



2023 CCF国际AIOps挑战赛决赛  
暨“大模型时代的AIOps”研讨会

# 基于图网络及LLM AGENT的微服务系统异常检测和根因定位方法

崔世彬 亿阳信通股份有限公司

主办单位：中国计算机学会（CCF）、清华大学、中国建设银行股份有限公司、南开大学

承办单位：中国计算机学会互联网专委会、清华大学计算机科学与技术系、中国建设银行股份有限公司运营数据中心、南开大学软件学院、北京必示科技有限公司

赞助单位：华为技术有限公司、国网宁夏电力有限公司电力科学研究院、软通动力信息技术（集团）股份有限公司

## 团队简介

成立于2016年，核心团队由公司CTO领队，多名硕士组成。

### 团队目标:

探索AI能力(包含LLM)在ICT智能运维领域中的落地应用，为行业提供先进的运维算法及实践案例。

### 研发成果:

- (1) 基于BERT和LLM的智能问答系统在多个运营商省份落地
- (2) 智能运维算法（故障检测、根因定位、图像识别等几十项能力）在OSS领域落地
- (3) 发布了MLOps训推一体化平台1.0
- (4) 研发了基于波动贡献度和多尺度移动差分等算法，多年获得国际AIOps挑战赛大奖。

**陈晓峰** 数据分析、算法设计 亿阳信通CTO

**崔世彬** 数据分析、算法设计 数据架构师

**白云飞** 算法设计 AI算法工程师

# 目录

# CONTENTS

第一章节 选题简述

第二章节 数据可视化

第三章节 异常检测

第四章节 根因定位

第五章节 大模型应用

第六章节 总结

## 第一章节

# 选题简述





## 微服务系统复杂

- 微服务系统架构复杂, 当异常出现时, 不能快速发现并且给出根因
- 各类数据 (业务指标、性能指标、调用链、日志等) 特征多种多样, 如何能实现统一化处理。



## 数据复杂

- 数据量比较大, 每天大约30G以上的数据量
- 微服务化后, 调用关系不固定。
- 需要快速给出异常检测结果。
- 需要快速给出根因定位结果。



## 与已有工具的关系

- 现在已有skywalking等调用链跟踪工具, 还要进行异常检测和根因定位的意义
- 节点运行状态, 可以通过业务拨测、skywalking设备监测获得。
- 哪些是上述工具无法实现的



## 大模型应用

如何借助大模型提升AIOps能力:

- Agent
- 思维链
- 生成报告
- 故障分类



## 资源限制

- 前期竞赛服务器资源紧张 (CPU、内存、磁盘)
- GPU等资源紧张, 给部署大模型, 带来难题



## 思路1：通过正常数据训练模型

- ① 通过正常数据进行学习和训练，得到各类数据的特征指标
- ② 对异常数据进行检测和根因定位。

## 思路2：异常检测和根因定位依据

- ① 以TC为主，Metric、Trace、日志为辅，进行异常检测。
- ② 以Metric、Trace和日志为主，进行异常根因定位。

## 思路3：大模型辅助应用

- ① 在资源有限的情况下部署大模型应用
- ② 大模型辅助根因定位

# 方案架构

异常  
检测

TC业务指标



模型训练



时序异常检测



异常数据

根因  
定位

异常数据

性能数据

调用链数据

拓扑数据

日志数据

调用链分析

动态建模

异常检测

时序挖掘

聚合网络

性能指标分析

指标放缩

VAE

可视化判定

联合定位

AIOps

图网络

综合判定

大模型

PROMPT

AGENT

思维链



## 方案创新性

- 1、实现时序异常检测，不需要传统的阈值设置，系统参数迭代优化。
- 2、使用动态建模和图网络进行根因定位
- 3、使用大模型辅助根因定位

## 方案实用性

- 1、模型训练和学习，可使用正常数据训练获得，不需要人工设置规则，便于落地。
- 2、能处理的异常类型多种多样，例如成功率、Full GC、CPU、内存、网络等多种故障，涵盖主要场景。
- 3、大模型部署落地，可以使用CPU方案和快速GPU方案

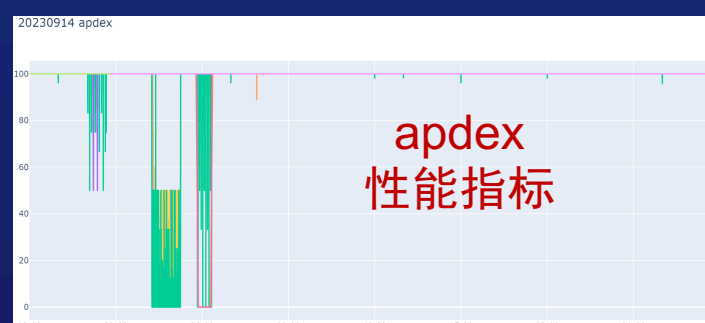
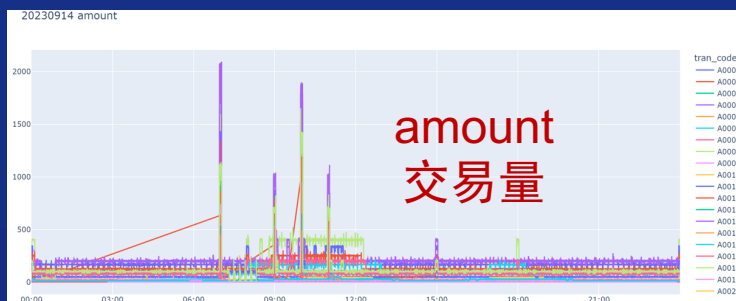




## 第二章节

# 数据可视化

根据每天的TC业务数据，给出图形化展示，便于直观判断是否存在异常。



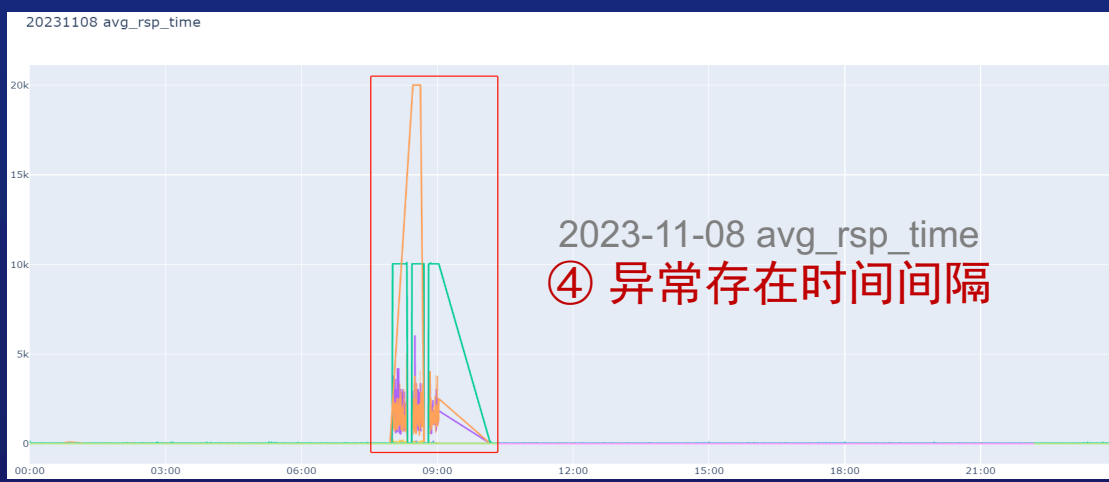
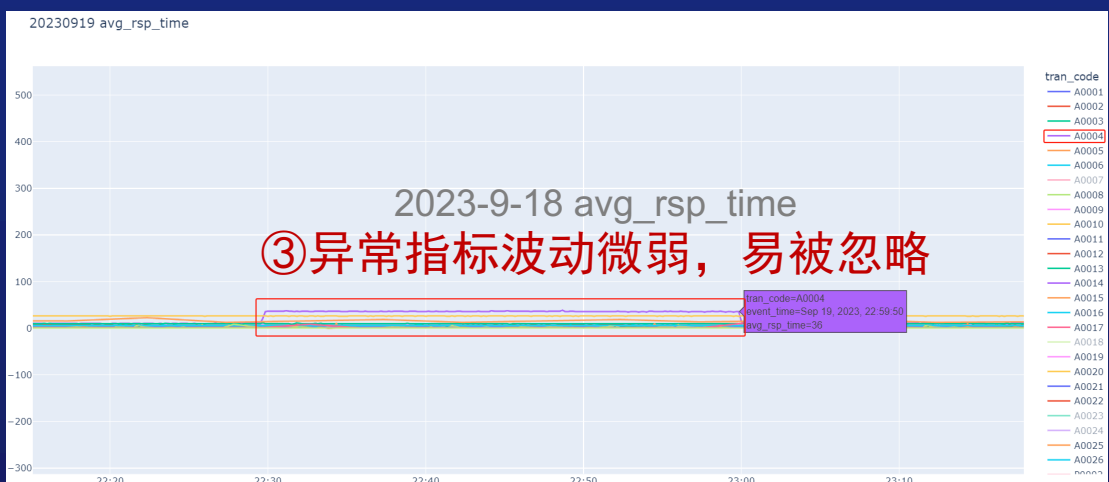
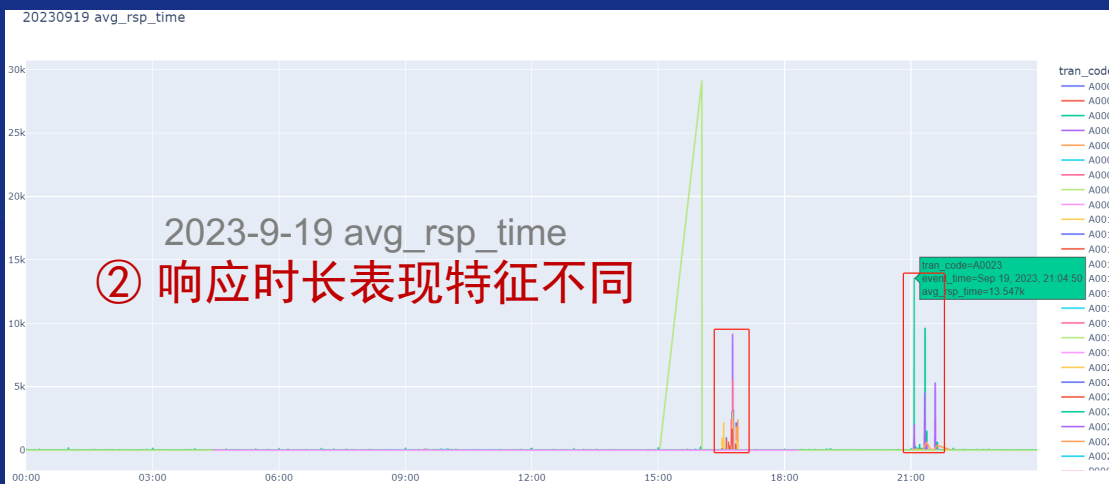
## 第三章节

# 异常检测



# 异常检测-异常类型

存在多种异常类型：例如：① 异常存在间歇性跳跃 ② 响应时长表现特征不同 ③ 异常波动指标微弱。④ 异常存在时间间隔等多种指标异常。



# 异常检测-算法流程



## 9月份结果

idx	dt1	dt2	tran_code	tc_kpi
20230914_1	2023-09-14 02:01:40	2023-09-14 02:40:20	A0023	apdex avg_proc_time avg_rsp_time
20230914_2	2023-09-14 04:15:20	2023-09-14 05:15:20	A0005 A0010 A0023	apdex avg_proc_time avg_rsp_time bus_success_rate sys_success_rate
20230914_3	2023-09-14 05:50:10	2023-09-14 06:20:40	A0013 A0023 A0025 P0013 P0014	apdex avg_proc_time avg_rsp_time bus_success_rate sys_success_rate
20230914_4	2023-09-14 07:29:10	2023-09-14 08:19:30	A0004 A0007 A0010 A0022	bus_success_rate sys_success_rate
20230915_1	2023-09-15 14:51:50	2023-09-15 16:18:40	A0010	bus_success_rate sys_success_rate
20230916_1	2023-09-16 18:02:10	2023-09-16 18:39:50	A0023	apdex avg_proc_time avg_rsp_time
20230916_2	2023-09-16 21:35:50	2023-09-16 22:05:50	A0004	avg_rsp_time
20230917_1	2023-09-17 07:40:10	2023-09-17 08:30:30	A0004 A0010 A0022	avg_rsp_time bus_success_rate sys_success_rate
20230917_2	2023-09-17 14:01:50	2023-09-17 14:31:50	A0004	avg_rsp_time
20230918_1	2023-09-18 17:30:10	2023-09-18 19:14:30	A0001 A0004 A0005 A0010 A0023 A0025	apdex avg_proc_time avg_rsp_time
20230919_1	2023-09-19 16:30:30	2023-09-19 16:53:30	A0001 A0005 A0010	apdex avg_rsp_time
20230919_2	2023-09-19 21:04:20	2023-09-19 21:35:20	A0003 A0006 A0011 A0012 A0013 A0016 A0019 A0020 A0021 A0023 A0024	apdex avg_proc_time avg_rsp_time
20230919_3	2023-09-19 22:29:50	2023-09-19 22:59:50	A0004	avg_rsp_time
20230920_1	2023-09-20 07:14:20	2023-09-20 08:14:30	A0005 A0010 A0023	apdex avg_proc_time avg_rsp_time bus_success_rate sys_success_rate
20230920_2	2023-09-20 22:19:20	2023-09-20 23:19:20	A0001 A0004 A0005 A0010 A0023 A0025	apdex avg_proc_time avg_rsp_time
20230921_1	2023-09-21 04:19:30	2023-09-21 04:58:30	A0023	avg_proc_time avg_rsp_time
20230921_2	2023-09-21 21:31:10	2023-09-21 22:21:20	A0005 A0007 A0010 A0022	avg_rsp_time bus_success_rate sys_success_rate
20230922_1	2023-09-22 06:17:10	2023-09-22 07:17:30	A0001 A0004 A0005 A0010 A0023	apdex avg_proc_time avg_rsp_time
20230922_2	2023-09-22 10:11:20	2023-09-22 12:01:20	A0005 A0010 A0023	apdex avg_proc_time avg_rsp_time bus_success_rate sys_success_rate

## 11月份结果

idx	dt1	dt2	tran_code	tc_kpi
20231103_1	2023-11-03 14:22:20	2023-11-03 15:00:20	A0010	apdex avg_rsp_time
20231103_2	2023-11-03 17:30:10	2023-11-03 19:20:30	A0001 A0004 A0005 A0010	apdex avg_rsp_time
20231104_1	2023-11-04 16:30:10	2023-11-04 17:46:10	A0001 A0004 A0005 A0007 A0010 A0022	apdex avg_rsp_time
20231104_2	2023-11-04 22:29:50	2023-11-04 23:45:50	A0004	avg_rsp_time
20231105_1	2023-11-05 07:14:10	2023-11-05 08:14:10	A0005 A0010 A0023	apdex avg_proc_time avg_rsp_time bus_success_rate sys_success_rate
20231105_2	2023-11-05 22:19:10	2023-11-05 23:54:30	A0001 A0004 A0005 A0010	apdex avg_rsp_time
20231106_1	2023-11-06 04:24:00	2023-11-06 04:48:30	A0023	apdex avg_proc_time avg_rsp_time
20231106_2	2023-11-06 18:19:10	2023-11-06 20:20:50	A0001 A0004 A0005 A0007 A0010 A0022	apdex avg_rsp_time bus_success_rate sys_success_rate
20231106_3	2023-11-06 21:31:10	2023-11-06 22:21:20	A0001 A0004 A0010	avg_rsp_time bus_success_rate sys_success_rate
20231107_1	2023-11-07 04:47:10	2023-11-07 05:37:30	A0001 A0004 A0005 A0007 A0010 A0022	apdex avg_rsp_time bus_success_rate sys_success_rate
20231107_2	2023-11-07 06:17:10	2023-11-07 07:17:20	A0001 A0004 A0005 A0010	apdex avg_rsp_time
20231107_3	2023-11-07 10:11:20	2023-11-07 12:01:20	A0005 A0010 A0023	apdex avg_proc_time avg_rsp_time bus_success_rate sys_success_rate
20231108_1	2023-11-08 02:15:50	2023-11-08 03:00:50	A0004	avg_rsp_time
20231108_2	2023-11-08 08:00:10	2023-11-08 09:02:00	A0001 A0004 A0005 A0010 A0023	apdex avg_proc_time avg_rsp_time
20231109_1	2023-11-09 08:30:10	2023-11-09 09:00:20	A0001 A0004 A0005 A0010 A0023	apdex avg_proc_time avg_rsp_time
20231109_2	2023-11-09 10:00:10	2023-11-09 11:00:20	A0001 A0004 A0005 A0010	apdex avg_rsp_time
20231109_3	2023-11-09 15:30:10	2023-11-09 16:21:00	A0001 A0004 A0005 A0007 A0010 A0022	apdex avg_rsp_time bus_success_rate sys_success_rate
20231110_1	2023-11-10 04:00:20	2023-11-10 04:30:40	A0010 A0013 P0013 P0014	apdex avg_proc_time avg_rsp_time bus_success_rate sys_success_rate
20231110_2	2023-11-10 07:14:20	2023-11-10 07:47:20	A0023 A0024	apdex avg_proc_time avg_rsp_time
20231110_3	2023-11-10 22:10:50	2023-11-10 22:55:50	A0004	avg_rsp_time



## 第四章

# 根因定位

## 业务指标

- 1、判断异常时间点
- 2、给出异常交易码
- 3、给出可能故障网元

## Trace数据

拓扑还原

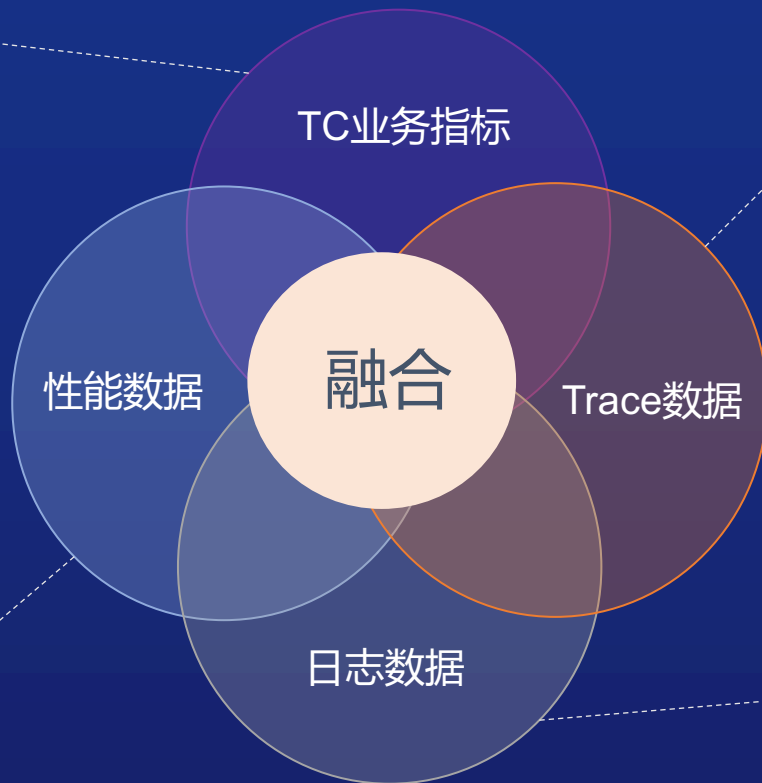
根据时间点和故障网元，找到对应的异常节点、端口、SQL等

## 性能指标

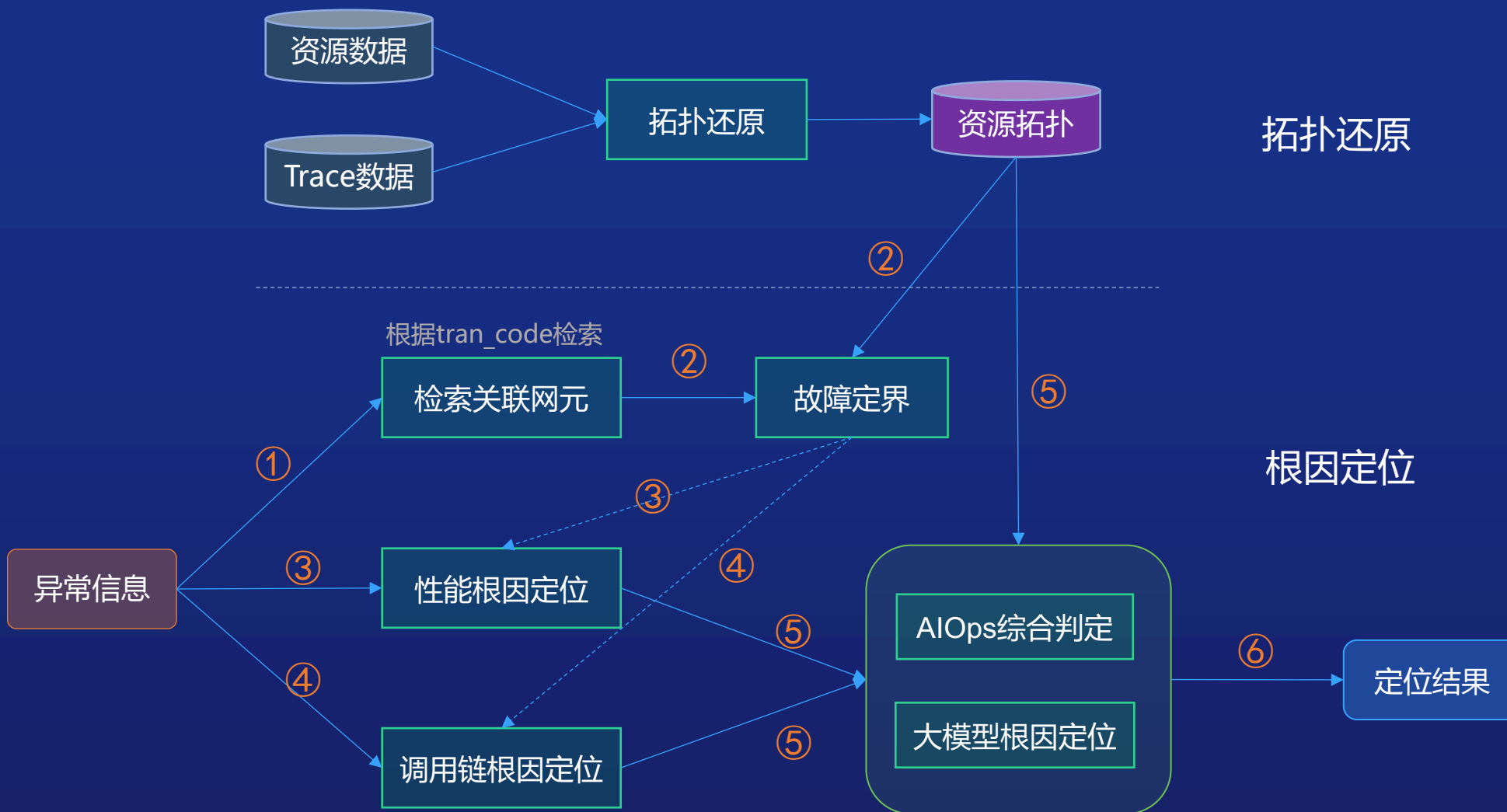
根据时间点和故障网元，找到对应的性能指标

## 日志数据

辅助验证异常根因



# 根因定位-总体流程



# 根因定位-资源拓扑



# 根因定位-基于图网络的调用链根因定位

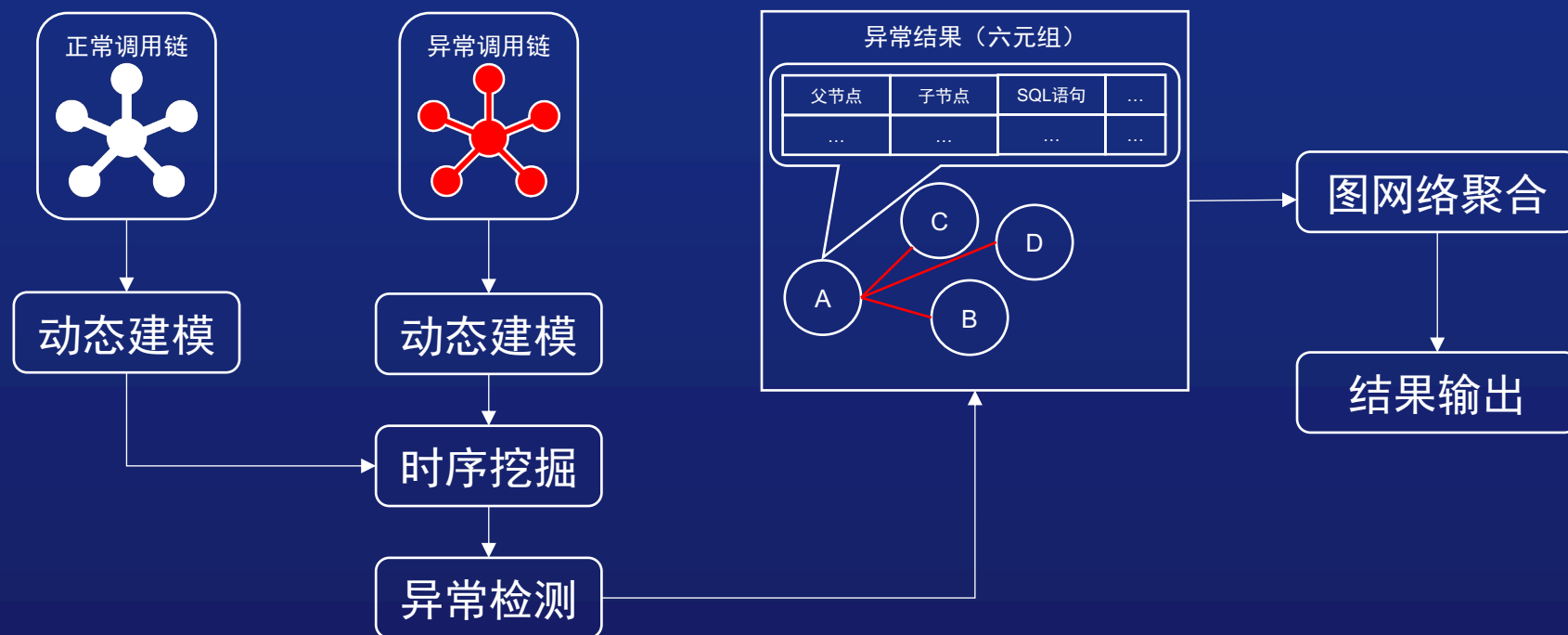
**面临的问题:** 1.trace是动态的 2. 图结构数据挖掘困难 3. 根因节点难以定位

**采用方法:**

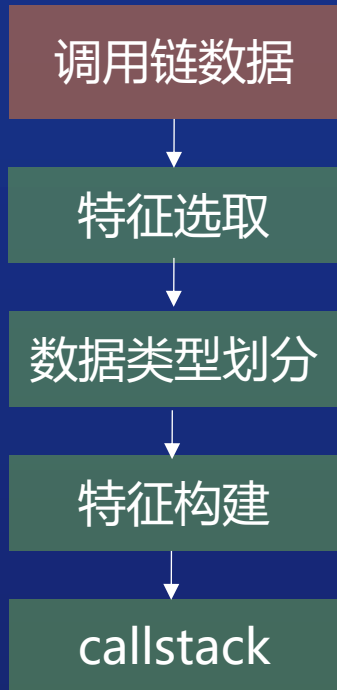
1. 动态建模: 构建六元组进行动态挖掘, 而不是使用调用模板
2. 异常检测: 通过差分获取异常信息指标, 进行异常指标筛选
3. 聚合网络: 追踪异常拓扑节点的信息, 图结构浓缩聚合, 确定根因节点

**技术亮点:**

1. 多元数据维度分析, 不局限于call和called两个指标, 对异常进行全面挖掘。
2. 使用GraphSAGE作为聚合网络, 准确发现异常根因。



基于callstack的动态建模适应于没有静态模板的情况，通过该方法构建动态调用链



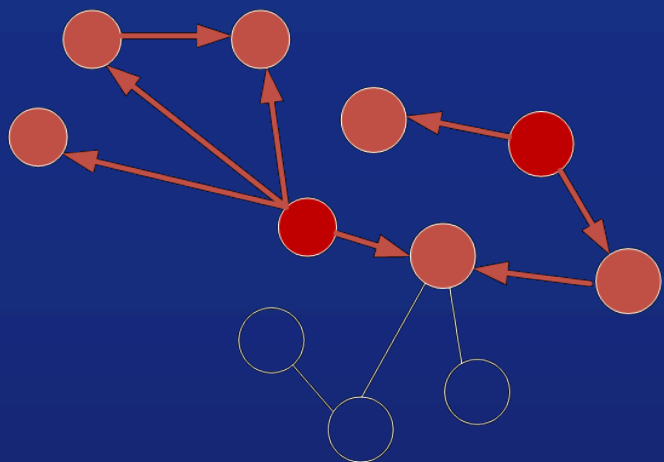
level	call	nginx	called	calledport	tcode	callkind	calloperation
0	0	LLMN_08	nginx_02	Weblogic_28	NaN	A0010	http /coupon/txCtrlPOSTcoupon/jym=
1	1	Weblogic_28	NaN	Weblogic_28	NaN	NaN	internal Render txCtrl
2	1	Weblogic_28	NaN	Weblogic_28	NaN	NaN	internal TxController.doTx
3	2	Weblogic_28	NaN	LLMN_18	8088.0	P0010	http HTTP POST/apibaisc/jym=
4	3	Weblogic_28	nginx_01	Weblogic_11	NaN	P0010	http /apibaisc/txCtrlPOSTapibaisc/jym=
5	4	Weblogic_11	NaN	Weblogic_11	NaN	NaN	internal Render txCtrl
6	⑥	Weblogic_11	②	Weblogic_11	④	⑤	internal TxControler.doTx ⑥
7	5	Weblogic_11	NaN	Oracle_01	11521.0	NaN	oracle SELECT nmcp10_yen00153_24.ccb.com.CLD_SYS_PARA...
8	5	Weblogic_11	NaN	Redis_02	6004.0	NaN	redis GET
9	2	Weblogic_28	NaN	LLMN_18	8088.0	P0001	http HTTP POST/apibaisccoup/jym=
10	3	Weblogic_28	nginx_01	Weblogic_16	NaN	P0001	http /apibaisccoup/txCtrlPOSTapibaisccoup/jym=
11	4	Weblogic_16	NaN	Weblogic_16	NaN	NaN	internal Render txCtrl
12	4	Weblogic_16	NaN	Weblogic_16	NaN	NaN	internal TxController.doTx
13	5	Weblogic_16	NaN	Oracle_01	11521.0	NaN	oracle SELECT nmcp10_yen00153_24.ccb.com.CLD_SHY_WHIT...
14	2	Weblogic_28	NaN	LLMN_18	18088.0	NaN	http HTTP POST/p5mcpp14
15	2	Weblogic_28	NaN	LLMN_18	18088.0	NaN	http HTTP POST/shyms/outBound

callstack: 由⑥的三列: level、callkind、calloperation及各级调用堆栈组成。

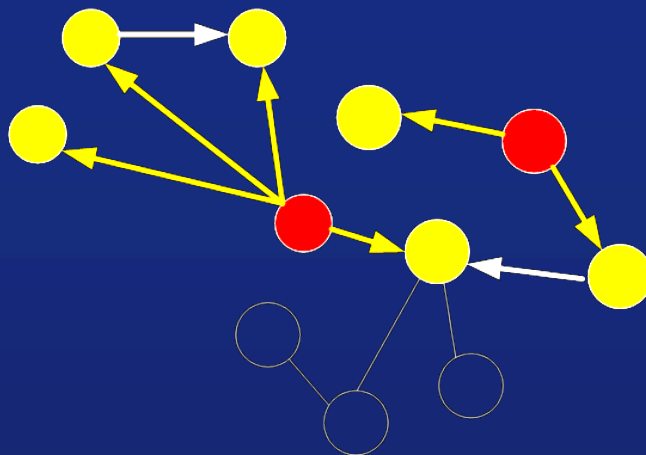
优势: 通过6元组可以唯一标识操作类型。



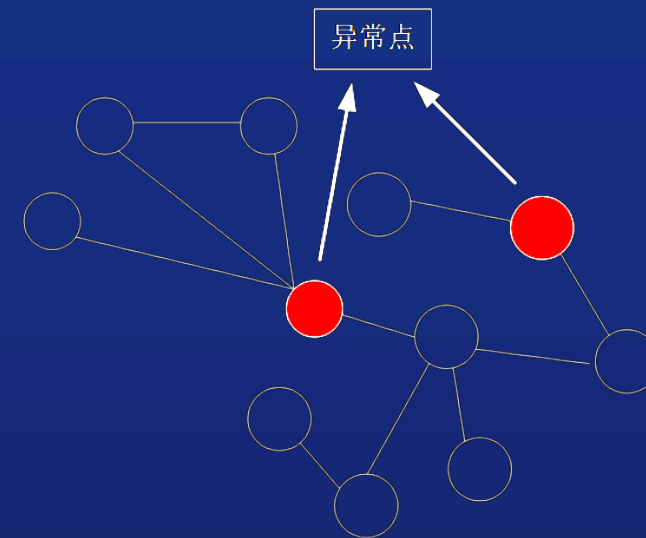
采用GraphSAGE的聚合方式，聚合根因节点周围的异常情况



拓扑异常分布



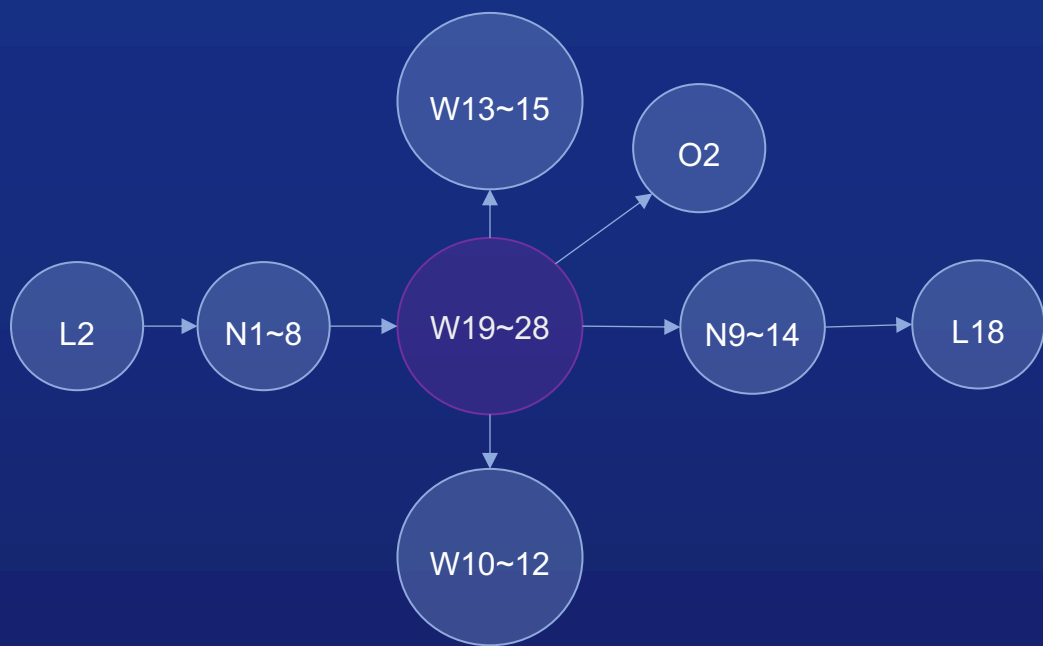
聚合邻近节点



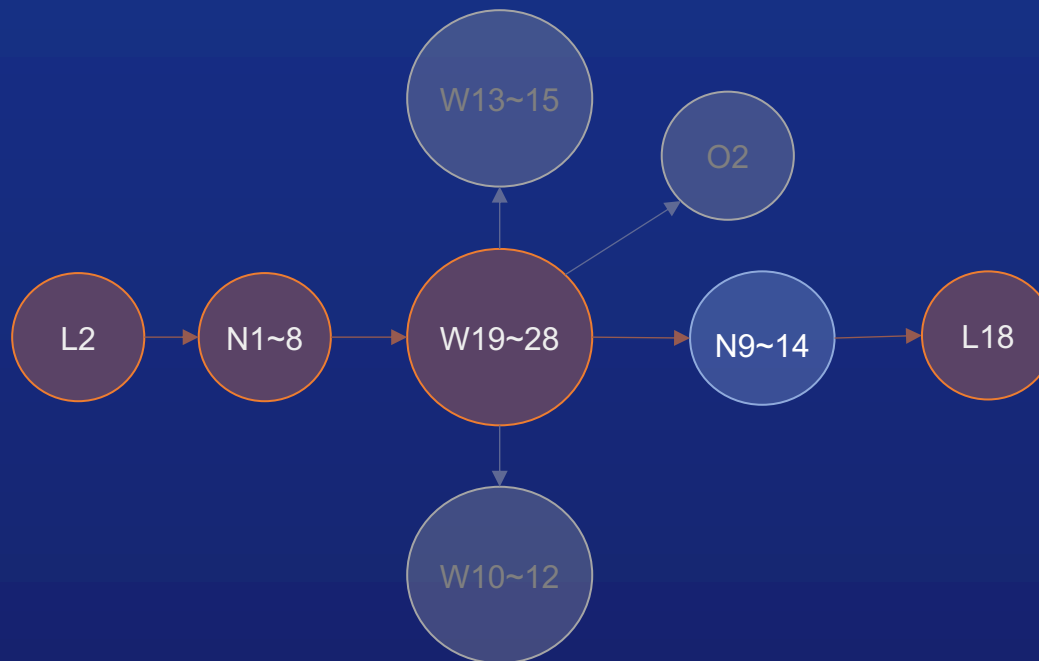
根因节点获取



以9月14日4:15~5:15为例，采用GraphSAGE的聚合方式

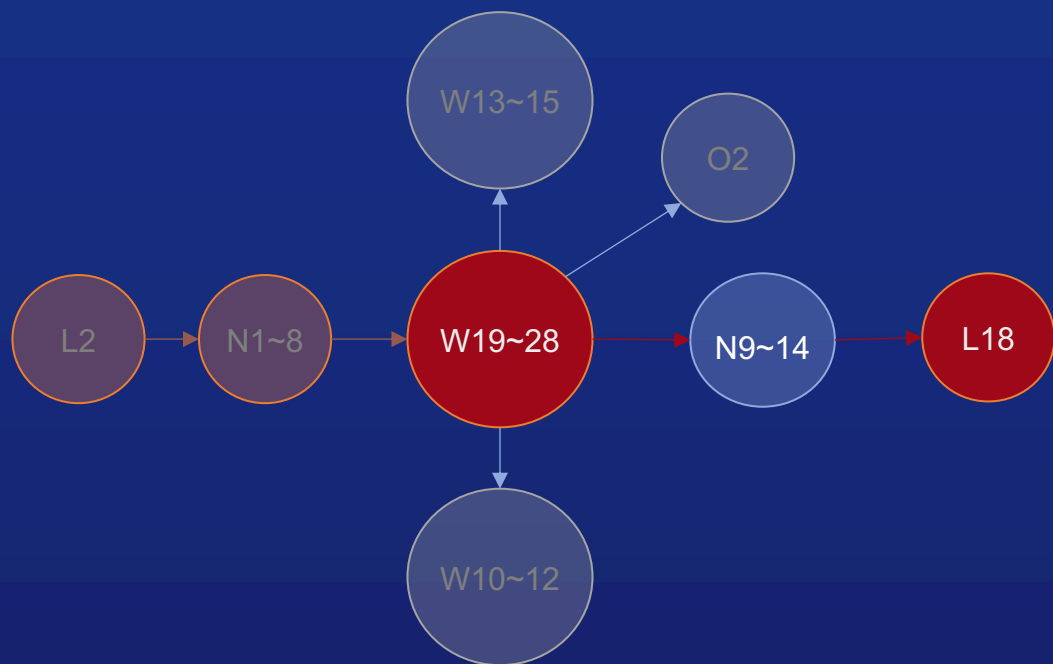


(1) 原始拓扑

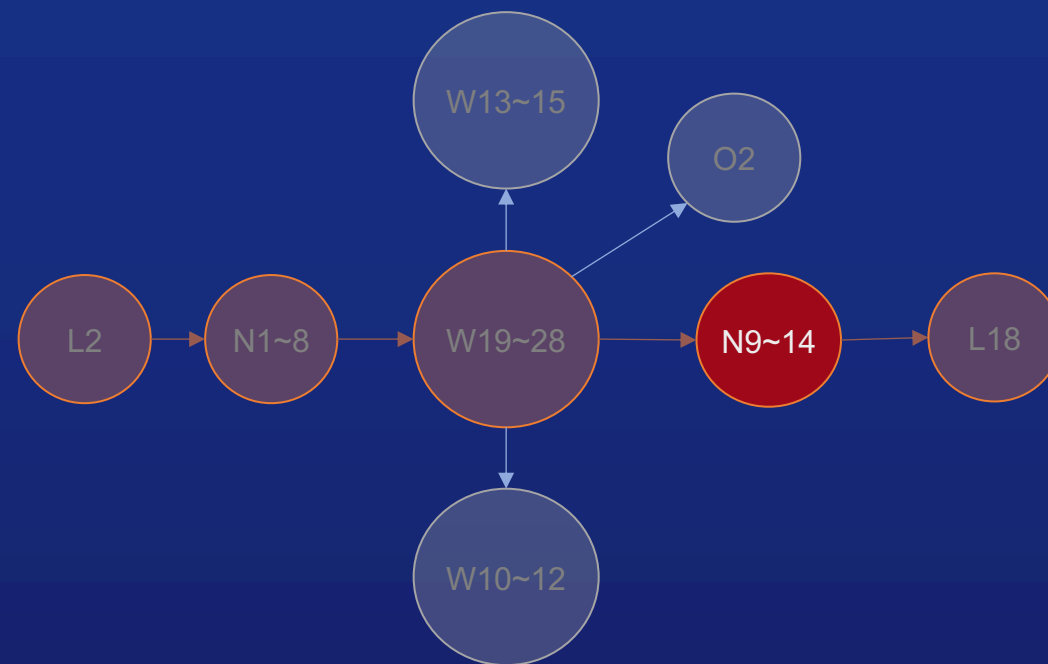


(2) 调用链异常检测

以9月14日4:15~5:15为例，采用GraphSAGE的聚合方式



(3) 聚合网络



(4) 拓扑+性能定位

# 根因定位-算法结果

9月份

idx	dt1	dt2	catalog	cmdb_id	kpi_name
20230914_1	2023-09-14 02:01:40	2023-09-14 02:40:20	FGC	Weblogic_30	system.processes.gc.InfoCount.FGC
20230914_2	2023-09-14 04:15:20	2023-09-14 05:15:20	NET	nginx_09 nginx_10 nginx_11 nginx_12 nginx_13 nginx_14	system.tcp.retrans_pct system.tcp.syn_recv
20230914_3	2023-09-14 05:50:10	2023-09-14 06:20:40	NET	Weblogic_13 Weblogic_14 Weblogic_15	system.tcp.close_wait
20230914_4	2023-09-14 07:29:10	2023-09-14 08:19:30	NET	Weblogic_19 Weblogic_20 Weblogic_21 Weblogic_22 Weblogic_23 Weblogic_24 Weblogic_25	system.tcp.close_wait
20230915_1	2023-09-15 14:51:50	2023-09-15 16:18:40	NET	Weblogic_22 Weblogic_23 Weblogic_24	system.tcp.close_wait
20230916_1	2023-09-16 18:02:10	2023-09-16 18:39:50	FGC	Weblogic_30 Weblogic_31	system.processes.gc.InfoCount.FGC
20230916_2	2023-09-16 21:35:50	2023-09-16 22:05:50	DB	Oracle_02	sql.md5: 7160570d148f231e962e272aed7d3795
20230917_1	2023-09-17 07:40:10	2023-09-17 08:30:30	NET	Weblogic_19 Weblogic_20 Weblogic_21 Weblogic_22 Weblogic_23 Weblogic_24 Weblogic_25	system.tcp.close_wait
20230917_2	2023-09-17 14:01:50	2023-09-17 14:31:50	DB	Oracle_02	sql.md5: 7160570d148f231e962e272aed7d3795
20230918_1	2023-09-18 17:30:10	2023-09-18 19:14:30	DB	Oracle_02	w19、w23、w27
20230919_1	2023-09-19 16:30:30	2023-09-19 16:53:30	FGC	Weblogic_22 Weblogic_23	system.processes.gc.InfoCount.FGC
20230919_2	2023-09-19 21:04:20	2023-09-19 21:35:20	MEM	Weblogic_01 Weblogic_03 Weblogic_04 Weblogic_30 Weblogic_31	system.swap.si system.swap.so
20230919_3	2023-09-19 22:29:50	2023-09-19 22:59:50	DB	Oracle_02	sql.md5: 7160570d148f231e962e272aed7d3795
20230920_1	2023-09-20 07:14:20	2023-09-20 08:14:30	NET	nginx_09 nginx_10 nginx_11 nginx_12 nginx_13 nginx_14	system.tcp.retrans_pct system.tcp.syn_recv
20230920_2	2023-09-20 22:19:20	2023-09-20 23:19:20	DB	Oracle_02	w19、w23、w27
20230921_1	2023-09-21 04:19:30	2023-09-21 04:58:30	FGC	Weblogic_30	system.processes.gc.InfoCount.FGC
20230921_2	2023-09-21 21:31:10	2023-09-21 22:21:20	NET	Weblogic_19 Weblogic_20 Weblogic_21 Weblogic_22	system.tcp.close_wait
20230922_1	2023-09-22 06:17:10	2023-09-22 07:17:30	DB	Oracle_02	w19、w23、w27
20230922_2	2023-09-22 10:11:20	2023-09-22 12:01:20	NET	nginx_09 nginx_10 nginx_11 nginx_12 nginx_13 nginx_14	system.tcp.retrans_pct system.tcp.syn_recv

包含根因分类和根因结果

11月份

idx	dt1	dt2	catalog	cmdb_id	kpi_name
20231103_1	2023-11-03 14:22:20	2023-11-03 15:00:20	CPU	Weblogic_16 Weblogic_17 Weblogic_18	system.cpu.iowait system.cpu.pct_usage
20231103_2	2023-11-03 17:30:10	2023-11-03 19:20:30	DB	Oracle_02	w19、w23、w27
20231104_1	2023-11-04 16:30:10	2023-11-04 17:46:10	FGC	Weblogic_22 Weblogic_23	system.processes.gc.InfoCount.FGC
20231104_2	2023-11-04 22:29:50	2023-11-04 23:45:50	DB	Oracle_02	sql.md5: 7160570d148f231e962e272aed7d3795
20231105_1	2023-11-05 07:14:10	2023-11-05 08:14:10	NET	nginx_09 nginx_10 nginx_11 nginx_12 nginx_13 nginx_14	system.tcp.retrans_pct system.tcp.syn_recv
20231105_2	2023-11-05 22:19:10	2023-11-05 23:54:30	DB	Oracle_02	w19、w23、w27
20231106_1	2023-11-06 04:24:00	2023-11-06 04:48:30	FGC	Weblogic_30 Weblogic_31	system.processes.gc.InfoCount.FGC
20231106_2	2023-11-06 18:19:10	2023-11-06 20:20:50	DB	Oracle_02	w19~w28
20231106_3	2023-11-06 21:31:10	2023-11-06 22:21:20	NET	Weblogic_19 Weblogic_20 Weblogic_21 Weblogic_22	system.tcp.close_wait
20231107_1	2023-11-07 04:47:10	2023-11-07 05:37:30	DB	Oracle_02	w19~w28,w26除外
20231107_2	2023-11-07 06:17:10	2023-11-07 07:17:20	DB	Oracle_02	other
20231107_3	2023-11-07 10:11:20	2023-11-07 12:01:20	NET	nginx_09 nginx_10 nginx_11 nginx_12 nginx_13 nginx_14	system.tcp.retrans_pct system.tcp.syn_recv
20231108_1	2023-11-08 02:15:50	2023-11-08 03:00:50	DB	Oracle_02	sql.md5: 7160570d148f231e962e272aed7d3795
20231108_2	2023-11-08 08:00:10	2023-11-08 09:02:00	DB	Oracle_02 Oracle_05	w23、w25、w27
20231109_1	2023-11-09 08:30:10	2023-11-09 09:00:20	DB	Oracle_02 Oracle_04 Oracle_05	w19~w28、w24除外
20231109_2	2023-11-09 10:00:10	2023-11-09 11:00:20	DB	Oracle_02	w19、w23、w27
20231109_3	2023-11-09 15:30:10	2023-11-09 16:21:00	DB	Oracle_02	w19~w28、w24除外
20231110_1	2023-11-10 04:00:20	2023-11-10 04:30:40	NET	Weblogic_13 Weblogic_14 Weblogic_15	system.tcp.close_wait
20231110_2	2023-11-10 07:14:20	2023-11-10 07:47:20	FGC	Weblogic_30 Weblogic_31	system.processes.gc.InfoCount.FGC
20231110_3	2023-11-10 22:10:50	2023-11-10 22:55:50	DB	Oracle_02	sql.md5: 7160570d148f231e962e272aed7d3795

包含根因分类和根因结果

## 第五章节

# 大模型应用



## 应用场景

- 1、Agent能力运用（调用前面的能力实现**异常检测**和**根因定位**）
- 2、故障根因定位，通过**思维链**进行辅助根因定位

## 大模型部署

- 1、模型本地化部署，本次部署使用**CPP CPU**方案
- 2、建议使用**CPP+GPU**方案，降低部署成本，部署推理效率提升10倍以上
- 3、LangChain+本地知识库





## Example

问：案例库中的指标分析发现的异常数据为{data1}，调用链分析发现的数据为{data2}，观察指标异常值数据的v\_err、v\_pre、v\_pst和trend列，可以根据指标异常值数据的指标突变值情况看出异常指标为outlier={result}，该数据包含kpi\_names和cmdb\_ids两列。再将outlier在调用链数据中进行查询，并结合调用链数据的指标v\_err、v\_pre、v\_pst和trend进行判断，确定出调用链数据的异常值组合为outlier2={result2}，该组合包含cmdb\_ids和ext\_net\_peer\_name两列。

现在指标分析发现的异常数据为{data\_now1}，调用链分析发现的数据为{data\_now2}，请根据以上案例进行分析并输出outlier1和outlier2。

Ps:上述问题中红色的数据由案例库给出，黄色的数据由指标分析结果和调用链分析结果给出

调用工具

定制报告

系统对接

问题分解

...



定义了两个工具，**网元状态工具**和**故障根因分析工具**。下面是工具的调用情况：

 请检查网元Oracle\_02的状态

 网元Oracle\_02的状态异常。

✓ 思考过程 ^

我应该使用网元状态工具来查询网元的状态。 Action: 网元状态工具 Action Input: Oracle\_02 Observation: 网元Oracle\_02的状态是正常的。 Thought: 我现在可以作答了。网元Oracle\_02的状态是正常的。 Final Answer: 网元Oracle\_02的状态是正常的。

工具名称：网元状态工具

工具状态：调用成功

工具输入：Oracle\_02

工具输出：网元Oracle\_02状态异常

我现在可以作答了。 Final Answer: 网元Oracle\_02的状态异常。

检查网元状态

 请给出2023-11-04 22:29:00 到2023-11-04 23:46:00的系统故障根因

 2023-11-04 22:29:00 到2023-11-04 23:46:00的系统故障根因是网元Oracle\_02的相关故障引起的。

✓ 思考过程 ^

问题与历史记录中的故障无关，我需要使用故障根因分析工具来查询这个时间段的故障。 Action: 故障根因分析工具 Action Input: 2023-11-04 22:29:00 到2023-11-04 23:46:00 Observation: 故障根因: 系统相关故障引起的 Thought: 我现在可以作答了。 Final Answer: 2023-11-04 22:29:00 到2023-11-04 23:46:00的系统故障根因是系统相关故障引起的。

工具名称：故障根因分析工具

工具状态：调用成功

工具输入：2023-11-04 22:29:00 到2023-11-04 23:46:00

工具输出：2023-11-04 22:29:00 到2023-11-04 23:46:00 故障根因：网元Oracle\_02

我现在知道最终答案了。 Final Answer: 2023-11-04 22:29:00 到2023-11-04 23:46:00的系统故障根因是网元Oracle\_02的相关故障引起的。

定位故障根因

问：案例库中的指标分析发现的异常数据data1为

	cmdb_id	kpi_name	v_err	v_pre	v_pst	trend
153	Weblogic_33	system.io.avg_rq_sz	14.299	5.037	8.767	up
154	Weblogic_33	system.io.wkb_s	21.116	13.150	11.450	up
155	Weblogic_33	system.net.bytes_rcvd	9944.388	3799.196	5579.503	up
156	Weblogic_33	system.net.packets_in.count	26.141	16.420	21.967	up
157	Weblogic_33	system.tcp.time_wait	140.428	82.666	109.833	up
158	Weblogic_34	system.tcp.time_wait	114.800	67.333	101.333	up
159	Weblogic_34	system.io.avg_rq_sz	5.566	13.162	32.327	down
160	Weblogic_34	system.io.await	0.160	0.390	0.827	down
161	Weblogic_34	system.io.w_await	0.160	0.390	0.827	down
162	Weblogic_34	system.io.w_s	1.033	2.766	3.800	down
163	Weblogic_34	system.io.wkb_s	5.933	18.400	50.866	down

性能指标数据

调用链分析发现的数据data2为

	cmdb_id	ext_net_peer_name	v_err	v_pre	v_pst	trend
Weblogic_33	nginx_05		46.50	14.00	0.00	up
	nginx_07		7.33	0.00	7.00	up
Weblogic_34	nginx_05		7.00	0.00	0.00	up
	nginx_07		46.50	0.00	86.00	up
	nginx_08		64.00	0.00	0.00	up

调用链分析数据

观察指标异常值数据的v\_err、v\_pre、v\_pst和trend列，可以根据指标异常值数据的指标突变值情况看出异常指标为

outlier1=[[Weblogic\_33, Weblogic\_34],[system.tcp.time\_wait]],

该数据包含kpi\_names和cmdb\_ids两列。再将outlier在调用链数据中进行查询，

并结合调用链数据的指标v\_err、v\_pre、v\_pst和trend进行判断，

确定出调用链数据的异常值组合为outlier2=[[Weblogic\_33, Weblogic\_34], [nginx\_03, nginx\_05, nginx\_06, nginx\_07, nginx\_08]],

该组合包含cmdb\_ids和ext\_net\_peer\_name两列。

指导如何得出结论

现在指标分析发现的异常数据now1为

	cmdb_id	kpi_name	v_err	v_pre	v_pst	trend
101	Weblogic_16	system.io.avg_rq_sz	2.803	11.246	22.030	down
102	Weblogic_16	system.io.w_s	1.333	7.300	3.333	down
103	Weblogic_16	system.io.wkb_s	6.316	39.100	57.166	down

给出待分析数据

调用链分析发现的数据now2为

	cmdb_id	ext_net_peer_name	v_err	v_pre	v_pst	trend
Weblogic_16	nginx_07		1.013605	0.00	1.01	up
	nginx_08		1.011905	0.00	1.00	up

请根据以上案例进行分析并输出now1和now2的outlier1和outlier2。



```
h You
问：案例库中的指标分析发现的异常数据data1为
cmdb_id      kpi_name      v_err  v_pre  v_pst trend
153 Weblogic_33 system.io.avg_rq_sz 14.299 5.037333 8.767 up
154 Weblogic_33 system.io.wkb_s 21.116667 13.15 11.45 up
155 Weblogic_33 system.net.bytes_rcvd 9944.388 3799.196 5579.503 up
156 Weblogic_33 system.net.packets_in.count 26141 16.42033 21.967 up
157 Weblogic_33 system.top.time_wait 140.428571 82.666 109.8333 up
158 Weblogic_34 system.tcp.time_wait 114.800 67.333 101.3333 up
159 Weblogic_34 system.io.avg_rq_sz 5.56633 13.1623 32.327 down
160 Weblogic_34 system.io.await 0.160667 0.390667 0.827 down
161 Weblogic_34 system.io.w_s 0.160667 0.390667 0.827 down
162 Weblogic_34 system.io.w_s 1.033333 2.766667 3.800 down
163 Weblogic_34 system.io.wkb_s 5.933333 18.4 50.866667 down
，调用链分析发现的数据data2为
v_err  v_pre  v_pst trend
cmdb_id ext_net_peer_name
Weblogic_33 nginx_05 46.5 14.0 0.0 up
nginx_07 7.33 0.00 7.00 up
Weblogic_34 nginx_05 7.00 0.00 0.00 up
nginx_07 46.5 0.00 86.00 up
nginx_08 64.00 0.00 0.00 up
，观察指标异常值数据的v_err、v_pre、v_pst和trend列，可以根据指标异常值数据的
指标突变值(情况差)出异常指标为outlier=[[Weblogic_33, Weblogic_34],
[system.top.time_wait]]，该数据包含kpi_names和cmdb_ids两列，再将outlier在调用链
数据中进行查询，并结合调用链数据的指标v_err、v_pre、v_pst和trend进行判断，确
定出调用链数据的异常值组合为outlier2=[[ Weblogic_33, Weblogic_34], [nginx_03,
nginx_05, nginx_06, nginx_07, nginx_08]]，该组合包含cmdb_ids和ext_net_peer_name
两列。
现在指标分析发现的异常数据now1为
cmdb_id      kpi_name      v_err  v_pre  v_pst trend
101 Weblogic_16 system.io.avg_rq_sz 2.803 11.246333 22.030667 down
102 Weblogic_16 system.io.w_s 1.333 7.30 3.333333 down
103 Weblogic_16 system.io.wkb_s 6.3166 391 57.166667 down
，调用链分析发现的数据now2为
v_err  v_pre  v_pst trend
cmdb_id ext_net_peer_name
Weblogic_16 nginx_07 1.013605 0.00 1.016575 up
nginx_08 1.011905 0.00 1.007874 up
，请根据以上案例进行分析并输出now1和now2的outlier1和outlier2。
```

## ChatGPT问题

```
g ChatGPT
根据以上案例进行分析，首先确定指标分析发现的异常数据 (now1) 的异常指标
(outlier1)：

异常指标为：

- cmdb_ids: Weblogic_16
- kpi_names: system.io.avg_rq_sz, system.io.w_s, system.io.wkb_s

接下来，根据调用链分析发现的数据 (now2) 的异常值组合 (outlier2)：

异常值组合为：

- cmdb_ids: Weblogic_16
- ext_net_peer_names: nginx_07, nginx_08

因此，输出结果为：

- outlier1: [[Weblogic_16], [system.io.avg_rq_sz, system.io.w_s, system.io.wkb_s]]
- outlier2: [[Weblogic_16], [nginx_07, nginx_08]]

这表示在now1中，Weblogic_16的指标 system.io.avg_rq_sz, system.io.w_s,
system.io.wkb_s 异常；在now2中，Weblogic_16与nginx_07, nginx_08 之间存在异常关
联。
```

## ChatGPT定位结论



问：案例库中的指标分析发现的异常数据data1为\n cmdb\_id kpi\_name v\_err v\_pre v\_pst trend\n153 Weblogic\_33 system.io.avg\_rq\_sz 14.299000 5.037333 8.767000 up\n154 Weblogic\_33 system.io.wkb\_s 21.116667 13.150000 11.450000 up\n155 Weblogic\_33 system.net.bytes\_rcvd 9944.388000 3799.196000 5579.503333 up\n156 Weblogic\_33 system.net.packets\_in.count 26.141000 16.420333 21.967333 up\n157 Weblogic\_33 system.tcp.time\_wait 140.428571 82.666667 109.833333 up\n158 Weblogic\_34 system.tcp.time\_wait 114.800000 67.333333 101.333333 up\n159 Weblogic\_34 system.io.avg\_rq\_sz 5.566333 13.162333 32.327000 down\n160 Weblogic\_34 system.io.await 0.160667 0.390667 0.827000 down\n161 Weblogic\_34 system.io.w\_await 0.160667 0.390667 0.827000 down\n162 Weblogic\_34 system.io.w\_s 1.033333 2.766667 3.800000 down\n163 Weblogic\_34 system.io.wkb\_s 5.933333 18.400000 50.866667 down\n, 调用链分析发现的数据data2为\n\n v\_err v\_pre v\_pst trend\ncmdb\_id ext\_net\_peer\_name \nWeblogic\_33 nginx\_05 46.500000 14.000000 0.000000 up\n nginx\_07 7.333333 0.000000 7.000000 up\nWeblogic\_34 nginx\_05 7.000000 0.000000 0.000000 up\n nginx\_07 46.500000 0.000000 86.000000 up\n nginx\_08 64.000000 0.000000 0.000000 up\n, 观察指标异常值数据的v\_err、v\_pre、v\_pst和trend列，可以根据指标异常值数据的指标突变值情况看出异常指标为 outlier=[[Weblogic\_33, Weblogic\_34],[system.tcp.time\_wait]], 该数据包含kpi\_names和 cmdb\_ids两列。再将outlier在调用链数据中进行查询，并结合调用链数据的指标v\_err、v\_pre、v\_pst和trend进行判断，确定出调用链数据的异常值组合为outlier2=[[ Weblogic\_33, Weblogic\_34],[nginx\_03, nginx\_05, nginx\_06, nginx\_07, nginx\_08]], 该组合包含cmdb\_ids和 ext\_net\_peer\_name两列。 \n现在指标分析发现的异常数据now1为\n cmdb\_id kpi\_name v\_err v\_pre v\_pst trend\n101 Weblogic\_16 system.io.avg\_rq\_sz 2.803000 11.246333 22.030667 down\n102 Weblogic\_16 system.io.w\_s 1.333333 7.300000 3.333333 down\n103 Weblogic\_16 system.io.wkb\_s 6.316667 39.100000 57.166667 down\n, 调用链分析发现的数据now2为\n v\_err v\_pre v\_pst trend\ncmdb\_id ext\_net\_peer\_name \nWeblogic\_16 nginx\_07 1.013605 0.000000 1.016575 up\n nginx\_08 1.011905 0.000000 1.007874 up\n, 请根据以上案例进行分析并输出now1和now2的outlier1和outlier2。请直接告诉我结论，不要python代码。

根据以上案例，可以得出以下结论：

- 现在指标分析发现的异常数据now1的outlier1为[[Weblogic\_16, system.io.avg\_rq\_sz], [Weblogic\_16, system.io.w\_s], [Weblogic\_16, system.io.wkb\_s]], 该组合包含cmdb\_ids和 kpi\_name两列。
- 调用链分析发现的数据now2的outlier2为[[Weblogic\_16, nginx\_07], [Weblogic\_16, nginx\_08]], 该组合包含cmdb\_ids和ext\_net\_peer\_name两列。

## 私域大模型定位结论

私域大模型问题

## 第六章节

# 总结



## 总结

- 1、通过**时序异常检测**算法，完成异常检测
- 2、使用**动态建模**和**图网络**完成故障根因定位
- 3、使用**大模型**辅助故障根因定位

## 后续工作

- 1、探索大模型**Agent**的**深度应用**，完成更多运维工作。
- 2、深入研究**思维链**，使能分析复杂的问题。
- 3、使用大模型+本地知识库，实现故障处理案例推荐等。



2023 CCF国际AIOps挑战赛决赛暨“大模型时代的AIOps”研讨会

# THANKS