先假设后验证：通过路径级并行实现微服务的推测式根因分析

Lingzhe Zhang 北京大学 北京，中国
zhang.lingzhe@stu.pku.edu.cn

Tong Jia*
北京大学
北京，中国
jia.tong@pku.edu.cn

Yunpeng Zhai
阿里巴巴集团
北京，中国
zhaiyunpeng.zyp@alibaba-inc.com

Leyi Pan 清华大学 北京，中国
panly24@mails.tsinghua.edu.cn

Chiming Duan
北京大学
北京，中国
duanchiming@stu.pku.edu.cn

Minghua He
北京大学
北京，中国
hemh2120@stu.pku.edu.cn

Pei Xiao 北京大学 北京，中国
xiaopei
@stu.pku.edu.cn

Ying Li*北京大学
北京，中国
li.ying
@pku.edu.cn

arXiv:2601.02736v1 [cs.SE] 6 Jan 2026

摘要

微服务系统凭借其资源弹性、松耦合架构和轻量级部署，已成为云原生企业应用的支柱。然而，此类系统固有的复杂性及其动态运行时交互不可避免地会引发异常。因此，确保系统可靠性取决于有效的根因分析（RCA），这不仅需要定位异常来源，还需要及时、可解释地描述其根本故障。智能根因分析技术的最新进展，尤其是由大语言模型（LLM）驱动的技术，展现出令人期待的能力：大语言模型减少了对手工设计特征的依赖，同时具备跨平台适应性、任务泛化能力和灵活性。然而，现有基于大语言模型的方法仍有两项关键局限：（a）探索多样性不足，削弱了准确性；（b）严重依赖大规模大语言模型，导致推理速度缓慢。为克服这些挑战，我们提出了 SpecRCA，这是一种面向微服务的推测式根因分析框架，采用先假设后验证 范式。SpecRCA 首先利用假设起草模块快速生成候选根因，然后通过并行根因验证器高效地对其进行验证。在 AIOps 2022 数据集上开展的初步实验表明，与现有方法相比，SpecRCA 在准确性和效率方面均表现出色，彰显了其作为复杂微服务环境中可扩展、可解释根因分析实用解决方案的潜力。

CCS 概念

• **软件及其工程** → **软件维护**。

关键词

根因分析、微服务、推测性验证

ACM 引用格式：

Lingzhe Zhang，Tong Jia*，Yunpeng Zhai，Leyi Pan，Chiming Duan，Minghua He，Pei Xiao 和 Ying Li*。2026。先假设后验证：基于路径级并行的微服务推测式根因分析。载于*2026*年 *IEEE/ACM* 第*48*届国际软件工程大会（*ICSE-NIER '26*），*2026*年*4*月*12*日至*18*日，巴西里约热内卢。ACM，纽约州纽约，美国，5页。https://doi.org/10.1145/3786582.3786803

1 引言

由于具备可扩展性、模块化和轻量级部署等优势，微服务系统已越来越广泛地应用于云原生企业应用中 [27]。然而，其复杂且高度互联的运行时行为使异常难以避免：单个故障可能迅速蔓延至依赖服务，并引发全系统中断 [7, 19, 36]。因此，要确保可靠性，就必须有效开展根因分析（RCA），及时且清晰易懂地识别异常源头并描述故障机制。

手动排查根因严重依赖专家知识，这不仅效率低下，而且随着系统规模扩大、业务场景日趋多样化，其局限性也愈发明显 [15, 16, 20, 23, 37, 41]。因此，实现智能化自动诊断已成为确保软件系统稳定性和可靠性的核心需求。现有诊断方法大致可分为基于规则的方法、结合特征工程的机器学习方法，以及深度学习模型。基于规则的方法依赖人工制定的规则，在高度动态的环境中难以维护 [6, 22]。机器学习方法在一定程度上提高了自动化程度，但仍高度依赖领域专业知识，特征提取和构建都需要投入大量人力且成本高昂 [1, 5, 32, 40]。深度学习方法减轻了对人工特征设计的依赖，但往往存在可解释性差、跨平台和跨任务可迁移性有限等问题 [3, 14, 25, 26, 30, 33]。总体而言，这些方法面临

*通讯作者。

本作品采用知识共享署名4.0国际许可协议授权。*ICSE-NIER '26*，巴西里约热内卢© 2026 版权所有者/作者保留所有权利。ACM ISBN 979-8-4007-2425-1/2026/04 https://doi.org/10.1145/3786582.3786803

一些共同挑战，包括缺乏跨平台通用性、对动态系统状态的适应能力不足，以及自动化程度不够，无法满足大规模系统运维的需求。

近年来，大语言模型的快速发展为故障诊断带来了新的机遇。凭借强大的知识表示和推理能力，大语言模型能够减少对手工设计特征的依赖，同时具备强大的跨平台适应性、任务泛化能力和灵活性，从而缓解传统方法的局限性 [2, 4, 8 10, 12, 13, 17, 18, – 。21, 24, 28, 31, 34, 35, 38, 39]

现有工作大致可分为三类：(i) 展示大语言模型用于根因分析潜力的早期探索 [2, 10, 11, 13, 21, 29, 35],(ii) 采用多智能体架构的系统化解决方案 [8, 9,17, 18, 31, 34, 38], 以及 (iii) 检索增强生成（RAG）等知识增强方法 [4, 11, 12, 28, 39]。尽管这些方向前景可期，但基于大语言模型的根因分析在真实微服务中的实际应用仍面临重大挑战：

• **探索多样性有限，导致准确率欠佳**。现有基于大语言模型的根因分析方法——包括多智能体投票（如 mABC [38]）和训练时探索增强（如 ThinkFL [34]）——在推理过程中仍倾向于收敛到相似的推理路径。因此，备选假设未得到充分探索，即便增加智能体数量或重复评估，准确率仍会受限。

• **过度依赖大规模大语言模型，导致推理速度缓慢**。近期方法仍依赖大型闭源或高参数量大语言模型（如 Claude、GPT-4o）来执行知识推理、长上下文处理或多轮交互（如 RCLAgent [31], COCA [4]）。即使采用中等规模模型（如 QwQ-32B），多智能体通信也会带来多轮交互延迟，使得现有方案难以满足实时微服务根因分析的速度要求。

为应对这些挑战，我们提出了**SpecRCA**，一种面向微服务的推测式根因分析框架，融合了路径级并行和先假设后验证范式。

针对第一个挑战，SpecRCA采用假设起草模型，其目标并非生成最准确的根因列表，而是枚举一组广泛且全面的潜在假设。每个候选根因都被明确视为一个假设。对于每个假设，根因验证器会独立评估其合理性，并提供支持性推理。最后，诊断综合器将所有验证结果整合为最终诊断报告。对每个假设进行强制评估，确保了充分的探索多样性，从而缓解以往方法过早收敛于浅层或同质化推理路径的倾向。

针对第二个挑战，SpecRCA 策略性地利用不同规模的模型，在效率与准确性之间取得平衡。假设起草模型采用轻量级机器学习模型，通过数据驱动分析枚举可能的根因。假设验证阶段使用经过微调的轻量级大语言模型（参数量为 < 3B），并行执行验证任务以加速推理。只有诊断综合器依赖规模更大的大语言模型，因为该阶段需要更强的推理与综合能力。通过协调不同能力的模型，SpecRCA 在降低推理成本和延迟的同时，实现了较高的探索多样性。

我们在 AIOps 2022 数据集上进行了初步实验。在准确性方面，SpecRCA 的故障定位效果比最先进的方法高出约 12.14%。在效率方面，SpecRCA 能够在 20 秒内生成完整的诊断报告，速度显著快于现有方法。这些初步结果表明，SpecRCA 不仅诊断准确性更高，还大幅降低了推理延迟，展现出其在现实场景中部署的潜力。

2 方法

图 1 展示了 SpecRCA 工作流程的整体概览。在微服务系统中检测到异常后，框架会收集异常运维数据以及此前一段时间的正常数据（指标、追踪数据和日志），以建立行为基线。随后，假设起草模块会处理这两组数据，并高效生成一系列广泛的候选根因。接着，这些假设将交由根因验证器进行细粒度验证，并得出推理结果和支持性证据。最后，诊断综合器会汇总验证结果、解决不一致之处，并输出连贯且易于理解的诊断报告。

2.1 假设起草

假设起草的目标是根据异构系统信号生成全面且经过优先级排序的候选根因集合。如图 2 所示，该模块由三条处理流水线（指标、追踪数据和日志）组成，分别生成特定于各模态的可疑度分数，随后在拓扑约束下将这些分数与服务对齐并进行融合。

指标分析。 该组件由两个主要模块组成：异常指标检测和格兰杰因果分析，二者共同基于偏离正常运行的程度和服务间依赖关系识别可疑服务。

异常指标检测 用于量化异常时段内服务级指标的偏差。对于每个服务 i，异常时段内采集的指标记为$\mathbf{X}^{(i)}$ abn $=\{x_t^{(i)}\}_{t=1}^{T_{\text{abn}}}$，正常运行期间的基线指标记为$\mathbf{X}_{\text{norm}}^{(i)} = \{x_t^{(i)}\}_{t=1}^{T_{\text{norm}}}$。服务级异常度分数通过 1-瓦瑟斯坦距离计算得出，如公式 1 所示，其中 $W_1(P,Q) = \inf_{\gamma\in\Gamma(P,Q)} \mathbb{E}_{(x,y)\sim\gamma}[|x-y|]$。

$$\text{Score}_{\text{metric}}(i) = W_1\left(\mathbf{X}_{\text{norm}}^{(i)}, \mathbf{X}_{\text{abn}}^{(i)}\right) \tag{1}$$

格兰杰因果分析 利用服务间的时间关系来细化候选根因。对于具有指标时间序列 $x_t^{(i)}$ 和 $x_t^{(j)}$ 的两个服务 i和 j，如果服务 i 满足公式 2 中的条件，则认为其格兰杰导致服务 j；其中 F_{t-1} 表示完整历史信息，$\mathrm{F}_{(-i)\,t-1}$ 不包含服务 i的历史信息。

$$\text{Var}\left(x_t^{(j)} \mid \mathcal{F}_{t-1}^{(-i)}\right) > \text{Var}\left(x_t^{(j)} \mid \mathcal{F}_{t-1}\right) \tag{2}$$

保留显著的因果链接，以调整异常度分数，从而得到因果加权指标分数，如公式 3 所示，其中 $\mathrm{C}(i)$ 是 i 的下游服务集合。

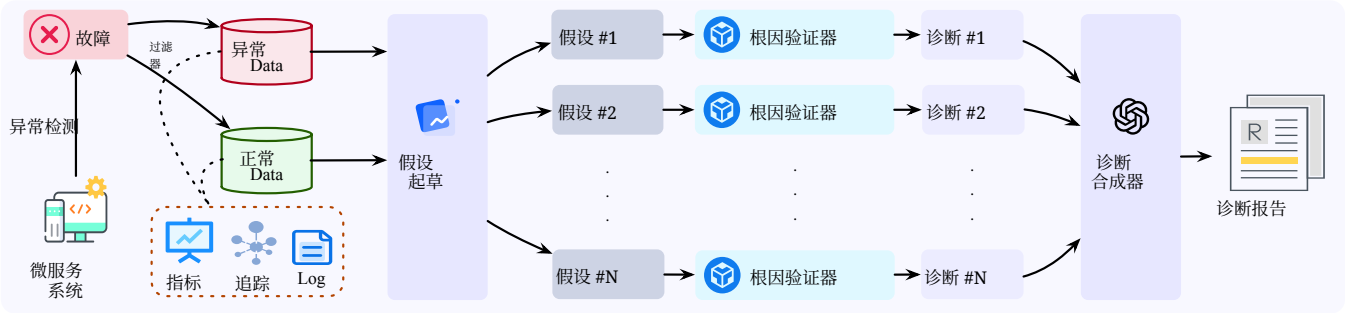

图 1：SpecRCA 概览。SpecRCA 遵循先假设后验证范式：首先使用假设起草模块快速生成潜在假设，然后使用根因验证器并行验证这些假设。

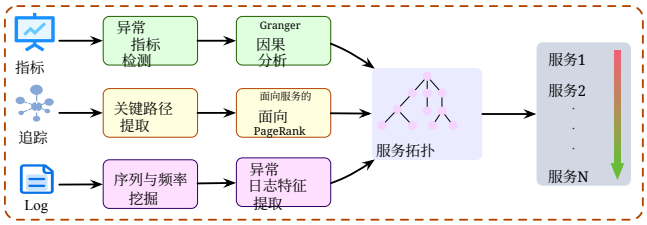

图 2：假设起草

在服务拓扑中，β 是可调节的影响系数，$\mathbf{1}_{\text{Granger}(i\to j)}$ 表示因果链接是否显著。

$$\text{Score}^{\text{causal}}_{\text{metric}}(i) = \text{Score}_{\text{metric}}(i) \cdot \left(1 + \sum_{j \in C(i)} \beta \cdot \mathbf{1}_{\text{Granger}(i\to j)}\right) \quad (3)$$

轨迹分析。 基于分布式追踪数据，我们构建调用图 $G = (V, E)$，其中 V 表示服务，E 表示观测到的调用。我们首先提取延迟贡献最大的关键路径 π^*。对于路径上的每个节点 v，将残差延迟定义为公式 4，其中 $L_\tau(v)$ 为观测到的延迟，$\bar{L}(v)$ 为基线均值。

$$r_\tau(v) = L_\tau(v) - \bar{L}(v), \quad (4)$$

为了对可疑服务进行排序，我们采用公式 5 所示的追踪加权面向服务的 *PageRank*，其中 $M(\tau_u)$ 是经过节点 u的关键路径上的累积残差延迟，$\mathrm{N}^-(v)$表示 v的入邻居，α是阻尼因子。

$$PR(v) = (1-\alpha)\frac{1}{|V|} + \alpha \sum_{u \in \mathcal{N}^-(v)} \frac{PR(u) \cdot M(\tau_u)}{\sum_{w \in \mathcal{N}^-(v)} M(\tau_w)} \quad (5)$$

日志分析。 首先将日志解析为结构化模板，以统一异构的日志格式。令 $\mathrm{L}_s = \{l_1, l_2, \ldots\ldots, l_n\}$表示服务 s的模板序列。

序列挖掘。 对于每个服务，我们使用长度为 k 的滑动窗口子序列对正常日志序列 $\mathrm{S}_{\text{norm}}(s)$ 进行建模。在异常期间，我们观察到序列 $\mathrm{S}_{\text{abn}}(s)$。子序列 q 的序列异常分数定义如公式 6 所示，其中 $P_{\text{norm}}(q)$ 和 $P_{\text{abn}}(q)$ 分别是在正常条件和异常条件下观察到 q 的概率。这可以捕捉到缺失或新出现的序列，它们可能表明存在异常行为。

$$\text{SeqAnom}(q) = \left|\log\left(\frac{P_{\text{abn}}(q)+\epsilon}{P_{\text{norm}}(q)+\epsilon}\right)\right|, \quad (6)$$

频率挖掘 为检测稀有或突发模板，我们按照公式 7 计算模板异常度分数，其中 $\text{freq}(t)$ 统计相应数据集中模板 t 的出现次数。

$$\text{Burst}(t) = \frac{\text{freq}_{\text{abn}}(t) - \text{freq}_{\text{norm}}(t)}{\sqrt{\text{freq}_{\text{norm}}(t)+\epsilon}}, \quad (7)$$

异常日志特征提取。服务级日志异常度整合了序列证据和频率证据，如公式 8 所示，其中 $\mathrm{T}(s)$ 表示与服务 s 相关联的模板集合。该分数为序列和单个模板出现情况中的异常模式提供了统一的度量。

$$\text{Anom}_{\log}(s) = \sum_{q \in \mathcal{S}_{\text{abn}}(s)} \text{SeqAnom}(q) + \sum_{t \in \mathcal{T}(s)} \text{Burst}(t), \quad (8)$$

拓扑引导集成。 将来自指标、追踪数据和日志的候选证据融合为服务级可疑度排名，并利用服务依赖拓扑 $\mathrm{T} = (S, E)$。

每个服务 $s \in S$都会获得一个初始分数，如公式 9 所示，其中 w_m，w_t，w_l为模态权重。

$$\text{Score}_0(s) = w_m \cdot \max_j \mathrm{GC}_{j\to s} + w_t \cdot PR(s) + w_l \cdot \text{Anom}_{\log}(s), \quad (9)$$

可疑度沿拓扑传播，以突出隐藏的根因，其计算方式如公式 10 所示，其中 $\tilde{A} = D^{-1}A$ 为归一化邻接矩阵，α用于控制传播强度。

$$\mathbf{Score}^{(k+1)} = \alpha \cdot \tilde{A}^\top \mathbf{Score}^{(k)} + (1-\alpha) \cdot \mathbf{Score}_0 \quad (10)$$

最后，为了捕捉共享下游依赖，通过共同子节点正则项调整分数，如公式 11 所示，其中 $\mathrm{C}(s)$ 是 s的直接下游子节点集合，λ用于平衡其影响。

$$\text{Score}(s_i) \leftarrow \text{Score}(s_i) + \lambda \sum_{j \neq i} \frac{|C(s_i) \cap C(s_j)|}{|C(s_i) \cup C(s_j)|} \cdot \text{Score}(s_j), \quad (11)$$

2.2 根因验证器

根因验证器以 **RCALite**模型为核心，该模型利用异构系统数据对候选根因假设进行验证推理。如图3所示，整体

工作流程分为三个阶段：从教师大语言模型进行蒸馏、奖励模型学习和强化微调。

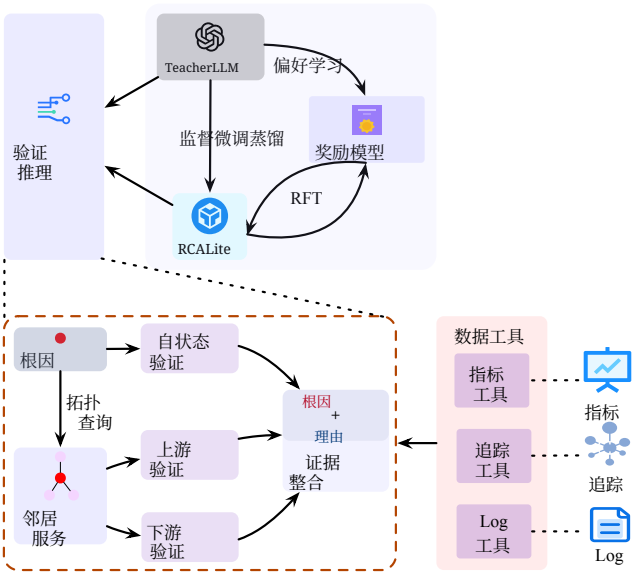

图3：根因验证器

教师大语言模型蒸馏。 调查量整个大语言模型 M_T（本文中使用 Claude-3.5-Sonnet）为每个候选假设 $h \in H$生成结构化验证推理。RCALite（记作 M_R）通过监督微调进行训练，以模仿 M_T，如公式 12 所示，其中 $\ell(\cdot,\ \cdot)$ 是词元级交叉熵损失。

$$\mathcal{L}_{\text{SFT}} = \sum_{h \in \mathcal{H}} \ell\Big(\mathcal{M}_R(h), \mathcal{M}_T(h)\Big) \tag{12}$$

奖励模型与强化微调。 为提高推理可靠性，我们利用来自 M_T 的偏好数据训练奖励模型 R_θ，并通过强化学习对 RCALite 进行微调。训练目标是最大化预期奖励，如公式 13 所示，其中π_ϕ 表示 RCALite 用于生成验证推理追踪数据的策略，并使用 GRPO 进行优化。

$$\mathcal{J}(\phi) = \mathbb{E}_{\mathcal{R}_R \sim \pi_\phi(\cdot|h)}\big[\mathcal{R}_\theta(\mathcal{R}_R)\big], \tag{13}$$

验证推理。 推理过程中，RCALite 接收候选根因假设 h 作为输入，并执行结构化验证推理以评估其有效性。验证过程包含四个主要组成部分：

(1) 自身状态验证： RCALite 会根据观测到的异常和正常系统数据，评估该假设的内部一致性，并检查该假设本身是否足以解释检测到的异常。

(2) 上游验证： 对于服务拓扑中候选服务的每个父服务 p，RCALite 会调查 p 中的异常是否能够解释观察到的系统行为，从而验证潜在的上游影响。(3) 下游验证： 对于每个子服务 c，RCALite 会分析 c 中观察到的异常是否表明，真正的根因可能位于下游，而非候选服务本身。

(4) 证据整合： RCALite 会整合自状态、上游和下游验证的结果，以生成诊断。根据收集到的证据，它会确定候选假设局部邻域内哪个组件最有可能是真正的根因，并为其结论提供详细的推理和支持性观察。

3 初步评估

为评估 SpecRCA，我们在 AIOPS 2022 数据集上考察其可行性和有效性。在这项初步评估中，我们以 Llama3.2-3B 作为基础模型，将 Claude-3.5 蒸馏为 RCALite。目前尚未进行后续的强化微调（RFT）过程。模态权重设为 $w_m = 0.3$、$w_t = 0.4$ 和 $w_l = 0.2$。作为对比，我们评估 SpecRCA 与 RCAgent 和 mABC 的表现；后两者均基于 Qwen-2.5-Plus。

表1：评估结果。准确率以 Recall@K 和 MRR 衡量；速度以每次查询所需秒数（s/q）衡量。

方法	Recall@1	Recall@3	Recall@5	MRR	速度（s/q）
RCAgent	22.10	28.40	30.25	23.95	52.79
mABC	34.19	42.13	44.51	38.46	83.17
SpecRCA	**61.34**	**75.72**	**81.63**	**62.64**	**9.89**

评估结果汇总于表 1。SpecRCA 在所有准确率指标上均显著优于 RCAgent 和 mABC，Recall@1 达到 61.34%，Recall@3 达到 75.72%，Recall@5 达到 81.63%，MRR 达到 62.64%。在效率方面，SpecRCA 也展现出显著优势，每次查询仅需 9.89 秒即可生成诊断结果，远快于基线方法。这些初步结果凸显了 SpecRCA 在准确率和推理速度方面的有效性。

4 结论

为了解决当前基于大语言模型的根因分析方法的主要局限——（a）探索多样性有限，以及（b）过度依赖大规模大语言模型——我们提出了 SpecRCA，这是一种面向微服务的推测式根因分析框架，利用路径级并行并遵循先假设后验证范式。通过在原型实现上进行的初步评估，验证了该方法的有效性。

5 未来计划

我们计划沿两个主要方向开展工作。首先，进一步完善 SpecRCA 的实现，包括对 RCALite 应用强化微调（RFT）——它目前仅经过监督微调（SFT）蒸馏——并在更多数据集上开展额外实验，以全面评估 SpecRCA 的性能。其次，诊断合成器目前是推理速度的主要瓶颈。为解决这一问题，我们计划利用基于扩散的大语言模型，通过并行解码加速生成，从而显著提升吞吐量。

致谢

本工作得到中国广东省重点研发计划（编号：2020B010164003）的支持。

参考文献

[1] Anunay Amar 和 Peter C Rigby。2019年。挖掘历史测试日志以预测缺陷并定位测试日志中的故障。载于 2019 IEEE/ACM 第41届软件工程国际会议（ICSE）。IEEE，140–151。[2] Yongqi Han、Qingfeng Du、Ying Huang、Jiaqi Wu、Fulong Tian 和 Cheng He。2024年。单次故障根因分析的潜力：大语言模型与小型分类器的协作。载于 IEEE/ACM 第39届自动化软件工程国际会议论文集。931–943。[3] Xiaoyun Li、Pengfei Chen、Linxiao Jing、Zilong He 和 Guangba Yu。2022年。Swiss-Log：面向交错非结构化日志的鲁棒异常检测与定位。IEEE《可靠与安全计算汇刊》20，4（2022），2762–2780。[4] Yichen Li、Yulun Wu、Jinyang Liu、Zhihan Jiang、Zhuangbin Chen、Guangba Yu 和 Michael R Lyu。2025年。COCA：利用代码知识进行分布式系统的生成式根因分析。载于 2025 IEEE/ACM 第47届软件工程国际会议（ICSE）。IEEE，1346–1358。[5] Qingwei Lin、Hongyu Zhang、Jian-Guang Lou、Yu Zhang 和 Xuewei Chen。2016年。面向在线服务系统的问题识别的日志聚类方法。载于第38届软件工程国际会议伴会论文集。102–111。[6] Siyang Lu、BingBing Rao、Xiang Wei、Byungchul Tak、Long Wang 和 Liqiang Wang。2017年。基于日志的 Spark 异常任务检测与根因分析。载于 2017 IEEE 国际 Web 服务会议（ICWS）。IEEE，389–396。[7] Nabor C Mendonça、Pooyan Jamshidi、David Garlan 和 Claus Pahl。2019年。开发自适应微服务系统：挑战与方向。IEEE《软件》38，2（2019），70–79。[8] Changhua Pei、Zexin Wang、Fengrui Liu、Zeyan Li、Yang Liu、Xiao He、Rong Kang、Tieying Zhang、Jianjun Chen、Jianhui Li 等。2025年。Flow-of-Action：标准操作程序增强的基于大语言模型的多智能体根因分析系统。载于 ACM 2025年 Web 会议伴会论文集。422–431。[9] Rui Ren。2025年。基于蒙特卡洛树搜索方法的多智能体故障定位系统。arXiv 预印本 arXiv:2507.22800（2025）。[10] Devjeet Roy、Xuchao Zhang、Rashi Bhave、Chetan Bansal、Pedro Las-Casas、Rodrigo Fonseca 和 Saravan Rajmohan。2024年。探索用于根因分析的基于大语言模型的智能体。arXiv 预印本 arXiv:2403.04123（2024）。[11] Komal Sarda、Zakeya Namrud、Marin Litoiu、Larisa Shwartz 和 Ian Watts。2024年。利用大语言模型实现微服务应用的自动修复：一项实验研究。载于 ACM 第32届软件工程基础国际会议伴会论文集。358–369。[12] Shiwen Shan、Yintong Huo、Yuxin Su、Yichen Li、Dan Li 和 Zibin Zheng。2024年。直面问题：一种基于大语言模型的两阶段策略，通过日志定位配置错误。arXiv 预印本 arXiv:2404.00640（2024）。[13] Honghao Shi、Longkai Cheng、Wenli Wu、Yuhang Wang、Xuan Liu、Shaokai Nie、Weixv Wang、Xuebin Min、Chunlei Men 和 Yonghua Lin。2024年。增强集群韧性：基于大语言模型智能体的自主智能集群诊断系统与评估框架。arXiv 预印本 arXiv:2411.05349（2024）。[14] Yicheng Sui、Yuzhe Zhang、Jianjun Sun、Ting Xu、Shenglin Zhang、Zhengdan Li、Yongqian Sun、Fangrui Guo、Junyu Shen、Yuzhi Zhang 等。2023年。Logkg：通过知识图谱进行日志故障诊断。IEEE《服务计算汇刊》16，5（2023），3493–3507。[15] Yongqian Sun、Zihan Lin、Binpeng Shi、Shenglin Zhang、Shiyu Ma、Pengxiang Jin、Zhenyu Zhong、Lemeng Pan、Yicheng Guo 和 Dan Pei。2025年。基于图自编码器的微服务系统可解释故障定位。ACM《软件工程与方法学汇刊》34，2（2025），1–28。[16] Dongjie Wang、Zhengzhang Chen、Jingchao Ni、Liang Tong、Zheng Wang、Yanjie Fu 和 Haifeng Chen。2023年。用于根因定位的相互依赖的因果网络。载于 ACM SIGKDD 第29届知识发现与数据挖掘会议论文集。5051–5060。[17] Qi Wang、Xiao Zhang、Mingyi Li、Yuan Yuan、Mengbai Xiao、Fuzhen Zhuang 和 Dongxiao Yu。2025年。TAMO：通过工具辅助的大语言模型智能体和云原生系统中的多模态观测数据进行细粒度根因分析。arXiv 预印本 arXiv:2504.20462（2025）。[18] Zefan Wang、Zichuan Liu、Yingying Zhang、Aoxiao Zhong、Jihong Wang、Fengbin Yin、Lunting Fan、Lingfei Wu 和 Qingsong Wen。2024年。RCAgent：利用工具增强型大语言模型的自主智能体进行云端根因分析。载于 ACM 第33届信息与知识管理国际会议论文集。4966–4974。[19] Muhammad Waseem、Peng Liang、Mojtaba Shahin、Amleto Di Salle 和 Gastón Márquez。2021年。微服务系统的设计、监控与测试：实践者视角。《系统与软件杂志》182（2021），111061。[20] Zhe Xie、Shenglin Zhang、Yitong Geng、Yao Zhang、Minghua Ma、Xiaohui Nie、Zhenhe Yao、Longlong Xu、Yongqian Sun、Wentao Li 等。2024年。通过潜在空间中的干预识别，在有限可观测性下进行微服务根因分析。载于 ACM SIGKDD 第30届知识发现与数据挖掘会议论文集。6049–6060。[21] Zhiqiang Xie、Yujia Zheng、Lizi Ottens、Kun Zhang、Christos Kozyrakis 和 Jonathan Mace。2024年。Cloud Atlas：利用语言模型和因果洞见实现云系统的高效故障定位。arXiv 预印本 arXiv:2407.08694（2024）。

[22] Jingmin Xu、Pengfei Chen、Lin Yang、Fanjing Meng 和 Ping Wang。2017年。Logdc：使用日志诊断声明式部署的云应用。在*2017*年 *IEEE* 第*14*届电子商务工程国际会议（*ICEBE*）。IEEE，282–287。[23] Qingyang Yu、Nengwen Zhao、Mingjie Li、Zeyan Li、Honglin Wang、Wenchi Zhang、Kaixin Sui 和 Dan Pei。2024年。IT服务中告警与事件智能管理综述。网络与计算机应用期刊（2024年），103842。[24] Yunpeng Zhai、Shuchang Tao、Cheng Chen、Anni Zou、Ziqian Chen、Qingxu Fu、Shinji Mai、Li Yu、Jiaji Deng、Zouying Cao 等。2025年。AgentEvolver：迈向高效的自演化智能体系统。*arXiv* 预印本 *arXiv:2511.10395*（2025年）。[25] Lingzhe Zhang、Tong Jia、Mengxi Jia、Ying Li、Yong Yang 和 Zhonghai Wu。2024年。分布式数据库的基于多变量日志的异常检测。在*ACM SIGKDD* 第*30*届知识发现与数据挖掘会议论文集。4256–4267。[26] Lingzhe Zhang、Tong Jia、Mengxi Jia、Hongyi Liu、Yong Yang、Zhonghai Wu 和 Ying Li。2024年。迈向运行时采集开销近乎为零：面向分布式存储系统系统故障的基于 Raft 的异常诊断。*IEEE*《服务计算汇刊》（2024年）。[27] Lingzhe Zhang、Tong Jia、Mengxi Jia、Yifan Wu、Aiwei Liu、Yong Yang、Zhonghai Wu、Xuming Hu、Philip Yu 和 Ying Li。2025年。大语言模型时代的 AIOps 综述。计算机调查（2025年）。[28] Lingzhe Zhang、Tong Jia、Mengxi Jia、Yifan Wu、Hongyi Liu 和 Ying Li。2025年。ScalaLog：使用大语言模型的可扩展基于日志的故障诊断。在*ICASSP 2025—2025*年 *IEEE* 声学、语音与信号处理国际会议（*ICASSP*）。IEEE，1–5。[29] Lingzhe Zhang、Tong Jia、Mengxi Jia、Yifan Wu、Hongyi Liu 和 Ying Li。2025年。XRAGLog：一种使用检索增强生成、资源高效且具备上下文感知能力的基于日志的异常检测方法。在*AAAI 2025*年预防与检测大语言模型错误信息研讨会（*PDLM*）。[30] Lingzhe Zhang、Tong Jia、Xinyu Tan、Xiangdong Huang、Mengxi Jia、Hongyi Liu、Zhonghai Wu 和 Ying Li。2025年。E-Log：面向数据库的细粒度弹性基于日志的异常检测与诊断。*IEEE*《服务计算汇刊》（2025年）。[31] Lingzhe Zhang、Tong Jia、Kangjin Wang、Weijie Hong、Chiming Duan、Minghua He 和 Ying Li。2025年。基于多智能体思维递归的微服务系统自适应根因定位。*arXiv* 预印本 *arXiv:2508.20370*（2025年）。[32] Lingzhe Zhang、Tong Jia、Kangjin Wang、Mengxi Jia、Yong Yang 和 Ying Li。2024年。通过规约事件增强基于日志的异常检测模型：一项实证研究。在第*18*届 *ACM/IEEE* 实证软件工程与度量国际研讨会论文集。538–548。[33] Lingzhe Zhang、Yunpeng Zhai、Tong Jia、Chiming Duan、Minghua He、Leyi Pan、Zhaoyang Liu、Bolin Ding 和 Ying Li。2025年。MicroRemed：微服务修复中的大语言模型基准测试。*arXiv* 预印本 *arXiv:2511.01166*（2025年）。[34] Lingzhe Zhang、Yunpeng Zhai、Tong Jia、Chiming Duan、Siyu Yu、Jinyang Gao、Bolin Ding、Zhonghai Wu 和 Ying Li。2025年。ThinkFL：通过强化微调实现微服务系统的自我精炼故障定位。*arXiv* 预印本 *arXiv:2504.18776*（2025年）。[35] Lingzhe Zhang、Yunpeng Zhai、Tong Jia、Xiaosong Huang、Chiming Duan 和 Ying Li。2025年。Agentfm：利用大语言模型驱动的多智能体对分布式数据库进行角色感知的故障管理。在第*33*届 *ACM* 软件工程基础国际会议论文集。525–529。[36] Ling-Zhe Zhang、Xiang-Dong Huang、Yan-Kai Wang、Jia-Lin Qiao、Shao-Xu Song 和 Jian-Min Wang。2024年。时间疲劳压缩：一种适用于基于 LSM树的时序数据库的弹性压缩方案。先进工程信息学59（2024年），102224。[37] Shenglin Zhang、Sibo Xia、Wenzhao Fan、Binpeng Shi、Xiao Xiong、Zhenyu Zhong、Minghua Ma、Yongqian Sun 和 Dan Pei。2024年。微服务系统中的故障诊断：综合综述与分析。*ACM*《软件工程与方法学汇刊》（2024年）。[38] Wei Zhang、Hongcheng Guo、Jian Yang、Zhoujin Tian、Yi Zhang、Yan Chaoran、Zhoujun Li、TongliangLi、Xu Shi、Liangfan Zheng 等。2024年。mABC：面向微服务架构根因分析的区块链启发式多智能体协作。在计算语言学协会论文集：*EMNLP 2024*。4017–4033。[39] Xuchao Zhang、Supriyo Ghosh、Chetan Bansal、Rujia Wang、Minghua Ma、Yu Kang 和 Saravan Rajmohan。2024年。使用 GPT-4 的上下文学习自动分析云事件根因。在第*32*届 *ACM* 软件工程基础国际会议伴随论文集。266–277。[40] Xu Zhang、Yong Xu、Si Qin、Shilin He、Bo Qiao、Ze Li、Hongyu Zhang、Xukun Li、Yingnong Dang、Qingwei Lin 等。2021年。Onion：识别云系统中指示事件的日志。在第*29*届 *ACM* 欧洲软件工程会议与软件工程基础研讨会联合会议论文集。1253–1263。[41] Zhouruixing Zhu、Cheryl Lee、Xiaoying Tang 和 Pinjia He。2024年。HeMiRCA：利用异构数据源对微服务进行细粒度根因分析。*ACM*《软件工程与方法学汇刊》33，8（2024年），1–25。