GOPS 2025 Shenzhen

10th ANNIVERSARY 2015-2025

第25届

GOPS 全球运维大会 2025

- XOps 风向标

深圳站

暨研运数智化技术峰会

时间：2025年4月25日-26日　　地址：中国·深圳

指导单位：DAOPS FOUNDATION　开放运维联盟 OOPSA Open OPS Alliance

主办单位：高效运维社区 GreatOPS Community　BizDevOps

承办单位：华佑科技 HUAYOU TECHNOLOGY

GOPS 2025 Shenzhen

10th ANNIVERSARY 2015-2025

大模型及DeepSeek在运维场景中的应用

王鹏

2025-04

王鹏

复旦大学计算机科学技术学院，教授

上海数据科学重点实验室，副主任

研究领域：大数据处理、智能运维

在数据库和数据挖掘领域顶级会议和期刊SIGMOD、VLDB、ICDE、SIGKDD、TKDE上发表论文100多篇。主持和参与科技部重点研发计划、国家青年973、自然科学重点、面上基金、上海市科委/经信委的多个项目，以及华为、蚂蚁金服、字节、中国移动等企业的资助项目

目录/ CONTENTS

1 大模型在运维领域应用前景

2 面临的挑战

3 若干应用场景

4 总结

百模大战

通用大模型

闭源

文心一言 | 讯飞星火 | 通义千问 | vivo BlueLM | oppo AndesGPT | 商汤日日新 sensenova | 出门问问 序列猴子
天工 | 字节云雀大模型 | 腾讯混元 | 西湖心辰 | 西湖大模型 | Moonshot AI | 言犀 | 360智脑
盘古大模型 | MiLM-6B | 云从科技从容大模型 | 深言科技 DeepLang AI | MINIMAX | 云知声 | 山海
4Paradigm SageGPT | 澜舟科技 孟子 | intellifusion 云天书 | AISPEECH | DFM | 玉言 | AiLMe

开源

百川Baichuan | ChatGLM2-6B | Qwen-7B | Yi-34B | XVERSE 元象 | InternLM | ChatYuan | Fengshenbang-LM
CPM-Bee | OpenBuddy | MOSS | TigerBot | RWKV-LM | Firefly | 中文LLaMA & Alpaca大模型 2 | FlagAI

行业大模型

部分领域

医疗	汽车	教育	金融	工业	文化/零售/交通
医联 / MedGPT	理想 MindGPT	MathGPT	蚂蚁金融大模型	AInno-15B	阅文集团 妙笔大模型
岐黄问道	DriveGPT	作业帮 银河大模型	轩辕大模型	COSMO-GPT	晓模型 XPT / 乐言科技
DISC-MedLLM	广汽AI大模型	子曰		SMore LrMo	PCI佳都科技

大模型在运维领域应用前景

GOPS 2025 Shenzhen | 10th ANNIVERSARY 2015-2025

L5 智能运维：实现系统自治，解放体力与脑力

专家经验运维

脚本编辑 ，人工执行

执行：人+脚本
决策：人

L1 - ScriptOps

工具化运维

使用多个独立工具
大部分工作工具化/流程化

执行：人+系统（20%）
决策：人

L2 - ToolsOps

高度自动化+单点智能化

运维开发融合运维

运维工具体系基本建设完成
运维数据体系建设完成

执行：人+系统（80%）
决策：人+系统（20%）

L3 - DevOps

高度自动化+串联智能化

数据化运维

主要运维场景实
现流程化免干预

执行：人+系统（95%）
决策：人+系统（80%）

L4 - DataOps

基于已有的经验知识
在不同场景下
自主决策处置

智能运维

AI分析与决策
自动化工具链+可视化

执行：人+系统（100%）
决策：人+系统（95%）

L5 - AIOPS

大模型的出现加速了实现终极L5智能运维的步伐

大模型在运维领域的应用前景

大语言模型是IT运维领域新质生产力的重要组成部分，推动数字化转型的颠覆式创新

01 **“自然语言”**成为连接运维人员、运维工具、运维文档、运维数据的通用接口

02 以**“聊天”**技术为载体实现人机协同完成运维任务

03 **提示词工程、检索增强、智能体**逐步在智能运维应用场景实践中凸显成效

应用场景

岗位助手
（近期）
运维知识问答
售后技术支持

数据工具
（近中期）
日志摘要/理解
告警摘要/标签
数据注释/标签

岗位培训教练
（近中期）
运维脚本解读/生成
培训题目自动生成/考核

数据探索/分析工具
（近中期）
Text2QL
（自然语言查询）
Text2API
（自然语言工具调用）

专业岗位专家智能体
（中期）
产品厂商智能体
专业领域智能体
（网络、数据库、应用）

虚拟作战室
（中期）
运维人员
工具智能体
岗位智能体
协同完成运维任务

基于“聊天”的人机交互协同应急处置

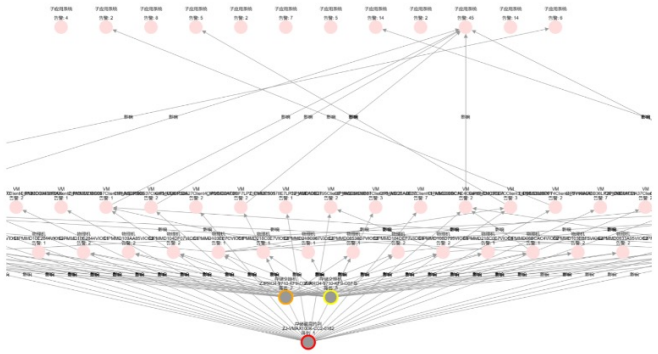
当前生产环境出现严重故障，请做一个初步分析。

过去半小时内，发生了 3456 条告警，涉及 50 套应用系统、100台物理机、200台虚拟机、50个数据库实例、50个中间件实例。
近期无相关生产变更。
正在执行拓扑根因定界分析，请稍等。

拓扑定界结果如下：

排名	CI/设备	根因分数	描述
1	STARY-47	80	EMC磁盘整列，非常可疑，所有故障告警都直接或间接连接该设备
2	STSWT-1	70	光纤交行机
3	STSWT-2	65	光纤交行机

根因分析拓扑图如下：



是否需要提供根因组件相关的信息？

请提供TOP1根因相关的异常信息

STRAY-47组件日志中有异常信息：

id	beginTime	level	status	content
4000415460	2023-12-16 18:53:34	30	4_notified	Symmetrix000496800182:Port state changed to Offline - Object
4000413835	2023-12-16 18:53:34	40	1_dispose	Symmetrix000496800182:Director state changed to Offline - Object
4000415459	2023-12-16 18:53:30	30	4_notified	Symmetrix000496800182:Power system AC line interruption detected.
4000415460	2023-12-16 18:42:22	30	4_notified	Symmetrix000496800182:Component state has change to Offline: Object is: 0004968000XXXX:SPS-3A
4000415069	2023-12-16 18:42:22	30	4_notified	Symmetrix000496800182:No real time data received for Systemtrix 0004968000XXX within the last 15 minutes
4000409991	2023-12-16 18:42:22	40	4_notified	Symmetrix000496800182:Director state changed to Offline - Object
4000410478	2023-12-16 18:42:22	30	4_notified	Symmetrix000496800182:Power system AC line interruption detected.
4000412333	2023-12-16 18:42:03	40	4_notified	Symmetrix000496800182:Director state changed to Offline - Object
4000411386	2023-12-16 18:41:57	30	4_notified	Symmetrix000496800182:Power system AC line interruption detected.
4000411379	2023-12-16 16:17:36	40	4_notified	Symmetrix000496800182:Director state changed to Offline - Object
4000409941	2023-12-16 16:16:33	30	4_notified	Symmetrix000496800182:No real time data received for Systemtrix 0004968000XXX within the last 15 minutes

请解读上述异常日志。

Director state has changed to Offline. - Object is: 000496800182:RF-3E
含义：这是一个状态变化的告警，指示一个Director（存储控制器）的状态从在线（Online）变为离线（Offline）。
影响：当Director离线时，相关的存储功能可能会受到影响，可能导致数据不可访问、性能下降或其他存储操作失败。
...

本次故障影响范围评估？

手机银行、柜面等交易受影响。期间成功交易量为0，上周同期交易量约244587笔。
需要生成更详尽的评估报告吗？

基于“聊天”的人机交互协同应急处置

大模型理解用户意图

当前生产环境出现严重故障，请做一个初步分析。

过去半小时内，发生了 3456 条告警，涉及 50 套应用系统、100台物理机、200台虚拟机、50个数据库实例、50个中间件实例。
近期无相关生产变更。
正在执行拓扑根因定界分析，请稍等。

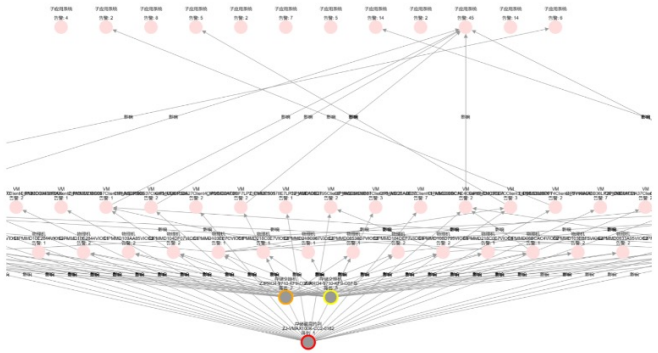
生成SQL，查询告警数据库，并根据结果和运维通识组织语言，提示下一步操作

拓扑定界结果如下：

排名	CI/设备	根因分数	描述
1	STARY-47	80	EMC磁盘整列，非常可疑，所有故障告警都直接或间接连接该设备
2	STSWT-1	70	光纤交行机
3	STSWT-2	65	光纤交行机

调用拓扑定位工具输出结果并解释

根因分析拓扑图如下：



是否需要提供根因组件相关的信息？

提示候选下一步操作

请提供TOP1根因相关的异常信息。

STRAY-47组件日志中有异常信息：

id	beginTime	level	status	content
4000415460	2023-12-16 18:53:34	30	4_notified	Symmetrix000496800182:Port state changed to Offline - Object
4000413835	2023-12-16 18:53:34	40	1_dispose	Symmetrix000496800182:Director state changed to Offline - Object
4000415459	2023-12-16 18:53:30	30	4_notified	Symmetrix000496800182:Power system AC line interruption detected.
4000415460	2023-12-16 18:42:22	30	4_notified	Symmetrix000496800182:Component state has change to Offline: Object is: 0004968000XXXX:SPS-3A
4000415069	2023-12-16 18:42:22	30	4_notified	Symmetrix000496800182:No real time data received for Systemtrix 0004968000XXX within the last 15 minutes
4000409991	2023-12-16 18:42:22	40	4_notified	Symmetrix000496800182:Director state changed to Offline - Object
4000410478	2023-12-16 18:42:22	30	4_notified	Symmetrix000496800182:Power system AC line interruption detected.
4000412333	2023-12-16 18:42:03	40	4_notified	Symmetrix000496800182:Director state changed to Offline - Object
4000411386	2023-12-16 18:41:57	30	4_notified	Symmetrix000496800182:Power system AC line interruption detected.
4000411379	2023-12-16 16:17:36	40	4_notified	Symmetrix000496800182:Director state changed to Offline - Object
4000409941	2023-12-16 16:16:33	30	4_notified	Symmetrix000496800182:No real time data received for Systemtrix 0004968000XXX within the last 15 minutes

自动查询ES中的设备日志，并过滤其中的异常日志

请解读上述异常日志。

Director state has changed to Offline. - Object is: 000496800182:RF-3E
含义：这是一个状态变化的告警，指示一个Director（存储控制器）的状态从在线（Online）变为离线（Offline）。
影响：当Director离线时，相关的存储功能可能会受到影响，可能导致数据不可访问、性能下降或其他存储操作失败。
...

查询厂商文档，对日志进行解释

本次故障影响范围评估？

手机银行、柜面等交易受影响。期间成功交易量为0，上周同期交易量约244587笔。
需要生成更详尽的评估报告吗？

查询业务指标信息，并汇总结果

大模型在运维场景中的落地仍面临众多挑战

挑战	场景示例
效率	• 日志解析 • RAG
准确性	• RAG • Text2SQL • Agent工具调用
有效性	• 故障定界/根因分析
扩展性	• RAG • Agent工具调用

大模型在运维场景中的落地仍面临众多挑战

挑战	场景示例	模型本身能力	模型外能力
效率	• 日志解析 • RAG	• 模型推理效率低 • DS更甚	• “产品经理”思维 • 模型能力+软件能力 • 融合大模型的整体系统架构 • 以终为始
准确性	• RAG • Text2SQL • Agent工具调用	• 幻觉/长上下文 • 简单问题：90分 • 复杂问题：50分	
有效性	• 故障定界/根因分析	• 运维思路 • 需综合全局多源信息	
扩展性	• RAG • Agent工具调用	• “玩具” vs “真实”场景	

大模型在运维场景中的落地仍面临众多挑战

- “产品经理”思维
  - 用户是谁
  - 产品功能、响应速度、输入/输出、并发数的清晰界定
  - 例如：大模型辅助的排障助手
- 模型能力+软件能力
  - 什么功能利用了大模型的什么能力
  - 软件层面要提供哪些能力，并发、容错、上下文超限等
- 融合大模型的整体系统架构
  - 传统的软件架构、设计模式仍需考虑

- **以终为始**

GOPS 2025
Shenzhen
ANNIVERSARY
2015-2025

1

基于异步大模型调度的高效日志解析

基于异步大模型调度的高效日志解析方法

系统日志广泛用于理解系统运行状态，支持故障检测与诊断。然而，日志数据常为半结构化文本，难以直接使用。因此，常常需要首先将日志转化为结构化的“模板”+“变量”的形式，一方面可以将日志序列简化为模板 ID 序列，降低分析复杂度，另一方面结构化的表达形式也更便于进行统一与自动化处理分析。因此日志解析常常作为日志处理的必要前置任务。

00d Creating NT trans (seq 13), object [6]
141 double-hummer alignment exceptions
instruction address: 0x00004ed8

提取变量，保留常量

<*> Creating NT trans (seq <*>), object [<*>]
<*> double-hummer alignment exceptions
instruction address: <*>

1.传统方法

通过规则匹配或统计特征（如频率、长度）进行解析，处理速度快，但缺乏语义理解能力，难以准确区分常量与变量，进而影响解析性能

2.深度学习方法

将解析建模为分类任务，识别日志中的变量位置，能提取一定语义信息，但依赖大量标注数据对模型进行训练，泛化能力有限

3.LLM方法

利用大语言模型强大的语义理解能力进行解析，精度高，但处理过程依赖LLM调用来进行解析，效率低，难以满足在线解析的效率要求

基于异步大模型调度的高效日志解析方法

关键瓶颈

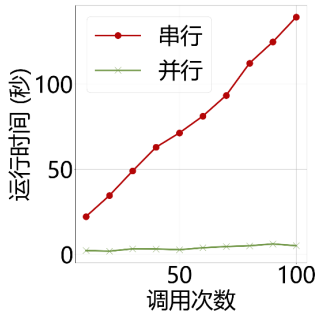
LLM调用开销

相较于传统方法与深度学习方法，每调用一次LLM的时间开销巨大。

串行解析

当前一条日志解析完成后才开始处理下一条，导致大量时间等待LLM返回结果。

然而，大多数LLM的部署方法支持并行调用，时间接近一次调用。



基于LLM的日志解析的常见操作

模板匹配

利用前缀树等结构进行快速匹配

是否涉及LLM：**否**　速度：**快**

模板抽取

利用LLM对给定日志生成模板

是否涉及LLM：**是**　速度：**慢**

模板更新

将当前日志合并已有模板进行更新

是否涉及LLM：**是**　速度：**较慢**

关键挑战

时间不一致

不同操作的速度差异，导致快速操作被迫等待，降低整体效率。

解析依赖顺序

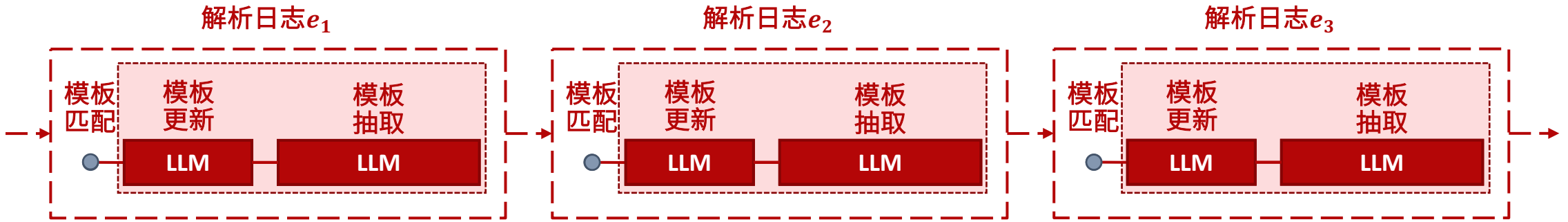
现有方法存在顺序依赖性，对于前面日志的解析，可能会影响后面日志解析的结果，直接并行会打乱这种顺序依赖。

重复解析

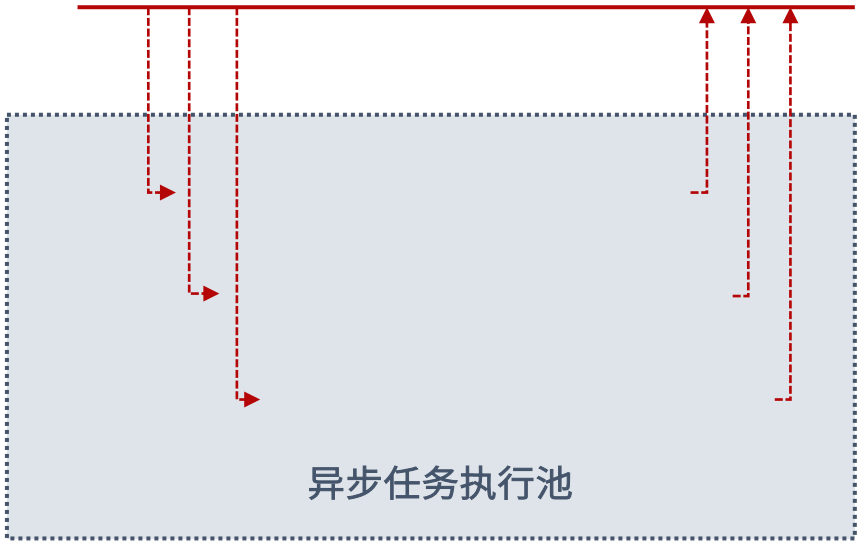
相似日志同时处理，模板尚未缓存进前缀树，导致重复触发LLM调用，资源浪费。

基于异步大模型调度的高效日志解析方法

串行解析



异步解析



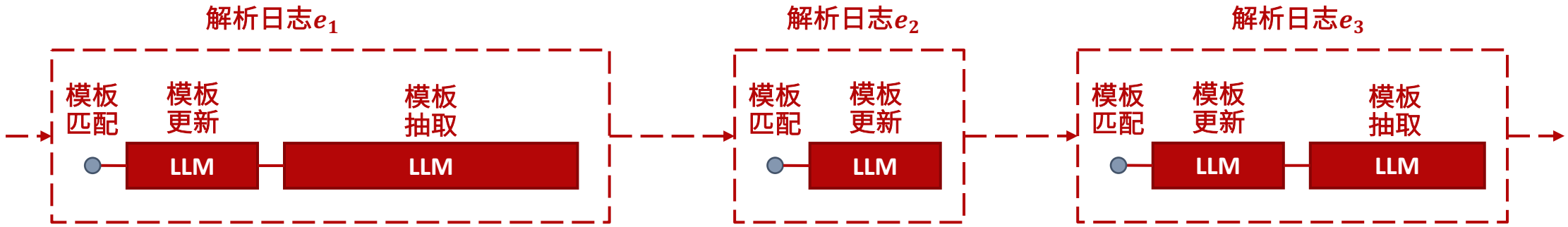
问题：串行执行，时间不一致

解决方案：异步并行

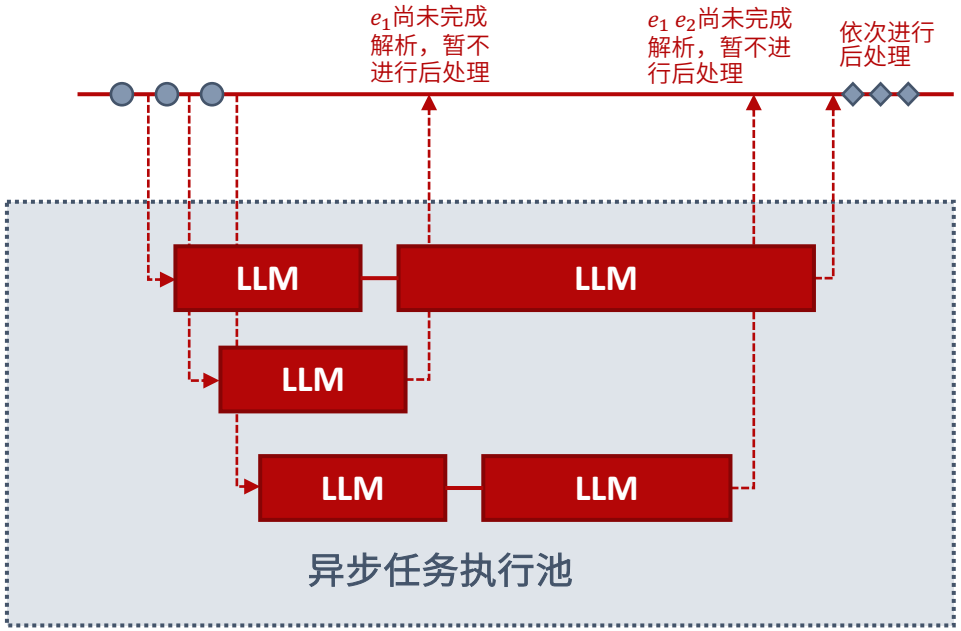
涉及LLM的操作统一进行异步并行执行，剩余操作包括模板匹配等，均在主流程中串行执行。在充分利用并行优势的情况下，规避不同操作时间不一致带来的延迟。

基于异步大模型调度的高效日志解析方法

串行解析



异步解析



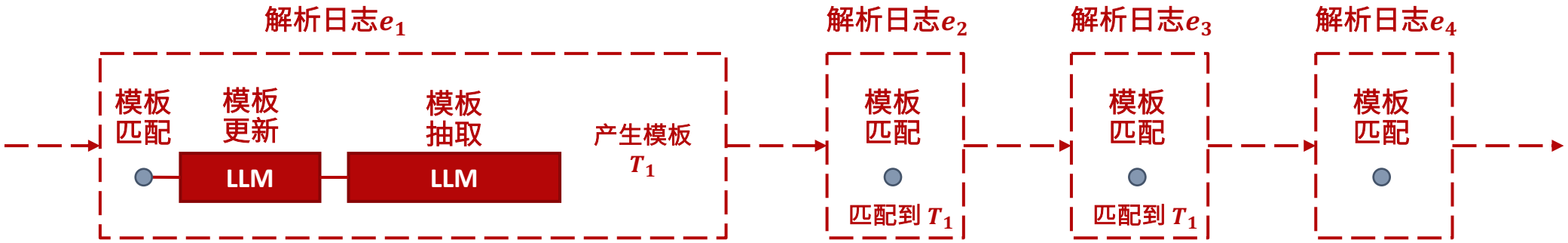
问题：解析顺序依赖

解决方案：统一调度

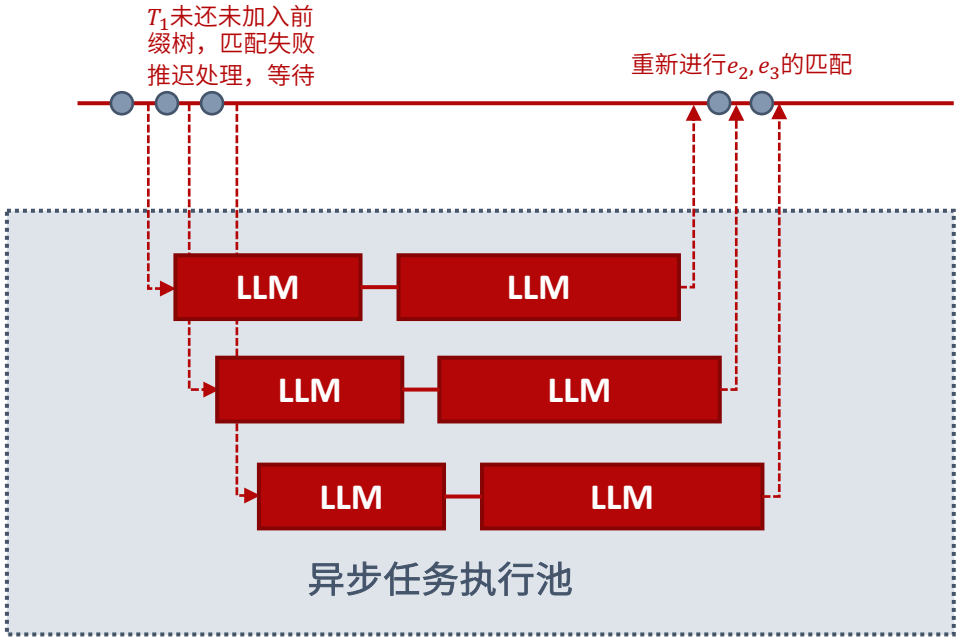
当LLM调用在异步执行池中完成时，不立即进行后序处理，而是交由全局的任务管理模块进行后处理，确定其后处理顺序，以保证顺序依赖。考虑到LLM操作和代码操作的巨大时间差异，这样推迟的成本非常小。

基于异步大模型调度的高效日志解析方法

串行解析



异步解析



问题：重复解析

解决方案：任务生成管理

引入等待机制，判断解析当前日志即将生成的LLM任务是否潜在与异步执行池中已有的任务潜在重叠，如果有重叠的可能性，则让当前任务“等待”，待前序任务完成，再重新开启对于当前任务的解析。

基于异步大模型调度的高效日志解析方法

异步并行

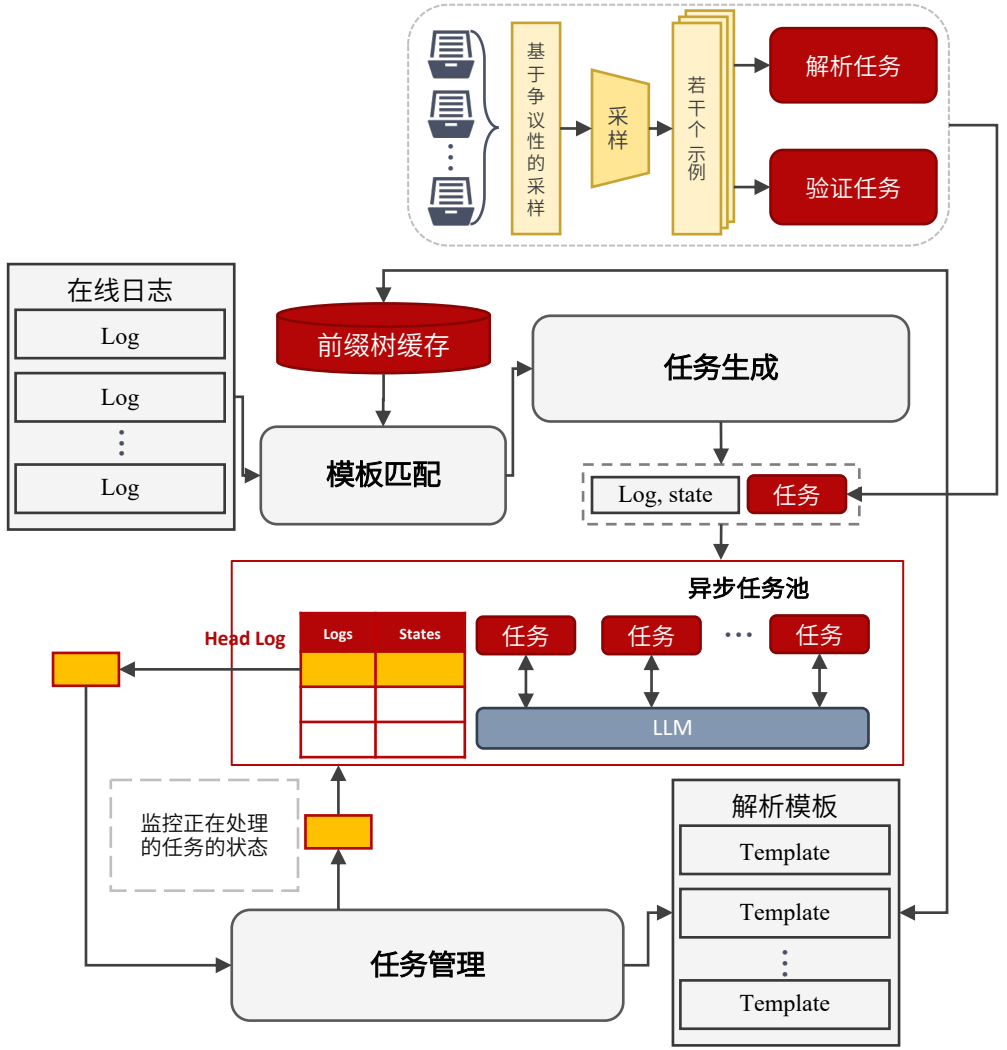
将任务异步提交至大模型，允许任务独立返回，从而解决了复杂日志拖慢整体进度的问题，提升整体效率。

统一调度

解析任务与调度逻辑解耦，日志动态分配任务，调度系统统一管理任务顺序与依赖，确保语义一致性和稳定性。

生成管理

通过等待机制避免重复调用大模型，对相似日志在模板尚未返回前暂缓处理，减少冗余生成任务，提升效率。



基于异步大模型调度的高效日志解析方法

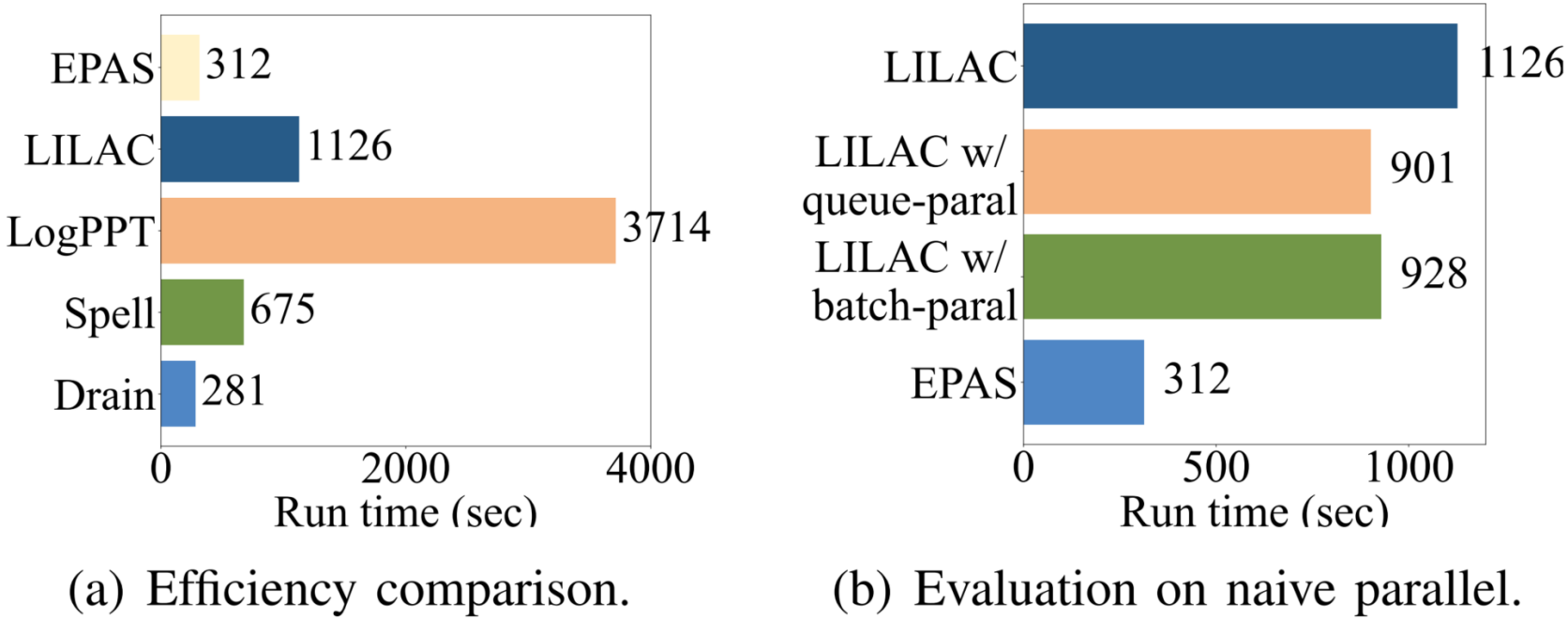
- **EPAS准确性均高于传统方法和现有基于大模型的方法**

Dataset	Drain				Spell				LogPPT				LILAC				EPAS			
	GA	FGA	PA	FTA	GA	FGA	PA	FTA	GA	FGA	PA	FTA	GA	FGA	PA	FTA	GA	FGA	PA	FTA
Apache	**100.0**	**100.0**	72.7	51.7	**100.0**	**100.0**	25.0	17.2	78.6	48.4	95.2	35.2	**100.0**	**100.0**	**99.9**	86.2	**100.0**	**100.0**	99.8	**89.7**
BGL	91.9	62.4	40.7	19.3	67.7	1.6	23.5	0.2	31.1	51.9	85.5	50.9	90.5	87.3	89.6	76.4	**96.9**	**90.9**	**95.3**	**84.0**
Hadoop	92.1	78.5	54.1	38.4	44.9	5.1	11.6	1.7	53.3	57.0	72.5	46.2	66.5	91.7	60.9	76.5	**99.0**	**97.3**	**94.8**	**85.1**
HDFS	99.9	**93.5**	62.1	60.9	96.1	62.5	29.0	22.5	69.4	38.5	89.7	30.8	**100.0**	78.0	**100.0**	**78.0**	99.9	76.2	96.7	64.3
HealthApp	86.2	1.0	31.2	0.4	65.2	1.4	16.2	0.5	83.9	86.3	98.7	78.7	**100.0**	**98.1**	98.8	**91.8**	99.8	97.1	**99.6**	90.8
HPC	79.3	30.9	72.1	15.2	59.4	26.4	49.9	14.3	78.2	81.2	**99.7**	76.2	**87.2**	**95.9**	64.0	**83.8**	85.8	91.4	77.2	82.1
Linux	68.6	77.8	11.1	25.9	62.3	78.4	3.3	10.8	20.0	68.6	62.1	39.9	95.9	**87.1**	87.2	55.9	**97.0**	83.7	**95.2**	**70.2**
Mac	76.1	22.9	35.7	6.9	76.0	28.3	6.9	3.3	53.6	45.9	41.0	27.5	78.0	84.0	58.9	55.1	**95.1**	**90.0**	**81.0**	**63.3**
OpenSSH	70.7	87.2	58.6	48.7	61.6	42.6	19.3	9.8	27.8	11.5	71.3	14.7	74.6	74.3	74.6	71.4	**78.0**	**96.1**	**77.9**	**85.7**
OpenStack	75.2	0.7	2.9	0.2	78.1	0.1	0.0	0.0	53.4	92.9	40.8	78.8	**100.0**	**100.0**	99.0	**93.8**	**100.0**	**100.0**	**100.0**	**95.8**
Proxifier	69.2	20.6	68.8	17.6	52.1	4.0	48.0	1.1	51.0	75.0	99.3	91.7	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
Spark	88.8	86.1	39.4	41.2	43.2	31.6	40.5	21.1	45.0	38.3	95.4	33.1	96.8	83.4	96.6	63.2	**100.0**	**91.4**	**99.9**	**78.0**
Thunderbird	83.1	23.7	21.6	7.1	42.9	12.9	12.1	2.8	41.6	33.4	28.3	13.1	79.7	79.3	64.4	52.9	**89.9**	**85.4**	**69.7**	**60.9**
Zookeeper	99.4	90.4	84.3	61.4	98.7	57.8	78.9	28.1	97.3	90.3	84.3	75.3	**100.0**	96.7	84.9	86.8	99.8	**98.3**	**99.6**	**92.7**
Average	84.3	55.4	46.8	28.2	67.7	32.3	26.0	9.5	56.0	58.5	76.0	49.4	90.7	89.7	84.2	76.6	**95.8**	**92.7**	**91.9**	**81.6**

- **EPAS: Efficient Online Log Parsing via Asynchronous Scheduling of LLM Queries**. In ICDE 2025.

基于异步大模型调度的高效日志解析方法

- **EPAS性能远优于现有基于大模型的方法，优于传统方法Spell**



(a) Efficiency comparison.

(b) Evaluation on naive parallel.

- **EPAS: Efficient Online Log Parsing via Asynchronous Scheduling of LLM Queries**. In ICDE 2025.

2

Text2SQL

Text2SQL

Text2SQL技术旨在将自然语言形式的查询自动转换为数据库上的结构化查询语言（SQL）。鉴于数据库在各应用领域的广泛应用，该技术被学术界和工业界广泛关注

总体而言，Text2SQL技术路线分为两类：基于精调小模型、基于大模型

数据的语义理解

自然语言描述的查询存有模糊性，加之算法对数据的理解也存在偏差，两者叠加，使数据语义理解变得尤为重要。

领域知识的依赖

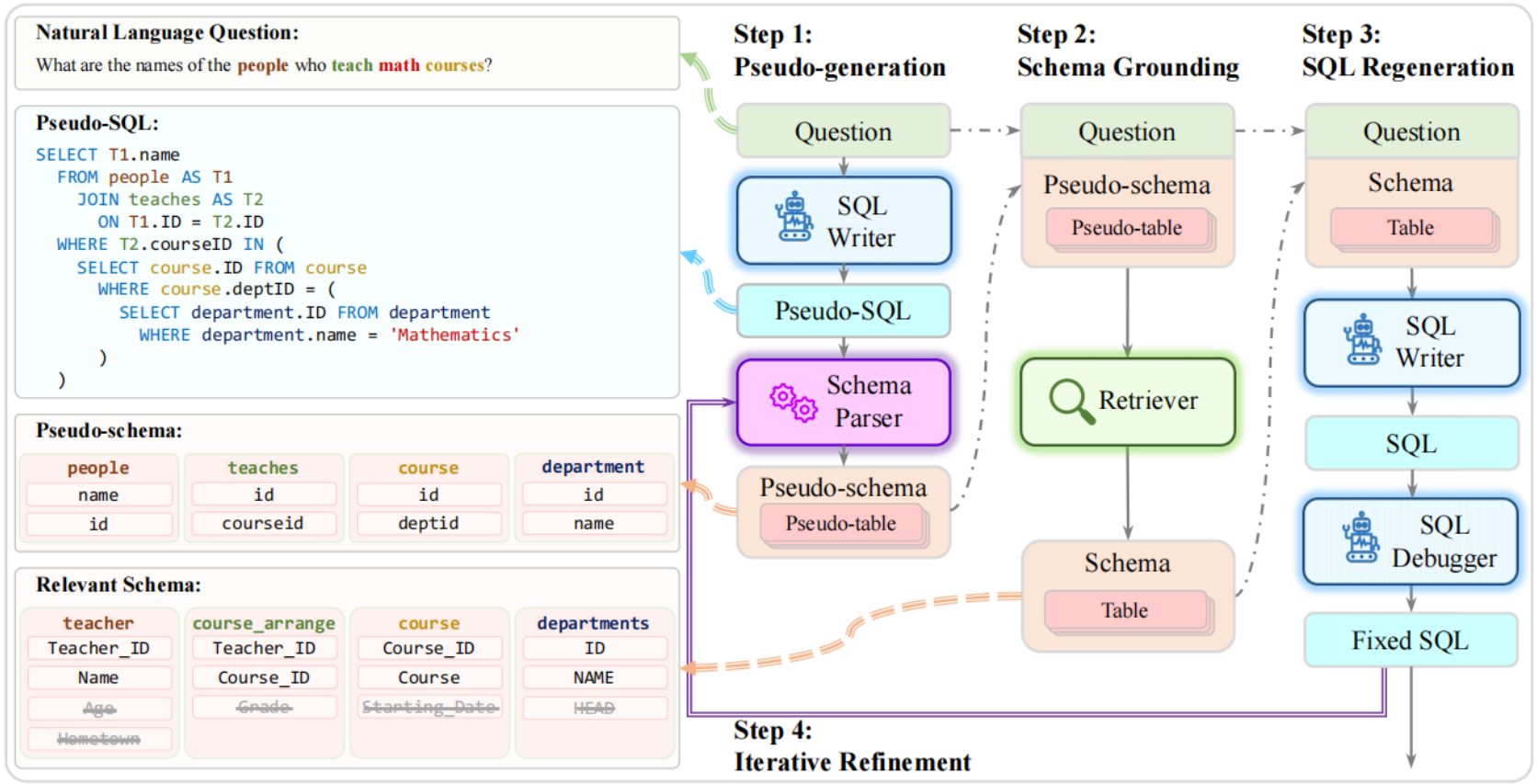
用户的提问可能包含行业黑话或特定的领域知识（如银行业务中的“头寸”，不同企业的 “财年”）。

操作逻辑的复杂性

数据库操作的固有复杂性使得难SQL的翻译效果不佳（如嵌套查询、复杂的聚集、多表连接等）。

基于大模型：Gen-SQL

- **先生成再检索**的范式
- 利用大模型预训练过程中取得的先验知识，先根据问题猜测所需表结构，再用向量检索器召回相关表



- **Gen-SQL: Efficient Text-to-SQL By Bridging Natural Language Question And Database Schema With Pseudo-Schema**, in COLING 2025

Text2SQL技术 – PURPLE

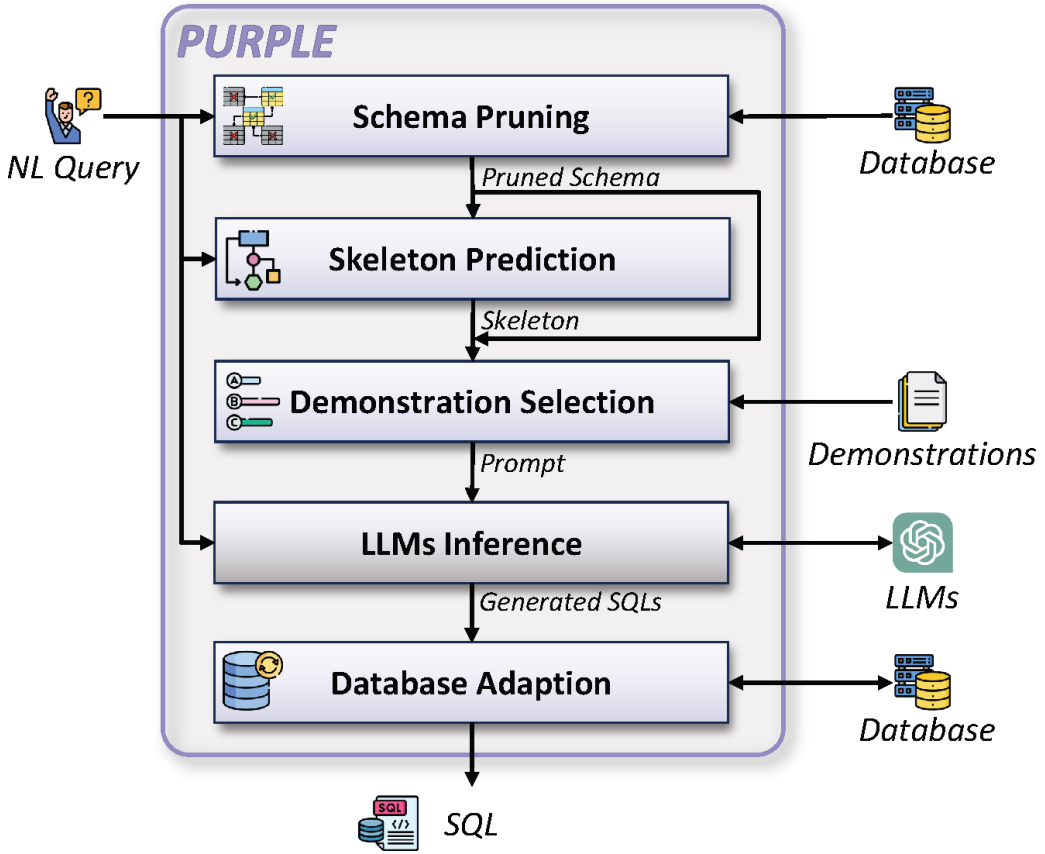
- LLM具有较好的语义理解能力（也有幻觉），直接利用LLM直接翻译，效果不佳；

对LLM精细化控制

控制大模型做其擅长的事情，如让它做选择题（从候选集中选SQL框架）

为LLM提供SQL算子组合

提供一系列的SQL算子组合（来自各种各样应用场景），让其进行选择



- **PURPLE: Making a Large Language Model a Better SQL Writer**. In ICDE 2024
- **The Power of Constraints in Natural Language to SQL Translation**. Accepted by VLDB 2025.

Text2SQL+RAG：三层索引检索

- **预处理阶段**：以**表、列、值**为粒度构建三层树状索引

- **推理阶段**：假如输入问题与表数据、列数据和值数据的相关度计算分别为 a、b 和 c，那么该路径上的表分数 S_t、列分数 S_c 和值分数 S_v 分别为：

$$S_t = a$$

$$S_c = k_{11}a + k_{12}b$$

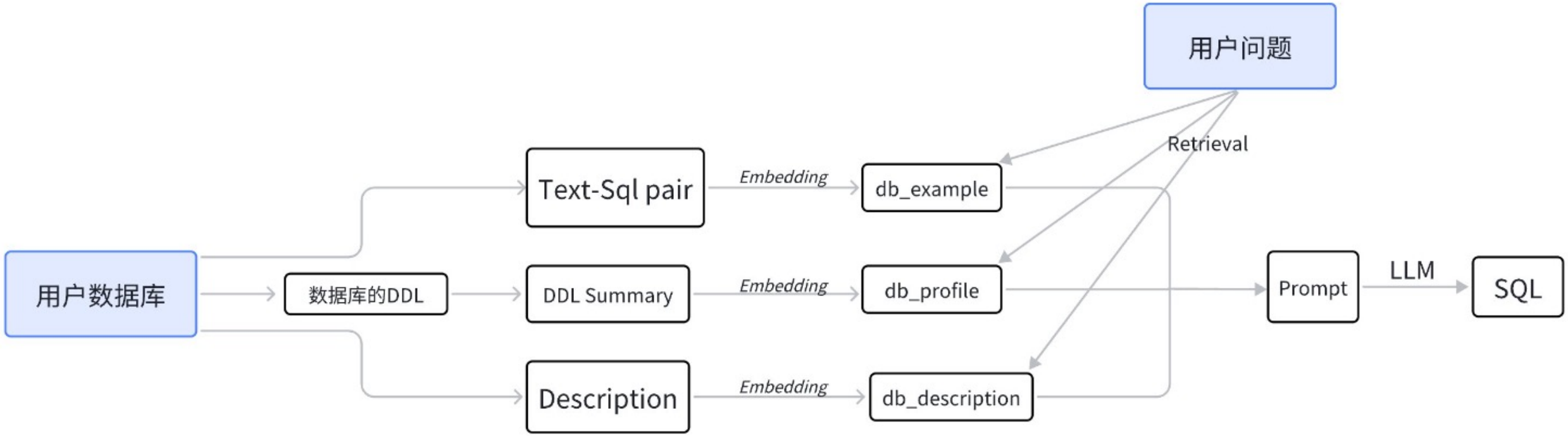
$$S_v = k_{21}a + k_{22}b + k_{23}c$$

- 实验结果表明，三层索引检索通过过滤数据库的表、列、值信息，可以有效增强大模型生成SQL的准确率

方法	千问 72b	ChatGPT-3.5-turbo	ChatGPT-4
实体编码检索方法	**82.1**	**83.5**	90.8
不带检索默认方法	65.3	72.7	82.3
表粒度检索方法	60.5	67.0	79.4
列粒度检索方法	63.2	70.2	80.8
值粒度检索方法	55.1	63.9	78.8
DAIL-SQL	N/A	78.1	86.6
PET-SQL	N/A	N/A	85.5
CHESS	N/A	N/A	87.2
MiniSeek	N/A	N/A	**91.2**

Text2SQL+RAG：外部知识反馈

- 方法
  - 补充和业务相关的专家知识
  - 描述性知识（正确的查询思路，正确的字段映射）
  - 可参考的正确查询样例
- 分而治之，对于复杂的问题，分为多个子问题，生成子查询后聚合



Text2SQL+RAG：外部知识反馈

描述性知识（Description）

sse_audit_online中的**字段terminal主要有三种取值：“and”，“ios”，“others”**，其中“and”即安卓

查询产品为L1，终端为安卓，金桥机房的根据不同的券商、信息商、产品，对应每个国家以及每个城市的用户连接数

用户想要了解产品为L1，终端为安卓，且在金桥机房的情况下，不同券商、信息商、产品在每个国家和城市的用户连接数。这需要我们对数据进行分组并计算每个分组的用户去重连接数。

Data SQL

```sql
SELECT
  broker,
  vendor,
  product,
  country,
  city,
  COUNT(DISTINCT uid) AS user_connections
FROM
  scene1.sse_audit_online
WHERE
  product = 'L1'
  AND terminal = 'and'
  AND home = '金桥'
GROUP BY
  broker,
  vendor,
  product,
  country,
  city
ORDER BY
  user_connections DESC
```

Text2SQL+RAG：外部知识反馈

Text-SQL对：

Text：查询用户 ID 为admin的用户在 2021年1月1日至2023年1月31日之间的类型为“变更管理”的待办任务

SQL：

```
。。。
SELECT GROUP_CONCAT(id) AS menu_ids
    FROM bpm_mo_menu
    WHERE parentid IN (
      SELECT getModelChildrenId(id)
      FROM menu_id
    )
。。。
```

查询用户 ID 为admin的用户在 2021年1月1日至2023年1月31日之间的类型为"服务请求"的待办任务

为了查询用户ID为admin的用户在指定日期范围内的类型为'服务请求'的待办任务，我需要从act_ru_task表中获取任务信息，同时连接bpm_process_fast_query表以获取流程的标题和类型，以及act_ru_identitylink表以获取处理人信息。我将使用WITH子句来找到'服务请求'相关的菜单ID，然后通过JOIN子句将这些信息与任务表关联。最后，我将使用WHERE子句来过滤出admin的待办任务和指定日期范围内的记录。

Data SQL

```
WITH
  menu_id AS (
    SELECT
      id
    FROM
      bpm_mo_menu
    WHERE
      name = '服务请求'
  ),
  sub_menu_ids AS (
    SELECT
      GROUP_CONCAT(id) AS menu_ids
    FROM
      bpm_mo_menu
    WHERE
      parentid IN (
        SELECT
          getModelChildrenId (id)
        FROM
          menu_id
      )
  )
SELECT
```

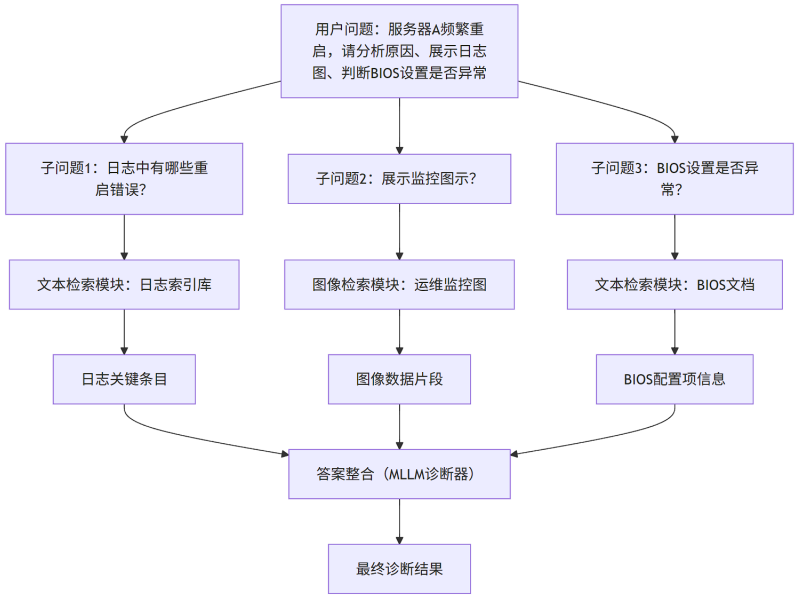
添加该例子前，对于指定类型有关的查询待办任务，不会使用预先定义好的getModelChildrenId存储过程，导致生成不完全准确的SQL。

3

基于静态DAG的多模态检索增强

多模态检索增强

检索增强生成（Retrieval-Augmented Generation, RAG）是一种结合语言模型和外部知识库的方法，能够显著提升大模型在问答、对话和文档理解中的知识覆盖率和回答准确性。然而传统RAG往往仅依赖“一轮检索+一轮生成”的流程，难以处理具有多跳逻辑、模态切换与强依赖结构的问题。我们提出一种基于静态有向无环图（DAG）对于用户的问题进行结构化分解，通过规划子问题之间的依赖关系，指导多模态检索器高效调度并且融合方案，提高问答的结构可控性、准确性以及过程可解释性。



传统RAG方法

没有结构规划，统一一次性检索，容易遗漏关键信息，难以应对多跳复杂问题。

动态规划的方法

动态调整路径，但是执行通常是线性，较为复杂效率较低；同时在调用链过长时LLM容易偏离原始查询意图。

基于静态DAG的方法

拆解子问题，规划依赖关系，可以并行解决子问题，信息整合准确，查询中不容易出现意图偏移。

静态DAG在复杂多模态问答中的应用流程与优势

静态DAG执行流程

1.用户提出问题

例如：查询设备功能、查看图云等

2.LLM进行DAG规划

根据问题复杂度进行结构化拆分

3.多模态执行检索

每个子问题根据内容执行对应的模态检索

4.答案整合和输出

整合所依赖的子问题结果得到答案

优势分析

结构清晰

合理拆解复杂问题提前规划

调度高效

子问题可以并行执行提高响应速度

自适应模态检索

子问题选择最合适的模态检索方式

可解释性强

每一步探索路径清晰可追朔，有利于故障定位和结果验证。

关键挑战

1.多模态异构信息分散

图像、文本、表格等信息来源往往不同，统一检索难度大

2.多跳推理路径复杂

问题隐含逻辑依赖关系，不易一步检索命中目标

3.复杂问题难以解答

LLM难以直接一步到位处理复杂问题中的依赖给出正确答案

静态DAG在复杂多模态问答中的应用流程与优势

动态规划的方法

- 效率问题
- “意图偏离”问题

基于静态DAG的方法

- 训练LLM，自动生成多个子问题
- 意图一致性的度量
- 样本分布一致性的考虑

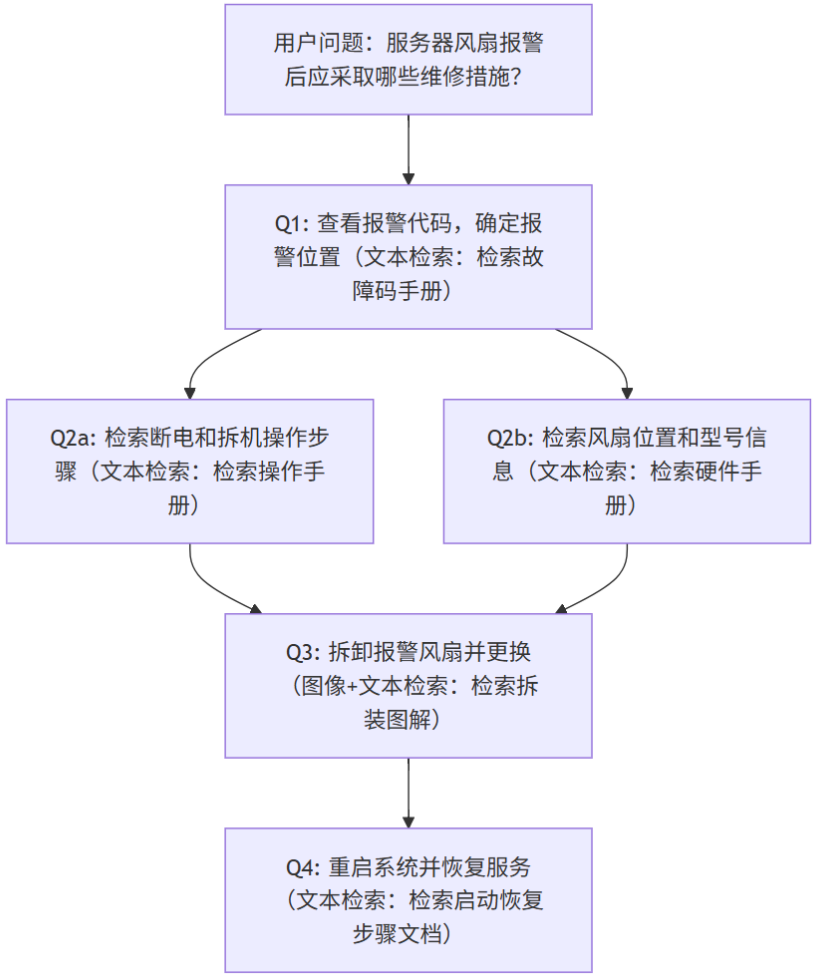
静态DAG在复杂多模态问答中的应用流程与优势

实验结果对比

我们在MMQA这个多模态多跳问答数据集上验证我们方法的准确性

方法	Exact Match	F1 Score
Binder	51.0	57.1
MMHQA-ICL	54.8	65.8
HPROPRO	59.0	66.7
Ours	60.61	69.65

Exact Match：回答与标准答案是否完全一致；
F1 Score：回答与标准答案的词级重合度（调和平均）。



4

基于多智能体的故障定位

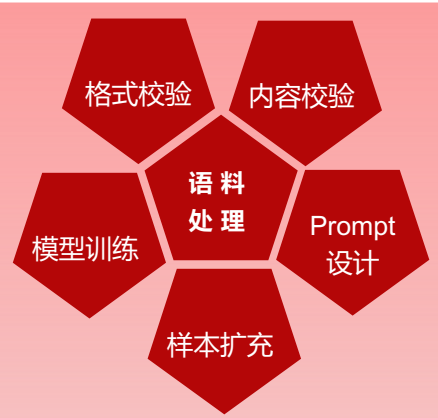
基于多智能体的故障定位

- **运维大模型训练**：通过收集故障处置过程中运维专家的思考、操作历史记录，训练运维大模型，使其可以构建出类似于运维专家的排障树思维，通过Agent规划运维步骤，并调用相关工具
- **多智能体交互框架**：多智能体的交互框架，使其能够协作完成复杂的运维任务

基于多智能体的故障定位

- 智能体语料样本准备策略
- 模型训练方案

- **运维流程数据范式:** 符合Agent的训练标准，兼顾运维专家的编写习惯
- **运维流程数据处理：**自动化处理收集到的运维流程数据，完成格式校验、内容校验、Prompt填充、样本转换等步骤
- **智能体增强训练：**利用专家运维流程数据，选取合适的基座模型进行训练，研究不同训练因素的影响



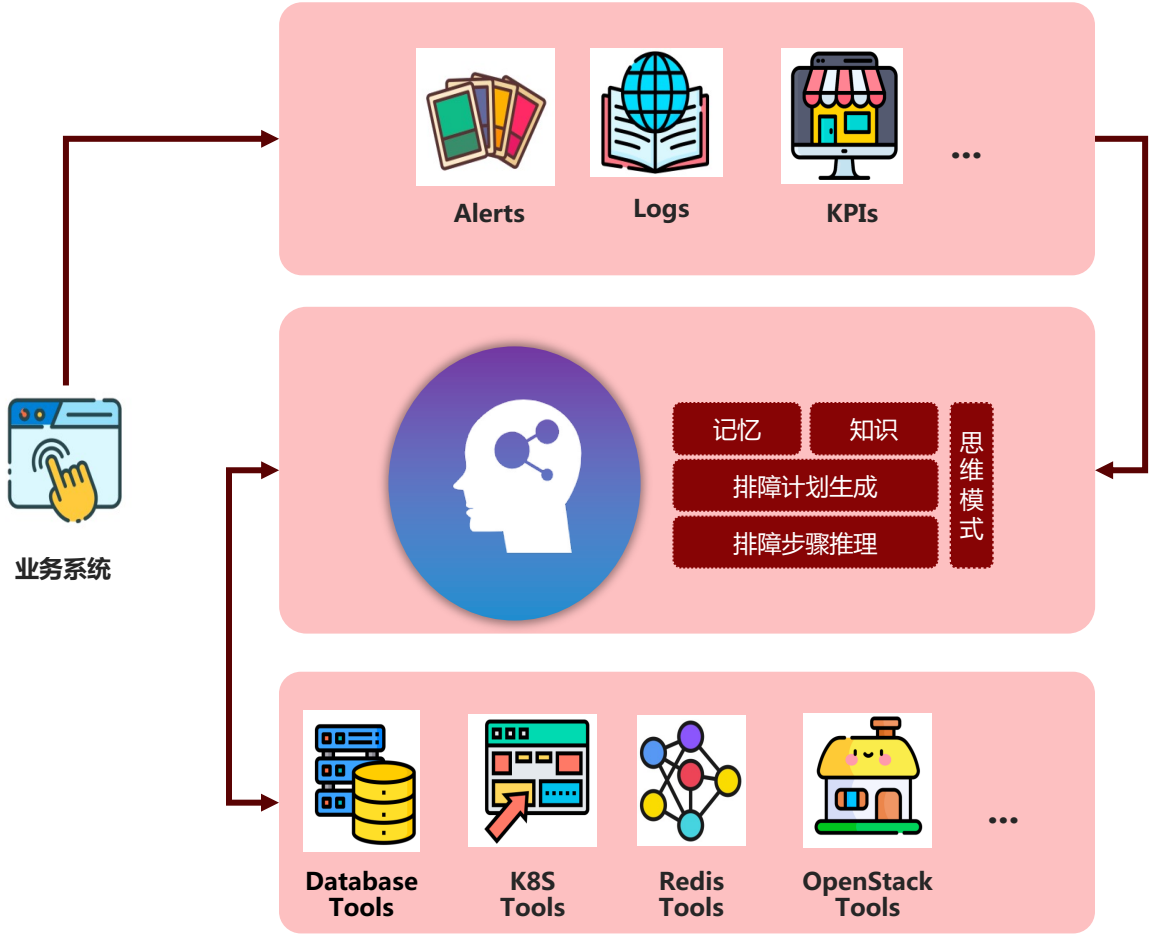
格式校验
自动校验样本格式是否符合定义的范式

内容校验
自动校验样本内容是否符合正确的排障逻辑

Prompt设计
设计合理的Prompt提示大模型完成任务

样本扩充
扩充样本的数量，增加样本的多样性

模型训练
采用合适策略训练模型，使之学会专家运维知识



基于多智能体的故障定位

- 构建复杂多智能体的通讯和交互组件，细分不同的智能体角色与能力
- 设计具体通讯和交互模式类型与内容

- **智能体角色与能力设计：**细分不同智能体的角色与能力，提高针对复杂任务的规划解决能力
- **智能体通讯交互内容设计：**规范智能体的通讯交互的内容与格式，保证信息传递稳定性和准确性
- **智能体通讯交互模式设计：**实现复杂通讯模式，包括一对多广播、多对多讨论等协作模式，提高整体智能

智能体角色与通讯交互内容设计

Role: Redis Solver

Ability: 解决Redis问题……

Memory:

我之前进行了如下操作：……

Observation:

当前的工具调用结果是……

Thought:

工具调用成功，但问题仍然……

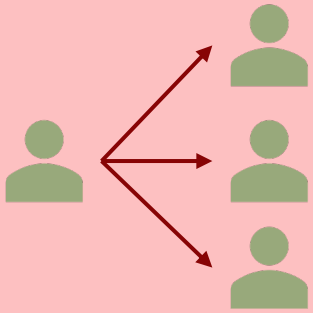
我需要调用XX工具….

To XX Agent（Structured):

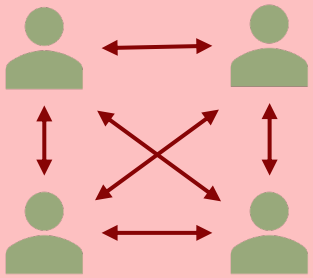
Summarized Thought: …….

Action: ……

智能体通讯交互模式设计

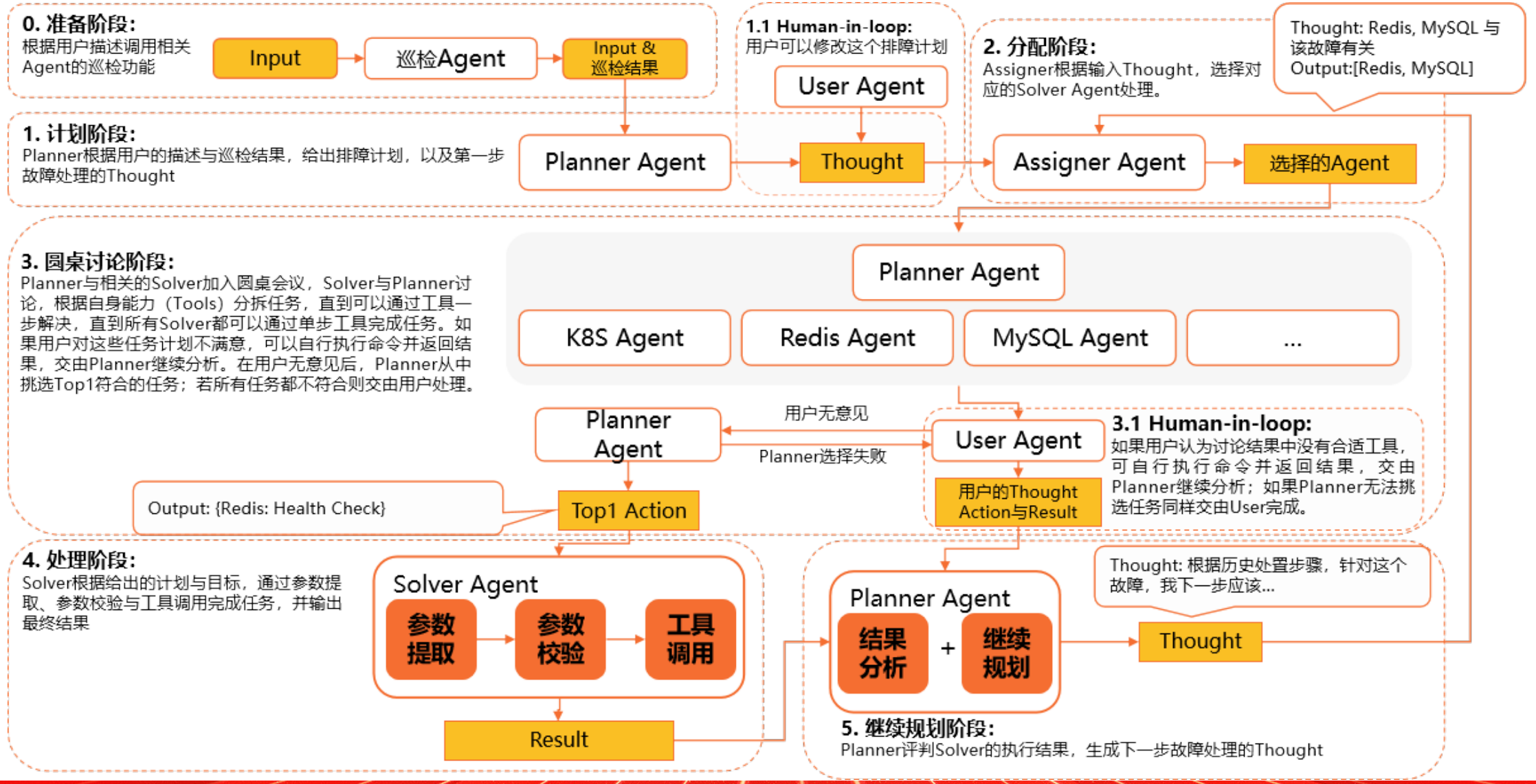


一对多广播



多对多讨论

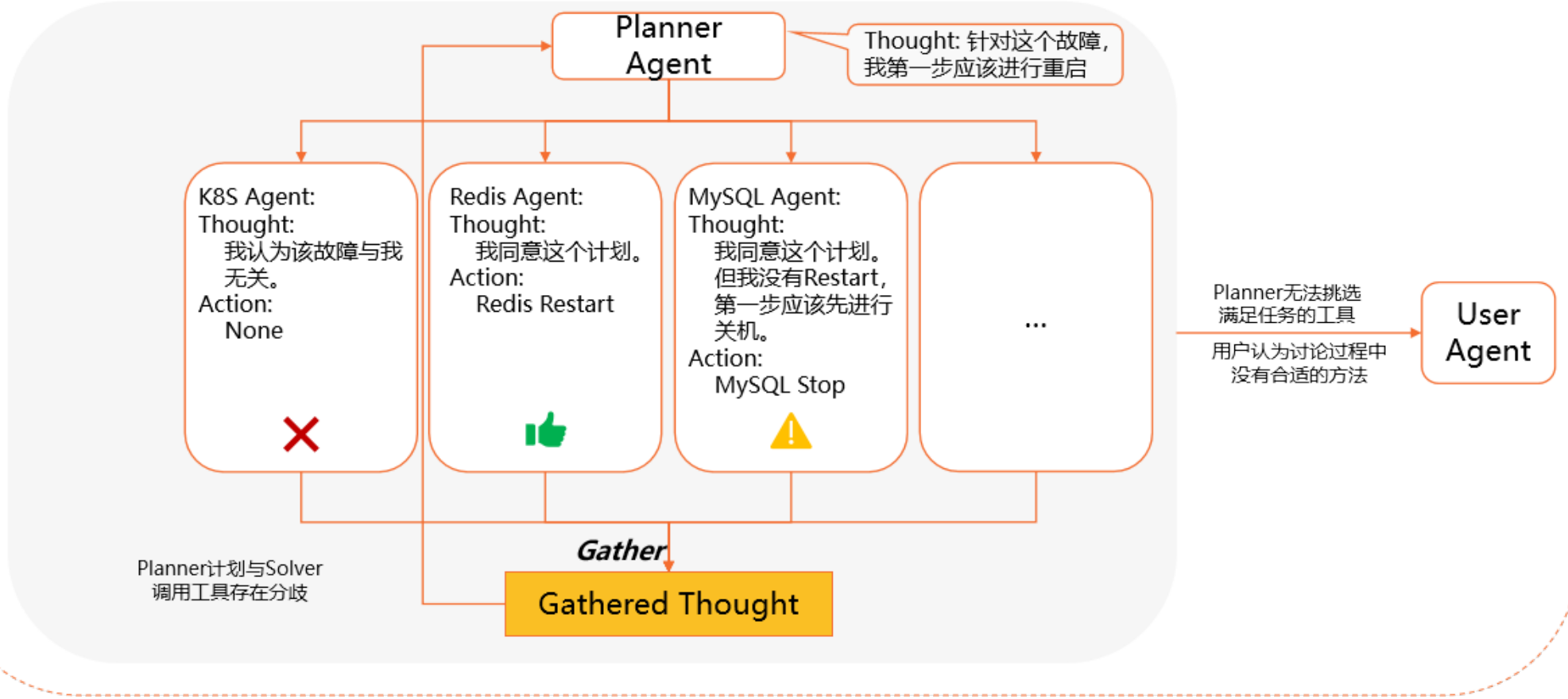
基于多智能体的故障定位



基于多智能体的故障定位

3. 圆桌讨论阶段：

Planner与相关的Solver加入圆桌会议，Solver与Planner讨论，根据自身能力（Tools）分拆任务，直到单个任务可以通过工具一步解决。
如果Planner无法挑选满足任务的工具或用户在讨论过程中认为没有合适的方法，则交给User Agent由用户处理



总结

- 大模型落地仍面临众多问题
  - “产品经理”思维
  - 模型能力+软件能力
  - 融合大模型的整体系统架构
  - 以终为始
- 大模型只是工具，不能为了大模型而大模型
  - 如同AIOps算法，场景、数据、迭代永远重要！

Thanks

高效运维社区

BizDevOps 社区

荣誉出品

GOPS 2025
Shenzhen

10th ANNIVERSARY
2015-2025

T H A N K S

感谢大家观看

2025.4