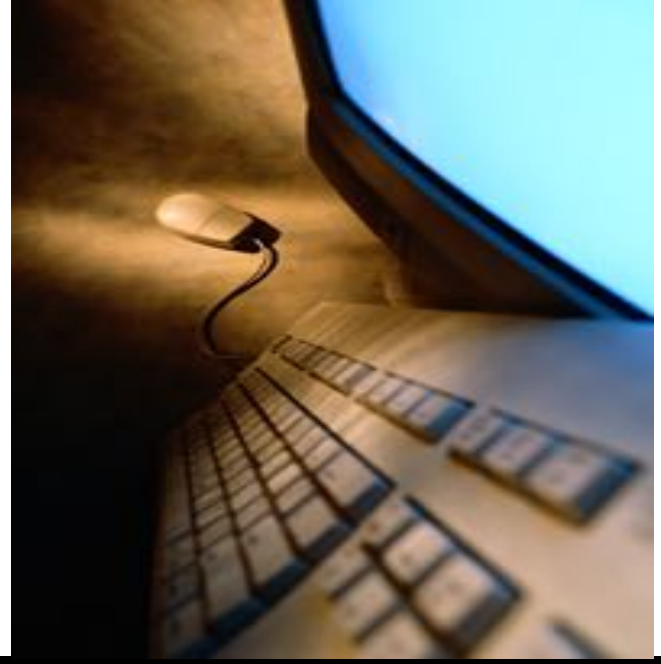
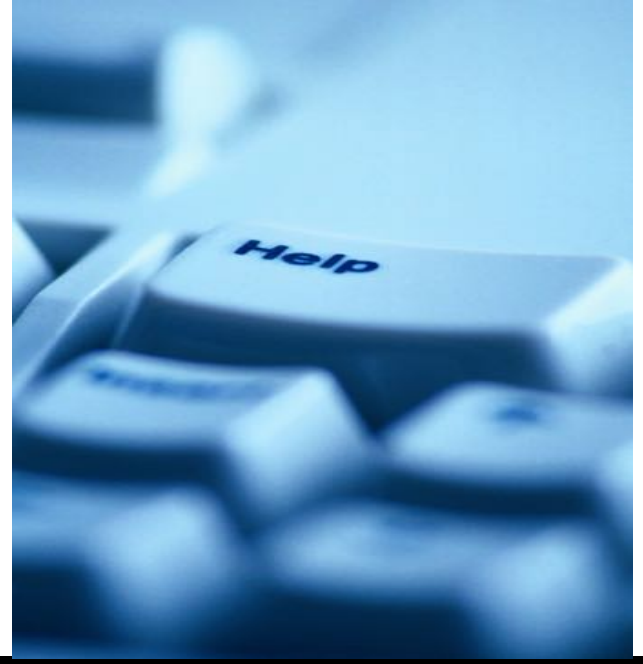
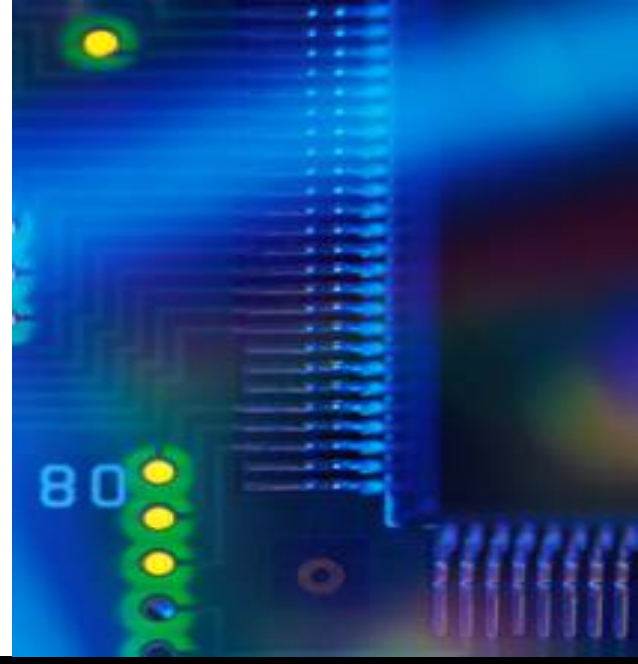





面向云原生系统的性能异常检测与根因定位

Anomaly detection and Root Cause Analysis in Cloud Native Systems

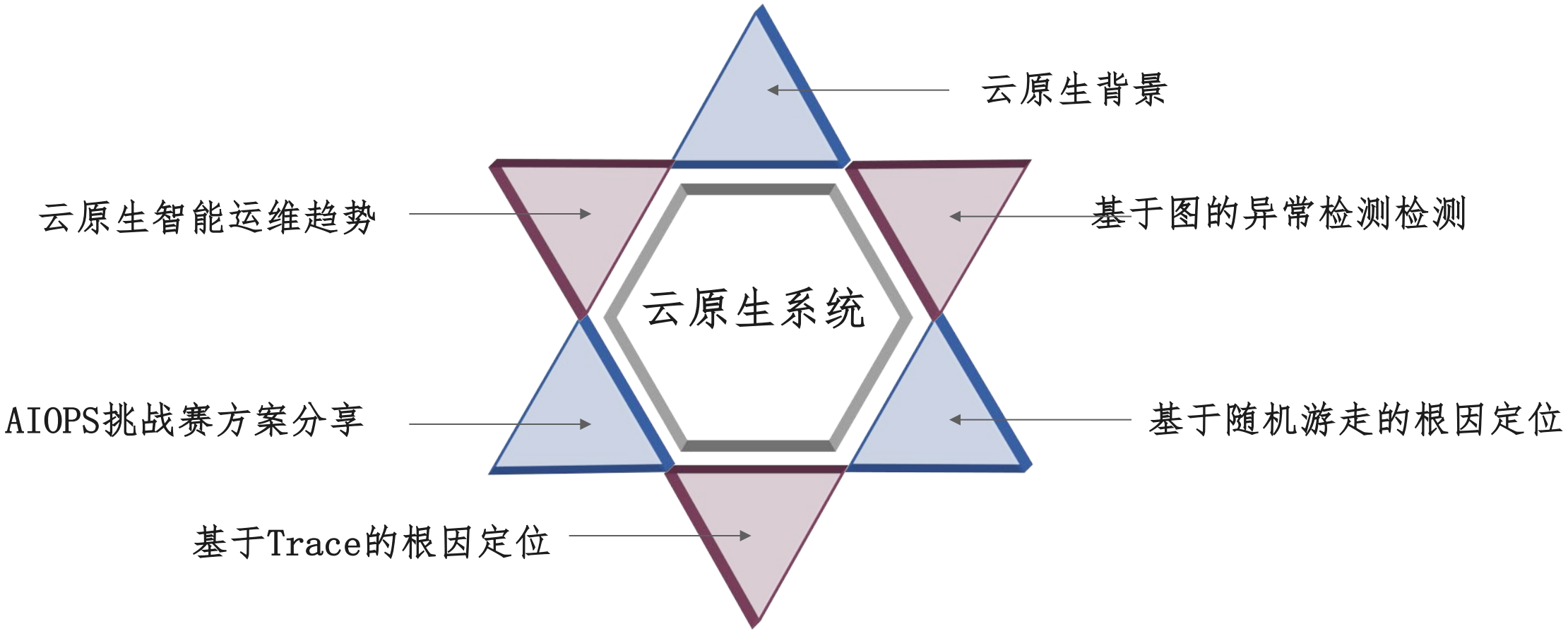
陈鹏飞
数据科学与计算机学院
中山大学
chenpf7@mail.sysu.edu.cn

● 引言

来源：https://www.moogsoft.com/blog/aiops/transforming-devops-itops-mttr

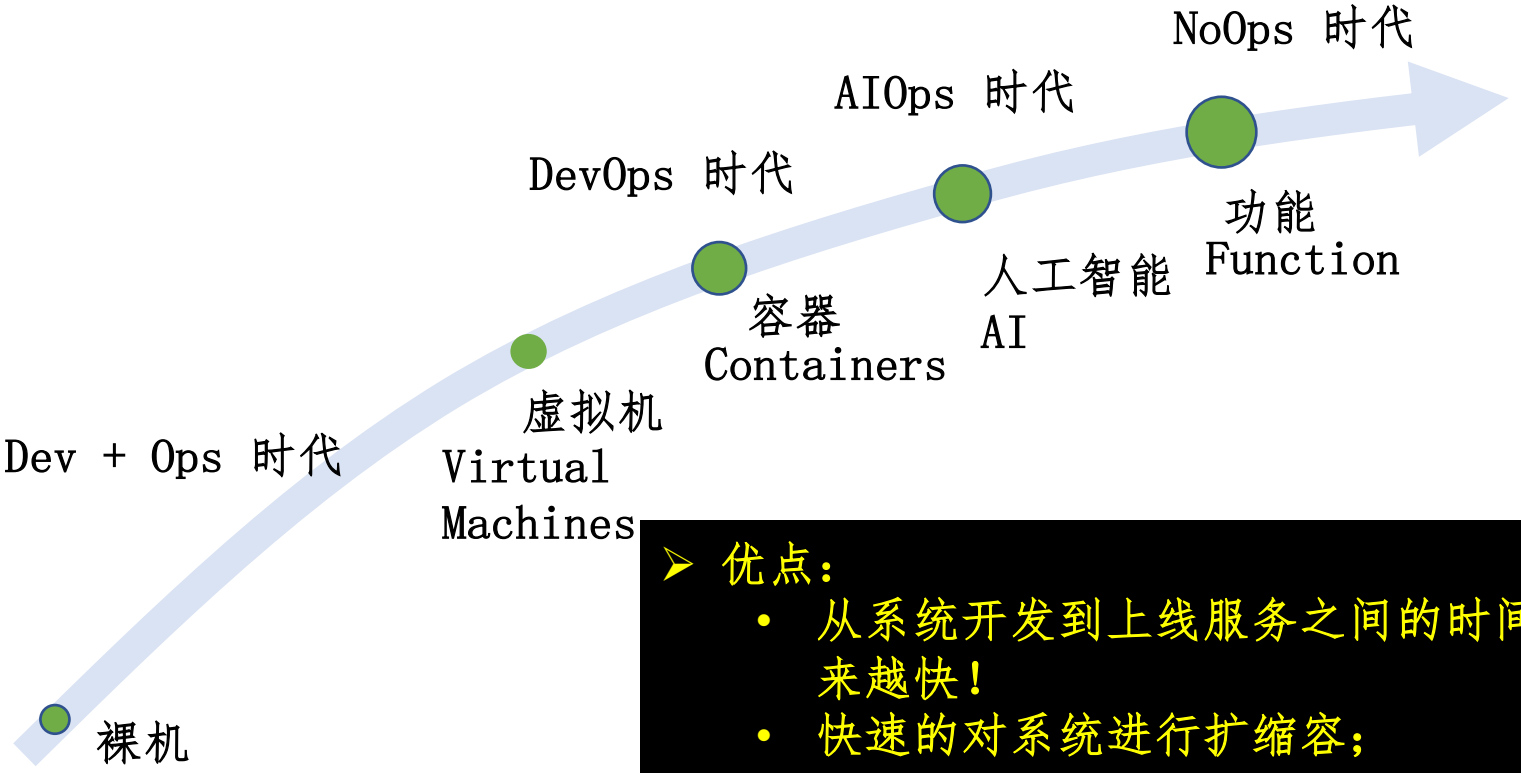
面向云原生系统的智能运维正在经历破茧为蝶的艰苦过程

内容提纲


云原生背景
云原生智能运维趋势
基于图的异常检测检测
云原生系统
AIOPS挑战赛方案分享
基于随机游走的根因定位
基于Trace的根因定位

● 云时代业务开发模式发展

专注业务逻辑
编程的灵活性



➢ 优点：
- 从系统开发到上线服务之间的时间越来越快！
- 快速的对系统进行扩缩容；

运营的敏捷性
在任何规模的快速响应能力

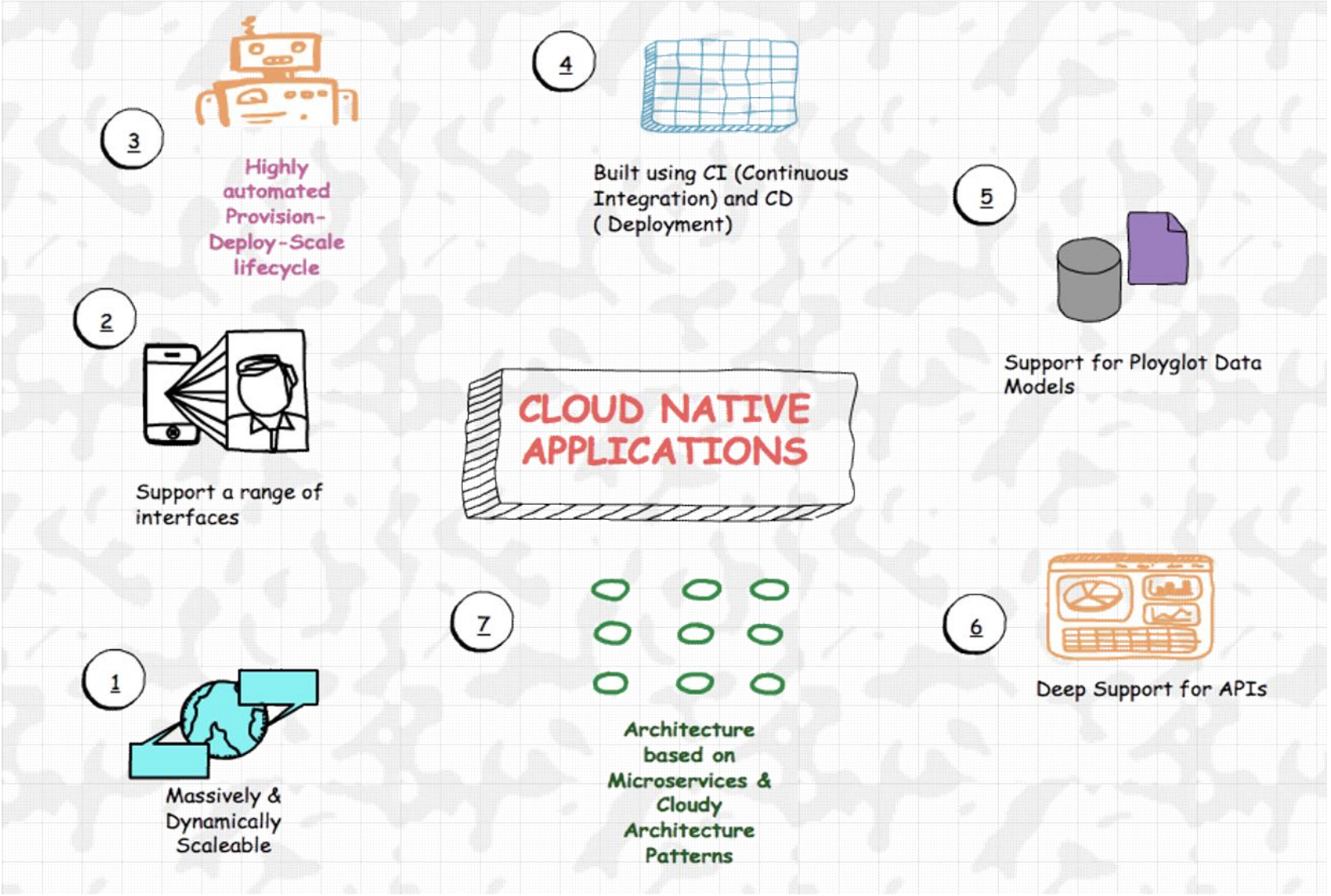
➢ 缺点：
- 开发的难度在降低，运维的难度在升高，运维成本高；压力和成本下沉

● 云原生系统

云原生技术帮助公司和机构在公有云、私有云和混合云等新型动态环境中，构建和运行科弹性扩展的应用。云原生的代表技术包括容器、服务网格、微服务、不可变基础设施和声明API。

● 云原生应用特点

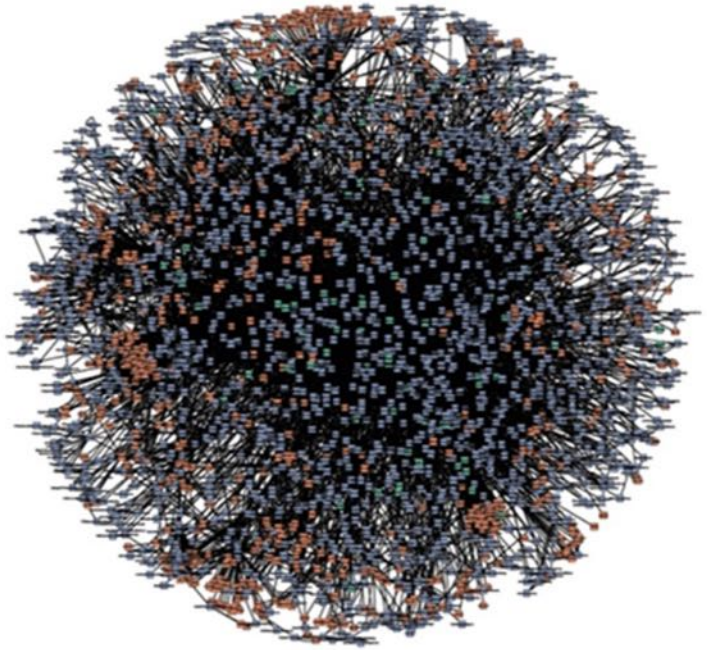
- ➢大规模动态可扩展
- ➢支持一系列的接口
- ➢高度自动化的分配、部署生命周期
- ➢ 基于CI/CD技术的连续集成和部署
- ➢ 支持多态数据模型
- ➢ 深度支持API
- ➢基于微服务的软件体系结构



● 云原生系统规模

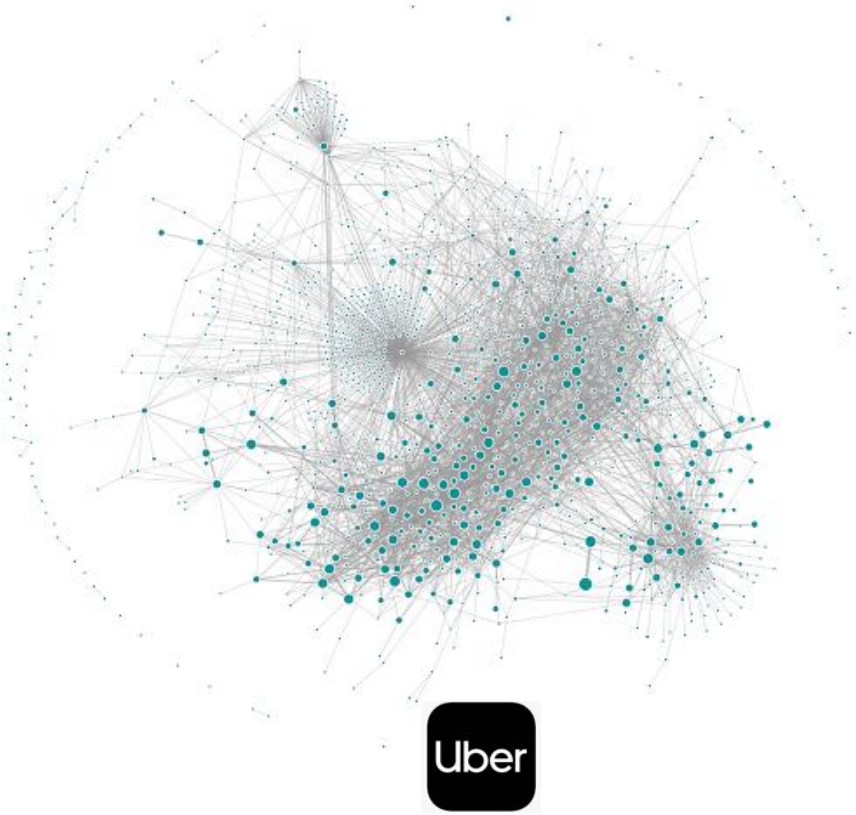
- 现代云原生软件系统的规模呈现指数级增加，动辄上千个微服务，例如：WeChat包含近3000个微服务，Netflix超过700种微服务，Uber包含的微服务多达2200个，……；

- 服务之间呈现复杂的调用关系，调用链跨越几十种微服务；



amazon.com

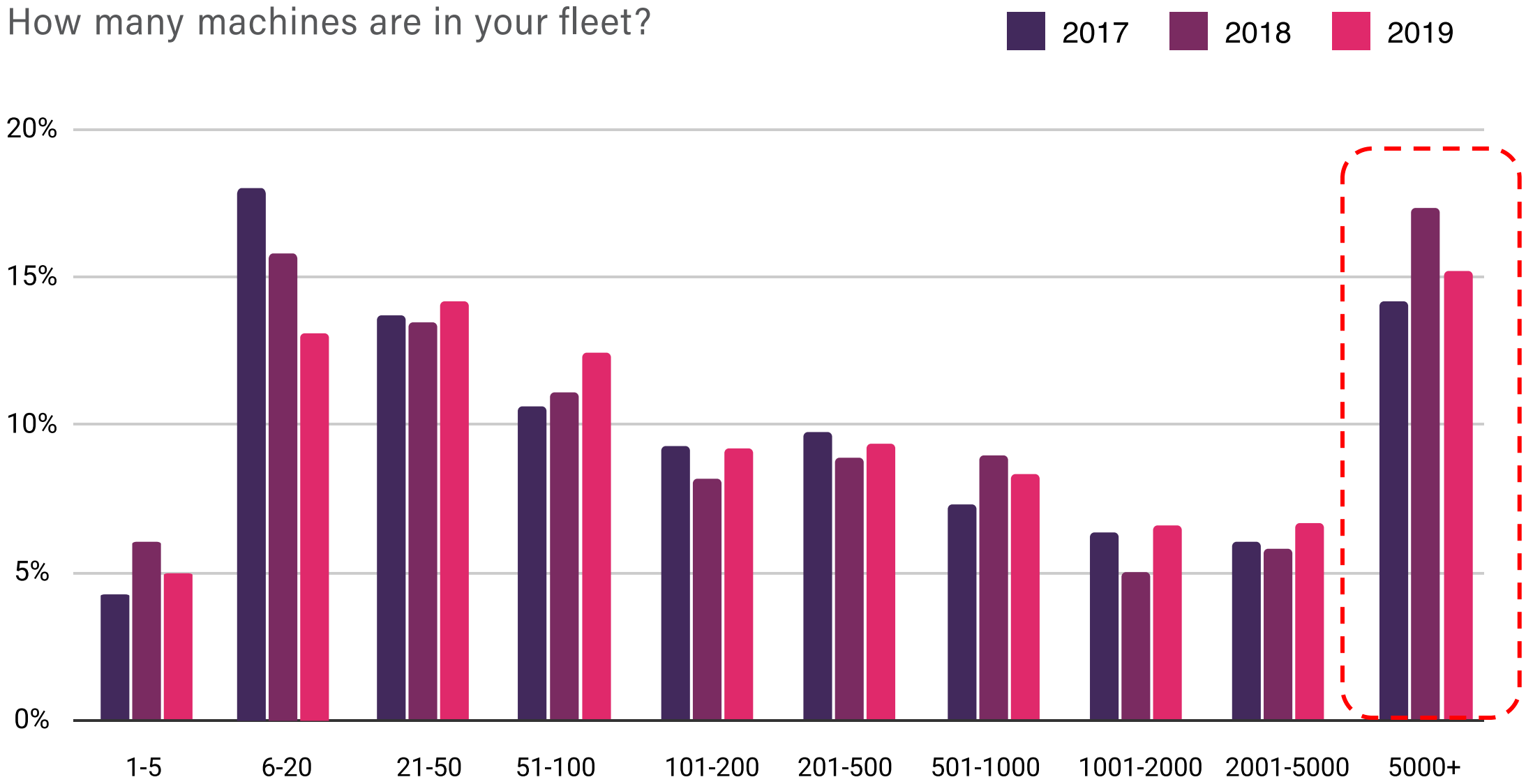
NETFLIX



Uber

● 云原生系统规模——物理机

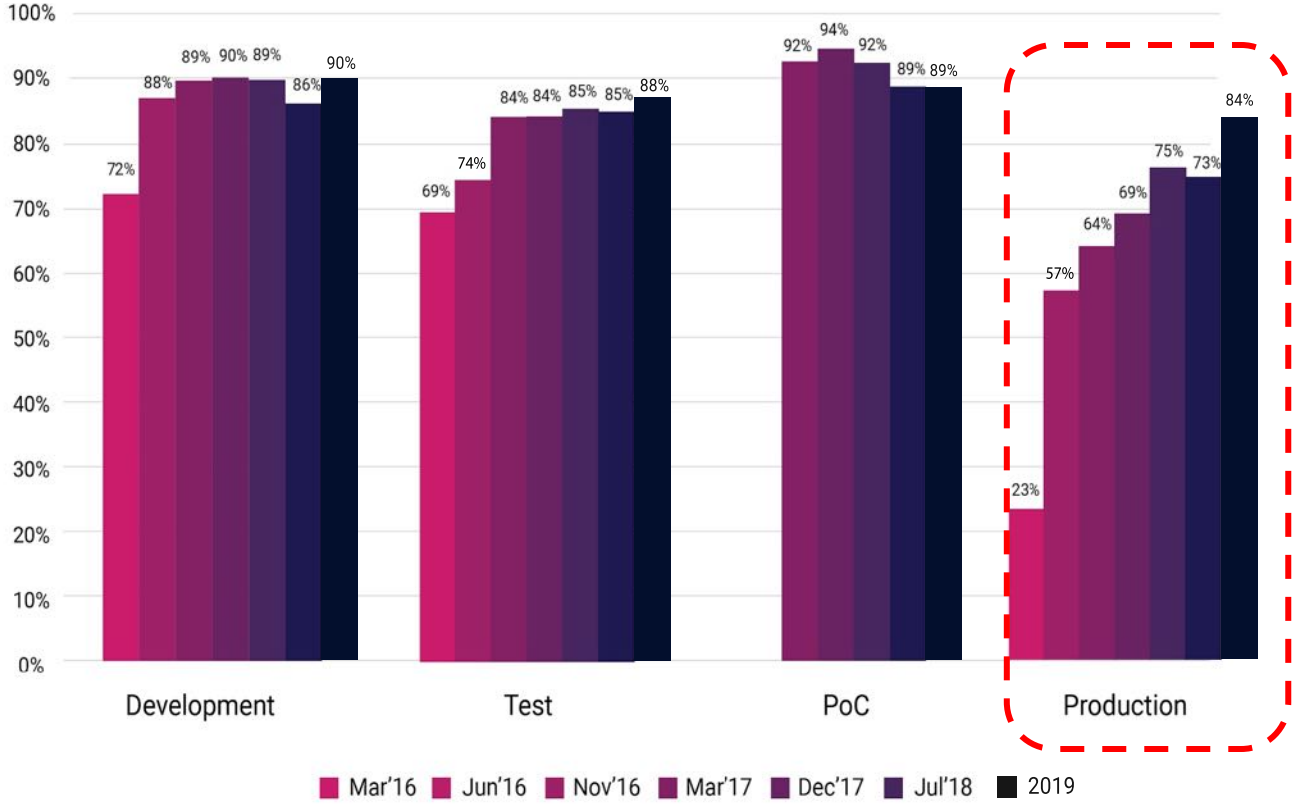
来源：CNCF Survey 2019



● 云原生系统规模—容器

➢ 超过15%的调查企业拥有超过5000台机器，有超过19%的调查企业拥有超过5000个容器；

Use of Containers since 2016



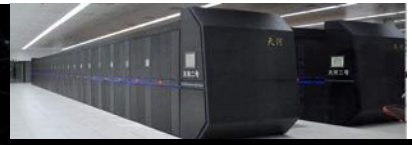
Number of Containers in Production

● 云系统双刃剑

➢ 大规模云计算系统的应用为开发、交付、扩展提供了极大的便利，但是**故障频发成为常态**，而且难以定位故障；

大规模分布式系统双刃剑





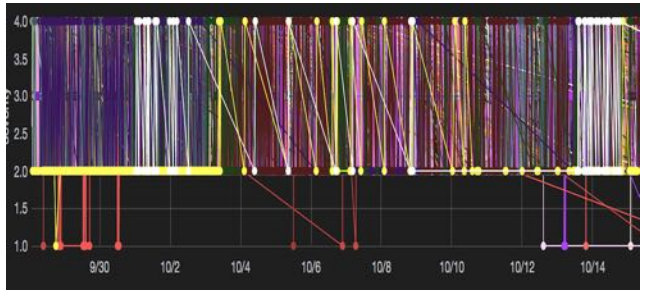
● 云原生系统中的故障

Xiang Zhou, Xin Peng, Tao Xie, Jun Sun, Chao Ji , et.al., Fault Analysis and Debugging of Microservice Systems: Industrial Survey, Benchmark System, and Empirical Study, IEEE TRANSACTION ON SOFTWARE ENGINEERING, VOL. 14, NO. 8, AUGUST 2018, pp:1-18.

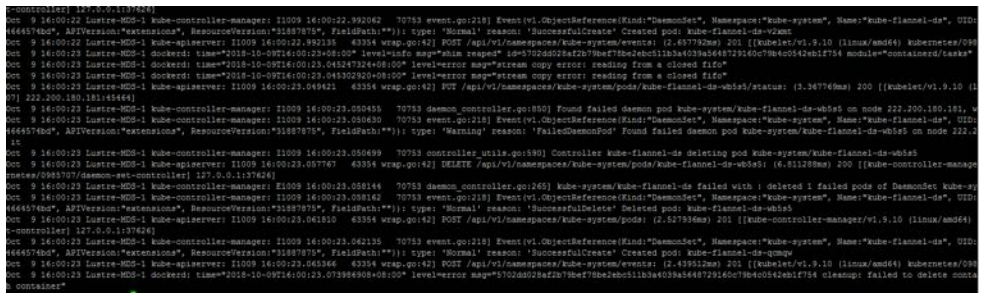
Microservice Fault Cases Reported by the Participants

Fault	Reporter	Symptom	Root Cause	Time
F1	P1 (A1)	Messages are displayed in wrong order	Asynchronous message delivery lacks sequence control	7D
F2	P2 (A2)	Some information displayed in a report is wrong	Different data requests for the same report are returned in an unexpected order	3D
F3	P2 (A2)	The system periodically returns server 500 error	JVM configurations are inconsistent with Docker configurations	10D
F4	P3 (A3)	The response time for some requests is very long	SSL offloading happens in a fine granularity (happening in almost each Docker instance)	7D
F5	P4 (A4)	A service sometimes returns timeout exceptions for user requests	The high load of a type of requests causes the timeout failure of another type of requests	6D
F6	P5 (A5)	A service is slowing down and returns error finally	Endless recursive requests of a microservice are caused by SQL errors of another dependent microservice	3D
F7	P6 (A6)	The payment service of the system fails	The overload of requests to a third-party service leads to denial of service	2D
F8	P7 (A7)	A default selection on the web page is changed unexpectedly	The key in the request of one microservice is not passed to its dependent microservice	5D
F9	P7 (A7)	There is a Right To Left (RTL) display error for UI words	There is a CSS display style error in bi-directional	0.5D
F10	P8 (A8)	The number of parts of a specific type in a bill of material (BOM) is wrong	An API used in a special case of BOM updating returns unexpected output	4D
F11	P9 (A8)	The bill of material (BOM) tree of a product is erroneous after updates	The BOM data is updated in an unexpected sequence	4D
F12	P10 (A9)	The price status shown in the optimized result table is wrong	Price status querying does not consider an unexpected output of a microservice in its call chain	6D
F13	P11 (A9)	The result of price optimization is wrong	Price optimization steps are executed in an unexpected order	6D
F14	P11 (A9)	The result of the Consumer Price Index (CPI) is wrong	There is a mistake in including the locked product in CPI calculation	2D
F15	P11 (A9)	The data-synchronization job quits unexpectedly	The spark actor is used for the configuration of actorSystem (part of Apache Spark) instead of the system actor	3D
F16	P11 (A9)	The file-uploading process fails	The “max-content-length” configuration of spray is only 2 Mb, not allowing to upload a bigger file	2D
F17	P12 (A10)	The grid-loading process takes too much time	Too many nested “select” and “from” clauses are in the constructed SQL statement	1D
F18	P13 (A11)	Loading the product-analysis chart is erroneous	One key of the returned JSON data for the UI chart includes the null value	0.5D
F19	P13 (A11)	The price is displayed in an unexpected format	The product price is not formatted correctly in the French format	1D
F20	P14 (A12)	Nothing is returned upon workflow data request	The JBoss startup classpath parameter does not include the right DB2 jar package	3D
F21	P15 (A13)	JAWS (a screen reader) misses reading some elements	The “aria-labeled-by” element for accessibility cannot be located by the JAWS	0.5D
F22	P16 (A13)	The error of SQL column missing is returned upon some data request	The constructed SQL statement includes a wrong column name in the “select” part according to its “from” part	1.5D

● 故障分析面临的挑战



KPI指标数据

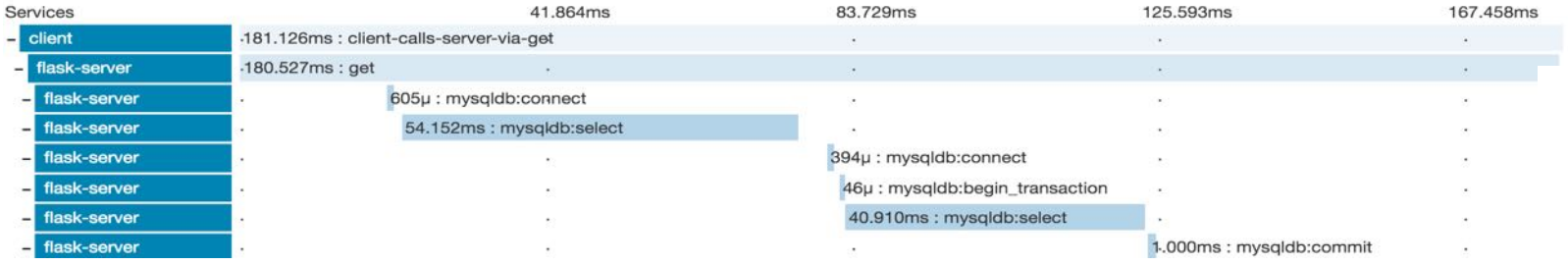


日志数据

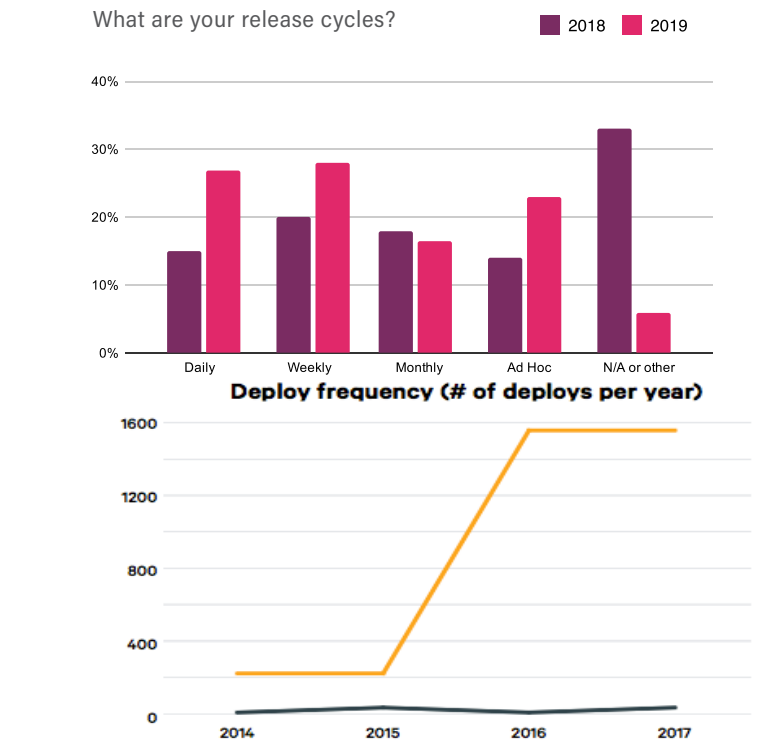
1527782760	#######	KH龙岗SH-3894-坪山工业区(腾讯客户)MUT_LOS
1527783180	#######	GD深圳龙岗麻岭工业区_BTS_900/1800_1无线基站设备和链路类告警合并
1527783180	#######	GD深圳龙岗麻岭工业区_BTS_900/1800_1载频与CMB的HDLC链路断
1527784020	#######	GD深圳横岗贤西_BTS_900/1800_1无线基站设备和链路类告警合并
1527784020	#######	GD深圳横岗贤西_BTS_900/1800_1载频与CMB的HDLC链路断
1527784260	#######	GD深圳平湖大街179号_BTS_900/1800_1无线基站设备和链路类告警合并
1527784260	#######	GD深圳平湖大街179号_BTS_900/1800_1载频与CMB的HDLC链路断
1527784260	#######	GD深圳平湖大街179号_BTS_900/1800_1载频与CMB的HDLC链路断

告警事件

监控数据**种类多、体量大、速度快**：LinkedIn的IT系统中，有多达400个服务、上千台物理机的性能数据需要监控。Netflix有2,000,000个服务性能指标，Uber有500,000,000个性能指标。



请求追踪数据

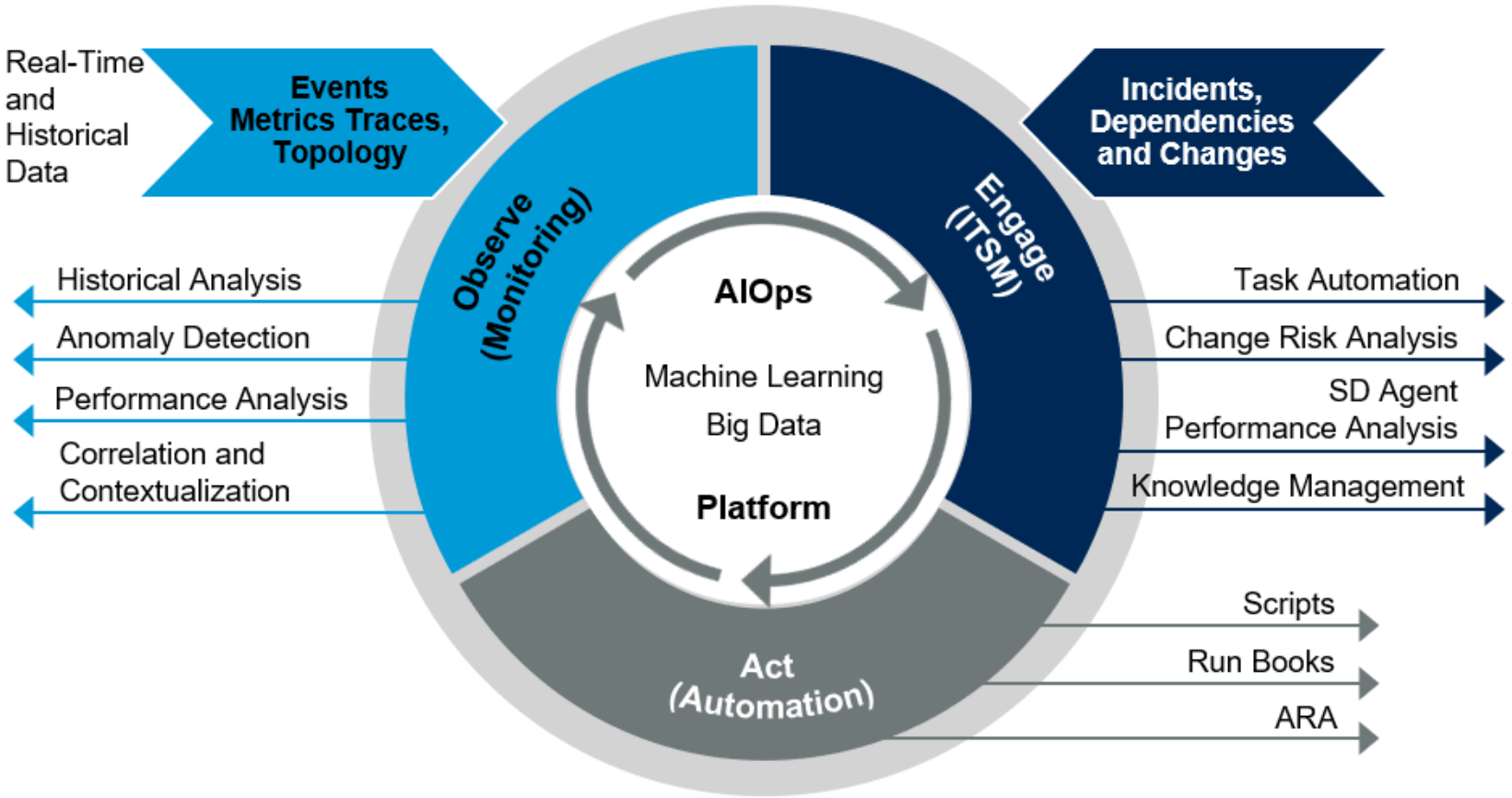


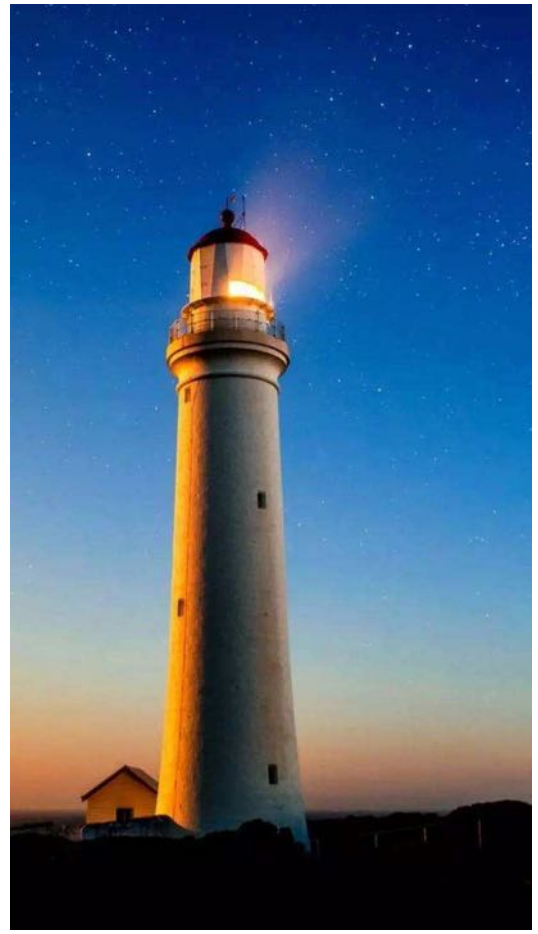
高频次的代码更新

问题出在什么地方？

● AIOps（Artificial Intelligence of IT Operations）

最初 “A” 的解释是Algorithmic， 即算法。现在指利用大数据、机器学习和其他分析技术，通过分析自动发现IT系统中的问题，并定位根因，甚至自动恢复系统，从而增强IT系统的稳定性、性能和可用性，为IT运维提供“端到端”的解决方法，最终做到“无人值守”。

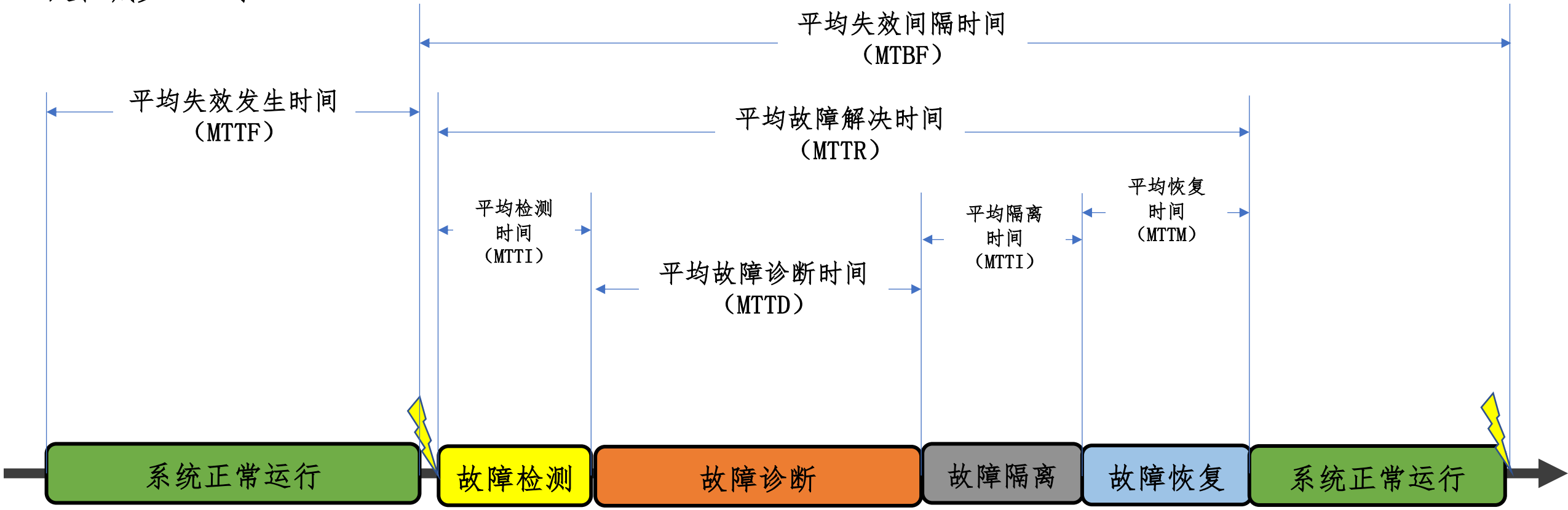




来源：Gartner: Market Guide for AIOps Platforms

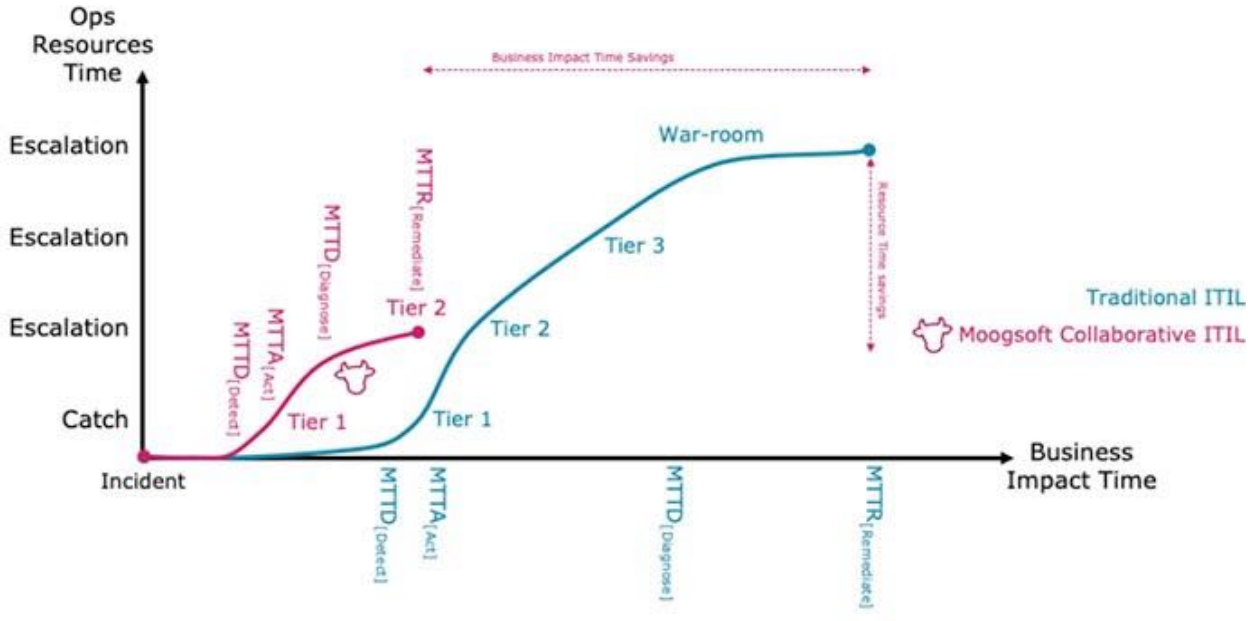
● 故障生命周期

故障从发生到解决经历检测、诊断、隔离、恢复等几个阶段，每个阶段都会产生造成一定的系统停机，影响系统整体的可用性，AIOPS所包含的不同方法、技术分别对应减少不同的阶段时间，如：异常检测减少MTTI，根因定位减少MTTD等

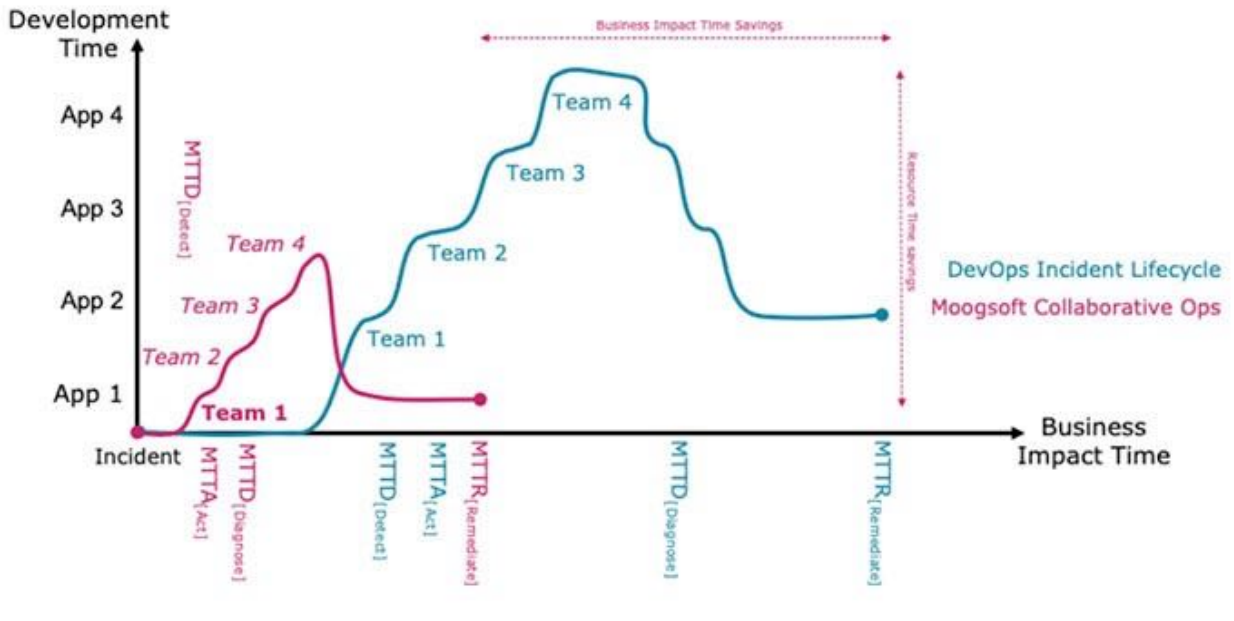


● AIOPS作用举例

来源：https://www.moogsoft.com/blog/aiops/transforming-devops-itops-mttr



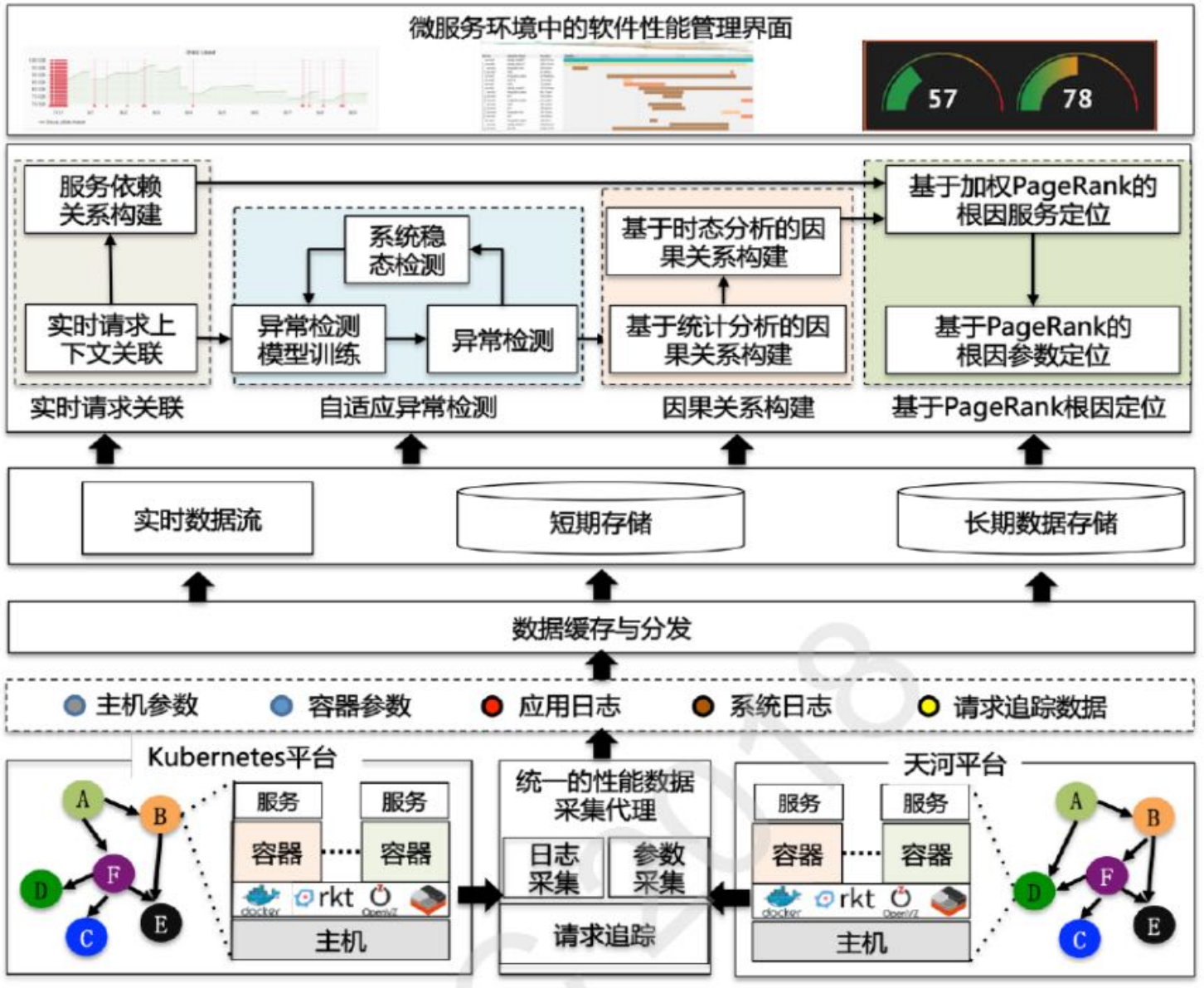
(a)使用了AIOPS后与传统的ITIL解决故障时间的对比

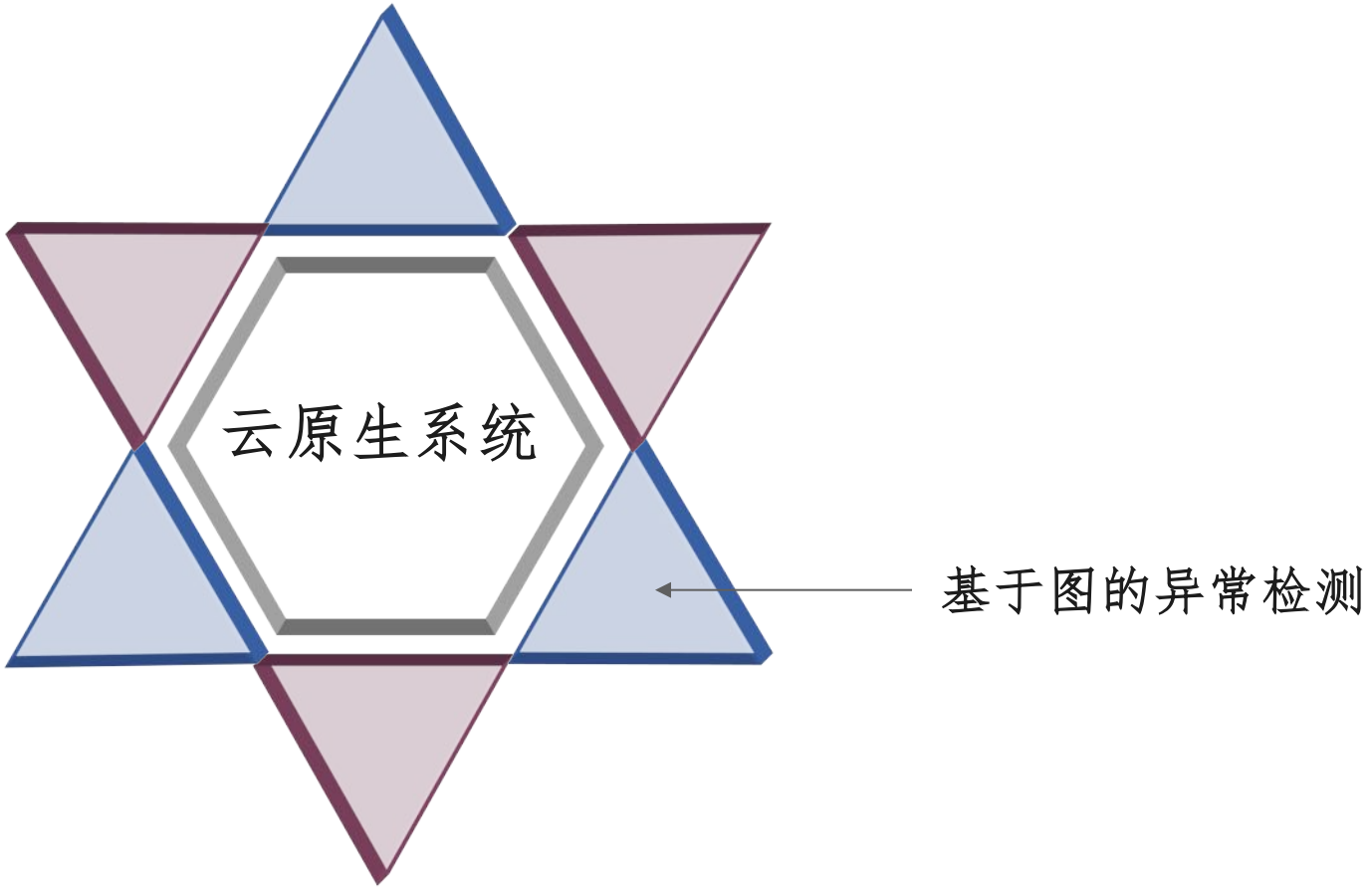


(b)将AIOPS与DevOps结合后与DevOps解决故障时间的对比

我们的研究

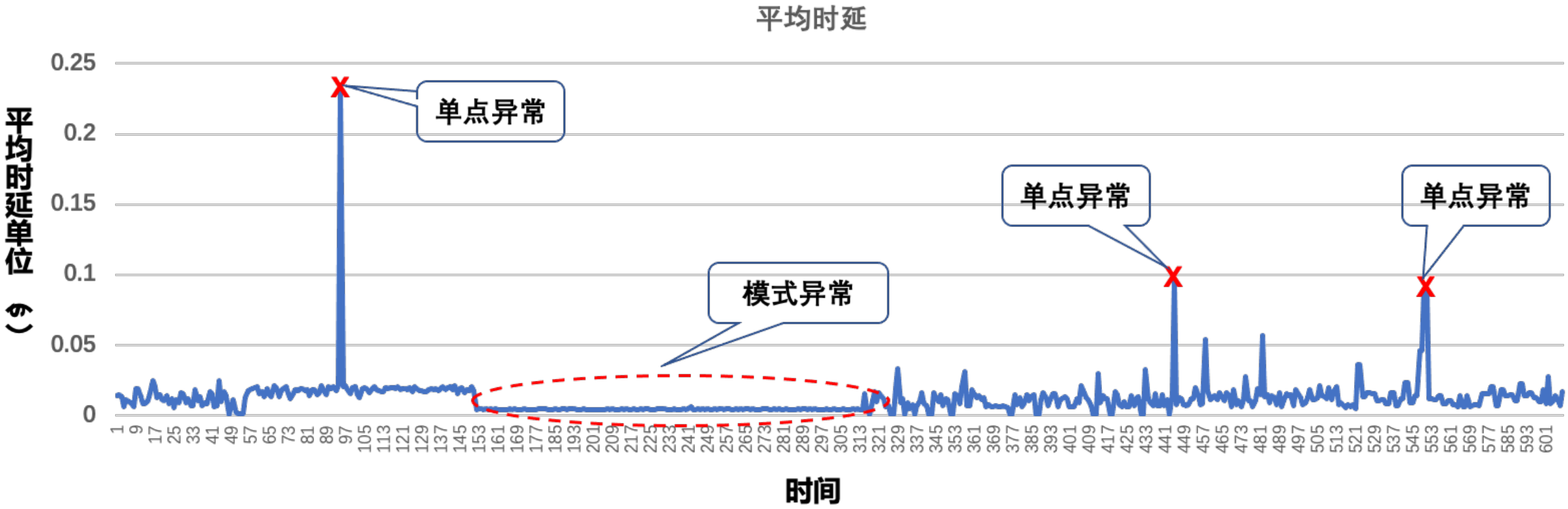
- 基于eBPF的性能监控
- 全链路上下文监控
- 基于日志的异常检测
- 基于调用链的异常检测及根因定位
- 调用链数据压缩；
- 基于因果关系的根因定位
- 告警关联和压缩




云原生系统
基于图的异常检测

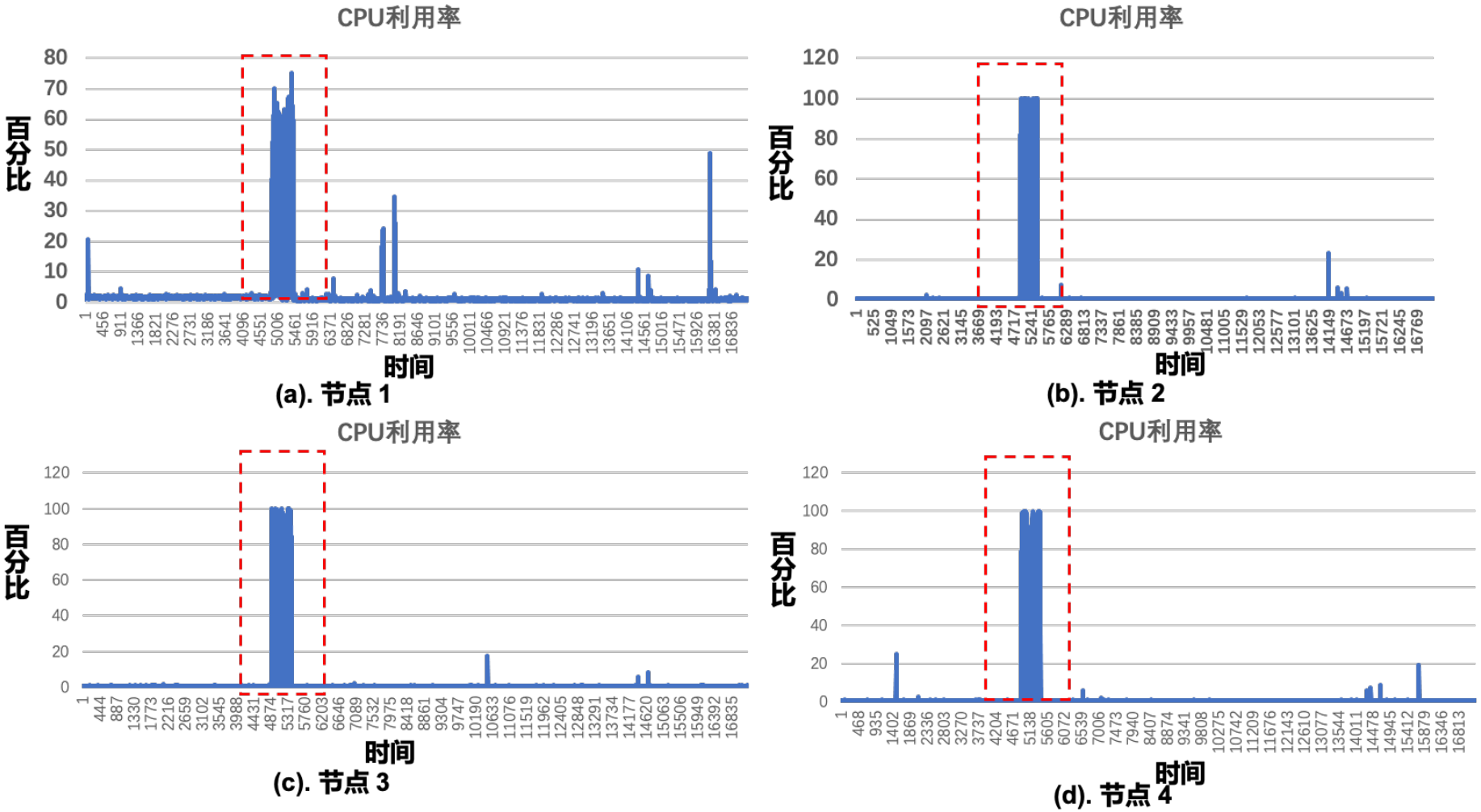
● 异常类型

- 软件系统异常类型可以分为三类：即单点异常、模式异常和上下文异常；
- 单点异常即某个数据点超出正常范围，可利用统计的方法检测异常；
- 模式异常即数据变化的模式发生变化，可通过聚类分析或者时间序列分析检测异常；

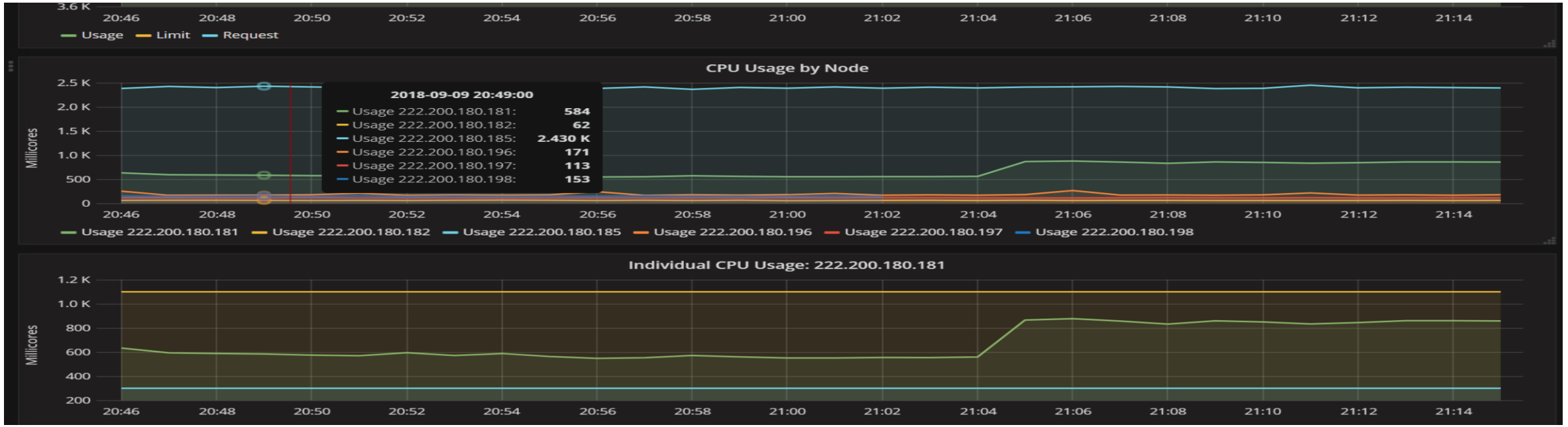


● 异常类型

➢ 上下文异常即数据点在特定的时空上下文的环境中才是异常，可以通过时空建模检测异常；



异常检测总览



基于阈值的方法
CUSUM
Shewhart Chart
Clustering、随机森林

单点异常检测

ARIMA, SARIMA, Prophet, SSA, Holt-Winer, LSTM, NN 、强化学习等时序分析

{1,3,4,5,7,4}

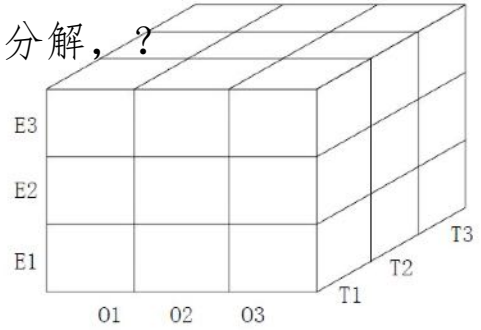
一维向量异常检测

PCA, RPCA, VAE
VAR, 矩阵分解机, Matrix sketch

1,3,4,5,7,4
2,4,5,6,7,8
4,6,7,9,3,0
…

二维矩阵异常检测

张量分解，？

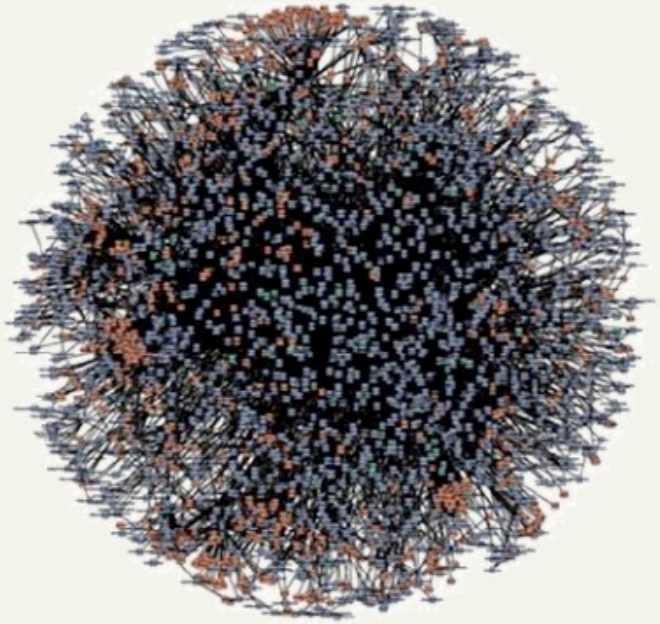


三维张量异常检测

● 服务之间复杂的依赖关系

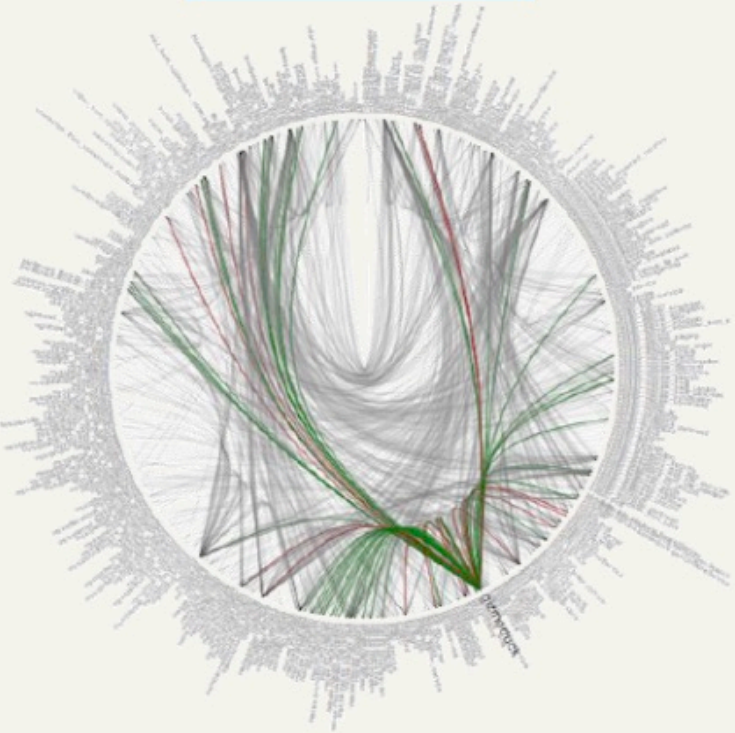
➢ 构成软件系统的服务之间具有复杂的依赖关系，且处于动态变化中；

amazon.com



NETFLIX

twitter



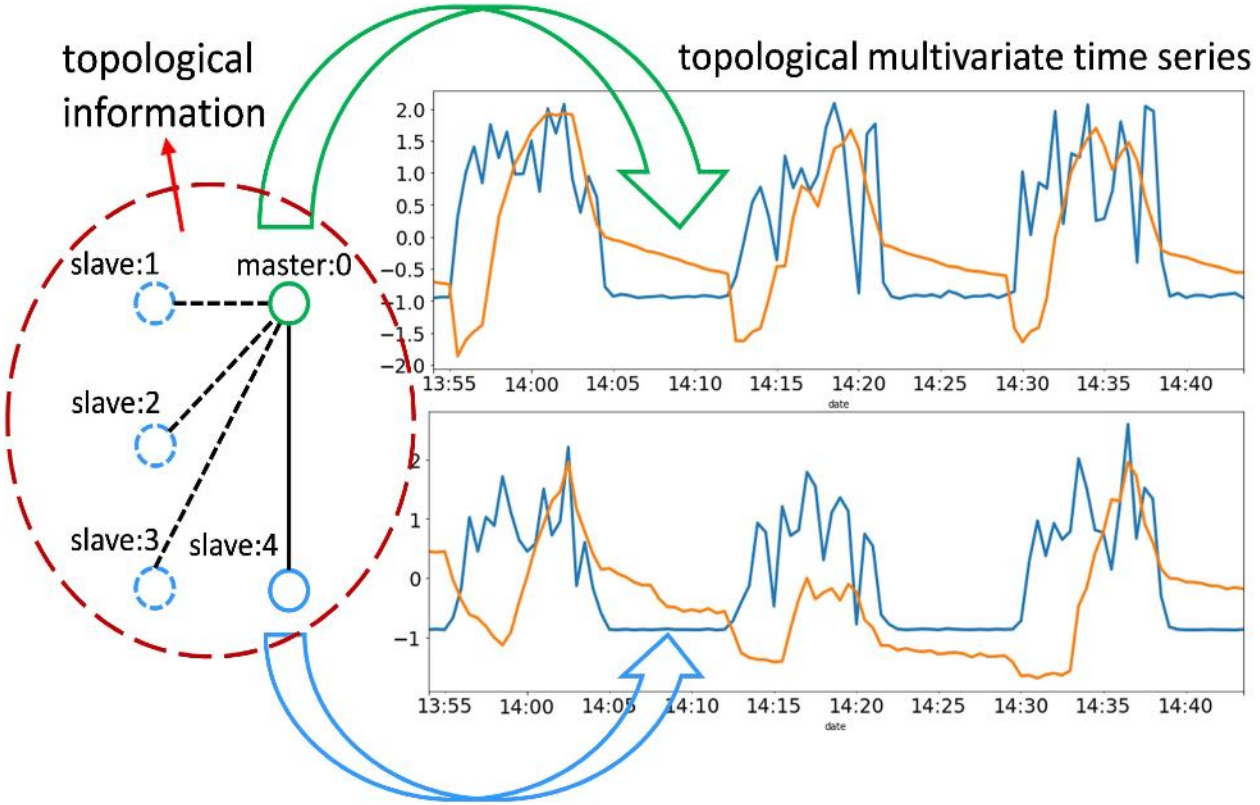
● 基于时空神经网络的无监督异常检测

➢ **算法满足以下特性：**

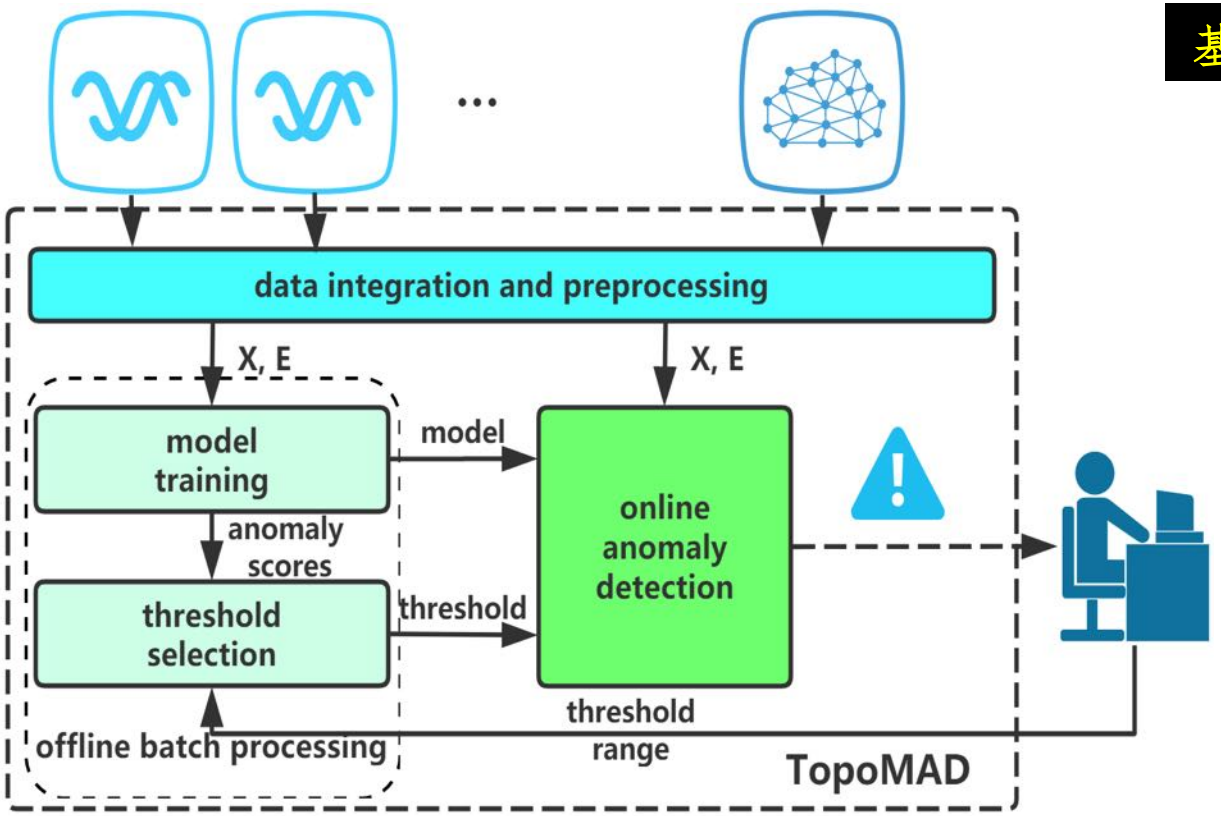
- 能够同时检测单点异常、模式异常和上下文异常·
- 无监督，不需要标签数据；
- 对系统的健康状态进行整体度量；
- 允许系统拓扑变化；

➢ **解决方案：**

- 同时建模指标数据的空间（即拓扑）和时间（即时序）特性；
- 采用图卷积神经网络模型；
- 采用自编码器模型；
- 自适应阈值调整；



● 基于时空神经网络的无监督异常检测



系统整体设计

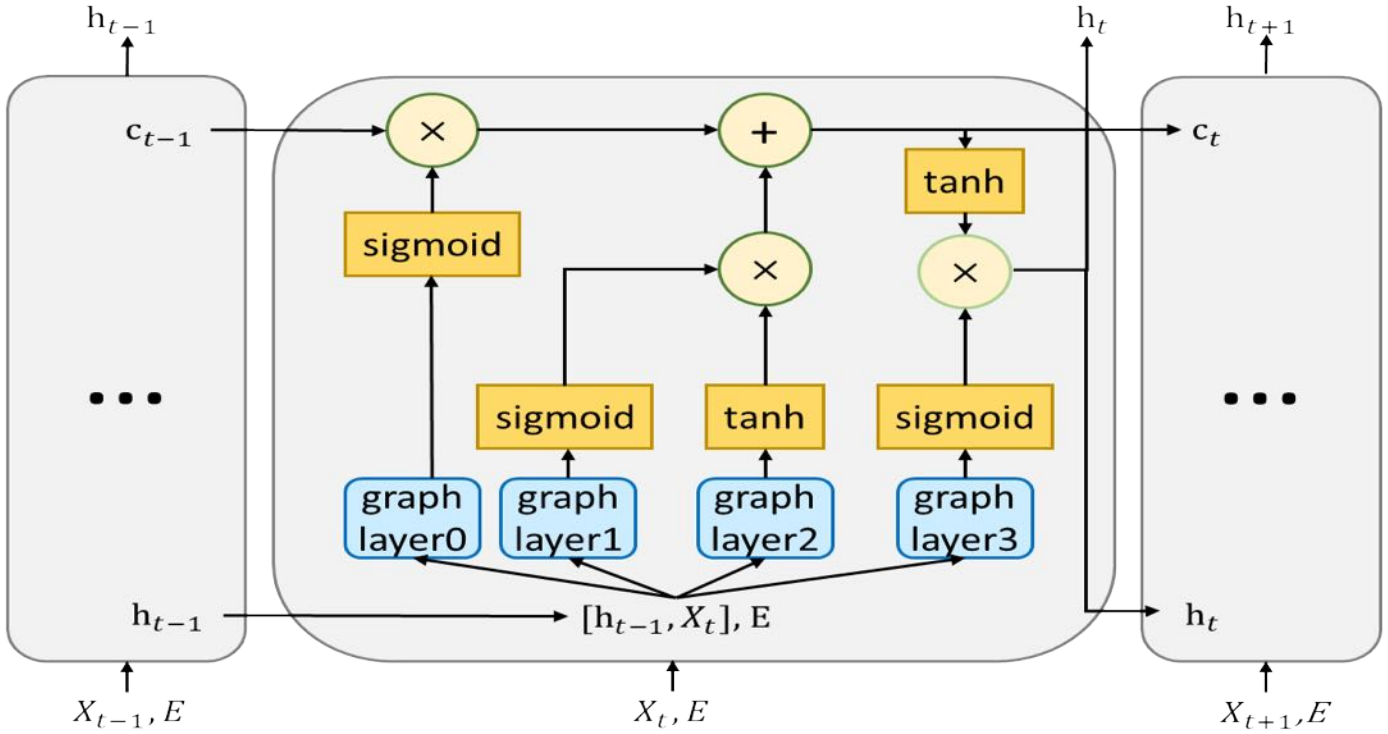
基于LSTM神经网络+图神经网络的变分自编码器异常检测方法

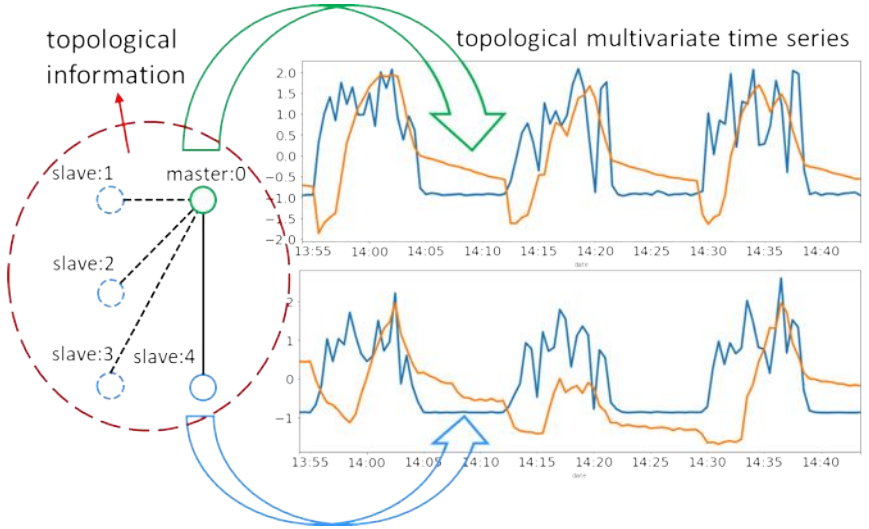
- ❖ LSTM神经网络：展开时序特征
  - 循环读取序列，当前隐状态由上一步隐状态和当前输入结合计算
- ❖ 图神经网络：展开空间特征（来自拓扑）
  - 节点表示由该节点本身特征和节点邻居节点的特征融合构成
  - 不同融合方式产生图卷积神经网络的不同变种：图卷积神经网络，图注意力神经网络等

算法原理

● GraphLSTM

➢ 融合LSTM神经网络和图神经网络同时提取数据中的时空信息





输入信息

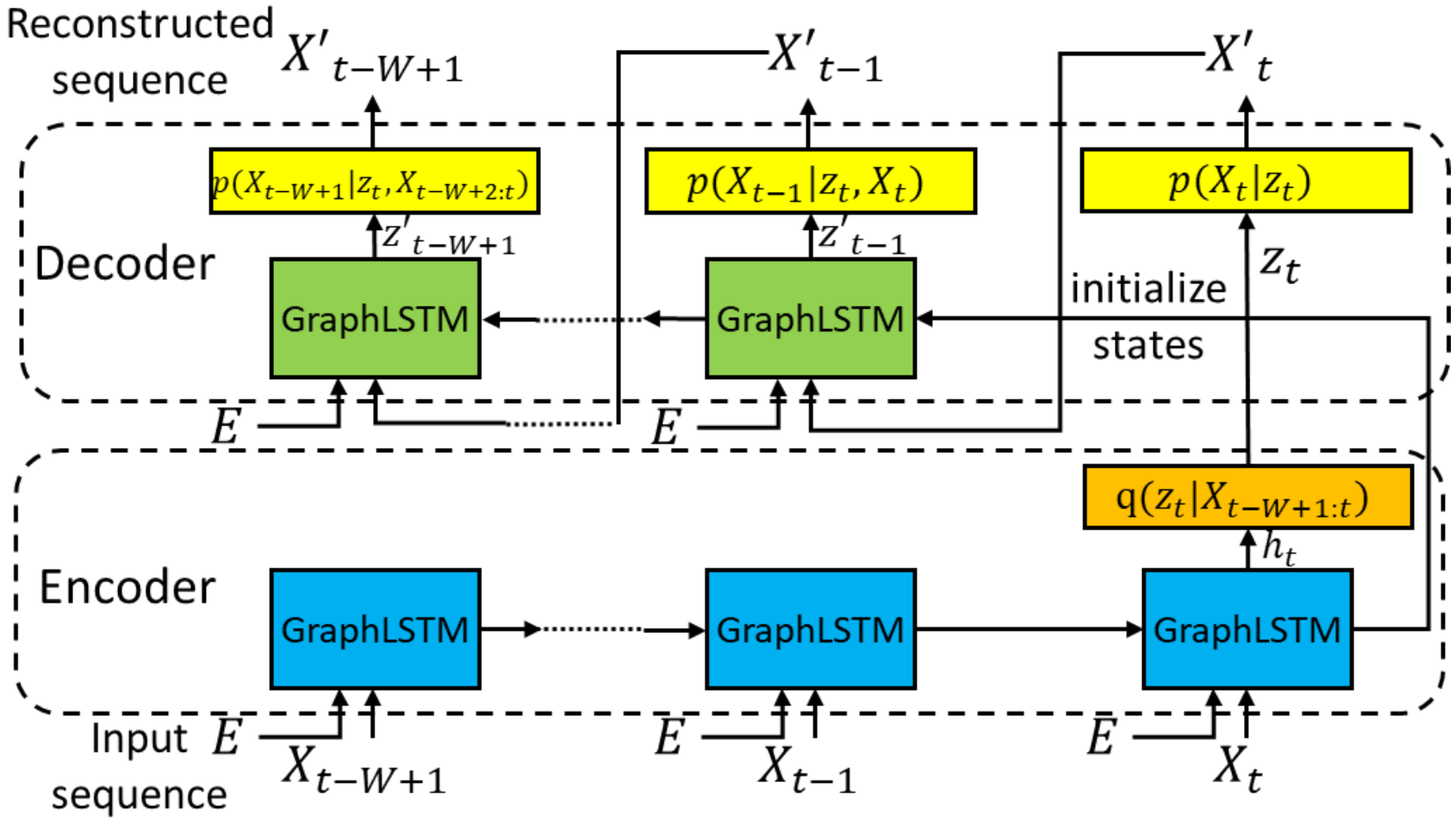
topology: A = $\begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$ or E = $\begin{bmatrix} 0 & 1 & 0 & 2 & 0 & 3 & 0 & 4 \\ 1 & 0 & 2 & 0 & 3 & 0 & 4 & 0 \end{bmatrix}$

+

metrics of each node: X = $\begin{bmatrix} cpu_0 & \cdots & socket_0 & \cdots \\ \vdots & \ddots & \ddots & \vdots \\ cpu_4 & \cdots & socket_4 & \cdots \end{bmatrix}$

27

● 自编码器模型



➢ 算法运行过程

利用重构概率作为异常得分：

$$tempS_t = -\mathbf{E}_{q_\phi(z_t|X_{t_0:t})}(\log(p_\theta(X_t|z_t)))$$

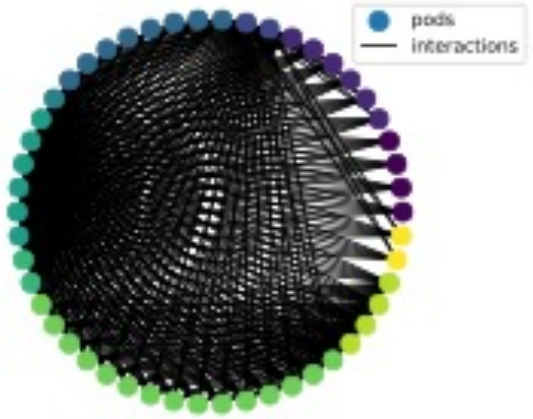
$$S_t = -\max_{0\le i<N} \frac{1}{L*D}\sum_{d=0}^{D-1}\sum_{l=1}^{L}\log(p_\theta(X_t^i|z_{t+d}^{(l)}, X_{t+1:t+d})),$$

选择一个合适的阈值报告异常；

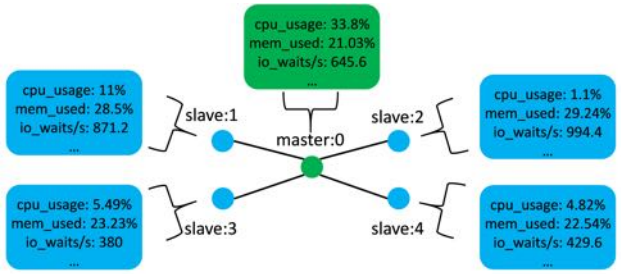
$$d(S_{<\tau}, S_{>\tau}) = \frac{\min(S_{>\tau}) - \max(S_{<\tau})}{\min(S_{>\tau}) + \max(S_{<\tau}) - 2*\min(S_{<\tau})},$$

结合LSTM神经网络，图神经网络的变分自编码器，用于进行非监督异常检测

● 实验结果



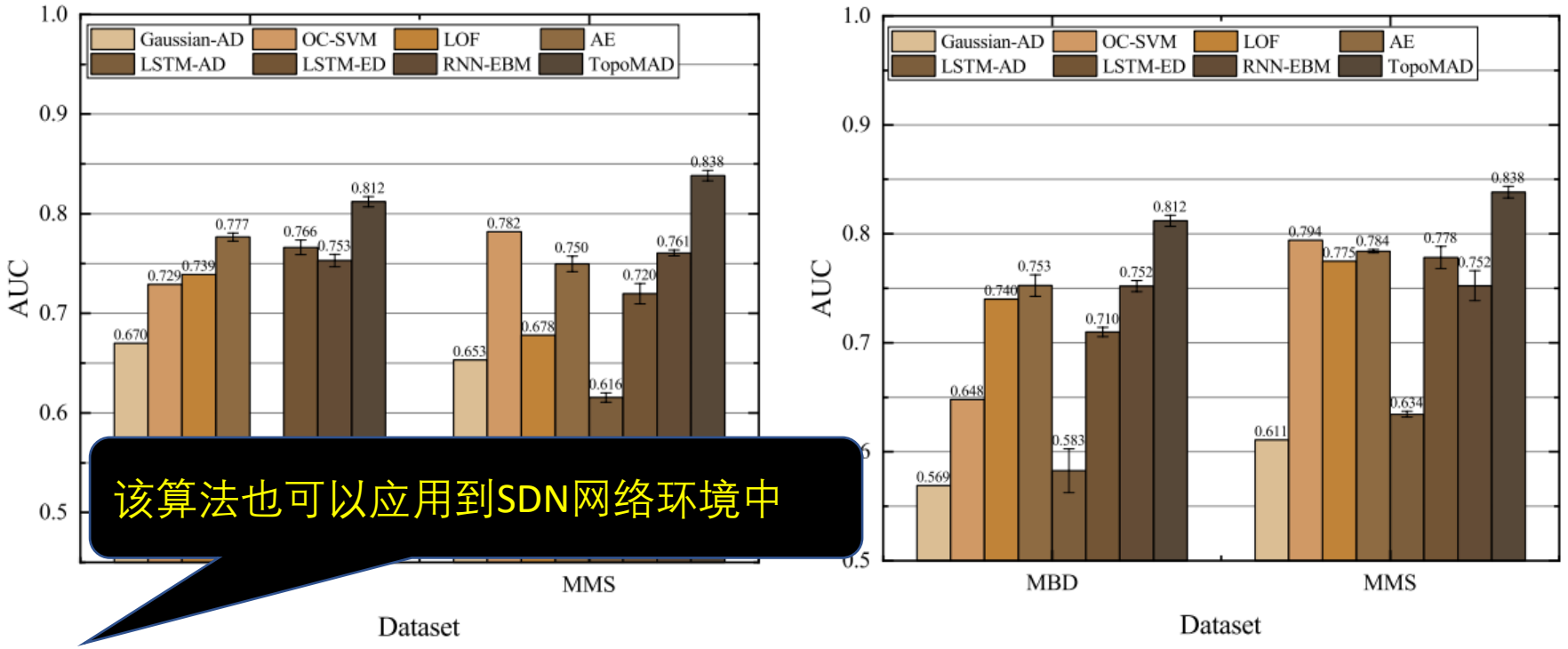
Hipster-shop Microservice benchmark



Hadoop system

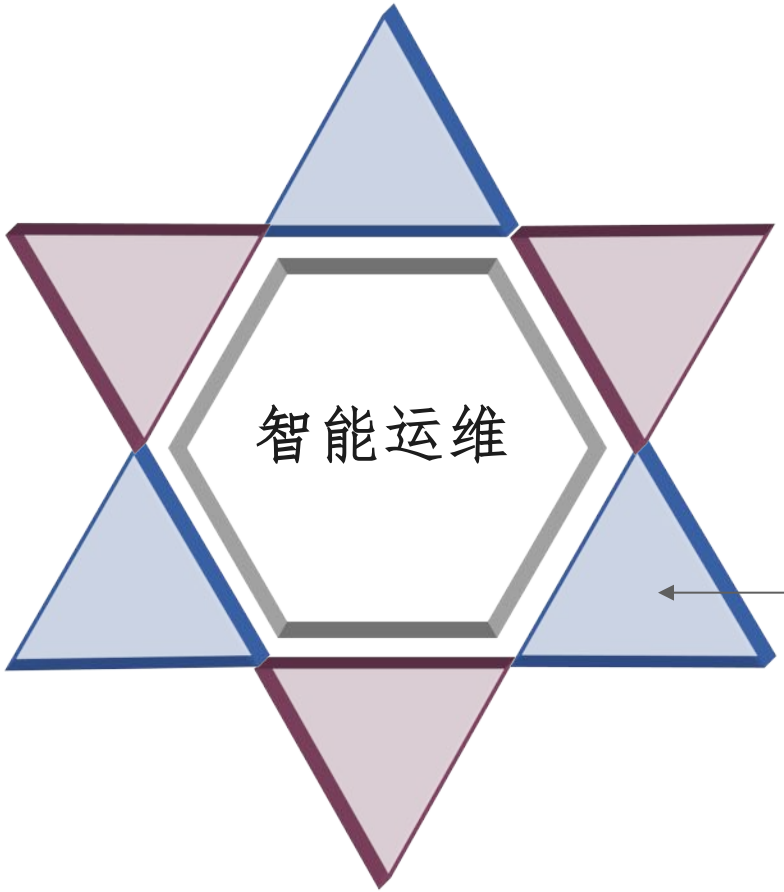
采用混沌工程的方法注入故障

分别与将各个组件指标拼接并应用其它异常检测器和对各个组件分别应用其它异常检测器做实验对比，结果如下：



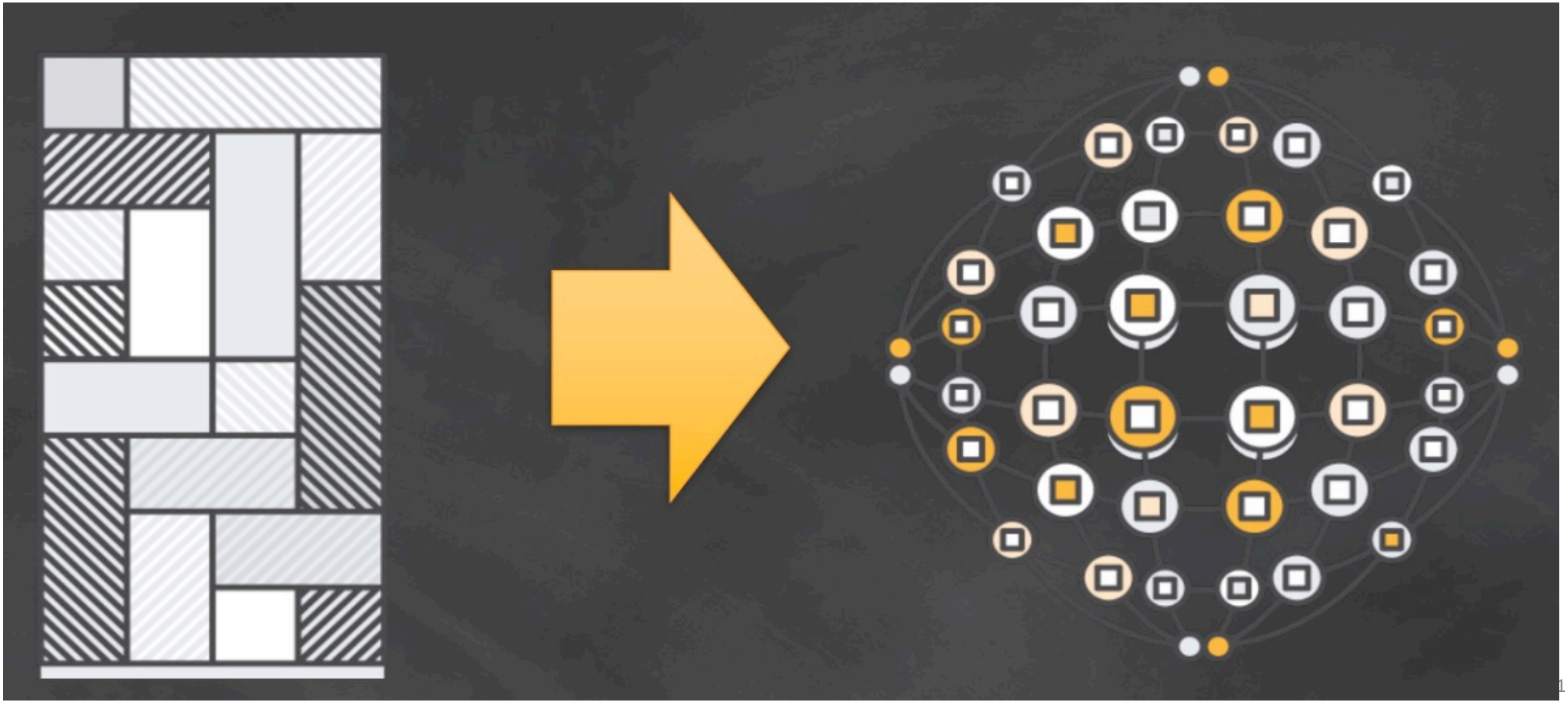
该算法也可以应用到SDN网络环境中

A Spatio-Temporal Deep Learning Approach for Unsupervised Anomaly Detection in Cloud Systems, IEEE Transactions on Neural Network and Learning System, 2020

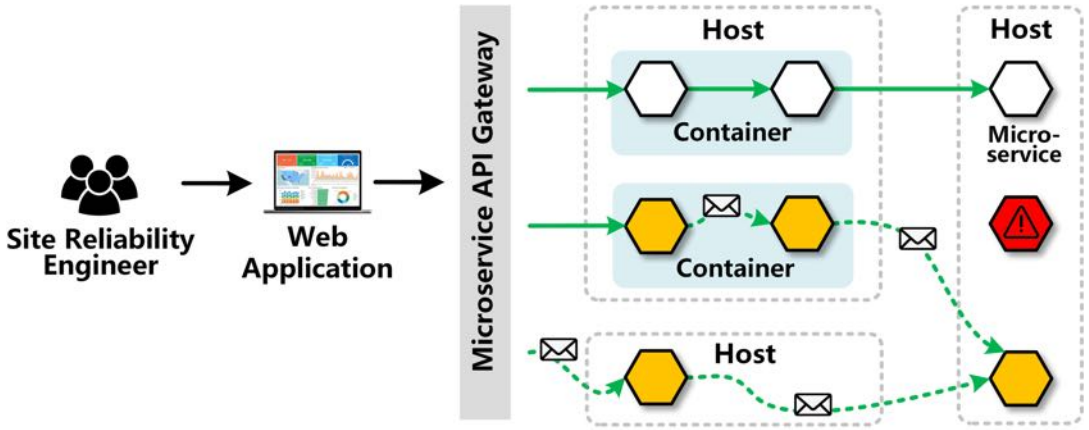


基于随机游走的根因定位

● 微服务系统



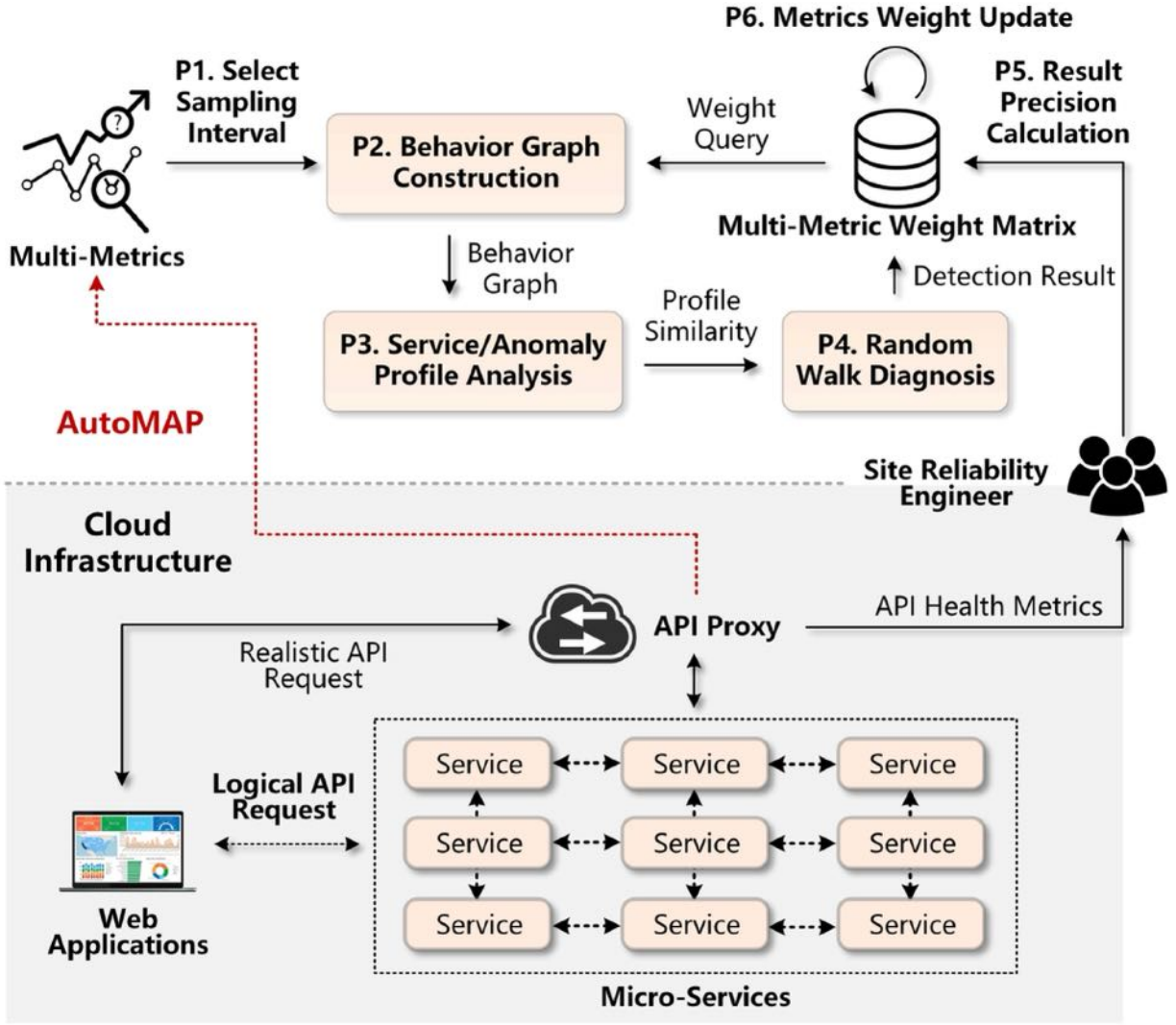
● 自动化诊断微服务性能问题



故障传播例子

➢ 动机和观察

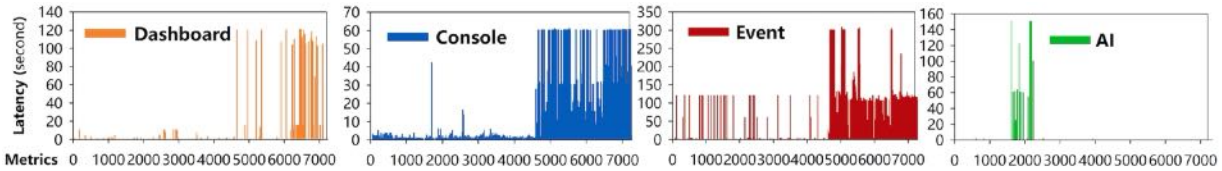
- 某些遗留系统中没有trace数据，而且难以获得拓扑；
- 微服务的计算环境一直处于动态变化中；
- 间接的故障传播路径（沿着非调用关系的路径）；
- 单类型的性能指标难以准确定位根因；
- 基于图谱遍历搜索的算法难以适应大规模微服务系统；



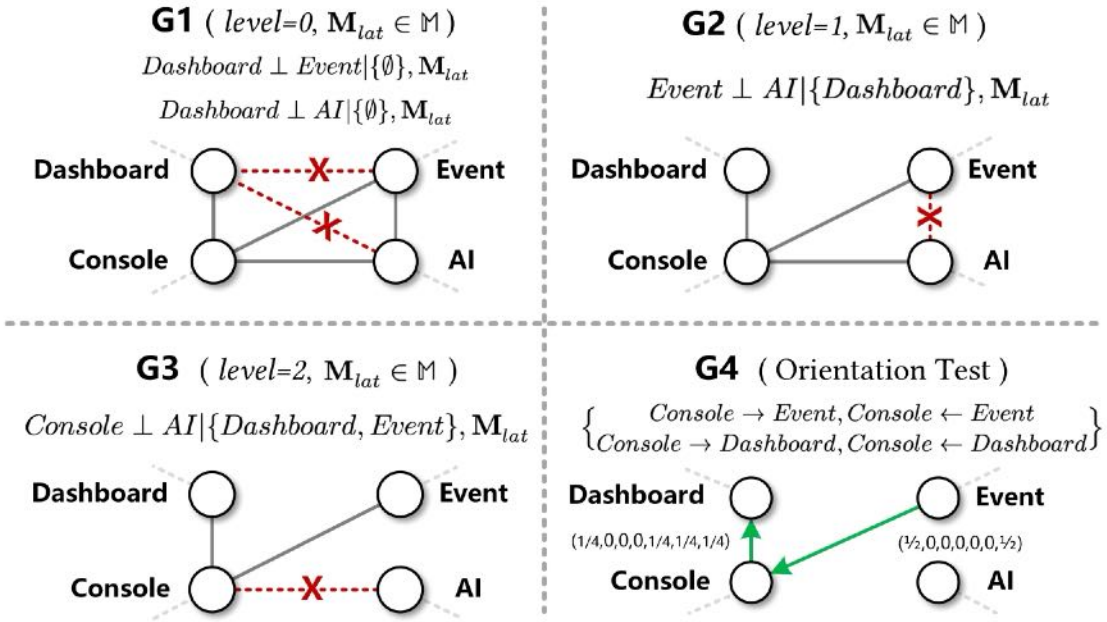
AutoMap：系统架构

● 自动化诊断过程

➢ 微服务行为图构建

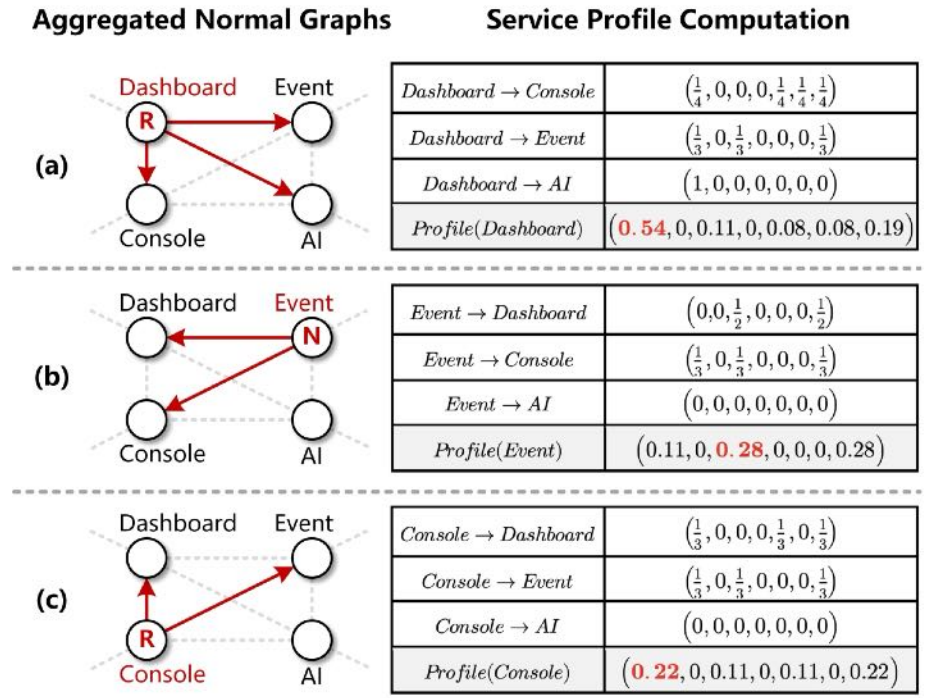


(a) Raw data

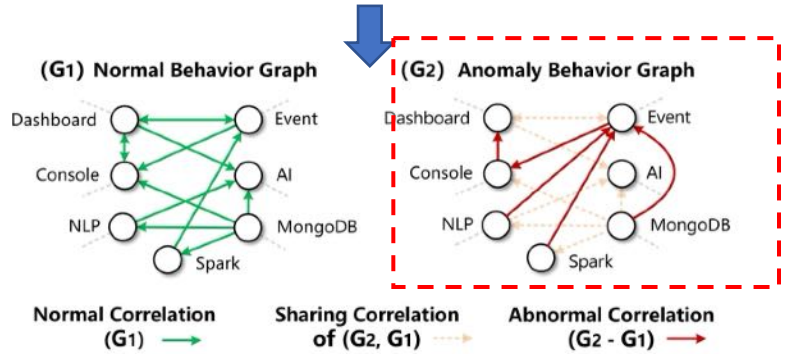


(b) Behavior graph Construction

	M_{lat}	M_{thr}	M_{con}	M_{cpu}	M_{io}	M_{mem}	M_{nel}	Overall Weight
Event → Console	1	0	0	0	1	1	1	$(\frac{1}{4},0,0,0,\frac{1}{4},\frac{1}{4},\frac{1}{4})$
Console → Dashboard	1	0	0	0	0	0	1	$(\frac{1}{2},0,0,0,0,0,\frac{1}{2})$



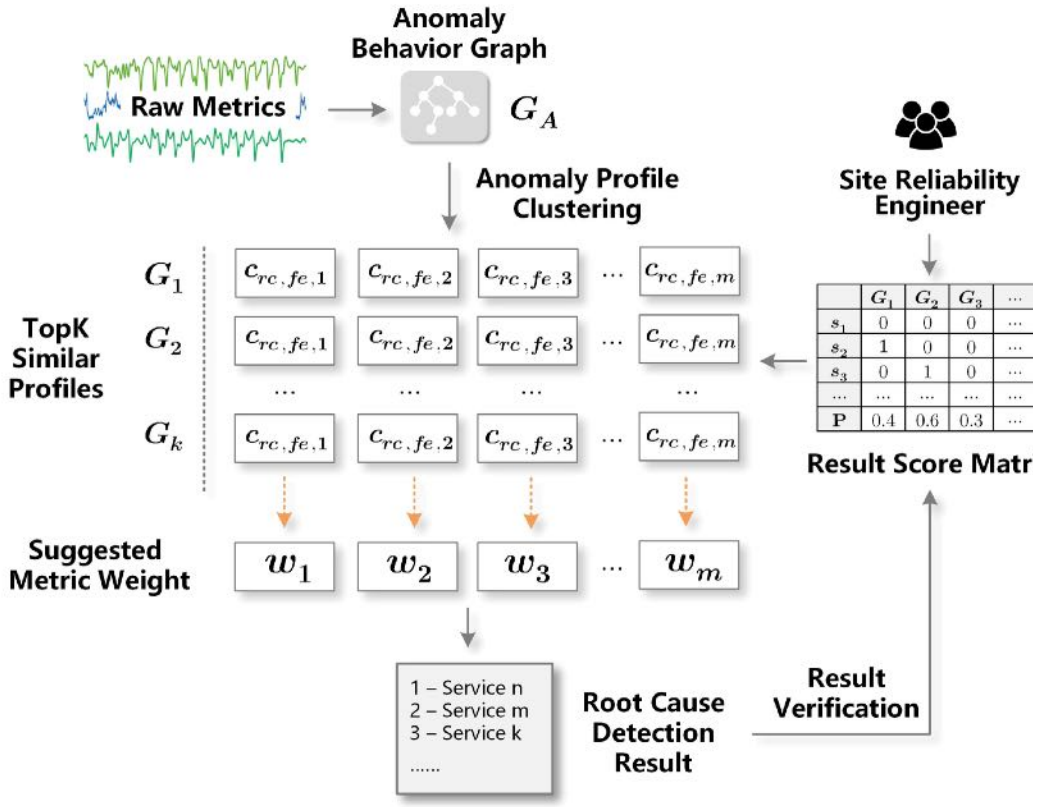
服务剖析

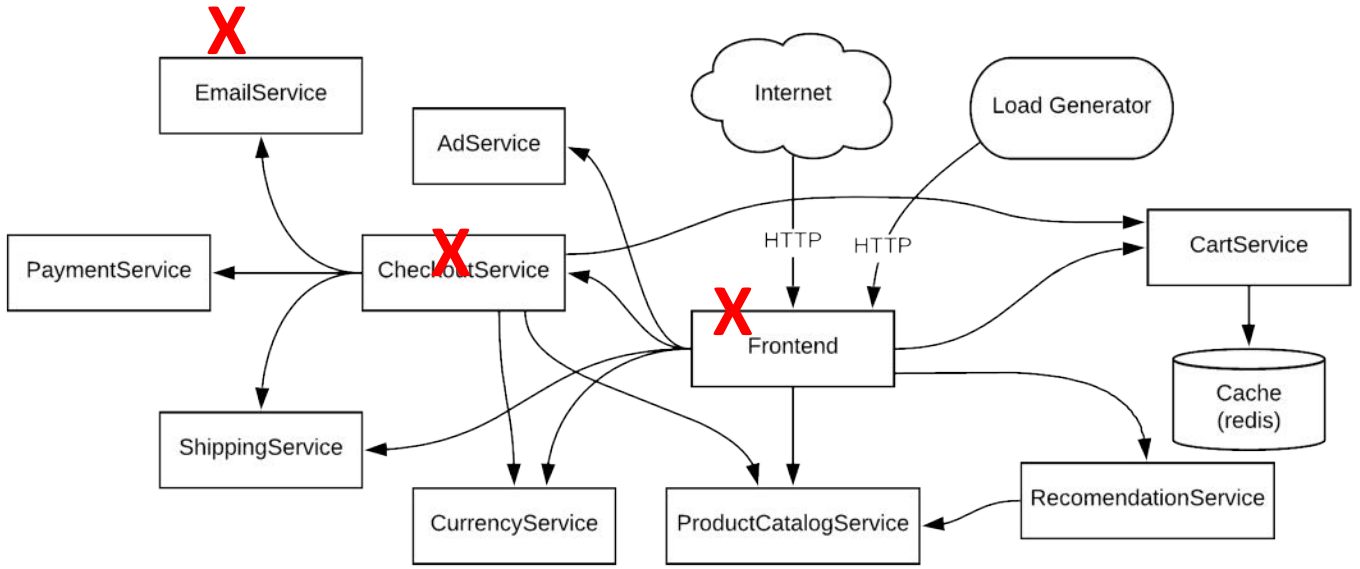


异常传播图

Figure 5: An example of behavior graph subtraction

● 自动化诊断过程

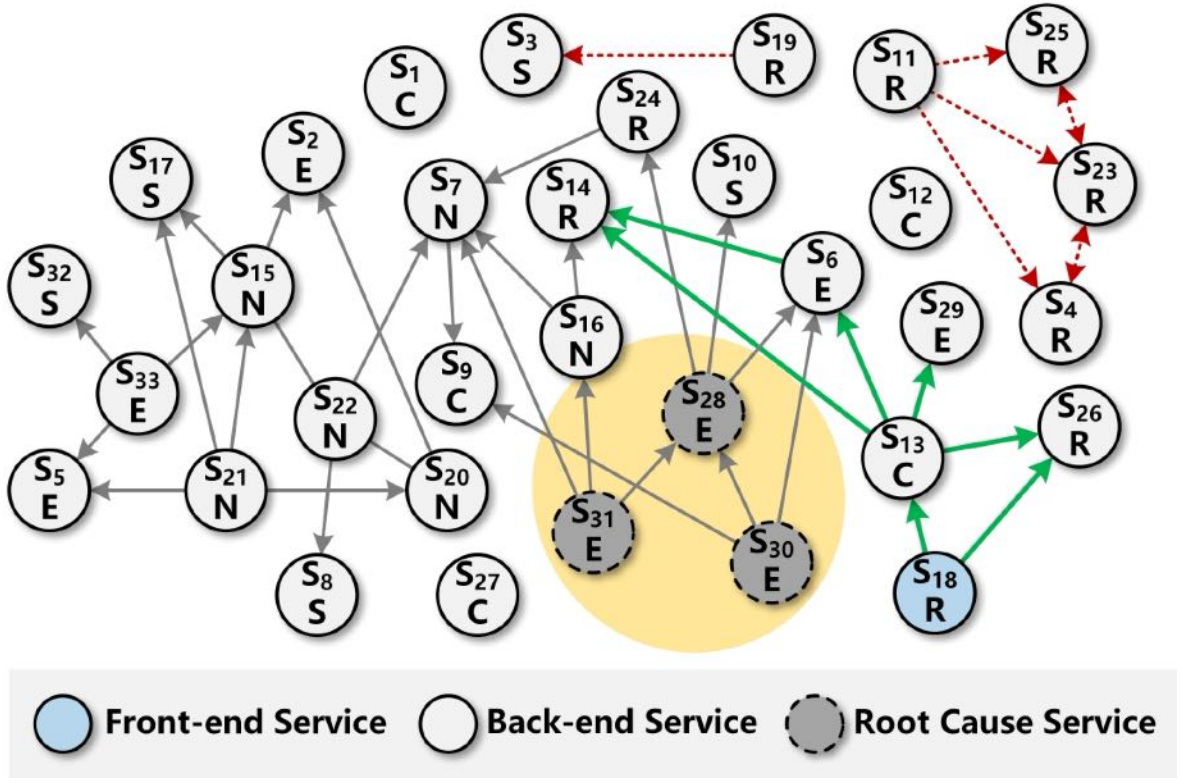




Google Hipster-shop Benchmark

业务指标筛选以及SRE反馈

● 自动化诊断过程



二阶随机游走，对节点重要性排序

➢ **前向搜索：**

利用跟前端节点的关联度作为搜索概率；

$$[\mathbf{P}]_{i,j} = p_{i,j} = \sum_{k=1}^{m} w_k W_{i,j,k} \frac{c_{j,fe,k}}{\sum_{l \in O_i} c_{l,fe,k}}, \forall e_{ij} = 1,$$

➢ **自我搜索：**

利用跟前端节点的关联度作为搜索概率；

$$p_i^s = \max\left(0, p_{i,i} - \max_{l \in I_i \cup O_i} p_{i,l}\right),$$

➢ **后向搜索：**

利用跟前端节点的关联度作为搜索概率；

$$p_{i,j}^b = \rho \frac{p_{i,j}}{\sum_{l \in I_i} p_{i,l}},$$

➢ **随机游走：**

一旦发生异常，从前端服务开始，利用前向、自我、后向等搜索过程，对根因进行定位；

● 自动化诊断结果

➢ 利用IBM Cloud的数据进行分析，发现经过多轮反馈后，诊断精度显著提高；

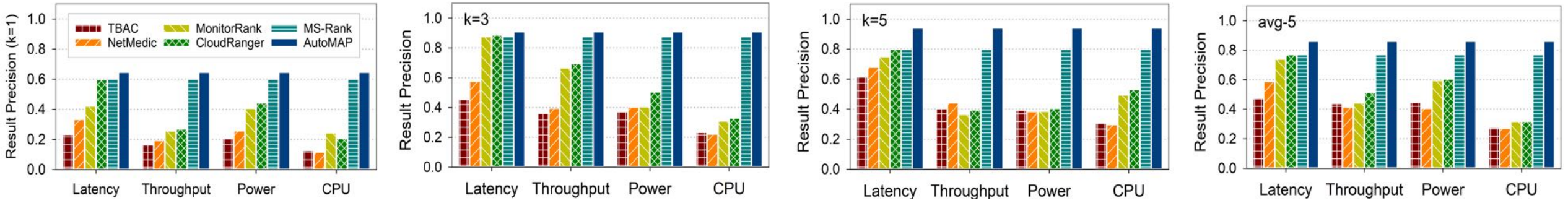


Figure 8: Top-1, 3, 5 and avg-5 precision of AutoMAP and different algorithms using real-world incidents

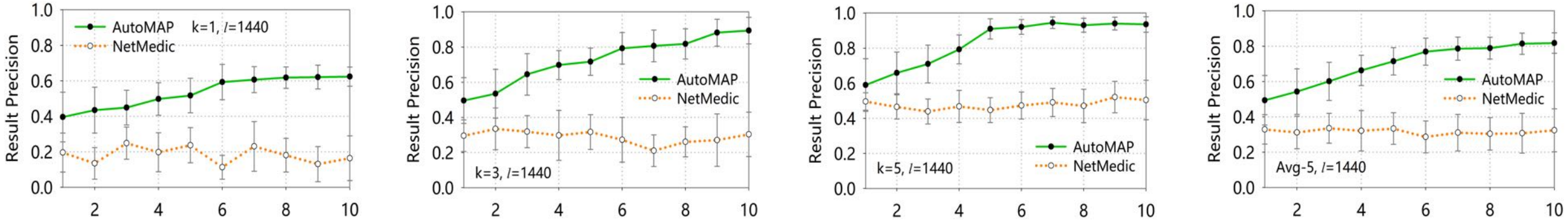
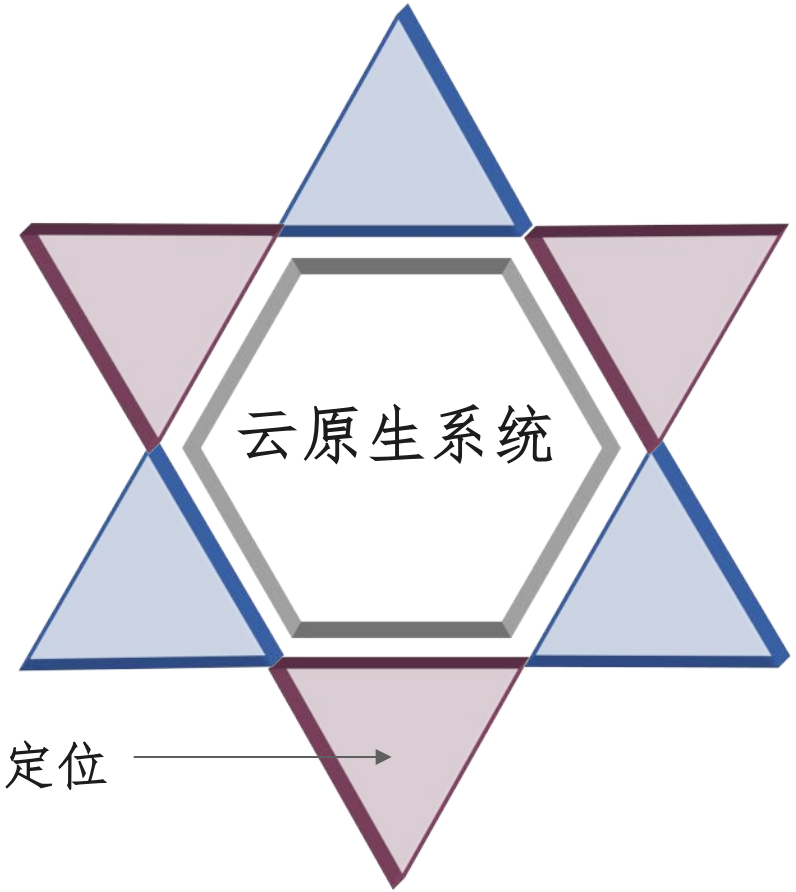


Figure 9: Precision of AutoMAP and NetMedic with different rounds of test using real-world incidents

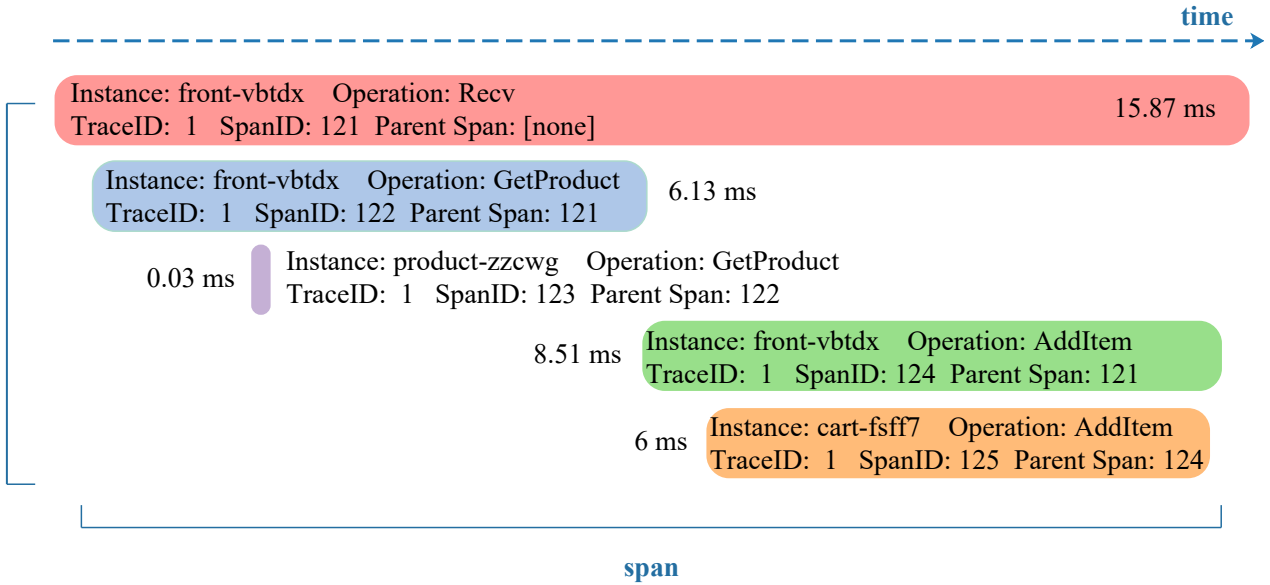
论文AutoMAP: Diagnose Your Microservice-based Web Applications Automatically发表在 （WWW 2020，CCF A）

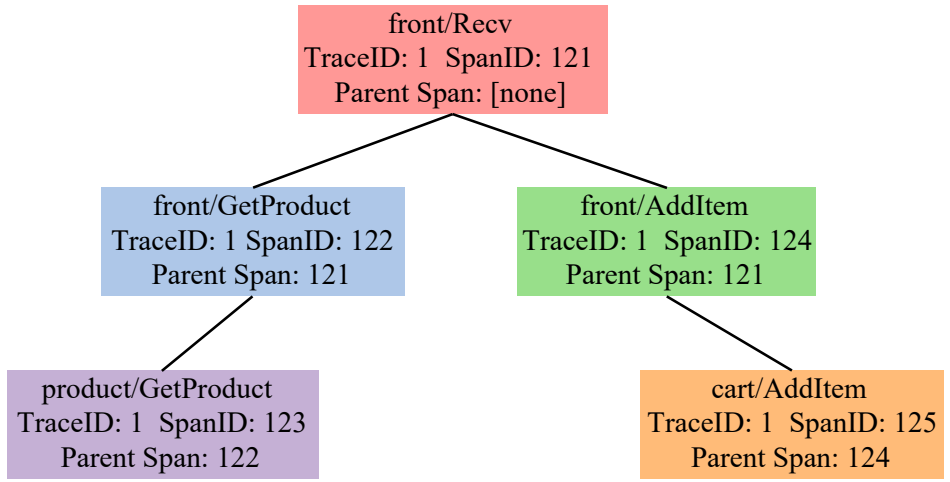


基于Trace的根因定位

● 基于调用链的根因定位

➢ 端到端的请求追踪成为现代微服务系统的默认配置，调用链蕴含丰富的信息对于问题定位作用巨大；





● 基于调用链的根因定位

➢ 已有的工作

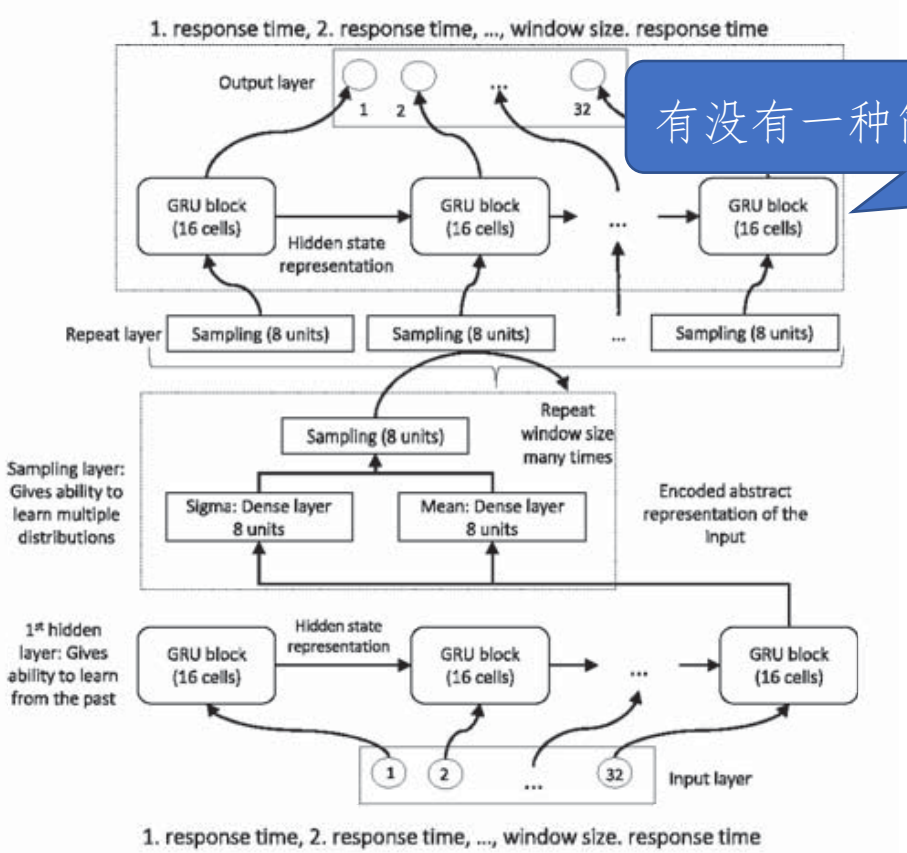
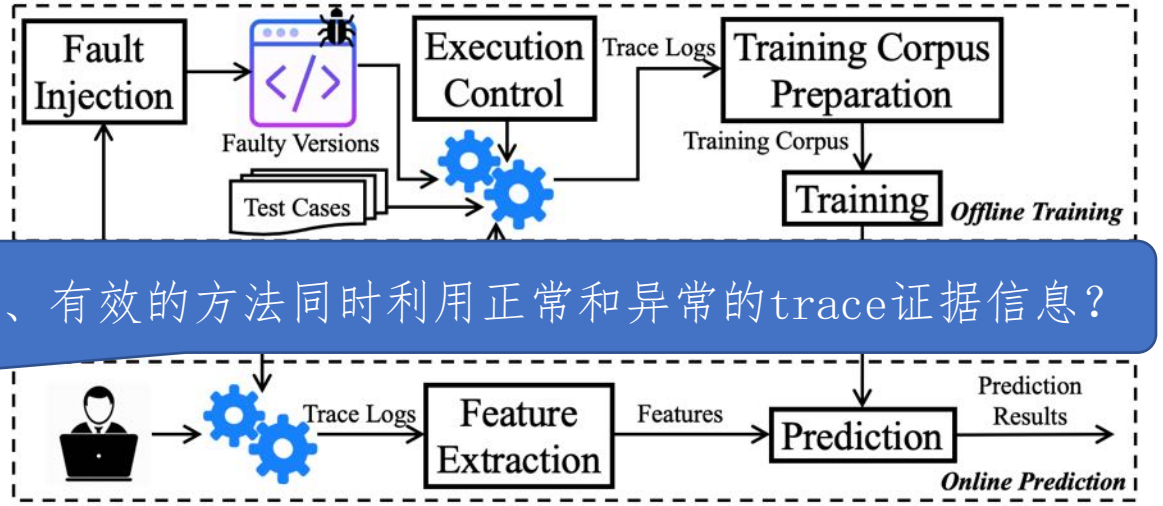


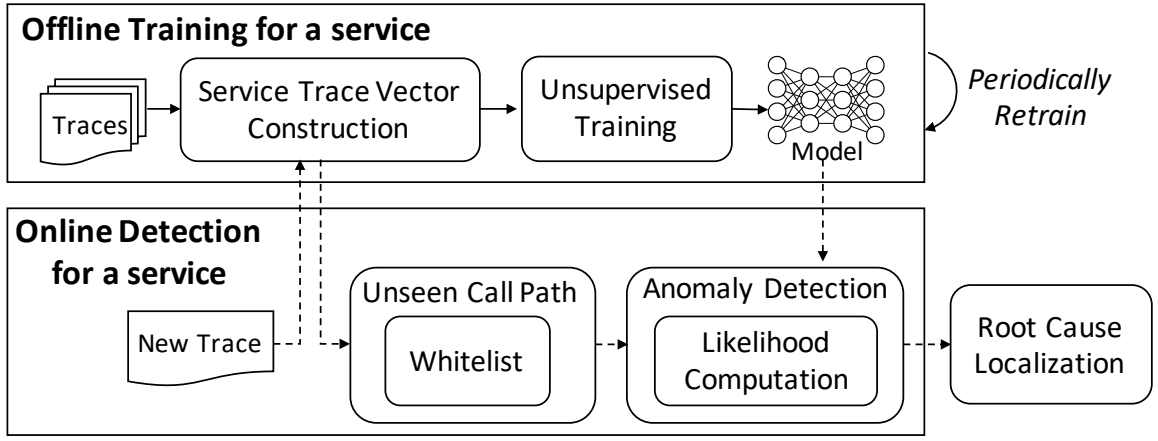
Fig. 3. Model architecture.

Anomaly Detection and Classification using Distributed Tracing and Deep Learning was accepted to CCGrid 2019, 14-17.05, 2019, Cyprus.



有没有一种简单、快速、有效的方法同时利用正常和异常的trace证据信息？

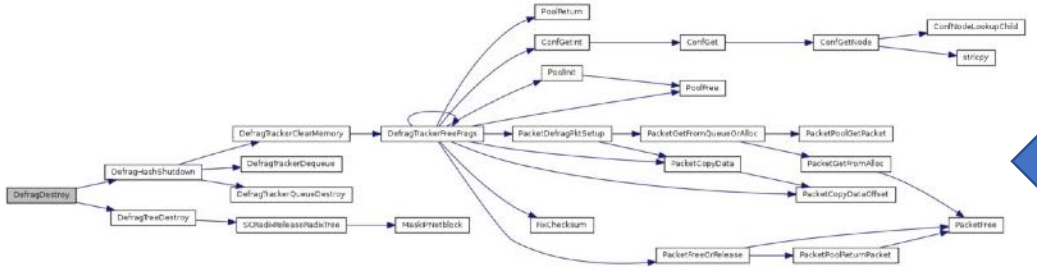
Latent Error Prediction and Fault Localization for Microservice Applications by Learning from System Trace Logs, FSE 2019



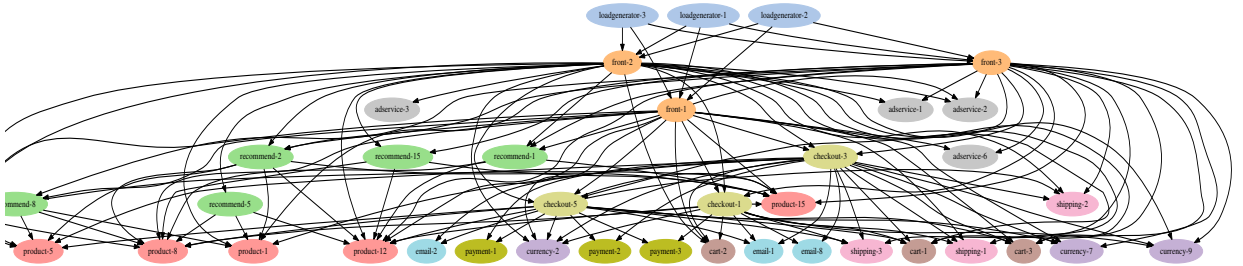
Unsupervised Detection of Microservice Trace Anomalies through Service-Level Deep Bayesian Networks, ISSRE 2020

● 基于调用链的根因定位

➢ 谱分析广泛用于代码故障根因定位，进行debug；



函数调用图



微服务调用图

➢ 利用基于谱分析的方法能够有效定位根因服务实例，但是，难以应对调用链种类稀疏的情况；

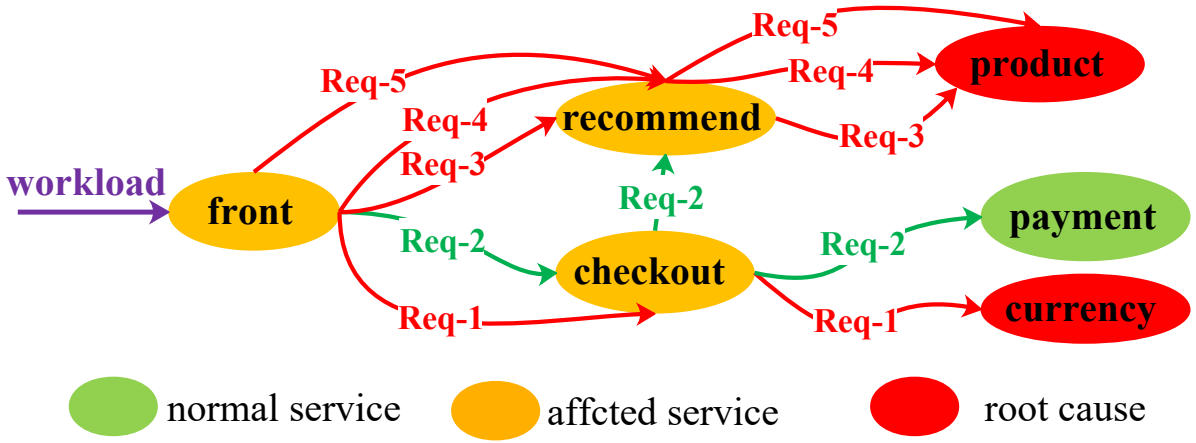
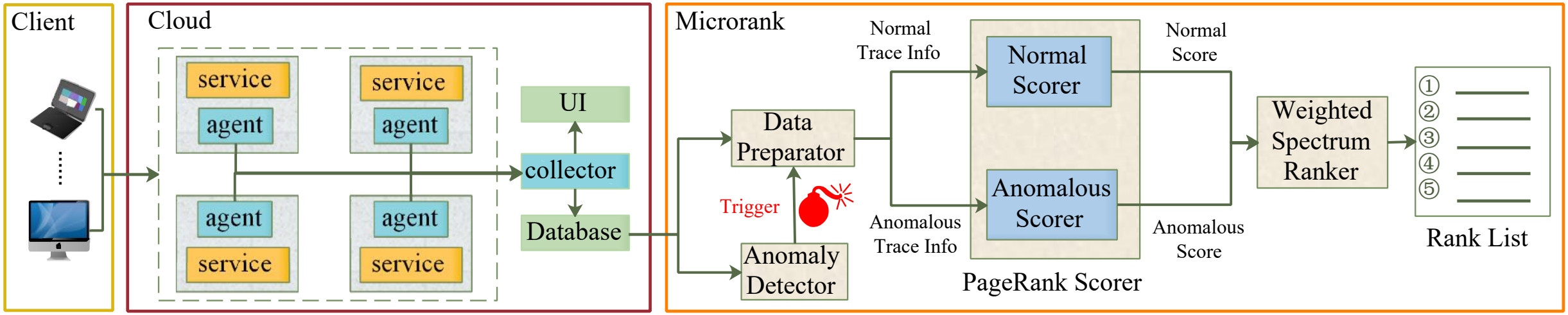


Table 1: Original Spectrum and *MicroRank* score in Fig. 3

	Initial Spectrum(Tarantula)					MicroRank (Tarantula)				
Service	O_{ef}	O_{ep}	O_{nf}	O_{np}	score	O_{ef}	O_{ep}	O_{nf}	O_{np}	score
front	3	1	0	0	0.5	2.97	0.990	0.0	0.0	0.5
checkout	1	1	2	0	0.25	0.328	0.328	0.656	0.198	0.25
recommend	1	0	2	1	1	0.328	0.0	0.686	0.0	0.4
product	3	0	0	1	1	3.0	0.0	0.0	0.0	0.666
payment	0	1	3	0	0	0.0	1.32	0.0	0.0	0.333

MicroRank

➢ MicroRank整合了基于调用链的谱分析以及PageRank方法能够更加准确定位根因，主要创新点为：同时利用正常调用链路和异常调用链路的信息；



➢ 异常Trace检测

1. Timeout；
2. Error code；
3. Performance deviation;

$$L_{excepted} = \sum count_o * (\mu_o + n * \sigma_o)$$

➢ PageRank score

$$v^{(q)} = d \cdot Av^{(q-1)} + (1-d) \cdot u$$

$$A = \begin{bmatrix} \overbrace{A_{oo}}^{operations} & \overbrace{A_{ot}}^{traces} \\ A_{to} & 0 \end{bmatrix}$$

转移概率矩阵

➢ 谱分析+PageRank

$$O_{ef} = F * N_{ef}, \quad O_{nf} = F * (N_f - N_{ef})$$
$$O_{ep} = P * N_{ep}, \quad O_{np} = P * (N_p - N_{ep})$$

MicroRank

- 基于Google的Hipster-shop在超过200个微服务实例上进行了实验；

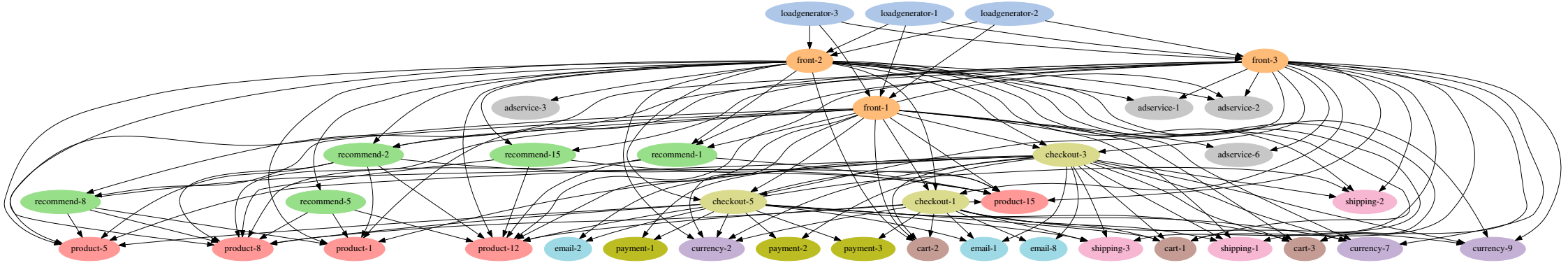


Figure 7: A sample of service instance dependency graph of Hipster-Shop microservice system

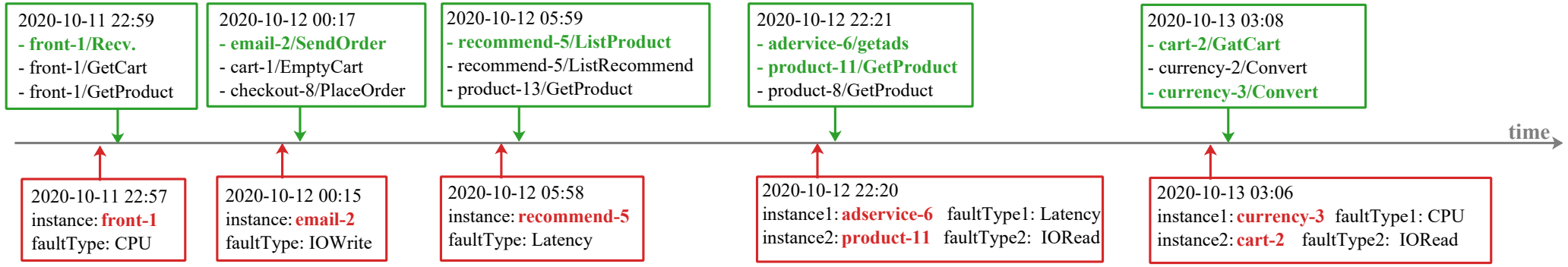
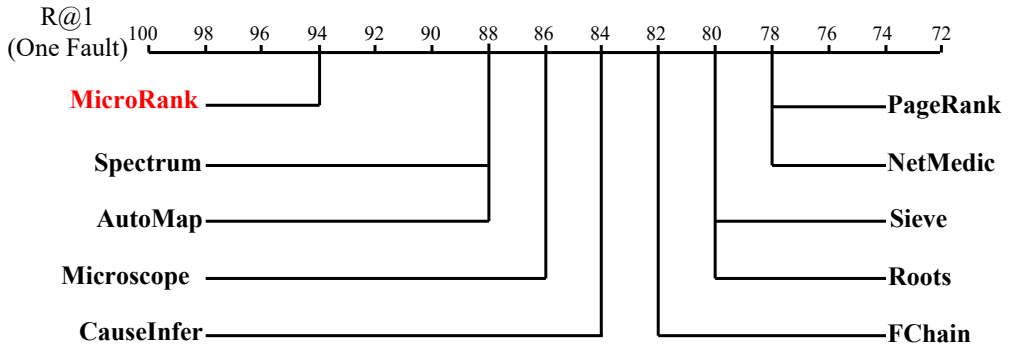


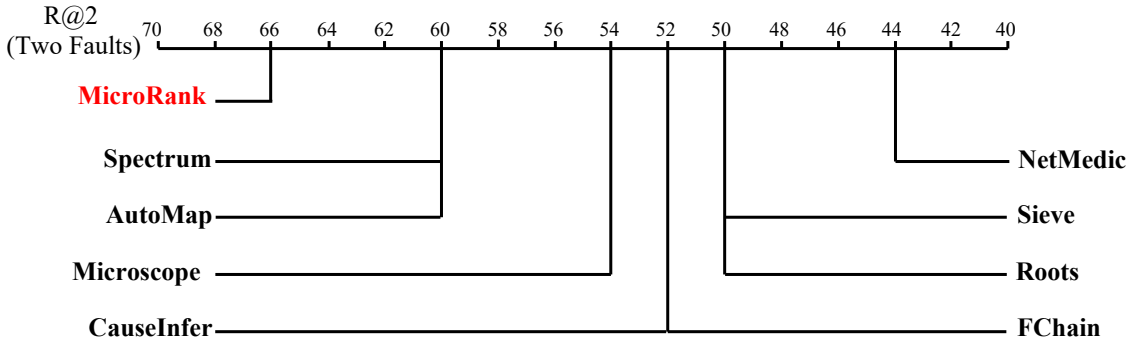
Figure 8: Examples of fault injection and root causes localization on Hipster-Shop

● 实验结果

该方法可用于SDN环境中

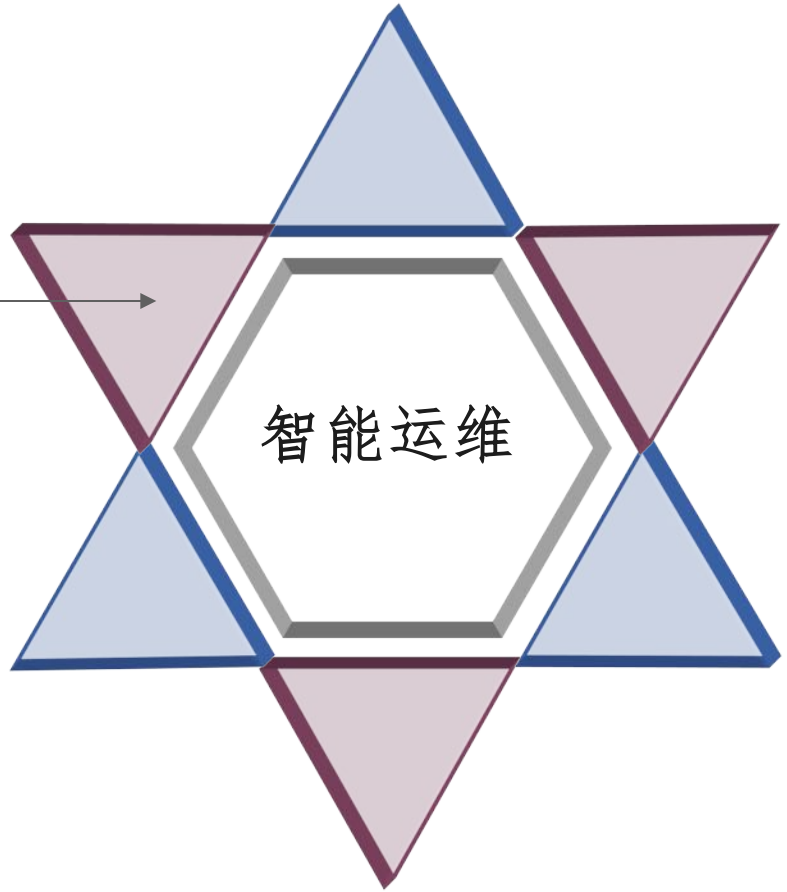
DataSet	Formula	R@1			R@3			R@5			Exam Score				
		PR	SP	MR	PR	SP	MR	PR	SP	MR	PR	SP	MR	IMP*	IMP†
$\mathcal{A}$	Dstar2	78	88	94	78	94	96	92	96	96	1.16	0.56	0.42	63.79%	25.00%
	Goodman	78	86	88	86	92	92	92	92	92	1.16	0.88	0.84	27.59%	4.45%
	Jaccard	78	86	88	86	92	92	90	92	92	1.28	0.88	0.84	34.38%	4.45%
	M2	78	90	94	86	94	96	92	94	96	1.16	0.60	0.42	63.79%	30.00%
	Ochiai	78	88	94	86	94	96	92	96	96	1.16	0.56	0.42	63.79%	25.00%
	Sørensen	78	86	88	86	90	92	92	92	92	1.16	0.88	0.84	34.38%	4.45%
	RussellRao	78	86	92	86	92	96	92	96	96	1.28	0.64	0.44	65.63%	31.25%
	Dice	78	86	88	86	92	92	90	92	92	1.32	0.88	0.84	36.36%	4.45%





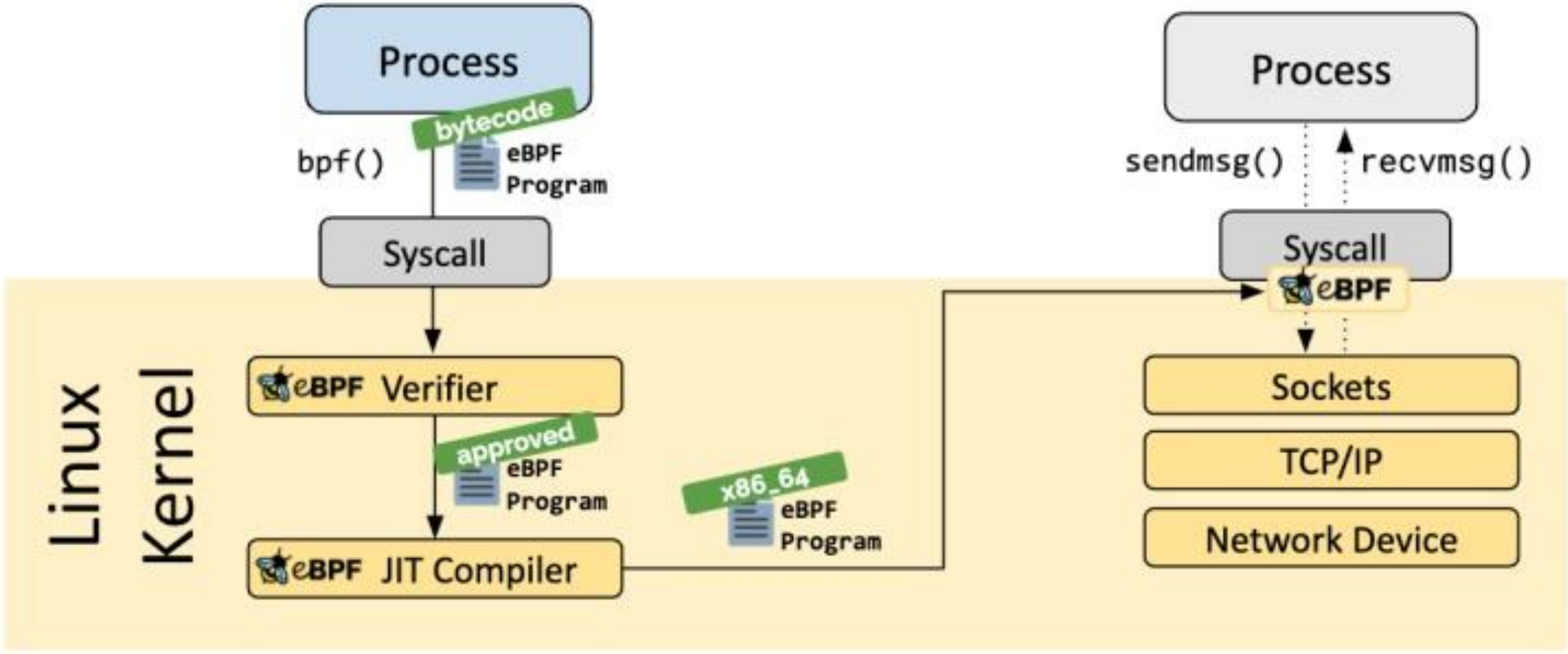
MicroRank: End-to-End Latency Issue Localization with Extended Spectrum Analysis in Microservice Environments, WWW 2020, under review.

面向云原生智能运维趋势



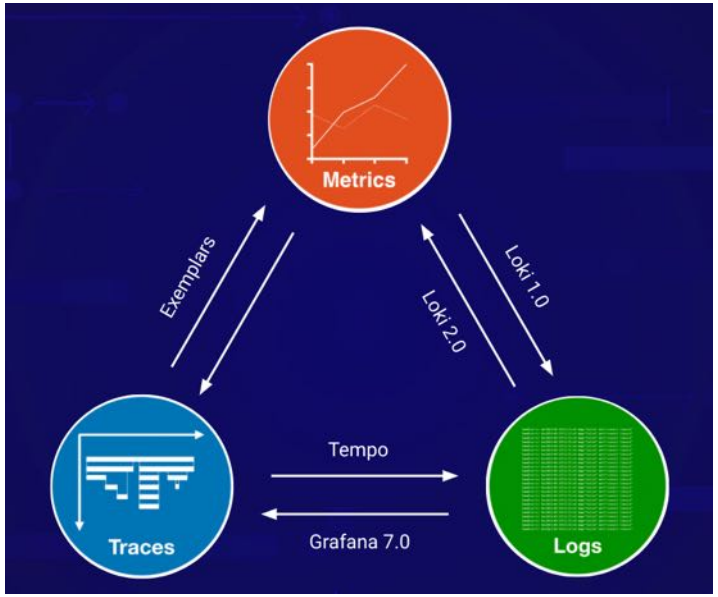
深度“观察”

- 基于eBPF的细粒度系统行为监控；

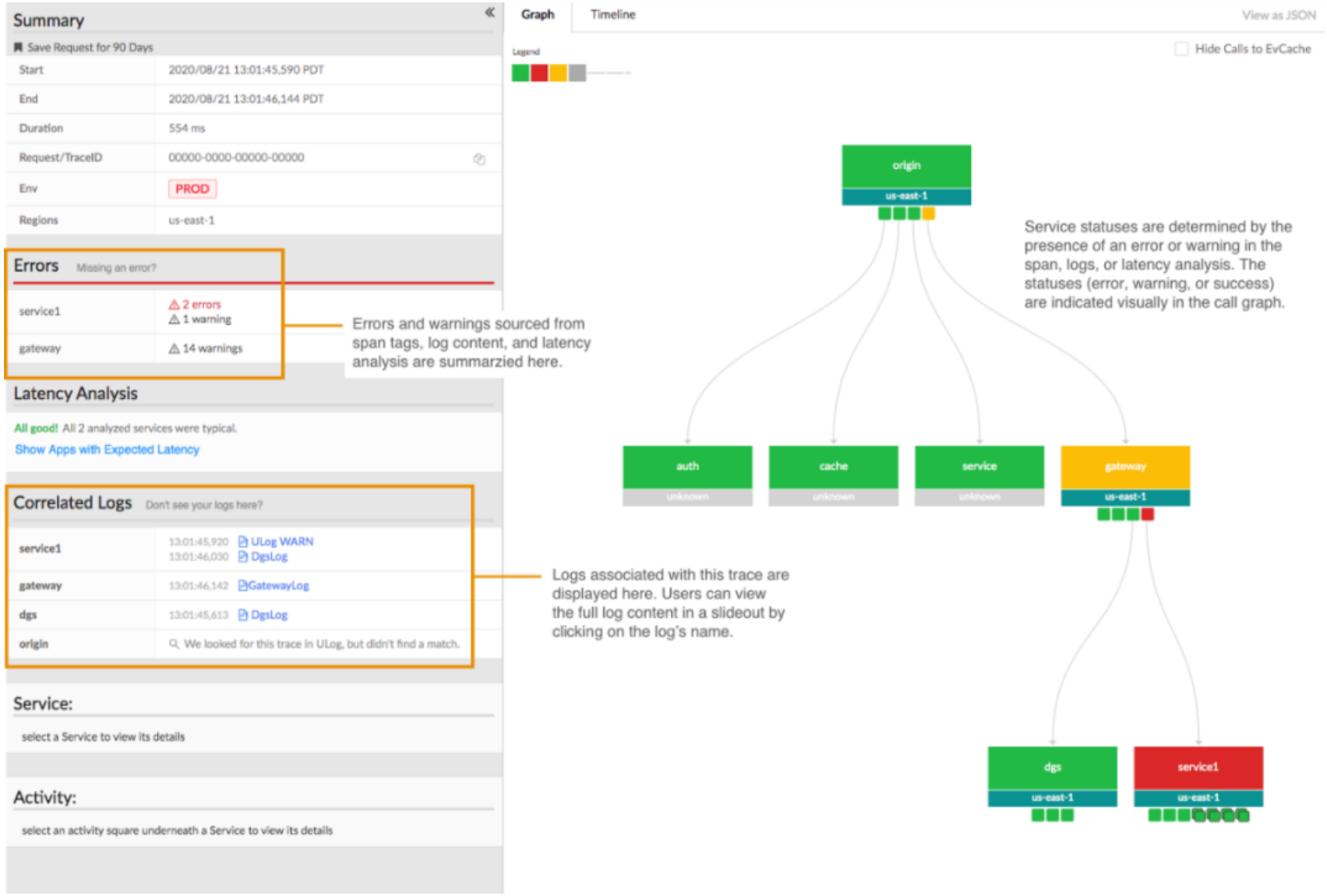


全链路上下文关联

- 将调用链、指标、日志进行关联统一，不仅可以检测异常还可以解释异常；

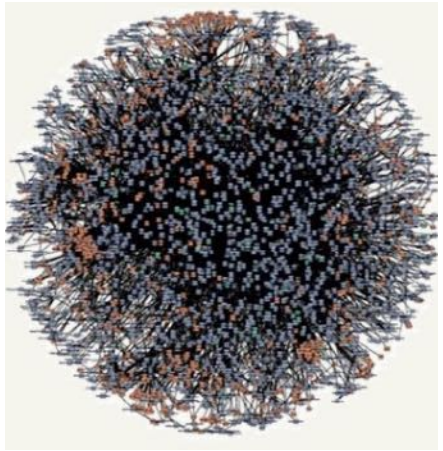


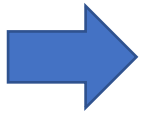
Netflix 全链路上下文关联

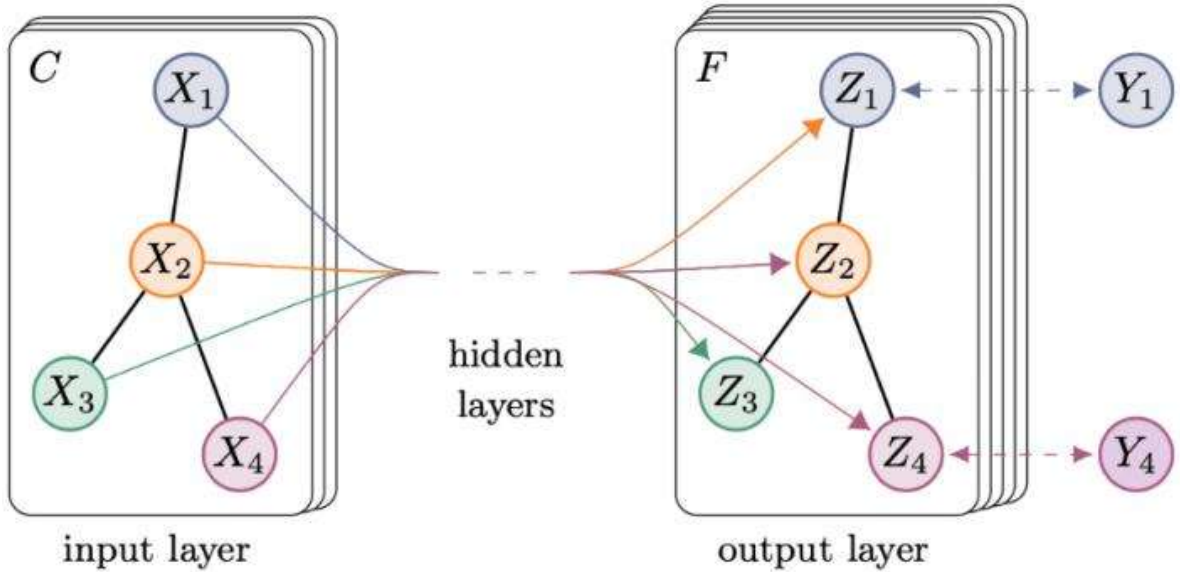


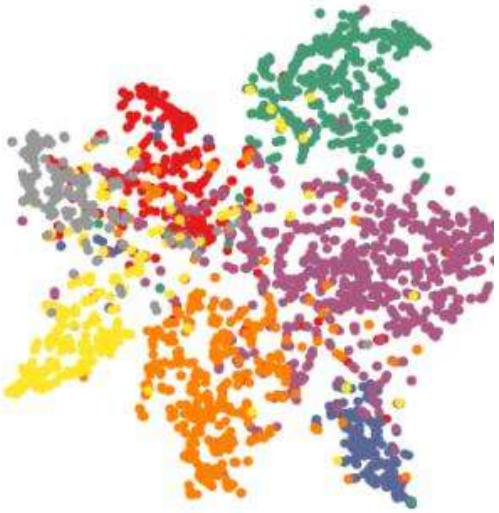
基于图的异常检测

- 基于图的时空结合的神经网络方法会成为云原生系统中检测异常的热门话题；



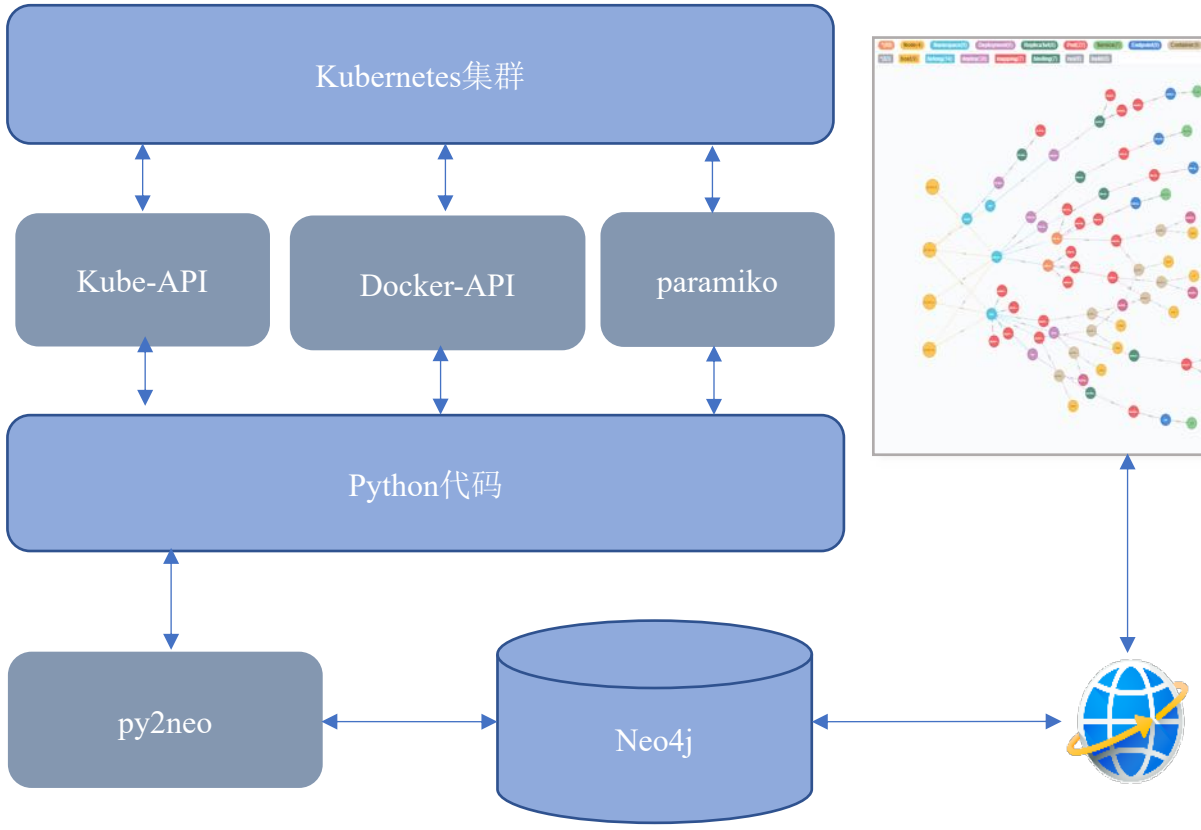


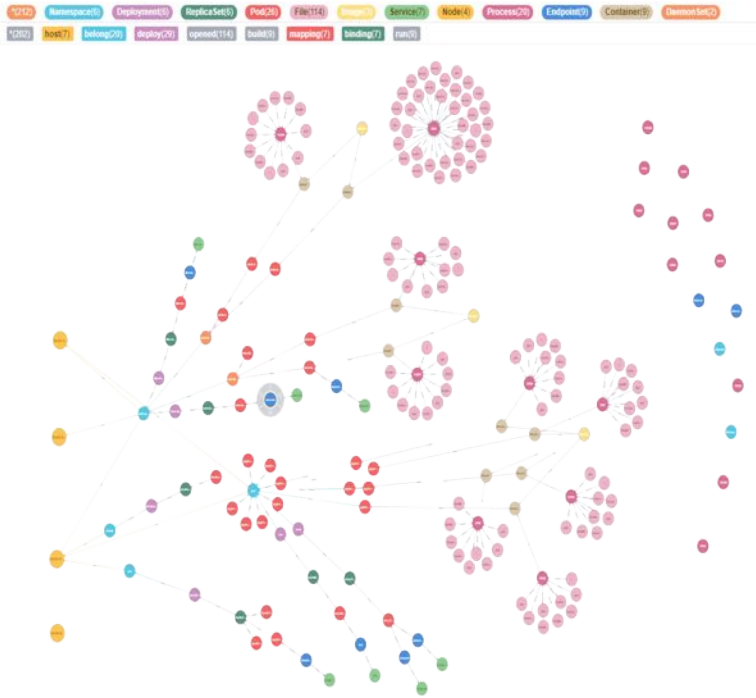




面向云原生的基于知识图谱的智能运维

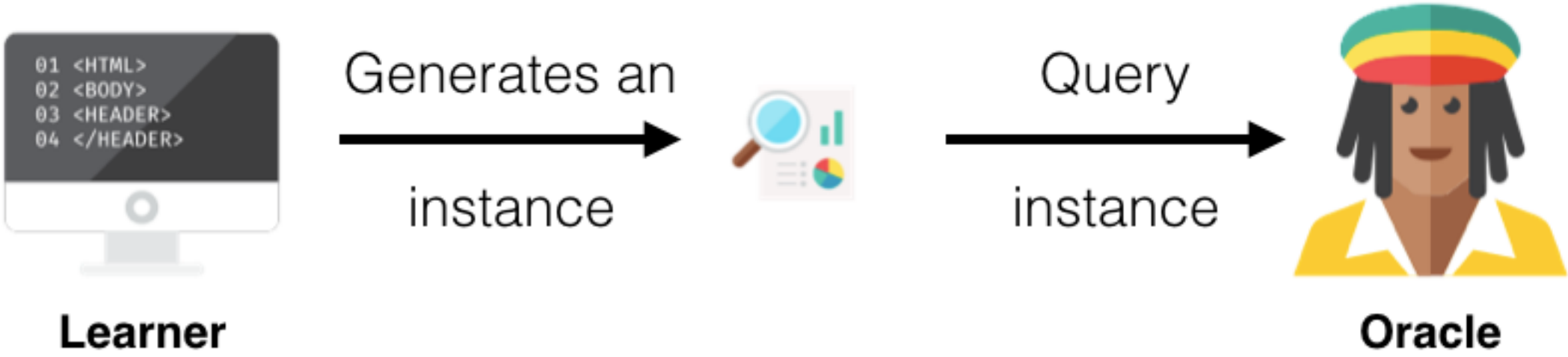
- 云原生提供了新的架构，以及天然的配置中心，为基于知识图谱的智能运维提供了重要基础；





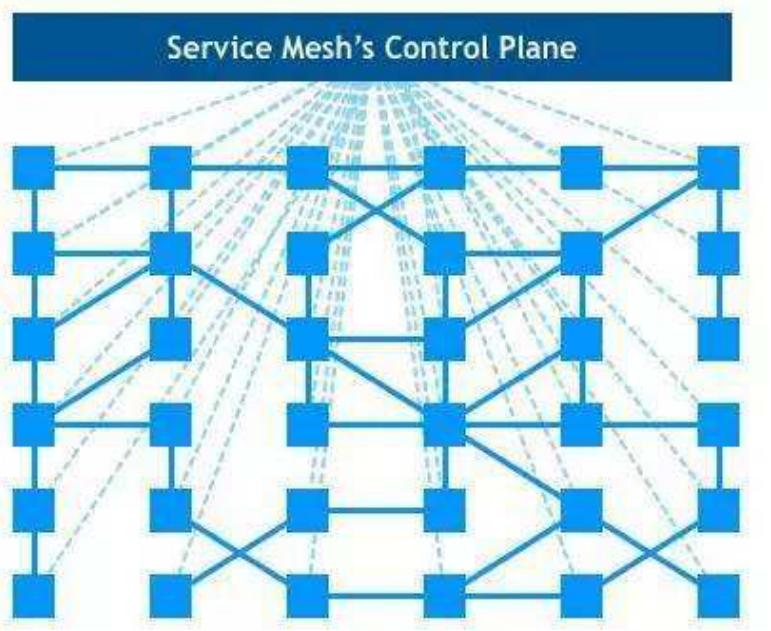
交互式智能运维

- 智能运维正在从无监督走向半监督、交互式运维，算法需要积极的反馈；

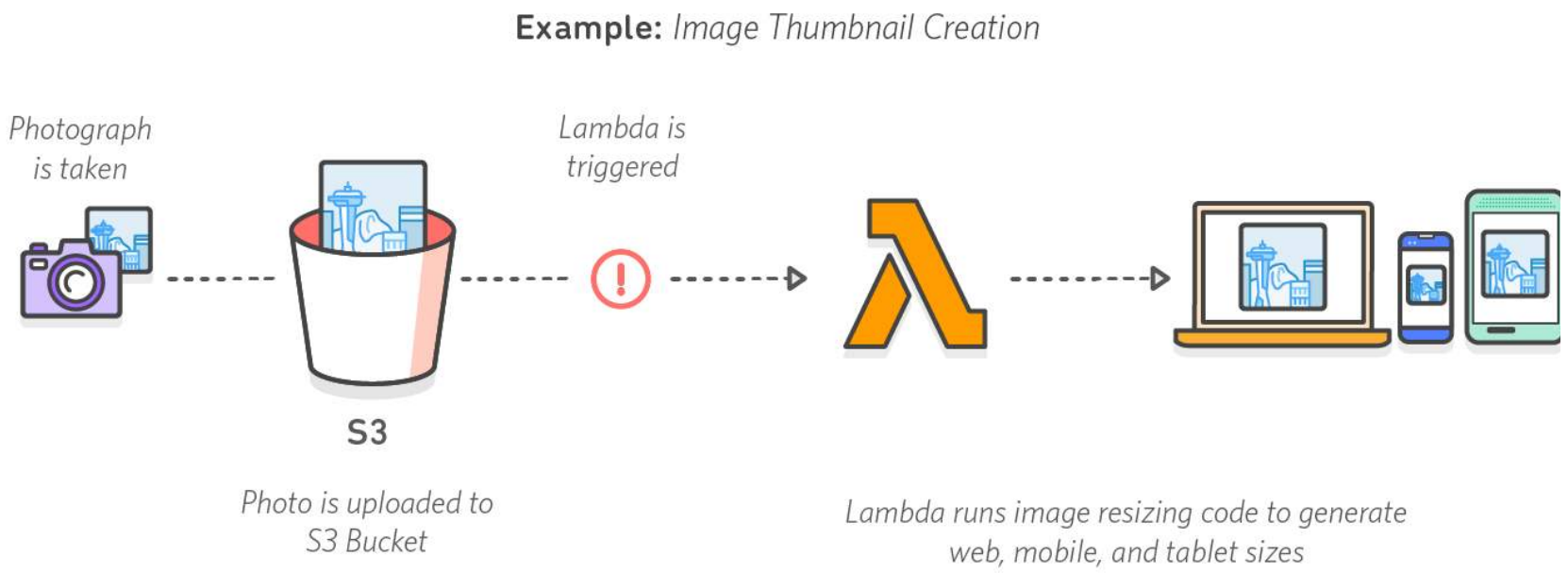


面向新型计算环境的智能运维

- 针对新型软件架构如ServiceMesh、Function as a service的智能运维；

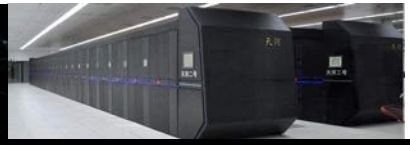


ServiceMesh



FaaS





● 科研项目

1. “Extending FaaS into a Function Delivery Network (FDN)” ——“基于FaaS的函数分发网络”， 中德合作项目（**NSFC-DFG**），300万，已经提交到国家自然基金委，正在评审中，负责人（**NEW**）。

2. “基于人工智能的政务网络精细化管理及精准运维”， 国家重点研发计划课题（课题编号：**2019YFB1804002**），总经费4400万，课题经费**416**万， **2020.01~2022.12**，负责人（**NEW**）。

3. "安全可靠云操作系统关键技术研究及应用示范"，广东省重大项目（项目编号：**2020B010165002**），项目总经费**2000**万，负责课题**300**万，**2020.01~2022.12**，课题负责人（**NEW**）。

4. “面向云原生系统的性能异常检测与定位方法研究"，广东省自然科学基金-面上项目（项目编号：2019A1515012229），10万，2019.10 ~ 2022.10， 主持。

5. “面向容器化微服务架构的软件性能诊断研究"，国家自然基金青年项目（项目编号：61802448），27万，2019.01 ~ 2021.12， 主持。

6. “面向云原生系统端到端性能问题定位与恢复方法研究"，广州市基础与应用基础研究项目，20万，2020.4 ~ 2022.3， 主持。

7. “基于大数据的地方金融安全智能预警与防控系统”，国家自然科学委员会（NSFC）-广东省人民政府大数据科学研究中心项目，课题经费123万，2019.01~2022.12， 子课题负责人。

8. “面向ICT环境的智能运维前沿技术研究”，校企合作，总经费98万，2020.07~2021.07， 主持。

9. “路由器智能运维方法研究”，校企合作，总经费86万，2020.07~2021.07， 主持。

10. “面向深度学习的云计算平台”，校企合作，2020.07~2020.10，主持。

谢谢