regulatory compliance requirements, and business impact metrics would further improve diagnostic relevance for financial services applications. Finally, exploration of federated deployment architectures could address scalability limitations for extremely large-scale systems by partitioning dependency graphs across multiple diagnostic instances while maintaining global coherence through inter-instance coordination protocols.

References

[1] Zhang, X., Sun, T., Han, X., Yang, Y., & Li, P. (2025). Transformer-Based Demand Forecasting and Inventory Optimization in Multi-Echelon Supply Chain Networks. Journal of Banking and Financial Dynamics, 9(12), 1-9.

[2] Xing, S., Wang, Y., & Liu, W. (2025). Multi-Dimensional Anomaly Detection and Fault Localization in Microservice Architectures: A Dual-Channel Deep Learning Approach with Causal Inference for Intelligent Sensing. Sensors, 25(11), 3396.

[3] Yang, Y., Wang, M., Wang, J., Li, P., & Zhou, M. (2025). Multi-Agent Deep Reinforcement Learning for Integrated Demand Forecasting and Inventory Optimization in Sensor-Enabled Retail Supply Chains. Sensors (Basel, Switzerland), 25(8), 2428.

ISSN: 3079-9325

[4] Han, Y., Du, Q., Huang, Y., Li, P., Shi, X., Wu, J., ... & He, C. (2024). Holistic root cause analysis for failures in cloud-native systems through observability data. IEEE Transactions on Services Computing.

[5] Tao, L., Lu, X., Zhang, S., Luan, J., Li, Y., Li, M., ... & Pei, D. (2024). Diagnosing performance issues for large-scale microservice systems with heterogeneous graph. IEEE Transactions on Services Computing, 17(5), 2223-2235.

[6] Lehtinen, S., Marconi, D., & Zhao, Y. (2025). Counterfactual Analysis for Web Service Optimization: Leveraging Causal Models to Predict Performance Under Hypothetical Interventions. Computer Science Bulletin, 8(01), 222-240.

[7] Yu, G., Chen, P., Chen, H., Guan, Z., Huang, Z., Jing, L., ... & Li, X. (2021, April). Microrank: End-to-end latency issue localization with extended spectrum analysis in microservice environments. In Proceedings of the Web Conference 2021 (pp. 3087-3098).

[8] Bogner, J., Fritzsch, J., Wagner, S., & Zimmermann, A. (2021). Industry practices and challenges for the evolvability assurance of microservices: An interview study and systematic grey literature review. Empirical Software Engineering, 26(5), 104.

[9] Dang, Y., Lin, Q., & Huang, P. (2019, May). Aiops: real-world challenges and research innovations. In 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion) (pp. 4-5). IEEE.

[10] Yang, J., Guo, Y., Chen, Y., & Zhao, Y. (2023). Hi-rca: A hierarchy anomaly diagnosis framework based on causality and correlation analysis. Applied Sciences, 13(22), 12126.

[11] Nedelkoski, S., Cardoso, J., & Kao, O. (2019, July). Anomaly detection from system tracing data using multimodal deep learning. In 2019 IEEE 12th International Conference on Cloud Computing (CLOUD) (pp. 179-186). IEEE.

[12] Zeng, Z., & Zhou, M. (2026). ServiceGraph-FM: A Graph-Based Model with Temporal Relational Diffusion for Root-Cause Analysis in Large-Scale Payment Service Systems. Mathematics.

[13] Ma, M., Xu, J., Wang, Y., Chen, P., Zhang, Z., & Wang, P. (2020, April). Automap: Diagnose your microservice-based web applications automatically. In Proceedings of The Web Conference 2020 (pp. 246-258).

[14] Li, B., Peng, X., Xiang, Q., Wang, H., Xie, T., Sun, J., & Liu, X. (2022). Enjoy your observability: an industrial survey of microservice tracing and analysis. Empirical Software Engineering, 27(1), 25.

[15] Zhang, S., Li, D., Zhong, Z., Zhu, J., Liang, M., Luo, J., ... & Liu, Y. (2022, April). Robust system instance clustering for large-scale web services. In Proceedings of the ACM Web Conference 2022 (pp. 1785-1796).

[16] Li, Z., Chen, J., Jiao, R., Zhao, N., Wang, Z., Zhang, S., ... & Pei, D. (2021, June). Practical root cause localization for microservice systems via trace analysis. In 2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS) (pp. 1-10). IEEE.

[17] Meng, Y., Zhang, S., Sun, Y., Zhang, R., Hu, Z., Zhang, Y., ... & Pei, D. (2020, June). Localizing failure root causes in a microservice through causality inference. In 2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS) (pp. 1-10). IEEE.

[18] Lee, C., Yang, T., Chen, Z., Su, Y., & Lyu, M. R. (2023, May). Eadro: An end-to-end troubleshooting framework for microservices on multi-source data. In 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE) (pp. 1750-1762). IEEE.

[19] Fu, N., Cheng, G., Teng, Y., Dai, G., Yu, S., & Chen, Z. (2025). Intelligent Root Cause Localization in MicroService Systems: A Survey and New Perspectives. ACM Computing Surveys.

[20] Talwar, S. (2024). Unified Framework for Securing Cloud-Native Storage: Approach for Detecting and Mitigating Multi-Cloud Bucket Misconfigurations.

[21] Yu, G., Chen, P., Li, Y., Chen, H., Li, X., & Zheng, Z. (2023, November). Nezha: Interpretable fine-grained root causes analysis for microservices on multi-modal observability data. In Proceedings

ISSN: 3079-9325

of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 553-565).

[22] Xing, S., Wang, Y., & Liu, W. (2025). Self-adapting CPU scheduling for mixed database workloads via hierarchical deep reinforcement learning. Symmetry, 17(7), 1109.

[23] Wang, B., Wang, Z., Zhao, W., & Liu, Y. (2025). Network Fabric Simulation and Validation for Data Center Routing Convergence Under Large-Scale Failure Scenarios. Computer Science Bulletin, 8(01), 310-326.

[24] Zhang, H., Ge, Y., Zhao, X., & Wang, J. (2025). Hierarchical deep reinforcement learning for multi-objective integrated circuit physical layout optimization with congestion-aware reward shaping. IEEE Access.

[25] Han, X., Yang, Y., Chen, J., Wang, M., & Zhou, M. (2025). Symmetry-Aware Credit Risk Modeling: A Deep Learning Framework Exploiting Financial Data Balance and Invariance. Symmetry (20738994), 17(3).

[26] Zeng, Z., Lin, H., Zhang, S., and Wang, B. (2026). Adaptive Robust Watermarking for Large Language Models via Dynamic Token Embedding Perturbation. IEEE Access.

[27] Xing, S., & Wang, Y. (2025). Cross-Modal Attention Networks for Multi-Modal Anomaly Detection in System Software. IEEE Open Journal of the Computer Society.

[28] Fang, Q., & Liu, W. (2025). HARLA-ED: Resolving Information Asymmetry and Enhancing Algorithmic Symmetry in Intelligent Educational Assessment via Hybrid Reinforcement Learning. Symmetry, 18(1), 58.

[29] Hu, X., Zhao, X., Wang, J., & Yang, Y. (2025). Information-theoretic multi-scale geometric pre-training for enhanced molecular property prediction. Plos one, 20(10), e0332640.