微服务架构退化的时态网络分析

Alexander Bakhtin[0000−0003−3513−7253]

University of Oulu, Oulu, Finland
alexander.bakhtin@oulu.fi

摘要 微服务架构可以被建模为一个相互调用的微服务网络，通常被称为服务依赖图。网络科学可以提供研究此类网络的方法。特别是，时态网络分析是网络科学的一个分支，用于分析随时间演变的网络。在微服务系统中，如果我们跨发布版本检查系统的架构或使用跟踪监控已部署的系统，就可能产生时态网络。
在这篇研究总结论文中，我讨论了从微服务系统中获取时间网络以及使用时间网络方法分析它们所面临的挑战。特别是，我们能够获得的最完整的时间网络包含 7 个时间实例和 42 个微服务，这限制了可以应用的潜在分析。

Keywords: 微服务架构退化网络科学时间网络 · · ·

1 介绍

微服务架构是一种促进将耦合的软件系统分解为小型、独立组件（称为微服务 [24]）的范式。文献中提出了许多模式和反模式以改进微服务系统的设计和架构 [13]。随着时间的推移，微服务系统的架构可能会退化。这种现象被称为微服务架构降解。

我们可以将微服务的构成，即它们的架构及其交互建模为网络。这些网络可以被研究为静态或随时间演变的，即所谓的时间的网络，利用来自网络科学 [3, 7] 的技术和方法。在微服务系统中，当考虑项目历史中的每个版本或提交时的微服务系统的网络 [6]，或者从正在运行的系统收集的实时监控追踪网络 [3] 时，时间网络就会出现。我们的目标是利用这种表示来检测架构退化并向架构师提供反馈。

2 相关工作

在本节中，我讨论了利用网络方法分析微服务架构及其退化的相关工作，尽管这些方法尚未广泛用于此类问题。此外，目前大多数方法都没有利用数据的时间维度。

例如，Cerny 等人。[13] 对微服务的架构反模式进行了三级研究，并提供了一个全面的目录。拓扑结构子类别代表了唯一一种可能利用微服务架构网络检测反模式的情况。目前检测此类反模式的方法要么考虑单个调用链，要么利用聚合的静态网络，其中忽略调用的顺序和频率 [3]。分析整个时序跟踪网络可以为这些反模式提供更详细和细粒度的洞察 [20]。例如，Saarimäki 等人。[31] 强调了在软件分析中纳入时间作为关键但未充分研究维度的重要性。

此外，Cerny 等人 [14] 主张采用微服务系统的演化研究来评估变更影响，这是目前在单体项目中进行的。他们认为当前的方法和工具侧重于单独研究每个微服务，这可能会忽略一个服务中的变更对系统中其他服务产生影响的情况 [30]。值得注意的是，他们确定了一些处理共演化分析和演化图的工作 [23]，并展示了对火车票基准系统的演化分析 [1]。然而，作者并没有在重构的微服务架构上执行任何类型的网络分析，而 Abufouda 和 Abukwaik [2] 则主张在经验软件工程中恰当地采用时间网络方法。在我们最近被当前版 ECSA [6] 接受的论文中，我们展示了利用时间中心性对火车票进行的时间分析，而前一版本的论文 [10] 则利用了时间社区检测来研究开发者协作网络。

3 提议的方法论

我们可以利用多种类型的时态网络方法来分析具有不同目标的微服务系统。在本节中，我考虑如何分析介绍中提出的两种微服务网络：微服务体系结构演化和微服务实时跟踪。

3.1 微服务架构演化

为了能够执行时间架构分析，我们需要一个合适的微服务系统和重建的架构 [9,15,32] 可用于该系统的多个版本 [8]。

时间中心性分析 [6,35] 可以提供一个指标，指示每个节点的时间重要性。如果某些节点变得过于重要和中心化，这可能表明它们正在成为系统 [3,13] 中的瓶颈和关键点。我们最近对火车票微服务基准测试 [6] 进行了此类分析，并发现中心性指标与大小、复杂度或质量指标不相关，因此可能是需要考虑的一种新颖指标 [7]，或者是一个正交视角。

此外，时间社区检测 [19] 可以揭示彼此紧密相连的节点群组，可能突出显示那些互相调用次数过多且高度耦合的服务。例如，Gauvin 等人 [19]

提出了一种社区检测方法，可以展示这种群体在哪一时段存在，即使这些时段不是连续的，从而揭示高度耦合群体的再现。事实上，我们之前曾在开源 (OSS) 微服务基准 [10] 的开发者合作网络中应用了他们的方法。我们发现每个版本都是由一个单一的紧密相连的人群准备的，在整个开发过程中只有一位核心开发者在场，而有另外几位在整个开发过程中的大部分时间也在场。

最后，我们还旨在研究诸如易感-感染-易感 [22] 等疾病传播模拟 [28] 是否可以应用于微服务架构版本的时间网络 [21] 中，给定一些质量属性，例如存在错误、代码异味或违规 [17]，以观察它们一旦开始影响一个服务 [16,30] 时是否会像疾病一样传播。

3.2 微服务实时跟踪

微服务追踪数据是监控微服务体系的重要方面 [11]。追踪数据允许实践者/研究者看到服务调用链，并分析不同的服务及其延迟或稳定性如何影响其他服务和整个微服务体系的健康状况 [3]。可以通过开发用于在线处理的时间网络方法来分析追踪数据，即实时消耗并分析传入的数据。

特别是，时间中心性有可能实现实时计算和更新。与 Taylor 等人 [35] 的算法不同，该算法要求整个时间网络一次性可用，Beres 等人 [12] 提供了一种算法，可以根据连接流计算时间网络中节点的中心性分数。每当注册新的连接时，这些分数就会得到更新。因此，我们旨在通过将微服务中心性与延迟或其他指标 [7] 相关联来研究实时中心性分数是否可以作为微服务监控的新指标。中心性分数可以突出显示接收到过多请求的问题服务，从而需要进行扩容或可能重构为多个服务。

类似于时间社区检测，可以检测不同的系统状态以识别异常。Masuda 和 Holm [27] 提供了一种方法来聚类时间网络的不同时间段，以识别网络似乎处于某种状态的时间段以及这种状态何时发生变化。一个可能的研究方向是观察是否可以根据诱导异常的实际情况，将微服务系统的检测到的状态与系统中的异常联系起来，或将该状态与其他在系统中观测到的指标相关联。利用这种方法可以提供一种轻量级、纯粹基于跟踪的方法来进行微服务异常检测。

最后，我们可以专注于检测在某一时间段内保持紧密连接的节点组，这被称为团。因此，在时态网络中挖掘团的过程类似于社区检测。林等人。[25] 提供了一种用于检测此类结构的算法，我们可以在微服务追踪网络中利用这种算法。在一个微服务监控网络中，这样的团表明一组服务之间持续进行大

量通信，从而可能显示出架构反模式的存在，如不适当的亲密服务、错误切割或贪婪微服务系统 [13]。

4 遇到的挑战

在我当前的博士研究工作中，我在追求本文概述的两个方向时遇到了挑战。这些挑战与工业和 OSS 微服务系统及其数据的可用性有关。具体细节如下。

4.1 微服务架构演化

作为我们小组研究工作的一部分，我调查了适用于分析 [15] 的 OSS 微服务系统以及用于微服务架构重构的工具 [9]，并利用静态分析 [32,33] 对这些工具进行了比较和评估。然而，试图将最有前景的原型工具*代码 2dfd* [34] 与识别数据集中的所有适用项目（即 Java Spring 项目）结合在一起，只得到了 24 个项目具有可靠重构的架构网络 [7]。这一结果连同工具评估 [32] 的结果，提出了以下挑战：

挑战 1 *缺乏提供架构图或可用可靠工具详细重构的微服务系统。*

在确定了至少最新版本 [7] 可以获取可靠架构的 24 个项目之后，我还试图重构这些项目的先前版本以获得时间上的架构网络。然而，由于大多数项目都是小型示例基准，它们没有显示架构的变化 [8]。唯一合适的项目，即拥有足够组件、连接和明确架构演化的项目，结果是*火车票* [38]，这本身就是微服务研究的实际标准对象 [3–5,36,37]。因此，这是我们最新论文中唯一可以分析的项目 [6]，不可避免地影响了研究的有效性和普遍性。此外，即使是*火车票*的时间网络也只包含 7 个时间实例和 42 个微服务。这提出了以下挑战：

挑战 2 *缺乏已知架构随时间变化的微服务系统。*

访问一个大型工业系统当然会提供一个显著进化的系统。然而，不能保证其架构在所有版本中都被精确记录，或者能够在其历史中被重建。

4.2 微服务实时跟踪

通过系统动态分析收集跟踪数据是一项耗时且费力的任务，需要生成并在部署有足够资源的运行系统上执行一些负载场景 [3]。因此，具有必要规

模并包含现实场景和分布的数据仅掌握在拥有大规模微服务系统的工业公司手中，而这些公司并未公开发布此类数据。

几家“科技巨头”，如 eBay [20]、阿里巴巴 [26] 和 IBM [29]，发布了关于其专有微服务系统分析和监控的研究；然而，由于行业惯例，他们并未分享这些分析数据。这种情况本身就带来了一定的挑战：

挑战 3 缺乏访问工业规模的微服务追踪数据集。

相反，大多数利用 OSS 基准项目的 [3–5, 36, 37] 研究似乎依赖于火车票 [38] 或死亡星基准测试 [18] 项目，这些项目似乎是就微服务数量而言最大的两个可用项目。其他包括范围显著较小的袜子店 [1]，以及需要部署到谷歌云平台的在线精品店 [2]。如此集中关注由研究人员创建的一小部分基准测试，对依赖它们的作品的有效性和普遍性构成了威胁，从而带来了一个挑战：

挑战 4 缺乏可以有效利用于动态跟踪分析的多样化 *OSS* 微服务基准。

作为基于多模态融合的微服务异常检测（MuFAno）项目的一部分，我们需要一个由日志、指标和跟踪数据组成的微服务监控数据集进行联合研究。鉴于挑战 3，我们尝试利用微服务项目的 [15] 数据集解决挑战 4，以识别一个非基准的 OSS 微服务系统，执行动态分析和重构，从而获取日志、指标和跟踪数据，提供一种新的微服务监控数据来源。我们对 *light-oauth2*[3] 项目的七个微服务进行了动态分析和测试，由 *networknt* [11] 进行。然而，该项目的作者没有依赖现有的 OpenTelemetry 实现，这要求我们必须注入 Jaeger 代理来收集跟踪数据，结果几乎没有任何输出。

我们尝试寻找并描述与微服务监控数据集相关的研究工作也突显了这样一个事实，即很难找到包含所有三种类型监控数据的可靠数据集，即日志、跟踪和指标 [11]。这又带来了另一个挑战：

挑战 5 缺乏包含跟踪、日志和指标的微服务监控数据的实际数据集。

总而言之，大多数遇到的挑战都与缺乏可靠和具有代表性的微服务系统及相关数据集有关。

---

[1] https://github.com/microservices-demo/microservices-demo

[2] https://github.com/GoogleCloudPlatform/microservices-demo

[3] https://github.com/networknt/light-oauth2

5 结论

在这篇研究总结论文中，我提出了作为我的研究基础的想法，该研究旨在应用时间网络方法来解决微服务架构退化问题。此外，我还概述了其实施的当前状态，并描述了几项遇到的挑战，其中大多数是由于大型工业玩家封闭性质以及开源微服务基准资源有限所导致。

参考文献

1. Abdelfattah, A.S., Cerny, T., Yero Salazar, J., Li, X., Taibi, D., Song, E.: Assessing evolution of microservices using static analysis. Applied Sciences **14**(22), 10725 (2024)
2. Abufouda, M., Abukwaik, H.: On using network science in mining developers collaboration in software engineering: A systematic literature review. International Journal of Data Mining & Knowledge Management Process (IJDKP) **7**(5/6), 17–34 (November 2017)
3. Al Maruf, A., Bakhtin, A., Cerny, T., Taibi, D.: Using microservice telemetry data for system dynamic analysis. In: 2022 IEEE International Conference on Service-Oriented System Engineering (SOSE). pp. 29–38. IEEE (2022)
4. Avritzer, A., Janes, A., Trubiani, C., Rodrigues, H., Cai, Y., Menasché, D.S., de Oliveira, Á.J.A.: Architecture and performance anti-patterns correlation in microservice architectures. In: 2025 IEEE 22nd International Conference on Software Architecture (ICSA). pp. 60–71. IEEE (2025)
5. Bakhtin, A.: Microservice api pattern detection : Using business processes and call graphs (2022)
6. Bakhtin, A., Esposito, M., Lenarduzzi, V., Taibi, D.: Centrality change proneness: an early indicator of microservice architectural degradation. In: 19th European Conference on Software Architecture (ECSA) (2025)
7. Bakhtin, A., Esposito, M., Lenarduzzi, V., Taibi, D.: Network centrality as a new perspective on microservice architecture. In: 2025 IEEE 22nd International Conference on Software Architecture (ICSA). pp. 72–83 (2025)
8. Bakhtin, A., Esposito, M., Taibi, D.: Challenges in constructing temporal architecture networks for microservice systems. Arctic AI 2024 (2024)
9. Bakhtin, A., Li, X., Soldani, J., Brogi, A., Cerny, T., Taibi, D.: Tools reconstructing microservice architecture: A systematic mapping study. In: European Conference on Software Architecture. pp. 3–18. Springer (2023)

10. Bakhtin, A., Li, X., Taibi, D.: Temporal community detection in developer collaboration networks of microservice projects. In: European Conference on Software Architecture. pp. 174–182. Springer (2024)

11. Bakhtin, A., Nyyssölä, J., Wang, Y., Ahmad, N., Ping, K., Esposito, M., Mäntylä, M., Taibi, D.: Lo2: Microservice api anomaly dataset of logs and metrics. In: 21st International Conference on Predictive Models and Data Analytics in Software Engineering (PROMISE) (2025)

12. Béres, F., Pálovics, R., Oláh, A., Benczúr, A.A.: Temporal walk based centrality metric for graph streams. Applied network science **3**, 1–26 (2018)

13. Cerny, T., Abdelfattah, A.S., Al Maruf, A., Janes, A., Taibi, D.: Catalog and detection techniques of microservice anti-patterns and bad smells: A tertiary study. Journal of Systems and Software **206**, 111829 (2023)

14. Cerny, T., Goulis, G., Abdelfattah, A.S.: Towards change impact analysis in microservices-based system evolution (Jan 2025). https://doi.org/10.48550/arXiv.2501.11778

15. d'Aragona, D.A., Bakhtin, A., Li, X., Su, R., Adams, L., Aponte, E., Boyle, F., Boyle, P., Koerner, R., Lee, J., et al.: A dataset of microservices-based open-source projects. In: Proceedings of the 21st International Conference on Mining Software Repositories. pp. 504–509 (2024)

16. Esposito, M., Falessi, D.: Uncovering the hidden risks: The importance of predicting bugginess in untouched methods. In: 2023 IEEE 23rd International Working Conference on Source Code Analysis and Manipulation (SCAM). pp. 277–282. IEEE (2023)

17. Esposito, M., Falessi, D.: Validate: A deep dive into vulnerability prediction datasets. Information and Software Technology p. 107448 (2024)

18. Gan, Y., Zhang, Y., Cheng, D., Shetty, A., Rathi, P., Katarki, N., Bruno, A., Hu, J., Ritchken, B., Jackson, B., et al.: An open-source benchmark suite for microservices and their hardware-software implications for cloud & edge systems. In: Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems. pp. 3–18 (2019)

19. Gauvin, L., Panisson, A., Cattuto, C.: Detecting the community structure and activity patterns of temporal networks: a non-negative tensor factorization approach. PloS one **9**(1), e86028 (2014)

20. Guo, X., Peng, X., Wang, H., Li, W., Jiang, H., Ding, D., Xie, T., Su, L.: Graph-based trace analysis for microservice architecture understanding and problem diagnosis. In: Proceedings of the 28th ACM Joint Meeting on European Software

Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 1387–1397 (2020)

21. Holme, P.: Temporal network structures controlling disease spreading. Physical Review E **94**(2), 022305 (2016)
22. Lee, H.K., Shim, P.S., Noh, J.D.: Epidemic threshold of the susceptible-infected-susceptible model on complex networks. Physical Review E—Statistical, Nonlinear, and Soft Matter Physics **87**(6), 062812 (2013)
23. Lelovic, L., Huzinga, A., Goulis, G., Kaur, A., Boone, R., Muzrapov, U., Abdelfattah, A.S., Cerny, T.: Change impact analysis in microservice systems: A systematic literature review. Journal of Systems and Software p. 112241 (2024)
24. Lewis, J., Fowler, M.: Microservices: a definition of this new architectural term (2014), https://martinfowler.com/articles/microservices.html
25. Lin, L., Yuan, P., Li, R.H., Wang, J., Liu, L., Jin, H.: Mining stable quasi-cliques on temporal networks. IEEE Transactions on Systems, Man, and Cybernetics: Systems **52**(6), 3731–3745 (2021)
26. Luo, S., Xu, H., Lu, C., Ye, K., Xu, G., Zhang, L., Ding, Y., He, J., Xu, C.: Characterizing microservice dependency and performance: Alibaba trace analysis. In: Proceedings of the ACM symposium on cloud computing. pp. 412–426 (2021)
27. Masuda, N., Holme, P.: Detecting sequences of system states in temporal networks. Scientific reports **9**(1), 795 (2019)
28. Newman, M.E.: Spread of epidemic disease on networks. Physical review E **66**(1), 016128 (2002)
29. Pourmajidi, W., Miranskyy, A., Steinbacher, J., Erwin, T., Godwin, D.: Dogfooding: Using ibm cloud services to monitor ibm cloud infrastructure. In: Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering. pp. 344–353 (2019)
30. Robredo Manero, M., Esposito, M., Palomba, F., Peñaloza, R., Lenarduzzi, V.: Analyzing the ripple effects of refactoring. a registered report. This Registered Report has been accepted at ICSME (2024)
31. Saarimäki, N., Moreschini, S., Lomio, F., Penaloza, R., Lenarduzzi, V.: Towards a robust approach to analyze time-dependent data in software engineering. In: 2022 SANER. pp. 36–40. IEEE (2022)
32. Schneider, S., Bakhtin, A., Li, X., Soldani, J., Brogi, A., Cerny, T., Scandariato, R., Taibi, D.: Comparison of static analysis architecture recovery tools for microservice applications: Preprint. arXiv preprint arXiv:2412.08352 (2024)

33. Schneider, S., Bakhtin, A., Li, X., Soldani, J., Brogi, A., Cerny, T., Scandariato, R., Taibi, D.: Comparison of static analysis architecture recovery tools for microservice applications: Registered report. arXiv preprint arXiv:2403.06941 (2024)

34. Schneider, S., Scandariato, R.: Automatic extraction of security-rich dataflow diagrams for microservice applications written in java. Journal of Systems and Software **202**, 111722 (2023)

35. Taylor, D., Myers, S.A., Clauset, A., Porter, M.A., Mucha, P.J.: Eigenvector-based centrality measures for temporal networks. Multiscale Modeling & Simulation **15**(1), 537–574 (2017)

36. Zhang, C., Peng, X., Zhou, T., Sha, C., Yan, Z., Chen, Y., Yang, H.: Tracecrl: contrastive representation learning for microservice trace analysis. In: Proceedings of the 30th ACM joint European software engineering conference and symposium on the foundations of software engineering. pp. 1221–1232 (2022)

37. Zhao, Y., De Matteis, T., Bogner, J.: How does microservice granularity impact energy consumption and performance? a controlled experiment. In: 2025 IEEE 22nd International Conference on Software Architecture (ICSA) (2025)

38. Zhou, X., Peng, X., Xie, T., Sun, J., Xu, C., Ji, C., Zhao, W.: Benchmarking microservice systems for software engineering research. In: Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings. pp. 323–324 (2018)