TAAF：一种协同知识图谱与大语言模型的轨迹抽象与分析框架

Alireza Ezaz 布鲁克大学 计算机科学 圣凯瑟琳斯，安大略省，加拿大
sezaz@brocku.ca

Ghazal Khodabandeh 布鲁克大学 计算机科学 圣凯瑟琳斯，安大略省，加拿大
gkhodobandeh@brocku.ca

Majid Babaei 麦吉尔大学 计算机科学 蒙特利尔，魁北克省，加拿大
majid.babaei@mcgill.ca

Naser Ezzati-Jivan 布鲁克大学 计算机科学 圣凯瑟琳斯，安大略省，加拿大
nezzatijivan@brocku.ca

arXiv:2601.02632v1 [cs.SE] 6 Jan 2026

摘要

执行跟踪是理解、调试和优化复杂软件系统的重要信息来源。然而，来自操作系统内核或 Chrome、MySQL 等大型应用程序的跟踪记录规模庞大，难以分析。现有工具依赖预定义分析，而获取定制洞见往往需要编写领域特定脚本，这既容易出错又耗时。本文提出了 **TAAF**（跟踪抽象与分析框架），这是一种新方法，结合时间索引、知识图谱（知识图谱）和大语言模型，将原始轨迹数据转化为可操作洞见。TAAF 根据轨迹事件构建时间索引知识图谱，以捕捉线程、中央处理器和系统资源等实体之间的关系。随后，大语言模型解读查询特定子图，以回答自然语言问题，从而减少人工检查和深厚系统专业知识的需求。为评估 TAAF，我们推出了 **TraceQA-100**，这是一个包含 100 个基于真实内核轨迹的问题的基准测试。在三种大语言模型和多种时间设置下进行的实验表明，TAAF 可将答案准确率提升至多 31.2%，尤其是在多跳和因果推理任务中。我们还进一步分析了图谱支撑推理在哪些方面有所助益、又存在哪些局限，为下一代轨迹分析工具奠定了基础。

1 引言

执行跟踪是理解、调试和优化软件系统的重要证据来源。它们捕获整个技术栈中的细粒度时间遥测数据，范围涵盖用户级函数调用和服务间消息，以及系统调用、上下文切换和中断处理等底层内核事件。Zipkin [4]、Jaeger [1]、LTTng [2] 和 Trace Compass [3] 等工具提供了对此类数据的访问，但要大规模解读这些数据仍然很困难。有效分析需要深入掌握操作系统内部机制、硬件调度或文件系统行为。现有工具的灵活性有限。用户必须依赖僵化的图表，或使用底层应用程序接口编写自定义脚本；这一过程容易出错，也让非专家难以使用。

已获准在 ICSE 2026 上发表。数字对象唯一标识符：10.1145/3744916.3787832。

有几个难题使轨迹分析难以扩展。跟踪记录规模庞大，即使是较短的运行也可能产生数亿个事件，无法全部载入内存，也超出大语言模型（LLM）的上下文窗口。它们还具有多维特性：事件涉及线程、进程、文件、套接字和中央处理器，并随时间相互作用。最后，推理往往需要关联长时间跨度内的事件，并在缺乏语义提示的情况下解读原始数值字段。即使是“为什么关闭了这个文件？”这样简单的问题，也可能需要大量人工分析。这些局限使轨迹分析难以普及。为明确问题，我们先从一个常见的轨迹分诊场景入手，介绍一个简短的实际操作情境。

动机性场景。 一位性能工程师正在调查负载测试期间 CPU_1 上反复出现的延迟尖峰。预热阶段已经结束，而尖峰会在短时间内再次出现。分诊的第一步是判断 CPU_1 是否由单个线程主导，还是由多个线程共同占用。工程师隔离出尖峰出现的时间段 $[t_1, t_2]$，并提出以下问题：

- 在 t_1 至 t_2 期间，哪个线程在 CPU_1 上运行的时间最长？
- 在同一时间段内，有多少个不同线程在 CPU_1 上运行？占用时间最长的线程是否占据了大部分运行时间？

这些检查体现了常见且耗时的任务，因为它们能揭示不平衡和迁移模式，而这些因素决定了下一步诊断方向。在这里，团队会遵循操作手册：先找出受影响的中央处理器上占用最多的对象，再衡量多样性，以决定是继续调查单个线程，还是排查系统级压力。手动完成这两项检查都很耗时，因为分析人员必须精确计算指定时间段内每个线程的运行时间总和，并考虑线程迁移。既有工作报告称，每次调查耗时 15–45 分钟 [23]。这只是此类调查的一个例子。其他事件也可能需要类似的时间段限定检查，例如在 CPU_0上进行同样的一组检查，或在 10 秒的时间窗口内执行相同的流程。

基于这一动机，我们提出了**TAAF**（基于基础模型的轨迹抽象与分析），这是一个三层框架，旨在实现语义上有意义且可扩展的轨迹分析。TAAF 并不

将原始数据输入大语言模型，而是构建一种紧凑的结构化抽象。时序状态系统将数百万个事件概括为带有时间索引的结构化转移；针对查询的知识图谱组织实体及其关系；大语言模型则对图进行解读，以回答自然语言问题。这种模块化设计实现了可解释推理，提高了可扩展性，并减少了复杂系统场景中的幻觉。

我们的研究探讨了状态系统和知识图谱等结构化抽象如何连接底层事件数据与高层问题。具体而言，我们提出以下问题：(1) 知识图谱如何对跟踪记录的时序性和多维特性建模？(2) 索引化跟踪输出能否提升大语言模型基于原始日志进行推理的能力？(3) 将知识图谱与大语言模型结合，会如何影响轨迹分析的准确率和可解释性？这些问题为 TAAF 的设计与评估提供了指引。

我们的贡献如下：

- 我们提出 **TAAF**，这是一种结合时序抽象、知识图谱和大语言模型推理的三层框架，用于语义跟踪分析。
- 我们提出一种方法，将内核轨迹转换为按时间索引的图，以编码实体交互和时序结构。
- 我们推出 **TraceQA-100**，这是一个包含 100 道由专家编写、基于真实 Linux 跟踪记录的轨迹分析问题的基准测试，重点关注与调度行为相关的线程–CPU 交互和活动模式。
- 我们通过实证表明，与原始输入或扁平化输入相比，图谱支撑推理在 GPT-4o、GPT-4.1 nano 和 o4-mini（推理）上均能提升答案准确率，最高可达 31%。

我们使用真实的 Linux 跟踪记录评估该方法，将轨迹划分为不同的时间位置和持续时间，以评估其可扩展性。实验涵盖了多种大语言模型和接地策略。TAAF 始终优于基线方法，表明结构化接地能够同时提升推理准确率和鲁棒性。我们还以匿名方式发布了所有代码、数据和评估材料，以支持可复现性和后续研究[1]。

2 相关工作

我们回顾与 TAAF 相关的四个关键领域的既有工作：跟踪抽象与有状态建模、用于系统理解的知识图谱构建、用于软件推理的大语言模型，以及衔接结构化技术与神经网络技术的集成策略。

2.1 跟踪抽象与有状态建模

执行跟踪通常规模庞大且语义密集，使得直接分析成本高昂且难以进行。既有工作提出了多种抽象技术，在缩减跟踪记录规模的同时保留可解释性。Pirzadeh 等人 [25] 依据格式塔原则将大型跟踪记录划分为不同的执行阶段，并采用分层抽样，以支持可扩展的探索。Feng 等人 [11] 提出了 *Sage*，这是一种层次抽象框架，能够从反复出现的底层模式中捕捉高层行为。Hamou-Lhadj 和 Lethbridge 提出了基于提示短语和频率分析的摘要生成方法 [12], ，以及 *Utilityhood* 指标 [13] ，用于根据结构相关性对组件进行排序。Cornelissen 等人 [9]对多种跟踪记录缩减技术进行了定量评估，包括采样、限制栈深度和分组。他们考察了这些方法在保留行为保真度方面的权衡。

作为对这些方法的补充，Trace Compass 平台 [14, 17, 29] 通过其 状态系统提供有状态抽象，该系统维护一个按时间索引的属性值区间数据库，以便高效查询。该系统通过事件重放和区间树构建持久化的轨迹状态模型，从而能够大规模提取系统级行为 [31, 34]。尽管这些技术提供了强大的索引和摘要生成功能，但它们缺乏语义结构，并且难以支持针对多维轨迹实体的灵活、由用户驱动的查询。

2.2 用于执行上下文建模的知识图谱

知识图谱（KG）提供了语义丰富的抽象，用于表示系统实体及其关系。在跟踪场景中，知识图谱能够以可解释且机器可读的形式对进程、线程、文件和中央处理器之间的复杂交互进行建模。Liang 等人 [21] 将知识图谱分为静态、时序和多模态类型。静态知识图谱表示三元组（e_1，r，e_2），不随时间演变；时序知识图谱引入不断演变的快照；多模态变体则纳入指标或图像、视频等其他数据类型。基于知识图谱的推理可以是传导式的，即受限于已知实体；也可以是归纳式的，支持泛化。尽管知识图谱已应用于网络安全和软件架构等领域，但很少有系统能从执行跟踪中动态构建知识图谱。TAAF 通过从有状态的跟踪输出中派生按时间索引、针对各个查询的知识图谱，填补了这一空白。

2.3 面向结构化软件推理的大语言模型

GPT-3 [8]和 GPT-4 [6]等大语言模型在翻译 [18], 摘要生成 [20], 和问答 [28]等自然语言处理任务中展现出强大的能力。然而，将其应用于轨迹分析面临两大障碍。首先，跟踪数据量往往超过大语言模型的令牌限制，使端到端摄取难以实现。其次，大语言模型以隐式且不透明的方式编码知识，导致结果难以验证，推理步骤也难以理解 [24]。这常表现为幻觉 ，即答案听起来似乎合理，实际却不正确 [16]。此外，大语言模型难以处理图或表格形式的状态历史等结构化输入，因此在需要显式因果推理的领域中作用有限。

2.4 大语言模型–知识图谱集成策略

为解决大语言模型的局限性，近期研究探索了将知识图谱整合进语言模型处理流程的混合方法。一种主流策略是检索增强生成（*RAG*），通过在推理时检索事实准确的子图，为大语言模型提供依据 [7, 10]。另一类研究则将知识图谱结构直接嵌入模型

[1] https://anonymous.4open.science/r/TAAF-LLM-KG-State-System--60EC/README.md

架构中，使用实体感知嵌入 [32, 35] ，或 QA-GNN 等图神经网络模块 [33]。第三类技术侧重于采用适配器 [30], 、低秩更新 [15], 或指令微调提示 [26], 以实现参数高效的领域适配。这些集成方式提升了事实准确率和可解释性，但也带来了延迟和对齐方面的挑战，尤其是在处理大型时序轨迹时 [19, 27]。近期研究尝试采用压缩历史树 [17], 以降低检索的内存和运行时间成本；这也促使 TAAF 等方法将高性能索引与结构化处理及大语言模型推理相结合。

3 方法

3.1 问题设置与动机

现代内核执行轨迹会记录数百万个带时间戳的事件，涉及线程、进程、中央处理器、文件和网络接口等各种实体，提供丰富的运行时洞察。然而，要从如此庞大且底层的数据中回答实际的轨迹分析问题，仍面临三项核心挑战。首先，分析人员必须在不同时间粒度上进行推理，从瞬时状态到细粒度区间（例如 1 秒窗口）以及完整轨迹的趋势。其次，许多查询涉及多维交互，例如跨调度行为、输入/输出流和资源争用进行推理，这需要解读复杂的实体间关系。第三，轨迹数据缺乏高层语义标注，而分析人员的查询既包括简单的事实查找，也包括比较性或因果问题，因此需要灵活而精准的语义理解。我们还希望了解，借助大语言模型和知识图谱，我们能在轨迹分析方面做到什么程度。

为应对这些挑战，我们提出了基于基础模型的轨迹抽象与分析框架（TAAF），该框架整合了可扩展的数据抽象、结构化知识表示和自然语言推理，从而实现灵活、准确且可解释的轨迹分析。图1展示了该框架的整体架构。

3.2 架构概述与从轨迹到答案的流水线

TAAF 是一个用于结构化轨迹分析的三层抽象框架。TAAF 不会直接分析原始内核跟踪事件，而是将其系统地转换为结构化、可解释的表示，以便进行结构化推理和基于大语言模型的回答。

总体而言，该系统定义了一个从轨迹到答案的转换流水线 T $\rightarrow$S $\rightarrow$G $\rightarrow$A，其中 T 是原始轨迹，S 是状态系统，G 是知识图谱，A 是最终答案。每一层都旨在限定范围、保留结构，并以模型可用的形式呈现轨迹语义。

- **第 1 层：从轨迹到状态**。原始事件流 T 通过确定性编码函数转换为结构化、按时间索引的状态系统 $S = (Q, H)$：

$$\mathcal{T} \xrightarrow{\phi_1} \mathcal{S}$$

这种转换将系统属性及其状态转换组织为同步树，从而实现可扩展的时序推理。

- **第2层：从状态到图**。给定自然语言查询 q，TAAF 从 S 中提取相关的时间区间，并构建语义知识图谱 G_q，以表示查询范围内的系统交互：

$$\mathcal{S}, q \xrightarrow{\phi_2} \mathcal{G}_q$$

该图针对特定查询构建，简洁精炼，并融入了时间和语义元数据。

- **第 3 层：从图到答案**。图 G_q 及其关联的模式提示 C_q 被序列化后传递给大语言模型，由其生成基于语义落地的自然语言答案 A_q：

$$\mathcal{G}_q, C_q, q \xrightarrow{\phi_3} \mathcal{A}_q$$

这一基于大语言模型的推理阶段支持定量分析、比较，以及基于轨迹语义的解释。

这些中间表示经过结构化处理和语义标注，能够捕捉系统行为，并支持不同轨迹类型间的推理、可组合性和可扩展性。

我们将在评估部分（第 4 节）重新审视这一分层设计，以评估其对可扩展性、语义落地和答案质量的影响。

下面我们将更详细地介绍各层。

3.3 层 1：从原始跟踪到状态系统抽象

内核跟踪 T 定义为按时间排序的细粒度系统事件集合：

$$\mathcal{T} = \{(e_i, t_i, c_i) \mid e_i \in \mathcal{E},\ t_i \in \mathbb{T},\ c_i \in C\}$$

每个元组由事件类型 e_i、时间戳 t_i以及一组涉及的系统组件 c_i 组成，例如线程、中央处理器、锁或 I/O 设备。这些事件反映了操作系统的底层行为，包括上下文切换、文件操作和同步事件。尽管这类跟踪记录表达能力很强，但数据量庞大且语义稀疏。即使只有几秒钟的系统活动，也可能产生数千万个事件；由于时间复杂度、交互实体的多维性以及缺乏结构化语义等因素，直接进行推理并不可行。

为解决这一问题，我们定义了一种形式化变换 ϕ_1，将原始跟踪 T 转换为一种时间索引结构化抽象，称为状态系统。状态系统最初由 Montplaisir 等人提出 [23], ，通过将跟踪组织为两种相互同步的树结构——属性树 和历史树——实现可扩展查询与语义丰富化。

形式上，状态系统定义为 $S = (Q, H)$，其中：

- $Q = \{q_1, q_2, \ldots, q_n\}$ 是与层次属性路径（例如 /中央处理器/2/线程/5130）对应的稳定整数标识符（夸克）集合，
- H 是历史树，用于记录值转换，如下所示：

$$H = \{t_{\text{start}}, t_{\text{end}}, q, v \mid q \in Q, v \in \text{V}\}$$

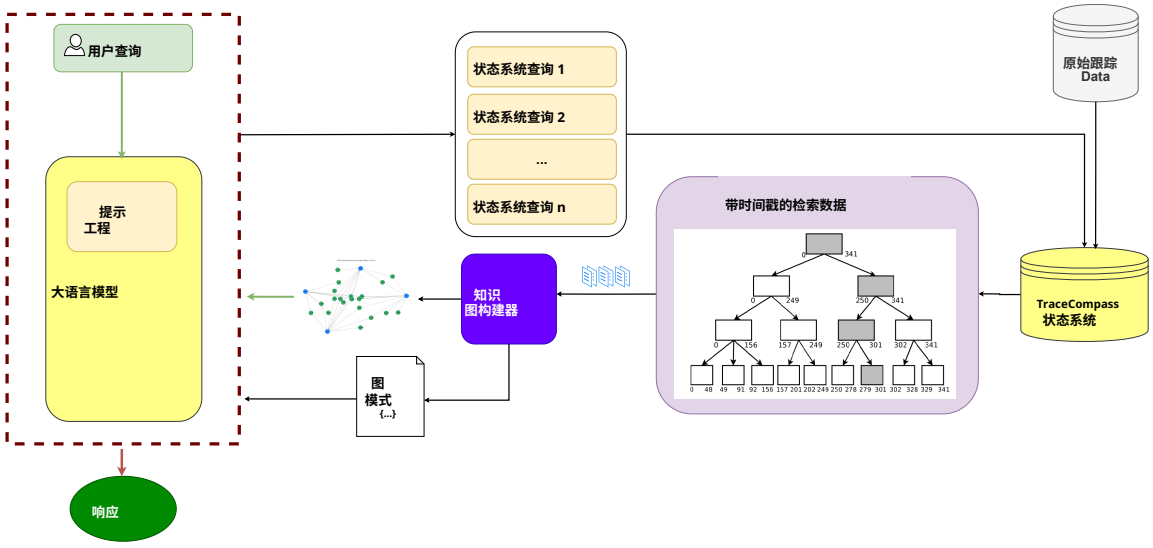

图1：　轨迹抽象与分析框架（TAAF）的端到端架构概览　.

每个条目都表示属性 q 在时间段 $[t_{\text{start}}, t_{\text{end}}$ 内的值为 v。

从原始事件到该结构的转换由一组预定义的、针对特定事件类型的函数控制。对于每种事件类型 $e \in \mathrm{E}$，我们定义一个映射：

$$\phi_1^{(e)} : \mathbb{T} \times C \to \mathcal{P}(H)$$

该映射描述了如何根据事件的时间戳和涉及的组件，将类型为 e 的事件映射为 H中的零个或多个条目。完整的转换 ϕ_1 是这些专用规则的并集：

$$\phi_1(\mathcal{T}) = \bigcup_{(e_i,t_i,c_i)\in\mathcal{T}} \phi_1^{(e_i)}(t_i, c_i)$$

这种表述既体现了事件到状态更新的确定性，也保留了不同事件类型之间的模块化特性。

这种双树设计支持高效的结构化查询，例如：

$$\text{query}(q, t) \to v \quad \text{and} \quad \text{query}(q, [t_1, t_2]) \to \{v_i\}$$

从而能够针对细粒度的执行区间进行局部推理，而无需扫描完整轨迹。

通过以这种形式紧凑地编码时间动态，状态系统能够以相对于轨迹大小 $[17, 23, 34]$ 的对数时间执行更高层次的语义查询。因此，该层将 T 转换为语义增强的结构 S，为 TAAF 中的下游推理任务奠定基础。

3.4 第2层：从状态系统到查询专用知识图谱

给定由ϕ_1生成的结构化状态表示 $\mathrm{S} = (Q, H)$，TAAF 的第二层会构建一个动态的查询专用知识图谱，以结构化且可解释的形式为跟踪语义提供依据。此转换由一个函数处理

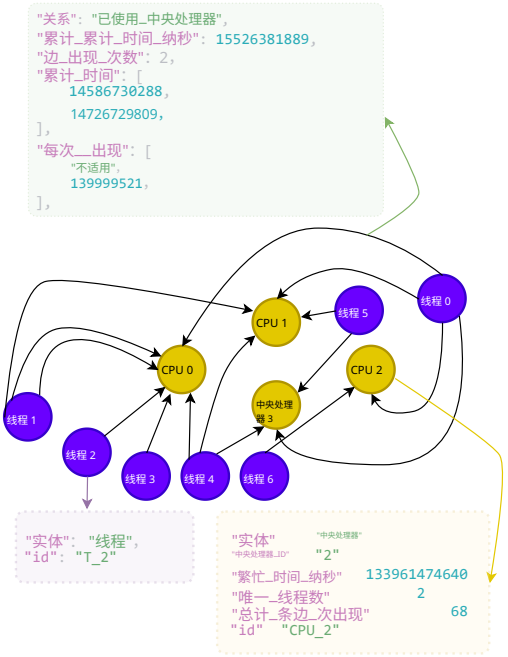

图2：由 1 秒轨迹区间生成的查询专用知识图谱示例。

ϕ_2以状态系统 S 和自然语言查询 q 作为输入，并生成查询专用知识图谱 G_q。

形式化地，我们将 G_q 定义为一个四元组：

$$\mathcal{G}_q = (V_q, E_q, \mu_q, \tau_q)$$

其中 V_q 表示代表系统实体的类型化节点集合，而 $E_q \subseteq V_q \times \mathrm{L} \times V_q$ 表示用于编码这些实体之间语义关系的带标签的有向边集合。标签

空间 L 包含诸如executes_on、reads_from 和 holds_lock 等关系类型。每条边 $e \in E_q$ 都通过函数 $\mu_q : E_q \to \mathbb{R}$ 赋予权重，通常表示持续时间或频率等定量值。时间范围通过映射 τ_q : $V_q \cup E_q \to \mathbb{T}$ 编码，该映射将每个实体或交互与一个从 H得出的时间窗口相关联。

该知识图谱按需构建，范围严格限定于查询 q 以及轨迹中的相关时间区间。例如，如果 q 询问："线程 5130 在 t_1 与 t_2 之间于 CPU 2 上运行了多久？"，系统会从 Q中识别相关夸克，查询它们在 H中的关联时间区间，并将结果映射为包含两个节点（分别表示线程 5130 和 CPU 2）的图，以及一条标记为 executes_on 的单向边；该边以总运行时间为权重（例如，$\mu(e) = 5.25$ 秒），并限定于时间区间 $\tau(e) = [t_1, t_2]$。图2展示了根据针对 1 秒轨迹窗口的查询生成的知识图谱示例。

通过仅提取 q所需的实体和关系，Gq避免了全局图或静态图所存在的过度泛化和可解释性问题。它以轻量、可解释且语义一致的形式支持结构化推理。边采用有向形式以表示因果关系，并根据状态系统提供的量化指标赋予权重。该图针对每个查询量身定制，并可通过添加元数据进行扩展，从而支持基于领域知识、聚焦且可量化并具有适应性的分析。

此步骤完成了 TAAF 的语义接地层，将时间序列轨迹数据转换为适用于大语言模型推理的结构化图格式。

3.5 第 3 层：通过大语言模型从知识图谱到答案

TAAF 的最终层在结构化表示 Gq 上执行语义推理，以生成自然语言答案 Aq。这一转换由函数 ϕ_3控制，其输入包括知识图谱 Gq、自然语言查询 q以及模式提示 Cq，后者明确编码了图元素的含义。输出是一个基于依据的、便于人类理解的答案：

$$\mathcal{A}_q = \phi_3(\mathcal{G}_q, C_q, q)$$

为最大限度地提高大语言模型响应的事实准确率和可解释性，TAAF 采用结构化提示策略。模式提示 Cq 正式定义了 Gq 中节点和边的类型及其属性（例如执行时间、文件路径），以及任何其他特定领域的提示信息。图 Gq 本身以结构化 JSON 格式进行序列化，同时保留图的拓扑结构和带注释的元数据。随后，这些组件与查询 q 一起，以模板化格式作为输入传递给大语言模型：

> { "schema": {...}, "graph": {...}, "**用户查询**": "
> 在 t_1 和 t_2之间，哪个线程在 CPU 1 上执行的
> 时间最长？" }

因此，该模型明确立足于范围界定清晰的语义上下文，从而最大限度地减少对开放式推理的需求。结果是答案 Aq，其中可能包含直接结果。此外，根据查询类型和模型能力，答案还可以包含论证或比较性洞见。

例如，针对上述用户查询，模型可能返回：

> "线程 *9127* 在 *CPU 1* 上执行了 *6.4* 秒，介于
> t_1和 t_2之间，是所有线程中最高的
> 执行时间。"

该输出展示了 TAAF 框架的核心优势。通过将跟踪理解分为结构化抽象层，并将推理延后交由基于这一结构化输入运行的大语言模型完成，我们无需依赖静态规则或脆弱的启发式方法，即可实现稳健且可解释的轨迹分析。

值得注意的是，TAAF 中的大语言模型接口旨在与具体模型无关。在我们的实现中，我们同时采用了通用基础模型和推理调优基础模型（例如 o4-mini），并观察到模式条件化输入能够显著提升事实准确性。此外，提示构造过程透明且可审计，从而更易于调试推理失败，或将系统扩展到新的查询类型或领域。

通过将结构与基础模型的推理能力相结合，这一最终层完善了从轨迹到答案的流水线，使 TAAF 能够灵活满足性能诊断、行为解释和根因调查等多种分析需求。

4 实证评估

本节详细介绍实验设计、基准测试构建、评估协议、评分指标、研究问题以及评估结果，这些内容共同用于评估 TAAF 的准确率、鲁棒性和可解释性。

4.1 评估框架：TraceQA-100

尽管利用人工智能和大语言模型进行轨迹分析已取得进展，但业界仍缺少用于内核级执行轨迹推理的公开真实标注数据集。现有的大语言模型基准测试（如 MMLU 或 Big-Bench）并未涉及轨迹分析所面临的独特挑战，包括细粒度时间、多实体交互和数值聚合。为填补这一空白，我们推出了 TraceQA–100，这是一个经过精心整理的基准测试，旨在考察这三个维度。

轨迹来源与切片。我们使用 SciMark 2.0 Java 基准测试的 LTTng 跟踪记录 [5], ，该基准测试会生成约 34M 个内核事件。对于每次运行，我们围绕三个典型时间位置（开始、中间和结束）提取 1s、10s 和 100s 的切片，生成一组轨迹片段，作为构建问题的事实依据。

问题编制。 问题通过四步双盲流程生成。两位专家独立检查轨迹切片，并拟定需要结合时间信息进行多实体推理的问题。草稿经同行评审后，统一为三种答案格式：解释型问题、多项选择或判断题。第三位专家在不知晓草稿内容的情况下，使用手写的 Trace Compass 脚本生成参考答案；意见分歧通过裁决解决。随后，根据推理范围，将每个问题标记为单跳或多跳 。

最终形成的问题集共包含100道题，其中40道解释型问题、30道多项选择题和30道判断题，比例均等；跳数范围也各占50%。由于每个提示都应用于9个不同的轨迹切片（3个时间位置 × × 3个时间窗口长度），该基准测试共包含 $100 \times 9 = 900$ 个独特的问题–轨迹片段对。代表性示例见表1。

模式与可复现性。TraceQA-100中的每个条目都包含文本提示、跟踪标识符、答案格式、参考答案及相关元数据。所有答案脚本均随项目提供，并作为持续集成流水线的一部分重新执行，以验证跟踪一致性。我们还提供了详细的标注指南，以支持将其扩展到SciMark或Linux之外的领域。

据我们所知，TraceQA-100 是首个将大规模内核轨迹与专家验证的问答任务相结合的公开基准测试。它着眼于时间、结构和数值推理，为未来采用结构化或神经方法进行跟踪理解提供了可重复使用的参考基准。

4.2 评估设置和指标

TraceQA-100 中的每个条目都是独立的问答任务，由自然语言提示、SciMark2.0 轨迹的一个时间切片以及一个标准答案组成。我们在三种跟踪表示设置下评估每项任务（表2）：仅事件 变体以流的形式提供原始内核事件，作为定性基线；基线 使用时间索引的数值输出；而 **TAAF** 则在此基础上，利用 ϕ_2构建的查询专用知识图谱进行增强。

为考虑大语言模型随机性，我们遵循最佳实践 [22]，对每种（问题、表示、模型）组合进行三次采样。对于 100 个基准问题，每种配置可得到 300 个评分回答。在三个模型、三个时间窗口（1秒、10秒、100秒）和两种结构化输入（基线、TAAF）下，核心网格共计 5,400 个标注输出。需要注意的是，仅事件运行仅用于定性比较，因为这类运行通常会超出大语言模型上下文窗口，因此不属于可扩展评估的一部分。

所有运行均在单个 GPU节点上执行。对最大的 100秒跟踪进行端到端处理（包括索引、知识图谱构建和推理）可在 40 秒内完成，证明其适用于交互式使用。

每个响应都按 3点评分量表进行标注：**0** 表示答案错误或无关；**0.5** 表示响应部分正确，但存在轻微错误或推理不完整；**1** 表示答案完全正确，且与标准答案一致。

我们报告两项指标。**准确率** 是按权重计算的得分平均值：

$$\text{Accuracy} = \frac{(N_0 \cdot 0) + (N_{0.5} \cdot 0.5) + (N_1 \cdot 1)}{N_{\text{total}}} \times 100 \tag{1}$$

其中，N_0 $N_{0.5}$ N_1 $N_{\text{total}} = 300$„表示各标签的计数（总数为 ）。

一致性 反映模型在重复采样时的稳定性。我们计算熵：

$$E = -\sum_{i \in \{0,0.5,1\}} P_i \log_2 P_i \tag{2}$$

$$\text{Consistency} = \left(1 - \frac{E}{\log_2 3}\right) \times 100 \tag{3}$$

其中，P_i 是得分 i 的经验频率。一致性越高，表示各次试验的方差越小。

模型配置。 我们通过应用程序接口评估三个模型：**GPT-4.1 nano**，一种紧凑型高吞吐量模型；**GPT-4o**，OpenAI 当前的旗舰模型；以及**o4-mini（推理）**，针对高推理准确率进行了优化。所有模型均使用相同的提示模板和少样本示例。除 RQ7 外，解码温度均固定为 0.5。对于 o4-mini，我们启用推理=高。

跨家族探测。 为评估超越单一供应商的可泛化性，我们还评估了 Google 和 Anthropic 的模型。对于 Google，我们使用 **Gemini 2.5 Pro** 作为旗舰推理模型，使用 **Gemini 2.5 Flash** 作为快速且经济高效的主力模型。对于 Anthropic，我们使用 **Claude Opus 4.1** 作为旗舰推理模型，使用 **Claude Haiku 4.5** 作为快速且经济高效的模型。我们选择这些模型对，是因为每一对都分别代表其供应商的顶级推理模型，以及更小、更注重速度和性价比的模型，而且两家供应商都公开将它们如此定位。四个模型均采用与 GPT 模型相同的提示模板和评分协议。

4.3 研究问题

为指导评估，我们提出了七个研究问题，并将其分为两个阶段。第1阶段探究 TAAF 在不同查询类型、时间窗口和大语言模型变体下的核心性能维度。第2阶段则单独考察模式包含、时间位置和采样行为等具体因素。

阶段 1 – 全网格评估

- **研究问题1:**TAAF 的准确率会如何随不同用户查询类型（如多项选择、判断题和解释型问题）以及不同图问题类型（如单跳与多跳）而变化?
- **RQ2:**时间窗口长度会如何影响模型响应的质量?
- **研究问题4:** 与原始状态系统输出相比，纳入知识图谱在多大程度上能提高准确率?
- **RQ4:** 不同大语言模型后端的性能有何差异，包括一项重点跨家族探测?

阶段 1 涵盖一个 3×3 评估设置网格，由三种大语言模型和三个时间窗口长度（1 秒、10 秒、100 秒）构成，所有数据均从轨迹的中点采样。对于网格中的每个单元，我们分别运行使用查询专用知识图谱的 TAAF 流水线，以及不采用图接地的基线流水线。每种配置涵盖 100 个基准问题，并对每个问题采样三次，从而得到 $9 \times 2 \times 300 = 5400$ 条带标签响应，为研究问题1至研究问题4提供依据。

第2阶段 – 聚焦因素

- **RQ5:** 向大语言模型提供图模式（例如节点类型和特征）能否提高其响应的准确率?
- **RQ6:** 窗口的时间位置（早期、中期、晚期）在多大程度上会影响准确率?
- **RQ7:** 温度参数（采样随机性）会如何影响准确率和响应一致性?

第2阶段 固定模型（GPT-4o）和查询锚定，然后改变关键因素。RQ5 在轨迹中段 1 秒切片中切换模式支持，生成 300 条响应。RQ6 比较时间窗口的位置，即轨迹中的早期与晚期位置，时间粒度为 10 秒，新增 600 条响应。RQ7 通过以四种采样温度（0.1、0.3、0.7、0.9）重新运行轨迹中段10秒切片，考察采样温度的影响，额外生成 1,200 条响应。第2阶段总共新增 2,100 个输出，使完整数据集达到 7,500 条带标签响应，供后续分析使用。

问题类型	跳数范围	示例问题	时间位置	时间窗口	参考答案
解释	单项	线程 5130 在 CPU 0 上累计使用的中央处理器时间是多少?	mid	1 s	5,247,200 纳秒
解释	多项	哪个中央处理器运行过的不同线程数量最多?	mid	100秒	CPU 0_
选择题	单项	线程 2559 主要使用：（A）中央处理器_0（B）中央处理器_1（C）中央处理器_2（D）无	开始	10 s	B
选择题	多项	线程 5236 主要使用：（A）中央处理器_0（B）中央处理器_1（C）中央处理器_2（D）中央处理器_3	end	10 s	C
对或错	单项	对或错？CPU 2 总繁忙时间 > 1e11 纳秒	mid	10 s	错
对或错	多项	对或错？最繁忙的中央处理器运行的不同线程也最多	开始	10 s	True

表1：示例条目　　取自 TraceQA-100，涵盖问题格式、跳数范围、时间位置和时间窗口　　.

变体	传递给大语言模型的输入	实际效果
仅事件	原始内核事件（$\geq 10^7$）	超出上下文限制；生成不连贯的回答。仅保留用于说明抽象的必要性。
基线	状态系统数值	符合预算，但仍处于低层级。
TAAF	特定查询的时序知识图谱	添加了明确的实体和关系；在保持简洁的同时为模型提供依据。

表2：轨迹表示　　提供给语言模型。只有后两种适合当前上下文窗口　　s.

以下将介绍这些研究问题的实证结果。

4.4 阶段1：完整网格实验

研究问题1：TAAF 的准确率在不同用户查询类型以及不同图问题类型（例如单跳与多跳）之间如何变化?

我们首先分析用户查询和图问题的形式如何影响 TAAF 的表现。我们的 TraceQA-100 基准测试包含三种自然语言格式：(1) 判断题，(2) 选择题，以及 (3) 解释型问题。每种格式均设置两种图问题情境：(1) 单跳问题，聚焦于一个中心实体（例如中央处理器）及其局部邻域；(2) 多跳问题，需要对多个不同中心进行推理，例如两个共享线程或迁移线程的中央处理器。

为了更好地理解用户查询风格与图查询结构如何相互作用，我们按问题类别对所有模型的响应进行分组，并可视化每个类别组合中的平均准确率。具体来说，我们汇总三个大语言模型的得分，然后计算每种用户问题类型 与图问题类型组合对应的平均准确率。图3总结了这项实验的结果。在 TAAF 中加入知识图谱后，每个类别组合的准确率都有所提高。提升最大的是判断题多跳查询，准确率上升 +30.12个百分点，达到 79.63%。解释型问题仍然是整体上最具挑战性的问题，但在基线设置下，TAAF 仍使其准确率提升了 23.89%。从绝对值来看，TAAF 设置下判断题单跳查询的准确率最高，达到 90.99%。另一方面，基线设置下解释型多跳查询的准确率最低，仅为 37.22%，反映出这类问题所需的复合推理难度。表3按模型和配置提供了完整的细分结果。

表 3：大语言模型在不同时间间隔和查询类型下，有无知识图谱时的准确率结果。

模型	区间	UQT	GQT	无知识图谱（基线）				使用知识图谱（TAAF）			
				0%	0.5%	1%	Acc%	0%	0.5%	1%	Acc%
GPT 4.1 nano	1 s	解释	单跳	45.00	13.33	41.67	48.33	33.33	8.33	58.33	62.50
GPT 4.1 nano	1 s	解释	多跳	61.67	20.00	18.33	28.33	46.67	20.00	33.33	43.33
GPT 4.1 nano	1 s	MC	单跳	35.56	17.78	46.67	55.56	11.11	5.56	83.33	**86.11**
GPT 4.1 nano	1 s	MC	多跳	40.00	11.11	48.89	54.44	26.67	4.44	68.89	71.11
GPT 4.1 nano	1 s	T/F	单跳	20.00	8.89	71.11	75.56	10.00	2.22	87.78	**88.89**
GPT 4.1 nano	1 s	T/F	多跳	41.11	21.11	37.78	48.33	16.67	12.22	71.11	77.22
GPT 4.1 nano	10 s	解释	单跳	46.67	16.67	36.67	45.00	43.33	5.00	51.67	54.17
GPT 4.1 nano	10 s	解释	多跳	56.67	21.67	21.67	32.50	46.67	13.33	40.00	46.67
GPT 4.1 nano	10 s	MC	单跳	50.00	13.33	36.67	43.33	29.17	5.83	65.00	67.92
GPT 4.1 nano	10 s	MC	多跳	55.67	13.00	31.33	37.83	37.14	4.29	58.57	60.71
GPT 4.1 nano	10 s	T/F	单跳	27.78	9.44	62.78	67.50	11.11	4.44	84.44	86.67
GPT 4.1 nano	10 s	T/F	多跳	55.56	16.67	27.78	36.11	29.17	9.17	61.67	66.25
GPT 4.1 nano	100秒	解释	单跳	49.03	15.97	34.99	42.98	45.10	7.84	47.06	50.00
GPT 4.1 nano	100秒	解释	多跳	59.15	21.13	19.72	30.29	49.30	19.72	30.99	40.85
GPT 4.1 nano	100秒	MC	单跳	43.40	19.81	36.79	46.70	26.61	6.42	66.97	70.18
GPT 4.1 nano	100秒	MC	多跳	48.50	13.50	38.00	44.75	33.58	6.72	59.70	62.06
GPT 4.1 nano	100秒	T/F	单跳	35.63	14.38	50.00	57.19	14.13	6.38	79.50	**82.69**
GPT 4.1 nano	100秒	T/F	多跳	55.10	17.65	27.25	36.07	26.13	10.92	62.95	67.41
GPT 4o	1 s	解释	单跳	30.00	8.00	62.00	66.00	18.00	2.67	79.33	**80.67**
GPT 4o	1 s	解释	多跳	46.00	12.00	42.00	48.00	29.33	9.33	61.33	66.00
GPT 4o	1 s	MC	单跳	26.00	6.00	68.00	71.00	7.00	1.33	91.67	**92.33**
GPT 4o	1 s	MC	多跳	33.33	8.33	58.33	62.50	13.33	3.33	83.33	**85.00**
GPT 4o	1 s	T/F	单跳	13.33	5.00	81.67	**84.17**	4.00	0.67	95.33	**95.67**
GPT 4o	1 s	T/F	多跳	32.00	14.67	53.33	60.67	12.67	3.33	84.00	**85.67**
GPT 4o	10 s	解释	单跳	34.48	11.49	54.02	59.76	33.33	3.45	63.22	64.94
GPT 4o	10 s	解释	多跳	50.00	12.50	37.50	43.75	38.46	11.54	50.00	55.77
GPT 4o	10 s	MC	单跳	34.55	12.73	52.73	58.09	13.64	2.73	83.64	**85.00**
GPT 4o	10 s	MC	多跳	46.43	11.90	41.67	47.62	22.32	3.57	74.11	75.89
GPT 4o	10 s	T/F	单跳	18.27	6.73	75.00	78.37	6.25	2.50	91.25	**92.50**
GPT 4o	10 s	T/F	多跳	41.67	16.67	41.67	50.00	17.44	10.47	72.09	76.32
GPT 4o	100秒	解释	单跳	39.39	12.12	48.48	54.55	29.54	8.41	62.05	66.26
GPT 4o	100秒	解释	多跳	55.17	15.52	29.31	36.07	35.79	14.74	49.47	56.84
GPT 4o	100秒	MC	单跳	39.22	14.71	46.08	53.44	11.76	3.92	84.31	**86.27**
GPT 4o	100秒	MC	多跳	50.88	14.04	35.09	42.11	23.81	5.95	70.24	72.22
GPT 4o	100秒	T/F	单跳	26.92	9.62	63.46	68.27	7.21	2.88	89.90	**91.35**
GPT 4o	100秒	T/F	多跳	47.83	17.39	34.78	43.48	15.85	6.71	77.44	**80.80**
GPT o4-mini	1 s	解释	S-hop	17.78	15.56	66.67	74.44	4.44	5.00	90.56	**93.06**
GPT o4-mini	1 s	解释	多跳	35.24	22.86	41.90	53.33	8.57	14.29	77.14	**84.29**
GPT o4-mini	1 s	MC	S-hop	17.02	8.51	74.47	78.72	3.19	1.06	95.74	**96.27**
GPT o4-mini	1 s	MC	多跳	21.21	14.14	64.65	71.72	5.41	2.70	91.89	**93.24**
GPT o4-mini	1 s	T/F	S-hop	8.33	7.22	84.44	**88.06**	2.22	1.11	96.67	**97.22**
GPT o4-mini	1 s	T/F	多跳	21.75	13.75	64.50	71.88	4.37	2.46	93.17	**94.35**
GPT o4-mini	10 s	解释	单跳	9.52	13.10	77.38	**83.93**	5.95	5.95	88.10	**91.07**
GPT o4-mini	10 s	解释	多跳	27.40	19.86	52.74	62.67	9.59	13.70	76.71	**83.56**
GPT o4-mini	10 s	MC	单跳	13.68	10.26	76.06	**81.19**	3.08	3.08	93.85	**95.38**
GPT o4-mini	10 s	MC	多跳	18.10	11.43	70.48	76.19	5.24	2.86	91.90	**93.33**
GPT o4-mini	10 s	T/F	单跳	3.33	5.00	91.67	**94.17**	0.56	0.56	98.89	**99.17**
GPT o4-mini	10 s	T/F	多跳	14.29	12.14	73.57	79.64	1.78	3.56	94.67	**96.44**
GPT o4-mini	100秒	解释	单跳	12.50	12.50	75.00	**81.25**	6.15	6.15	87.69	**90.76**
GPT o4-mini	100秒	解释	多跳	25.64	17.95	56.41	65.38	7.69	12.82	79.49	**85.90**
GPT o4-mini	100秒	MC	单跳	17.39	11.59	71.01	76.81	4.96	2.48	92.56	**94.05**
GPT o4-mini	100秒	MC	多跳	22.22	11.11	66.67	72.22	5.88	2.94	91.18	**92.65**
GPT o4-mini	100秒	T/F	单跳	6.58	7.89	85.53	**89.47**	1.54	1.92	96.54	**97.50**
GPT o4-mini	100秒	T/F	多跳	18.25	13.14	68.61	75.18	2.55	3.63	93.83	**95.64**

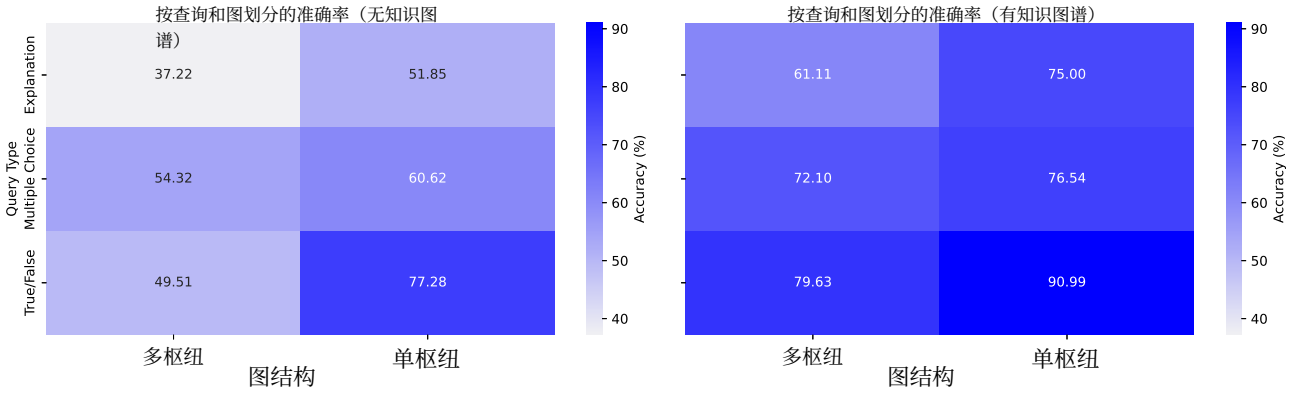

图3：准确率按　　查询类型和跳数划分。左：基线；右：TAAF。白色文字显示准确率提升（+个百分点）　　.

研究问题1的关键观察结果。

- 判断题单跳问题的整体准确率最高（90.99%）。
- 多跳查询通常更难，但从图 grounding 中获益更多。
- 尽管取得了显著提升，解释性提示仍然最为困难，尤其是在多跳问题上。

RQ2：时间间隔长度对 TAAF 答案性能有何影响?

接下来，我们考察时间间隔长度如何影响性能。为单独分析时间间隔长度的影响，我们在所有三种大语言模型设置下，分别比较了有无知识图谱时的三个时间窗口（1s、10s 和 100s）（TAAF 与基线）。图4绘制了由此得到的准确率曲线。实心标记表示 TAAF变体，虚线标记表示基线。结果（TAAF）显示出一种清晰且与模型无关的趋势：窗口越宽，准确率越低，但下降速率取决于模型的能力。

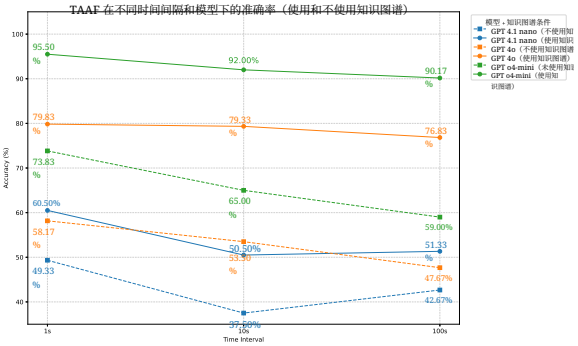

图4：各模型在不同时间间隔下的准确率

数值下降（TAAF设置）。

- **o4-mini** 95.5（总体下降 2.33 个百分点）• **GPT-4o** 79.83 % → 79.33 % → 76.83 %（下降 3.0 个百分点）
- **GPT-4.1 nano** 60.5 % → 50.5 % → 51.33 %（–9.17 个百分点）

数值下降（基线设置）。

- **o4-mini**73.83 % → 65.0 % → 59.0 %（总体下降 14.83 个百分点）
- **GPT-4o** 58.17 % → 53.5 % → 47.67 %（下降 10.5 个百分点）•
- **GPT-4.1 nano** 49.33 % → 37.5 % → 42.67 %（下降 6.66 个百分点）

这里有两种力量在相互竞争。更长的时间窗口提供了更丰富的关系证据，但也会扩大状态系统输出或图的规模。o4-mini 和 GPT-4o 等能力更强的模型能够利用额外的上下文，性能下降幅度较小；而轻量级的 GPT-4.1 nano 则会更快地失去落地信号。基线的性能下降更为陡峭，因为它必须处理规模不断增大的原始状态系统数据，而 TAAF 在图规模扩大时，大多只需更新节点和边特征。我们还观察到，蓝色的 GPT-

4 1 50 5 → . nano 曲

51 33 37 5 → 42 67

线有两处小幅回升：一处出现在启用知识图谱时（. . %），另一处出现在基线中（. . %），均发生在时间窗口从 10s 扩大到 100s 时。虽然具体原因仍属推测，有待进一步研究，但一种可能的解释是，较宽的时间窗口使模型能够获取更多累积证据，从而有利于依赖聚合行为的一些查询。此外，GPT-4.1 nano 可能难以处理 10s 的中间规模输入：这种输入过于复杂，无法充分推理，但规模又不足以凸显主导模式。

RQ2 关键观察。

- 较短的窗口最稳妥。 所有模型在 1 秒切片上的表现都达到峰值；超过 10 秒后，噪声累积的影响大于额外上下文带来的收益。• *o4-mini* 仍然稳健。 即使在 100 秒时，它仍保有 90%+准确率，这表明精心设计的推理目标能够应对图扩展。• 基线退化得更快。 在基线设置中，准确率下降速度约为 TAAF 的两到三倍，这凸显了随着时间间隔增大，原始状态系统输出会变得难以驾驭。

RQ3：引入知识图谱后，TAAF 在回答轨迹相关查询时的准确率提升了多少?

第三，我们量化 KG 中的显式结构相比纯数值状态系统数据所带来的帮助。图5量化了在 TAAF 中加入知识图谱上下文后相较于基线所获得的绝对准确率提升。在全部九种模型—时间间隔组合中，KG 始终带来帮助，提升幅度从 GPT-4.1 nano 在 100 秒切片上的 modest + 8.67% 到 o4-mini 在 100 秒切片上的显著 + 31.17% 不等。整个网格的平均增益为 21。5%，这证实了图 grounding 是我们处理流程中影响最大的因素。

图6更详细地说明了准确率如何提升。对于每个模型–区间对，左侧的堆叠条形图表示基线，右侧的堆叠条形图表示 TAAF 变体。知识图谱主要将部分正确或错误的答案转化为完全正确的答案。得分为 0.5 的比例也有所下降，这表明模型并非只是在猜测，而是更频繁地得出明确且正确的结论。

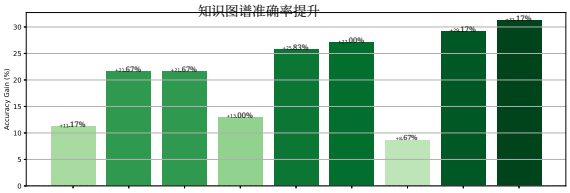

图5：TAAF 中知识图谱带来的准确率提升。

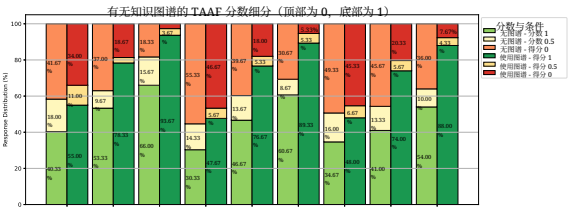

图6：有知识图谱与无知识图谱时的得分细分（0/0.5/1）。

RQ3 关键观察。
• 时间间隔越长，收益越大。 </style}随着时间间隔从 1 秒延长到 100 秒，提升幅度稳步增长，因为多跳关系变得更加密集，也更难仅凭原始数值重建。
• 更强的模型仍能带来提升。即使是更大型的模型（GPT-4o），在 1s 和 100s 时也分别提升约 ＋22% 和 ＋29%，这表明参数化知识无法取代显式结构。• 推理调优模型最能发挥知识图谱的优势。o4-mini 的增量收益最高，这表明轻量级推理目标与以图为基础的设置十分契合。

研究问题4：TAAF 的响应在不同大语言模型后端（包括针对性的跨家族探测）上的准确率和质量有何差异?

最后，在阶段1中，我们探讨观察到的收益是否取决于底层大语言模型。由于相同的实验网格涵盖了全部三种大语言模型，我们可以直接从图4中看出时间间隔长度的影响。

总体而言，采用 TAAF 设置的 o4-mini 在每个时间间隔下都名列前茅，在 1 秒切片上达到 95.5% 的峰值，在 100 秒切片上仍超过 90.17%。GPT-4o 落后约 15 个百分点，而小型 GPT-4.1nano 即使有图支持，准确率仍维持在 50%–60% 区间。采用基线设置时，各模型间的差距缩小，这证实了推理能力更强的模型比能力较弱的模型更能充分利用显式结构。

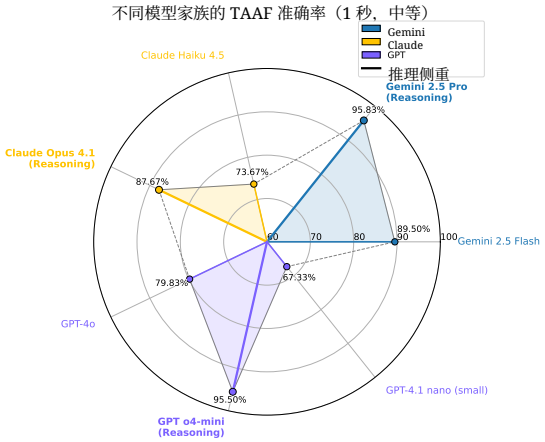

图7：不同模型家族的 TAAF 准确率。

跨家族探测。 为检验 TAAF 的准确率和研究结果是否依赖于单一厂商家族，我们还在相同的提示和评分协议下，对另外两个模型家族进行了有针对性的探测。每个家族分别选取一个旗舰模型和一个推理调优模型，以对应 GPT层级。该探测采用 **TAAF**设置，并使用 **1 秒轨迹中段** 切片，以限制运行时间并隔离模型家族这一因素。准确率分别为：Gemini 2.5 Pro（推理）**95.83%**、Gemini 2.5 Flash **89.50%**、Claude Opus 4.1（推理）**87.67%**、Claude Haiku 4.5 **73.67%**。这一模式与 GPT层级相一致。面向推理的旗舰模型在各自家族中表现最佳，而紧凑型快速模型则以小幅准确率下降换取速度。图 7 更直观地展示了这些数字，并表明准确率源于 TAAF图谱锚定，而非厂商特性。这一结果表明，TAAF 能够跨越不同模型家族发挥作用，并支持其广泛应用。

研究问题4：关键观察
• 采用 *TAAF*设置的 *o4-mini* 领先。 其准确率在每个区间都超过 90%。这表明，在具备图上下文时，明确的推理目标具有重要价值。• 随着结构增强，差距扩大。 一旦提供 KG，GPT-4o 与 GPT-4.1 nano 之间的性能差距便会扩大，这表明更大的模型能够提取更丰富的关系线索。

• 供应商无关模式。 跨家族探测呈现出与 GPT 分层相似的趋势。在匹配层级时，差距始终处于个位数范围内，这表明 TAAF 的图接地是性能提升的主要来源。

4.5 第二阶段：聚焦因素

RQ5：向大语言模型提供图模式（节点类型和特征）是否能提高其响应的准确率?

在确认知识图谱的价值后，我们测试了向模型提供轻量级 JSON 模式是否能进一步改善接地。在运行完整实验套件之前，我们首先询问大语言模型是否真的使用模式中包含的结构提示（节点类型和边特征）。因此，我们在 GPT-4o 中间轨迹、1 秒实验中移除了提示里的模式字符串，但保留了图三元组。表4展示了结果。

表4：传递图模式的影响。

提示变体	0	0.5	1	TAAF 准确率
无模式	20.0 %	16.7 %	63.3 %	71.7 %
使用模式	18.7 %	3.0 %	78.3 %	**79.8 %**↑

RQ5 关键观察。提供轻量级模式可使绝对准确率额外提高 +8.1%；因此，所有后续实验均启用该模式。

RQ6：时间位置（即时间窗口在轨迹中的位置）如何影响 TAAF 准确率?

时间点很重要：我们现在检查在轨迹的早期、中间或后期选取时间窗口时，准确率是否会发生变化。对于 RQ6，我们在 TAAF 设置中固定使用 GPT-4o，并在三个时间位置应用 10秒的分析窗口。我们测试了每条轨迹中该窗口的三种位置。**起始**位置覆盖轨迹开始后 5 秒至 15 秒的区间。**中间**位置覆盖每条轨迹的中点至该中点之后 10 秒的区间。**末尾**位置覆盖轨迹结束前 15 秒至结束前 5 秒的区间。图8展示了这项实验的详情。

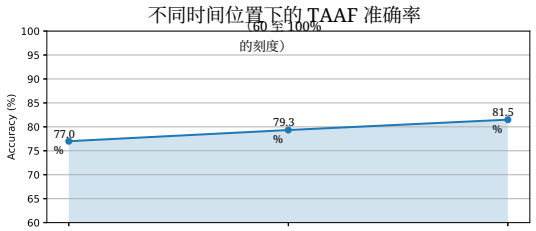

图8：TAAF 在不同时间位置的准确率

如图8所示，准确率从起始位置的 77.0% 上升至中间位置的 79.3%，并在接近末尾的位置达到 81.5%。较晚的时间窗口通常包含更长、更稳定的活动突发，因此更容易发现调度模式。较早的时间窗口涵盖工作负载的预热阶段，瞬态行为也更多。整体差异较小，约为 4%，因此用户几乎可以在轨迹中的任意位置设置时间窗口，仍能获得可靠的结果。

RQ6 关键观察。
• 随着时间窗口从开头移向末尾，准确率略有提升，整体提升了 4.5%。
• 较晚的片段具有更稳定的执行阶段，为模型提供了更清晰的模式。

• 差距较小，表明 TAAF 对时间窗口的位置具有较强的鲁棒性，因此用户无需精细调整时间戳。

RQ7：采样温度如何影响准确率和一致性? 最后，我们探究大语言模型的采样温度对准确率和稳定性的影响。

与 RQ6 中一样，我们在 TAAF 设置中使用 GPT-4o，并采用 10 秒的跟踪记录中段窗口，然后改变控制大语言模型输出随机性的采样温度。测试了五个取值：0.1、0.3、0.5、0.7 和 0.9。对于每个取值，我们都计算了准确率以及第 4.2 节中定义的一致性指标。结果如图9所示。

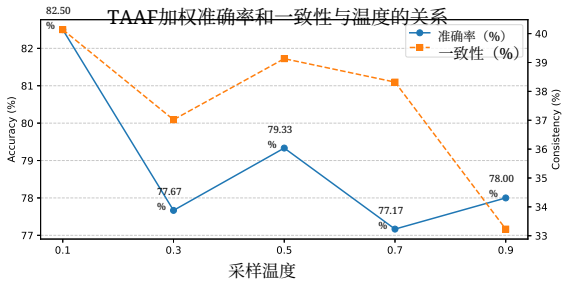

图9：准确率和一致性随采样温度的变化。

根据结果，准确率在最低温度下最高，0.1 时为 82.5%，随后在 0.3 时降至 77.67%，在 0.5 时略有回升，并在 0.7 时再次下降。一致性呈现出类似但

更平滑的曲线：在 0.1 时最佳，在 0.5 附近保持稳定，在 0.9 时最低。低温度可以减少幻觉，但可能导致确定性错误；而温度过高则会引入过多噪声。

RQ7 关键观察。
• 最佳权衡出现在 0.1 至 0.3 之间，此时准确率较高，一致性得分也位居前列。• 温度高于 0.7 时，准确率和一致性都会下降，说明采样过于随机时，模型容易偏离。• 接近 0.5 的适中设置兼顾了多样性与可靠性，因此此前的实验将其设为默认值。

4.6 讨论

两阶段评估表明，在所有设置下，TAAF 均能提高准确率，并且在不同模型、窗口大小和查询风格下保持稳定。图 grounding 提供了明确的结构，短窗口减少了噪声，而轻量级模式则以很小的额外成本增强了接地能力。这些结果证明了第 3 节所采用设计选择的合理性。状态系统支持快速的时间索引访问，查询专用知识图谱使上下文保持精简且相关，而大语言模型则将这些上下文转化为可操作洞见。

我们也指出了性能仍然较弱的方面，例如解释型多跳问题和超长窗口。这些不足为未来的优化指明了方向，也有助于说明下文将讨论的局限性。

在获得实证证据后，我们现在来探讨可能影响本研究有效性的因素。

5 有效性威胁

我们概述可能影响结果解读的潜在局限，并提出未来改进方向。

5.1 构念效度

问题多样性。我们的基准测试包含 100 个根据 SciMark 2.0 跟踪记录手工设计的问题。虽然问题在形式和范围上具有多样性，但可能未涵盖快速上下文切换或深度嵌套的中断链等边界情况。未来工作可以通过自动生成来扩大覆盖范围。

评分分辨率。 我们的三级评分标准 {0, 0.5, 1} 能够衡量总体正确性，但无法区分细微错误（例如，时间差一）。采用更精细的指标（例如 BLEU、数值容差）或许能提升灵敏度。

5.2 内部效度

标注偏差。 作者标注可能会使评分产生偏差，尤其是在面对边界性输出时。采用多名独立评审者并报告标注者间一致性，将有助于提高客观性。

提示长度限制。 某些大型跟踪记录或图会超出大语言模型上下文限制。我们会根据需要截断边属性，这可能会降低答案质量。未来的解决方案包括分层提示或滑动窗口。

5.3 外部效度

轨迹泛化性。 所有问题均基于 Linux 下的 SciMark 2.0。其他内核或工作负载可能会改变行为。需要使用多种跟踪记录（例如 eBPF、高性能计算、移动端）进行更广泛的评估。

模型稳定性。 通过应用程序接口访问的大语言模型可能在不知不觉中发生变化，从而影响可复现性。缓存模型版本和输出结果有助于缓解这一威胁。

范围限制。 当查询限定在一个窗口内时，TAAF 的效果最佳。全局且无界的查询可能生成规模过大的知识图谱，超出现代上下文的处理能力；如果进行过度摘要，还可能破坏时间顺序。对于这类情况，全局预聚合或近似抽样更为合适。

大语言模型作为推理器的局限。 我们观察到，即使知识图谱上下文正确，模型有时仍会在多步聚合任务上失败。例如，查询“哪个中央处理器在线程中其最高累计时间与最低累计时间之差最小”需要执行多个聚合步骤。我们观察到，模型可能会跳过某些步骤，或出现细小的算术错误。

5.4 结论效度

抽样变异性。 每个查询均采样三次。虽然这符合标准做法，但增加重复次数或采用配对检验可提升统计置信度。

基线范围。 我们仅将 TAAF 与原始状态系统进行比较。摘要或静态图等中间设计可进一步帮助理解 TAAF 的优势。

6 结论与未来工作

本文介绍了 **TAAF**，这是一种通过整合时间索引、知识图谱和大语言模型，将低层内核轨迹转化为与查询相关的答案的框架。TAAF 将抽象与结构化上下文相结合以进行推理，从而应对轨迹分析中的关键挑战，包括数据量庞大、时间复杂度、语义歧义以及对领域专长的需求。

在涵盖三种 GPT 模型、三个时间窗口和六种查询类型的 7,800 条响应中，基于图的 grounding 将加权准确率平均提升了 21.5%，在较短时间区间内最高达到 95.5%。这些提升证实，注入显式结构能够增强事实准确性和稳定性，即使对于表现出色的模型也是如此。

评估还发现，较长的窗口和多跳查询仍然具有挑战性，不过像 o4-mini 这样的推理型模型依然表现稳健。TAAF 在早期、中期和后期的轨迹切片中表现稳定，而 0.1 到 0.3 之间的采样温度则能在准确率与多样性之间实现良好的平衡。

展望未来，颇具前景的方向包括整合时序知识图谱以实现持续推理，以及将 TAAF 嵌入自主智能体工作流。后续工作将把该框架应用于生产级系统，以发现真实缺陷，并通过检索增强或分层提示突破当前的上下文限制。另一个互补方向是开展系统性的延迟审查，从端到端分析每个处理流程阶段，同时探索图划分与采样、流式处理或增量知识图谱构建、批处理以及提前退出启发式等优化方法。

参考文献

[1] Jaeger：开源端到端分布式追踪。https://www. jaegertracing.io/。访问日期：2025-05-22。[2] [n. d.]。LTTng：Linux 下一代追踪工具包。https://lttng.org/。访问日期：2025-05-22。[3] [n. d.]。TraceCompass：追踪可视化工具。https://www.eclipse.org/ tracecompass/。访问日期：2025-05-22。[4] [n. d.]。Zipkin。https://zipkin.io/。访问日期：2025-05-22。[5] 2020。SciMark 2.0 追踪数据。https://zenodo.org/record/437170。访问日期：2025-05- 01。[6] Josh Achiam、Steven Adler、Sandhini Agarwal、Lama Ahmad、Ilge Akkaya、Florencia Leoni Aleman、Diogo Almeida、Janko Altenschmidt、Sam Altman、Shyamal Anadkat 等。2023年。GPT-4 技术报告。*arXiv* 预印本 *arXiv:2303.08774* (2023 2303 08774)。https://arxiv.org/abs/。Jinheon Baek、[7] Chingyu Chung、Sung Ju Kim 和 Seung-won Hwang。2023年。KAP-ING：知识感知提示提升抽象摘要生成的事实准确性。*arXiv* 预印本 *arXiv:2305.13952* (2023年)。https://arxiv.org/abs/2305.13952[8] Tom Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared D Kaplan、Prafulla Dhariwal、Arvind Neelakantan、Pranav Shyam、Girish Sastry、Amanda Askell 等。2020年。语言模型是少样本学习者。神经信息处理系统进展33 (2020年)，1877–1901。[9] Bas Cornelissen 和 Leon Moonen。2008年。关于大型执行跟踪与跟踪抽象技术。软件工程研究组，代尔夫特(2008年)。[10] Yihao Feng、Zeyu Chen、Yue Zhang 等。2023年。知识求解器：教大语言模型从知识图谱中搜索领域知识。*arXiv* 预印本 *arXiv: 2305 16581* 2023 2305 16581 。( )。[11] https://arxiv.org/abs/。Yang Feng、Kaj Dreef、James A Jones 和 Arie van Deursen。2018年。用于程序理解的执行跟踪层次抽象。载于第*26*届程序理解会议论文集。86–96。[12] Abdelwahab Hamou-Lhadj。2005年。追踪摘要生成的概念。*PCODA* 论文集(2005年)。[13] Abdelwahab Hamou-Lhadj 和 Timothy Lethbridge。2006年。总结大型跟踪记录的内容，以促进对软件系统行为的理解。载于第*14*届 *IEEE* 国际程序理解会议（*ICPC'06*）。IEEE，181–190。[14] Weijiang Hong、Zhenbang Chen、Yide Du 和 Ji Wang。2021年。基于跟踪抽象的未解释程序验证。载于形式化方法：第*24*届国际研讨会，*FM 2021*，虚拟活动，*2021*年*11*月*20*日至*26*日，论文集第*24*卷。Springer，545–562。[15] Edward Hu、Yelong Shen、Phil Wallis 等。2022年。LoRA：大语言模型的低秩适配。载于国际学习表征会议论文集。https://openreview.net/forum?id=nZeVKeeFYf9[16] Ziwei Ji、Nayeon Lee、Rita Frieske、Tiezheng Yu、Dan Su、Yan Xu、Etsuko Ishii、Ye Jin Bang、Andrea Madotto 和 Pascale Fung。2023年。自然语言生成中的幻觉综述。《计算调查》55，12 (2023年)，1–38。[17] Herve Mbikayi Kabamba、Matthew Khouzam 和 Michel Dagenais。2023年。Node Compass：对基于 JavaScript 的 Web 服务器中的请求执行进行多层级追踪与调试。*arXiv* 预印本 *arXiv:2401.08595* (2023年)。https://arxiv.org/ abs/2401.08595[18] Mike Lewis、Yinhan Liu、Naman Goyal、Marjan Ghazvininejad、Abdelrahman Mohamed、Omer Levy、Veselin Stoyanov 和 Luke Zettlemoyer。2020年。BART：用于自然语言生成、翻译和理解的去噪序列到序列预训练。载于第*58*届计算语言学协会年会论文集。7871–7880。[19] Patrick Lewis、Ethan Perez、Aleksandra Piktus、Fabio Petroni、Vladimir Karpukhin、Naman Goyal、Heinrich Kanal、Sebastian Riedel、Edward Grefenstette、Mike Lewis 等。2020年。面向知识密集型自然语言处理任务的检索增强生成。载于神经信息处理系统进展，第33卷。9459–9474 。[20] Junyi Li、Tianyi Tang、Wayne Xin Zhao、Jian-Yun Nie 和 Ji-Rong Wen。2024年。用于文本生成的预训练语言模型：综述。《计算调查》56，9（2024年），1–39。[21] Ke Liang、Lingyuan Meng、Meng Liu、Yue Liu、Wenxuan Tu、Siwei Wang、Sihang Zhou、Xinwang Liu、Fuchun Sun 和 Kunlun He。2024年。图类型上的知识图谱推理综述：静态、动态和多模态。《*IEEE*模式分析与机器智能汇刊》（2024年）。[22] Gili Lior、Eliya Habba、Shahar Levy、Avi Caciularu 和 Gabriel Stanovsky。2025年。ReliableEval：通过矩估计法进行随机大语言模型评估的方法。*arXiv*预印本 *arXiv*: *2505 22169* 2025 2505 22169 。（）。[23] https://arxiv.org/abs/。Alexandre Montplaisir、Naser Ezzati-Jivan、Florian Wininger 和 Michel Dagenais。2013年。用于查询和可视化从跟踪数据中提取的系统状态的高效模型。载于国际运行时验证会议。Springer，219–234。[24] Shirui Pan、Linhao Luo、Yufei Wang、Chen Chen、Jiapu Wang 和 Xindong Wu。2024年。统一大语言模型与知识图谱：路线图。《*IEEE*知识与数据工程汇刊》（2024年）。[25] Heidar Pirzadeh Tabari。2012年。跟踪抽象框架与技术。博士论文。康考迪亚大学。[26] Parth Sen、Anirudh Ramesh、Pritish Mukerji 等。2023年。用于药物发现的知识增强图机器学习：从精准性到可解释性的综述。*arXiv*预印本 *arXiv:2307.00559*（2023年）。https://arxiv.org/abs/2307.00559[27] Weijia Shi、Sewon Min、Michihiro Yasunaga、Minjoon Seo、Richard James、Mike Lewis、Luke Zettlemoyer 和 Wen-tau Yih。2024年。REPLUG：检索增强黑盒语言模型。载于*2024*年计算语言学协会北美分会会议论文集：人类语言技术。8371–8384。https://aclanthology.org/2024.naacl-long.463[28] Dan Su、Yan Xu、Genta Indra Winata、Peng Xu、Hyeondey Kim、Zihan Liu 和 Pascale Fung。2019年。通过预训练语言模型微调实现问答系统的泛化。载于第*2*届机器阅读问答研讨会论文集。203–211。[29] The Eclipse Foundation。2025年。通用状态系统。The Eclipse Foundation。https://archive.eclipse.org/tracecompass/doc/stable/org.eclipse.tracecompass.doc.dev/Generic-State-System.html[30] Ruize Wang、Duyu Tang、Nan Duan、Zhongyu Wei、Xuanjing Huang、Jian-shu Ji、Guihong Cao、Daxin Jiang 和 Ming Zhou。2021年。K-Adapter：利用适配器将知识注入预训练模型。载于计算语言学协会论文集：*ACL-IJCNLP 2021*年会议论文精选。1405–1418。https://aclanthology.org/ 2021 121 .findings-acl。Florian [31] Wininger、Naser Ezzati-Jivan 和 Michel R Dagenais。2017年。一种用于有状态分析执行跟踪的声明式框架。《软件质量期刊》25（2017年），201–229。[32] Ikuya Yamada、Akari Asai、Hiroyuki Shindo、Hideaki Takeda 和 Yuji Matsumoto。2020年。LUKE：借助实体感知自注意力实现深度上下文化实体表示。载于*2020*年自然语言处理实证方法会议论文集。6442–6454。[33] Michihiro Yasunaga、Hongyu Ren、Antoine Bosselut 等。2022年。QA-GNN：利用语言模型和知识图谱进行问答推理。载于北美计算语言学协会年会。 582 609 2022 43 –。[34] 2023 https://aclanthology.org/.naacl-main。Siwei Zhang。分布式与集中式应用的跟踪可视化。硕士论文。瑞典斯德哥尔摩瑞典皇家理工学院。[35] Zhengyan Zhang、Xu Han、Zhiyuan Liu、Xin Jiang、Maosong Sun 和 Qun Liu。2019年。ERNIE：利用信息丰富的实体增强语言表示。载于计算语言学协会第*57*届年会论文集。1441–1451。