TKG-Thinker: Towards Dynamic Reasoning over Temporal Knowledge Graphs via Agentic Reinforcement Learning

Anonymous ACL submission

Abstract

Temporal knowledge graph question answering (TKGQA) aims to answer time-sensitive questions by leveraging temporal knowledge bases. While Large Language Models (LLMs) demonstrate significant potential in TKGQA, current prompting strategies constrain their efficacy in two primary ways. First, they are prone to reasoning hallucinations under complex temporal constraints. Second, static prompting limits model autonomy and generalization, as it lacks optimization through dynamic interaction with temporal knowledge graph (TKG) environments. To address these limitations, we propose **TKG-Thinker**, a novel agent equipped with autonomous planning and adaptive retrieval capabilities for reasoning over TKGs. Specifically, TKG-Thinker performs in-depth temporal reasoning through dynamic multi-turn interactions with TKGs via a dual-training strategy. We first apply Supervised Fine-Tuning (SFT) with chain of thought data to instill core planning capabilities, followed by a Reinforcement Learning (RL) stage that leverages multi-dimensional rewards to refine reasoning policies under intricate temporal constraints. Experimental results on benchmark datasets with three open-source LLMs show that TKG-Thinker achieves state-of-the-art performance and exhibits strong generalization across complex TKGQA settings.[1]

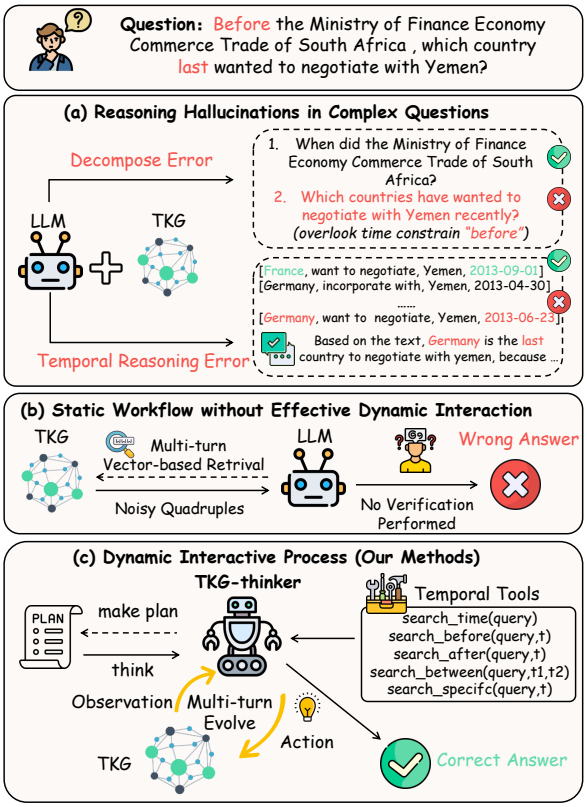

Figure 1: Comparison between **TKG-Thinker** and existing LLM-based methods. TKG-Thinker employs a think–action–observation loop for autonomous interaction with TKGs, enabling verified temporal reasoning.

1 Introduction

Temporal knowledge graphs (TKGs) organize factual knowledge over time and serve as an essential foundation for a wide range of knowledge-driven applications, such as recommendation systems (Li et al., 2025c; Chen et al., 2025) and question answering (Liu et al., 2025b; Gao et al., 2024). In TKGs, facts are represented as quadruples (*subject*, *relation*, *object*, *timestamp*). Building upon this representation, temporal knowledge graph question answering (TKGQA) focuses on answering time-sensitive questions by leveraging the knowledge stored in TKGs. For instance, the question *"Which team did Luka Dončić play for on 2025-02-03?"* can be answered by the quadruple *(Luka Dončić, play for, Los Angeles Lakers, 2025-02-03)*.

Recently, large language models (LLMs) have demonstrated remarkable performance in tackling complex tasks (DeepSeek-AI, 2024; Yang et al., 2025; Peng et al., 2025). Building on this success, recent studies have increasingly focused on exploring the potential of LLMs for addressing TKGQA. For instance, some methods (QianyiHu et al., 2025;

[1] https://anonymous.4open.science/r/TKG-Thinker-51ED

Chen et al., 2024b) employ few-shot prompting to guide LLM-based agents in performing temporal reasoning over TKGs, while others (Gong et al., 2025; Qian et al., 2025, 2024) decompose temporal questions into a sequence of sub-questions and combine retrieval-augmented generation (RAG) mechanisms to support step-by-step reasoning. Despite substantial progress, both paradigms struggle in complex settings for two main reasons.

First, existing LLM-based methods are **prone to reasoning hallucinations when handling complex temporal constraints in TKGs,** including incorrect sub-question decomposition and insensitivity to fine-grained temporal constraints. As illustrated in Figure 1(a), when faced with the question *"Before the Ministry of Finance Economy Commerce Trade of South Africa, which country last wanted to negotiate with Yemen?"*, LLMs often fail to account for critical temporal constraints such as *before* and *last*. As a result, even when the relevant evidence is explicitly available in the temporal context, these methods tend to generate logically inconsistent reasoning steps and ultimately produce incorrect answers. Second, current methods suffer from **limited autonomy and suboptimal generalization** due to their reliance on static, manually-engineered workflows. More fundamentally, these models lack optimization through dynamic interaction with TKG environments, hindering the development of grounded temporal reasoning. As shown in Figure 1(b), this limitation leads to two critical failures: (1) retrievers often provide context that violates temporal constraints, and (2) the lack of internal verification mechanisms prevents LLMs from autonomously detecting and correcting such misaligned evidence during inference.

Regarding the limitations of static workflows, Reinforcement Learning (RL) (Shang et al., 2025; Yue et al., 2025) provides a promising paradigm for shifting to autonomous optimization. By utilizing dynamic reward signals (Shao et al., 2024a), RL allows LLMs to acquire complex reasoning behaviors, such as self-correction and strategic search, which are essential for navigating temporal environments (Jin et al., 2025). Such emergent capabilities provide a robust mechanism to bridge the gap between static retrieval and temporally-grounded reasoning over TKGs, thereby facilitating the mitigation of hallucinations and limited exploration. Motivated by this perspective, we pose the following research question to guide our study: *Can LLMs be effectively trained to autonomously perform dynamic reasoning and retrieval in complex TKGQA scenarios via RL-based optimization?*

To address these challenges, we propose **TKG-Thinker**, a novel agent that reformulates TKGQA as a multi-step interactive process within a dynamic environment. TKG-Thinker performs principled question decomposition and temporal analysis through a two-stage optimization. Specifically, we first employ supervised fine-tuning on a customized dataset with chain-of-thought reasoning paths to equip the model with planning and ReAct-style capabilities (Yao et al., 2023). This stage establishes the fundamental "think–action–observation" loop and effectively alleviates the cold-start problem for subsequent optimization. In the second stage, we optimize TKG-Thinker via RL, formalizing temporal reasoning as a sequential decision-making process. To ensure effective policy optimization, we implement a structured interaction protocol where the agent must provide explicit reasoning steps before executing predefined temporal actions (e.g., planning, time-aware retrieval), ensuring that trajectories are fully observable. Specifically, we employ a multi-objective reward mechanism that incorporates an outcome reward for factual correctness, a format reward for structured reasoning, and a retrieval reward for information coverage. This scheme enables TKG-Thinker to internalize autonomous and dynamic reasoning behaviors in complex TKGQA scenarios. In summary, the contributions of this paper are as follows:

- We introduce **TKG-Thinker**, a novel agent capable of autonomously performing dynamic, multi-step temporal reasoning.
- To the best of our knowledge, this is the first work to explore modeling TKGQA as an RL-driven interleaved decision-making process with a time-aware interaction protocol and multi-dimensional reward design.
- Extensive experiments on benchmark datasets with three open-source LLMs demonstrate significant improvements over state-of-the-art TKGQA methods across multiple metrics.

2 Related Work

2.1 TKGQA

Temporal Knowledge Graph Question Answering is a challenging task that requires models to jointly reason over entities and temporal information in

TKGs. Early approaches, such as: MultiQA (Chen et al., 2023), TempoQR (Mavromatis et al., 2022), and TSQA (Shang et al., 2022), typically formulate TKGQA as a temporal knowledge graph completion task, relying on scoring functions to assess the plausibility of candidate facts. Recent LLM-based methods mainly treat the question as a query over the TKGs and use retrieved evidence for reasoning. Specifically, ARI (Chen et al., 2024b) enhances the temporal adaptability of LLMs through time-aware training and reasoning signals, while TempAgent (QianyiHu et al., 2025) treats the LLM as an agent that performs interaction. TimeR^4 (Qian et al., 2024) and PoK (Qian et al., 2025) strengthen LLM reasoning by improving the retrieval component, whereas RTQA (Gong et al., 2025) decomposes questions into sub-problems solved in a bottom-up manner with LLMs and TKGs. Nevertheless, these LLM-based methods still rely on manually crafted prompts, which limit their ability to autonomously detect and correct evidence.

2.2 Agentic Reasoning with RL

While prompting-based methods facilitate search capabilities in a training-free manner (Li et al., 2025b), the landscape is shifting toward training-centric agentic reasoning. Advanced approaches have demonstrated that Reinforcement Learning with Verifiable Rewards (RLVR) can unlock superior reasoning abilities in LLMs (Shao et al., 2024b; Sheng et al., 2024). This has catalyzed efforts to optimize agentic workflows—such as multi-step search (Jin et al., 2025; Team et al., 2025) and external tool integration (Shang et al., 2025)—using RL. Despite progress in long-horizon tasks through self-reflection (Shi et al., 2025) and structured memory (Yan et al., 2025), these methods primarily focus on logical or mathematical tasks, but lack specialized mechanisms to navigate the intricate temporal constraints inherent in TKGs.

3 Preliminary

To enhance the reasoning capabilities of LLMs for TKGQA, we employ the RLVR framework, which optimizes the policy π_θ using deterministic rewards r (e.g., execution results or exact-match accuracy). Formally, the objective is to maximize:

$$\mathcal{J}(\theta) = \hat{\mathbb{E}}_{Q,\mathbf{y}\sim\pi_{old}}\left[\frac{1}{G}\sum_{i=1}^{G} f_\epsilon(\rho_i(\theta), \hat{A}_i)\right] - \beta \cdot \hat{\mathbb{E}}_Q\left[\mathbb{D}_{KL}[\pi_\theta(\cdot|Q)||\pi_{\text{ref}}(\cdot|Q)]\right],$$

where G denotes the number of sampled trajectories per prompt ($G > 1$ for GRPO). $\rho_i(\theta) = \frac{\pi_\theta(y_i|Q)}{\pi_{old}(y_i|Q)}$ represents the importance sampling ratio. f_ϵ denotes the clipping function used in PPO/GRPO to stabilize updates, while $\hat{A}_i$ represents the advantage of trajectory y_i computed based on verifiable rewards. The KL divergence term, scaled by β, penalizes deviations from a reference policy π_{ref} to prevent model collapse.

4 Methodology

In this section, we introduce TKG-Thinker, a novel agent equipped with autonomous planning and adaptive retrieval capabilities for reasoning over TKGs. As illustrated in Figure 2, TKG-Thinker is trained through two complementary stages: (1) Supervised Fine-Tuning (SFT) for cold start (§ 4.1), and (2) Online Reinforcement Learning with Temporal Tool Calls (§ 4.2).

4.1 Supervised Fine-Tuning for Cold Start

TKG-Thinker relies on meaningful exploration over tool-augmented trajectories, but a generic base model does not yet know how to plan or invoke tools in the expected format, leading to low-quality rollouts (Shao et al., 2025). To address this, we perform SFT on trajectories generated by a strong teacher model (e.g., GPT-4o) acting as a tool-augmented agent (Li et al., 2025a), thereby initializing TKG-Thinker with a reasonable search and citation strategy before online RL. To enable the teacher model to produce tool-augmented CoT trajectories suitable for SFT, we carefully construct a prompting pipeline. Specifically, we first adopt a few-shot prompting strategy to elicit structured CoT trajectories from the teacher model, as detailed in Appendix A.1. However, not all generated trajectories are reliable. Therefore, we apply a two-stage rejection sampling pipeline:

- **Format validity filtering**: We discard trajectories that violate structural constraints, ensuring consistent CoT patterns.
- **Answer correctness filtering**: We filter out trajectories whose final answer does not match the ground-truth label in the training set.

In this way, we final curated set forms a high-quality CoT dataset for reasoning activation, and statistical details are provided in Appendix A.3. With this dataset in hand, given a question Q and

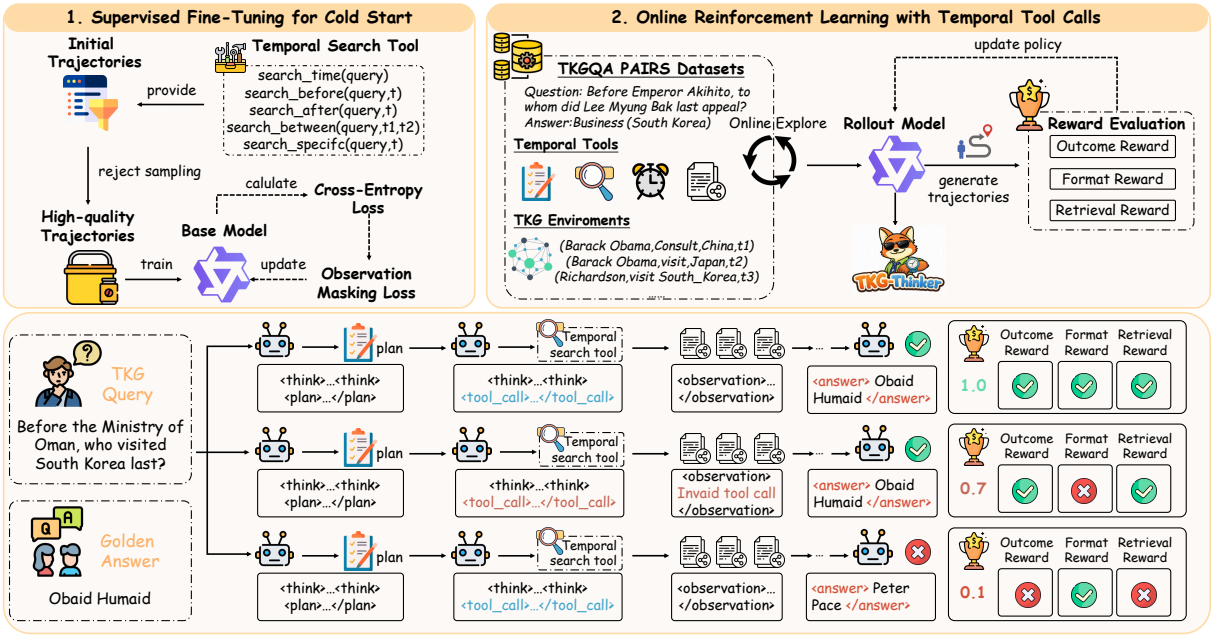

Figure 2: The overview of our proposed TKG-Thinker. We first apply supervised fine-tuning on high-quality trajectories to mitigate the cold-start problem, and further refine the model via online reinforcement learning with temporal tool calls. The bottom panel illustrates three rollouts: a complete success, a partial success, and a failure.

a filtered trajectory $y = [y_1, \ldots, y_T]$, where y concatenates the structured reasoning steps, tool calls, and final answer, the SFT objective maximizes the likelihood of the teacher trajectory:

$$\mathcal{L}_{\text{SFT}} = -\sum_{t=1}^{T} \log \pi_\theta(y_t \mid Q, y_{<t}), \qquad (1)$$

where π_θ denotes the model's token distribution. Following prior work (Li et al., 2025a), we compute the loss only over model-generated tokens and exclude environment feedback (e.g., observations). The resulting model π_{SFT} serves as the initialization for the second online RL stage, substantially alleviating cold-start issues.

4.2 Online RL with Temporal Tool Calls

After the supervised fine-tuning stage, we further apply the online reinforcement learning with temporal tool calls, which enhances its multi-hop and time-sensitive reasoning capabilities. To enable effective learning in this setting, we address the problem from three key perspectives: action space, reward design, and training objective.

4.2.1 Action Space

As we can observe, existing retrievers are insensitive to temporal constraints and LLMs tend to overlook temporal requirements and hallucinate during question decomposition (Qian et al., 2025; Guo et al., 2025), we identify the critical reasoning characteristics of temporal questions, and we categorize the reasoning paradigms of temporal reasoning over TKGs into specific taxonomies according to timestamps. Crucially, we formalize the agent's action space as a collection of temporal functional tools. By doing so, we transform temporal reasoning into actionable primitives that directly mirror the TKG structure, moving beyond the limitations of the LLM's implicit reasoning capabilities. In detail, our action space includes *think*, *plan*, temporal search actions, and *answer*, with the temporal search actions defined as follows:

- *Search_time(query)*. This action returns timestamps or time intervals associated with relevant quadruples.
- *Search_specific(query, t)*. This action returns relevant quadruples at the specified time t.
- *Search_before(query, t)*. This action returns relevant quadruples occurring strictly before the specified time t.
- *Search_after(query, t)*. This action returns relevant quadruples occurring strictly after the specified time t.
- *Search_between(query, t_1, t_2)*. This action returns relevant quadruples occurring within the time interval $[t_1, t_2]$.

Based on this action space and treating TKGs as the environment, temporal search tool returns structured feedback enclosed within <*observation*> and </*observation*>. Under this formulation, we adopt a ReAct-style (Yao et al., 2023; Liu et al., 2025a) interaction protocol, in which the reasoning process proceeds as a sequence of planning, internal thought generation, temporal search tool calls, and environment feedback. Formally, the interaction trajectory at step n can be represented as:

$$\mathcal{H}_n = (\tau_0, p, \tau_1, a_1, o_1, \ldots, \tau_{n-1}, a_{n-1}, o_{n-1}), \tag{2}$$

where τ denotes the agent's internal thought, a is an action selected from the temporal search actions, with p being the initial planning action, and o is the observation obtained by executing the action a over the TKGs. Based on the historical trajectory $\mathcal{H}_n$, the generation process for the next thought τ_n and action a_n can be formulated as:

$$\pi_\theta(\tau_n \mid \mathcal{H}_n) = \prod_{i=1}^{|\tau_n|} \pi_\theta(\tau_n^i \mid \mathcal{H}_n, \tau_n^{<i}), \tag{3}$$

$$\pi_\theta(a_n \mid \mathcal{H}_n, \tau_n) = \prod_{j=1}^{|a_n|} \pi_\theta(a_n^j \mid \mathcal{H}_n, \tau_n, a_n^{<j}), \tag{4}$$

where $\pi_\theta = \pi_{\text{SFT}}$, τ_n^i and $|\tau_n|$ denote the i-th token and the length of τ_n, and a_n^j and $|a_n|$ denote the j-th token and the length of a_n. The interaction loop terminates when either the *answer* action is invoked or the interaction-turn budget $B_{\max}$ is reached.

4.2.2 Reward Design

To optimize the above interactive process, we adopt the RLVR framework equipped with a novel multi-reward formulation. We incorporate three key components into our reward design: the *format reward*, the *retrieval reward*, and the *outcome reward*.

Format Reward verifies whether the generated rollout adheres to the structured interaction protocol. Specifically, we ensure that the entire rollout follows the temporal interaction pattern defined in Eq. 2. Formally, the format reward is defined as:

$$R_{\text{fmt}} = \alpha\, \mathbb{I}_{\text{fmt}}, \tag{5}$$

where $\mathbb{I}_{\text{fmt}} \in \{0, 1\}$ is a binary format validity indicator and $\alpha \in (0, 1)$ is a scaling coefficient.

Retrieval Reward measures whether the retriever successfully retrieves evidence containing the correct answer, defined as:

$$R_{\text{ret}} = \gamma\, \mathbb{I}_{\text{ret}}, \tag{6}$$

where $\mathbb{I}_{\text{ret}} \in \{0, 1\}$ is a binary retrieval indicator, and $\gamma \in (0, 1)$ is a tunable scaling coefficient.

Outcome Reward evaluates the correctness of the final output answer a_{pred} by comparing it against the ground truth answer a_{gold} using a rule-based criteria exact match (EM):

$$R_{\text{out}} = \text{EM}(a_{\text{pred}}, a_{\text{gold}}), \tag{7}$$

where $\text{EM}(\cdot, \cdot)$ returns 1 if the two strings match exactly and 0 otherwise, and thus $R_{\text{out}} \in \{0, 1\}$. Finally, we combine the above components into the overall reward. The final reward R_{all} is defined as:

$$\begin{aligned} R_{\text{all}} = R_{\text{out}}(1 - (1 - \mathbb{I}_{\text{fmt}})\lambda) \\ + (1 - R_{\text{out}})(R_{\text{fmt}} + R_{\text{ret}}) \\ + (1 - R_{\text{out}})\, \delta\, (1 - \mathbb{I}_{\text{fmt}}), \end{aligned} \tag{8}$$

where $\lambda > 0$ denotes the penalty applied when the answer is correct but the format is invalid, and $\delta > 0$ serves as a fallback reward granted when both the answer and format are incorrect. In this way, R_{fmt} enforces adherence to the temporal interaction protocol, R_{ret} promotes effective problem decomposition and evidence retrieval over TKGs, and R_{out} ensures factual correctness.

4.2.3 Training Objective

With the multi-dimensional reward formulation defined, we formalize the overall training objective of TKG-Thinker framework. To ensure robust policy optimization, we adopt the RLVR paradigm, which can be instantiated through either PPO or GRPO. The policy π_θ is optimized by maximizing the objective function $\mathcal{J}(\theta)$, which encourages trajectories with higher-than-average rewards while maintaining stability via importance sampling and Kullback–Leibler (KL) divergence constraints. The overall objective is defined as:

$$\begin{aligned} \mathcal{J}(\theta) = & \hat{\mathbb{E}}_{Q, y_{i=1}^{G} \sim \pi_{\text{old}}} \left[\frac{1}{G} \sum_{i=1}^{G} f_\epsilon(\rho_i(\theta), \hat{A}_i) \right] \\ & - \beta \cdot \hat{\mathbb{E}}_Q \left[\mathbb{D}_{KL} \left[\pi_\theta(\cdot|Q) || \pi_{\text{ref}}(\cdot|Q) \right] \right], \end{aligned} \tag{9}$$

where $\rho_i(\theta) = \frac{\pi_\theta(y_i|Q)}{\pi_{\text{old}}(y_i|Q)}$ denotes the importance sampling ratio for the i-th trajectory, and $f_\epsilon(\rho_i(\theta), \hat{A}) = \min(\rho_i(\theta)\hat{A}, \text{clip}(\rho_i(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A})$ is the clipping function. For standard PPO, $G = 1$ and the advantage $\hat{A}_i$ is estimated using a learned value function, whereas for GRPO, $G > 1$ and the advantage $\hat{A}_i$ is computed group-relatively.

5 Experiments

In this section, we evaluate TKG-Thinker on widely used datasets. We conduct extensive experiments to demonstrate the effectiveness of our method by answering the following research questions (RQ): (1) **RQ1:** How does TKG-Thinker perform compared to state-of-the-art baselines on complex TKGQA datasets? (2) **RQ2:** What is the contribution of each key module in the TKG-Thinker framework to the overall performance? (3) **RQ3:** How do different retrieval configurations affect reasoning performance? (4) **RQ4:** How does RL optimization shape the model's behavior in TKGQA scenarios? We also conduct a cross-domain generalization study in Appendix B, and present a case study in Appendix E to further demonstrate the robustness and the advantages of our proposed method.

5.1 Experimental Settings

We evaluate TKG-Thinker on representative TKGQA benchmarks, including MULTITQ (Chen et al., 2023) and CronQuestions (Saxena et al., 2021). Detailed descriptions of these datasets are provided in Appendix A.2, while additional results on the cross-domain benchmark TimelineKGQA (Sun et al., 2025) are reported in Appendix B. We adopt Hits@1 as the evaluation metric, measuring the proportion of questions for which the top-ranked prediction is correct. We compare TKG-Thinker against three categories of baselines: PLM-based methods, Embedding-based methods, and LLM-based methods. A detailed description of all baseline models is provided in Appendix A.4. For training, we use GPT-4o as the teacher model to generate trajectories and adopt e5-base-v2 (Wang et al., 2022) for evidence retrieval, retrieving the top-15 most relevant quadruples per query. TKG-Thinker is instantiated with Llama3-8B-Instruct, Qwen2.5-7B-Instruct, and Qwen3-4B-Instruct-2507 as backbone models. Additional implementation details are provided in Appendix A.5.

5.2 Main Results (RQ1)

In this section, we compare TKG-Thinker with representative baselines on MULTITQ and CronQuestions. As shown in Table 1, TKG-Thinker achieves consistently superior overall performance, outperforming diverse baselines across different model families, parameter scales (4B, 7B, 8B), and RL training strategies (trained with GRPO or PPO), demonstrating strong model-agnostic applicability. Compared to the strongest baseline, TKG-Thinker achieves absolute overall Hits@1 improvements of 7.60% and 7.30% on MULTITQ and CronQuestions, respectively. These results suggest that enabling LLMs to dynamically interact with TKGs via RAG mechanisms facilitates effective search strategies and temporally grounded reasoning capabilities. Notably, TKG-Thinker exhibits substantial improvements on complex multi-step TKGQA tasks, surpassing the best-performing baselines on the corresponding complex settings by 29.70% on MULTITQ (Multiple) and 23.50% on CronQuestions (Complex). This further confirms that our approach significantly enhances temporal multi-hop reasoning through explicit planning and time-aware retrieval tool usage. While TKG-Thinker shows slightly lower performance on Single-type questions in MULTITQ, this difference can be reasonably attributed to PoK's use of a retriever specifically optimized for single-step temporal retrieval.

5.3 Ablation Study (RQ2)

In this section, we conduct a series of ablation experiments to examine the contribution of each component in TKG-Thinker, as summarized in Table 2, including the SFT stage, the planning mechanism, and the temporal retrievers. Specifically, we systematically remove or replace individual components to construct corresponding model variants for comparison. Additional ablation results on the CronQuestions are reported in Appendix C.

Effect of the SFT Stage. We further analyze the role of SFT, which initializes the model's ability to execute structured reasoning protocols prior to reinforcement learning. When the SFT stage is removed and the model is trained with RL alone, the overall performance drops drastically by 26.40%. This confirms that SFT provides essential scaffolding that stabilizes learning, reduces temporal hallucination, enables verifiable temporal reasoning rather than unconstrained free-form generation.

Effect of the Plan Action. We remove the Plan action and prompt the model to interact directly with the environment using the original queries. As a result, eliminating the planning component leads to an overall performance drop of 5.90%. In particular, performance on Multiple-type temporal questions decreases by 8.80%, 14.00%, and 12.00% on Equal Multi, After First, and Before Last, respectively. These results suggest that the planning component plays an important role in reliable temporal reasoning over TKGs, as removing

Methods	MULTITQ					CronQuestions				
	Overall	Question Type		Answer Type		Overall	Question Type		Answer Type	
		Single	Multiple	Entity	Time		Simple	Complex	Entity	Time
PLM-based Methods										
BERT (Devlin et al., 2019)	0.083	0.092	0.061	0.101	0.040	0.243	0.249	0.239	0.277	0.179
DistillBERT (Sanh et al., 2019)	0.083	0.087	0.074	0.102	0.037	–	–	–	–	–
ALBERT (Lan et al., 2020)	0.108	0.116	0.086	0.139	0.032	0.248	0.255	0.235	0.279	0.177
Embedding-based Methods										
EmbedKGQA (Saxena et al., 2020)	0.206	0.235	0.134	0.290	0.001	0.288	0.290	0.286	0.411	0.057
CronKGQA (Saxena et al., 2021)	0.279	0.337	0.134	0.328	0.156	0.647	0.987	0.392	0.699	0.549
MultiQA (Chen et al., 2023)	0.293	0.347	0.159	0.349	0.157	–	–	–	–	–
LLM-based Methods										
ARI (Chen et al., 2024b)	0.380	0.680	0.210	0.394	0.344	0.707	0.860	0.570	0.660	0.800
Naive RAG (Chen et al., 2024a)	0.379	0.469	0.155	0.242	0.672	0.633	0.726	0.280	0.610	0.684
ReAct RAG (Yao et al., 2023)	0.398	0.506	0.130	0.243	0.735	0.809	0.863	0.600	0.768	0.895
TempAgent (QianyiHu et al., 2025)	0.702	0.857	0.316	0.624	0.870	0.842	0.895	0.640	0.805	0.921
TimeR^4 (Qian et al., 2024)	0.728	0.887	0.335	0.639	0.945	–	–	–	–	–
RTQA (Gong et al., 2025)	0.765	0.902	0.424	0.692	0.942	–	–	–	–	–
PoK (Qian et al., 2025)	0.779	**0.929**	0.409	0.696	0.962	–	–	–	–	–
TKG-Thinker (Ours)										
♣ Qwen2.5-7B-Instruct	**0.855**	0.910	**0.721**	**0.814**	0.955	0.893	0.958	0.844	0.863	0.949
♠ Qwen2.5-7B-Instruct	0.824	0.881	0.683	0.774	0.945	0.868	0.937	0.817	0.838	0.925
♣ Llama3-8B-Instruct	0.810	0.872	0.659	0.756	0.943	**0.915**	**0.968**	**0.875**	**0.893**	**0.955**
♠ Llama3-8B-Instruct	0.799	0.861	0.644	0.709	0.739	0.873	0.949	0.817	0.844	0.928
♣ Qwen3-4B-Instruct	0.804	0.914	0.533	0.739	**0.963**	0.889	0.953	0.837	0.857	0.948
♠ Qwen3-4B-Instruct	0.780	0.901	0.480	0.708	0.956	0.877	0.953	0.820	0.841	0.944

Table 1: Performance comparison of baselines and TKG-Thinker in Hits@1 across different question and answer types on MULTITQ and CronQuestions. ♣ denotes TKG-Thinker trained with SFT+GRPO, while ♠ denotes training with SFT+PPO. The best and second-best scores are marked in **bold** and underline, respectively.

Model	Overall	Answer Type		Fine-grained Question Type					
		Entity	Time	Equal	Before/After	First/Last	Equal Multi	After First	Before Last
TKG-Thinker	**0.855**	**0.814**	0.955	**0.939**	**0.851**	**0.920**	**0.792**	**0.678**	**0.729**
w/o SFT Stage	0.591	0.465	0.897	0.869	0.609	0.836	0.213	0.078	0.084
w/o Plan Action	0.796	0.766	0.871	0.933	0.822	0.838	0.704	0.538	0.609
w/o Temporal Retrievers	0.458	0.252	**0.959**	0.466	0.456	0.909	0.205	0.133	0.139

Table 2: Ablation study on the MULTITQ dataset. **Bold** indicates the best performance, while underline marks the second-best . Single-type questions include Equal, Before/After, and First/Last; Multiple-type questions include Equal Multi, After First, and Before Last. "*w/o*" means removing or replacing the corresponding module.

it tends to hallucinate intermediate reasoning steps.

Effect of the Temporal Retrievers. To assess the role of temporal retrievers, we replace them with a purely semantic retriever that ignores temporal constraints. This substitution yields the largest performance degradation (–39.70%), highlighting the importance of temporal alignment between questions and evidence. Notably, performance drops sharply on multiple-type temporal questions, demonstrating that temporal retrieval is indispensable for providing fine-grained, temporally grounded evidence to support reliable temporal reasoning.

5.4 Retrieval Analysis (RQ3)

Effect of Retriever Model. Since retrieval quality directly determines the availability of temporal evidence, we first evaluate TKG-Thinker under four representative retrievers: `e5-base-v2`, `bge-m3`, `contriever`, and `qwen3-embedding-4B`. As shown in Figure 3 (Left), all four retrievers achieve competitive performance, substantially surpassing the strongest baseline (PoK). Among them, contrastively trained retrievers (e.g., `e5-base-v2`, `bge-m3`) deliver superior performance, with the advantage being more evident on Multiple questions.

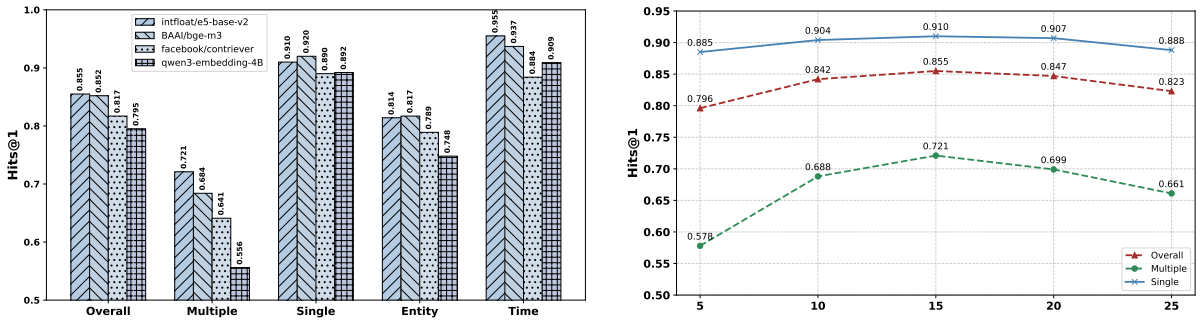

Figure 3: Retriever analysis on the MULTITQ dataset. **Left**: Performance comparison of different retriever models. **Right**: Effect of retrieval depth, measured by the number of top-k retrieved quadruples.

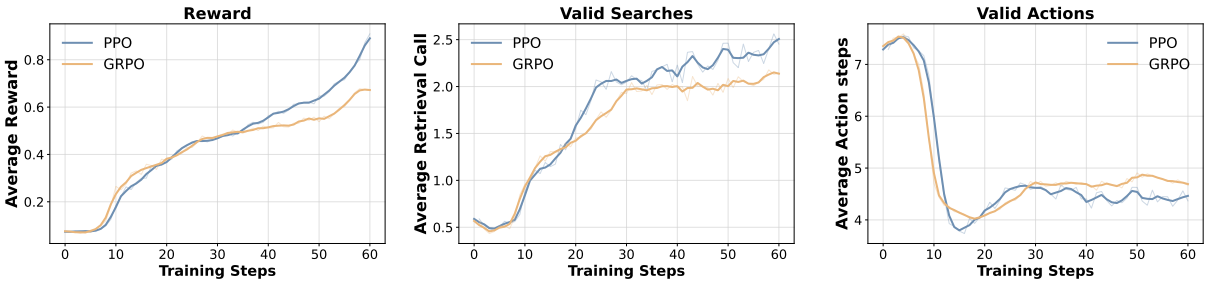

Figure 4: Training dynamics of TKG-Thinker implemented with GRPO and PPO on MULTITQ. **Left**: Training Reward; **Middle**: Retrieval Call Steps; **Right**: Action Steps.

Effect of Retrieval Depth. The hyperparameter k controls how many top-ranked quadruples the temporal search tools return as environmental feedback. As illustrated in Figure 3 (Right), performance increases with k and then declines. This reflects a trade-off: larger k improves the likelihood of retrieving useful evidence, whereas excessively large k introduces distractors that impede LLM reasoning. Notably, performance degradation at larger k is more pronounced on Multiple questions. We attribute this phenomenon to the accumulation of errors across successive reasoning steps, as Multiple-type questions require iterative retrieval and multi-step reasoning, where early errors are progressively amplified in later stages. In practice, we find that $k = 15$ offers the best balance between evidence coverage and distractor noise.

5.5 Training Dynamics (RQ4)

To investigate how TKG-Thinker evolves during the training process, we illustrate its training dynamics in Figure 4 (with additional entropy and response details in Appendix D). As shown in the left panel, both PPO and GRPO exhibit a steady increase in training rewards. This demonstrates that our fine-grained reward design provides stable reinforcement signals, facilitating consistent policy optimization. Regarding action and retrieval dynamics, we observe a clear "decrease–then–increase" pattern. Specifically, the average number of action steps initially drops sharply as the model learns to follow the required output format and eliminates redundant or invalid actions. Subsequently, both action steps and retrieval calls gradually increase and stabilize, indicating that TKG-Thinker strategically invokes additional temporal tool calls to acquire necessary evidence and thereby strengthens its agentic reasoning capability. Notably, while both algorithms converge well, PPO achieves a higher reward ceiling and more frequent retrieval calls in the later stages of training.

6 Conclusion

We introduce TKG-Thinker, a novel agent equipped with autonomous planning and adaptive retrieval capabilities for TKGQA. By modeling the TKG as a dynamic environment, TKG-Thinker integrates supervised fine-tuning and reinforcement learning with a multi-reward optimization scheme to enhance temporal reasoning. Experiments show that TKG-Thinker consistently outperforms baselines, demonstrating the effectiveness of explicit interaction and RL-driven optimization in reducing hallucination and improving multi-step reasoning.

Limitations

Despite TKG-Thinker's strong performance achieved in TKGQA, this work has several limitations. The current reward mechanism relies heavily on binary indicators and rule-based criteria, such as Exact Match (EM) for outcomes and basic format verification. This outcome-based reward lacks a nuanced evaluation of the intermediate reasoning process. Future iterations could incorporate an LLM Judge with detailed rubrics to qualitatively assess the logical and temporal consistency of the think and plan steps, ensuring that the model understands complex temporal constraints rather than just optimizing for a specific output format. Besides, while the model demonstrates effective multi-step reasoning, the relative simplicity of current datasets and benchmarks—particularly their limited reasoning hops—restricts the training of temporal agents capable of long-range planning and inference. Future work should thus explore more complex synthetic multi-hop tasks and open-world settings to foster greater model robustness.

Ethics Statement

In constructing the CoT-based SFT datasets, we have taken into account ethical considerations and limitations commonly associated with large language models. All data used in this work are publicly available and do not contain personal or sensitive information. Nonetheless, we acknowledge that, despite our best efforts, the datasets may still contain gaps or unintended biases. To mitigate these concerns, the source data have been curated to ensure diversity and reduce potential bias. Through careful dataset construction, review, and testing procedures, we strive to uphold ethical AI principles while advancing research in TKGQA.

References

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2024a. Benchmarking large language models in retrieval-augmented generation. In *AAAI*, pages 17754–17762. AAAI Press.

Wei Chen, Haoyu Huang, Zhiyu Zhang, Tianyi Wang, Youfang Lin, Liang Chang, and Huaiyu Wan. 2025. Next-poi recommendation via spatial-temporal knowledge graph contrastive learning and trajectory prompt. *IEEE Trans. Knowl. Data Eng.*, 37(6):3570–3582.

Ziyang Chen, Dongfang Li, Xiang Zhao, Baotian Hu, and Min Zhang. 2024b. Temporal knowledge question answering via abstract reasoning induction. In *ACL (1)*, pages 4872–4889. Association for Computational Linguistics.

Ziyang Chen, Jinzhi Liao, and Xiang Zhao. 2023. Multi-granularity temporal question answering over knowledge graphs. In *Proc. of ACL*, pages 11378–11392.

DeepSeek-AI. 2024. Deepseek-v3 technical report. *CoRR*, abs/2412.19437.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics.

Yifu Gao, Linbo Qiao, Zhigang Kan, Zhihua Wen, Yongquan He, and Dongsheng Li. 2024. Two-stage generative question answering on temporal knowledge graph using large language models. In *ACL (Findings)*, pages 6719–6734. Association for Computational Linguistics.

Zhaoyan Gong, Juan Li, Zhiqiang Liu, Lei Liang, Huajun Chen, and Wen Zhang. 2025. RTQA : Recursive thinking for complex temporal knowledge graph question answering with large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9853–9870, Suzhou, China. Association for Computational Linguistics.

Kai Guo, Harry Shomer, Shenglai Zeng, Haoyu Han, Yu Wang, and Jiliang Tang. 2025. Empowering graphrag with knowledge filtering and integration. *arXiv preprint arXiv:2503.13804*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *CoRR*, abs/2503.09516.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*. OpenReview.net.

Weizhen Li, Jianbo Lin, Zhuosong Jiang, Jingyi Cao, Xinpeng Liu, Jiayu Zhang, Zhenqiang Huang, Qianben Chen, Weichen Sun, Qiexiang Wang, Hongxuan Lu, Tianrui Qin, Chenghao Zhu, Yi Yao, Shuying Fan, Xiaowan Li, Tiannan Wang, Pai Liu, King Zhu, and 11 others. 2025a. Chain-of-agents: End-to-end agent foundation models via multi-agent distillation and agentic RL. *CoRR*, abs/2508.13167.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025b. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438, Suzhou, China. Association for Computational Linguistics.

Yuhan Li, Xinni Zhang, Linhao Luo, Heng Chang, Yuxiang Ren, Irwin King, and Jia Li. 2025c. G-refer: Graph retrieval-augmented large language model for explainable recommendation. In *WWW*, pages 240–251. ACM.

Ben Liu, Jihai Zhang, Fangquan Lin, Cheng Yang, Min Peng, and Wotao Yin. 2025a. Symagent: A neural-symbolic self-learning agent framework for complex reasoning over knowledge graphs. In *WWW*, pages 98–108. ACM.

Runxuan Liu, Luobei Luobei, Jiaqi Li, Baoxin Wang, Ming Liu, Dayong Wu, Shijin Wang, and Bing Qin. 2025b. Ontology-guided reverse thinking makes large language models stronger on knowledge graph question answering. In *ACL (1)*, pages 15269–15284. Association for Computational Linguistics.

Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N. Ioannidis, Adesoji Adeshina, Phillip Ryan Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. 2022. Tempoqr: Temporal question reasoning over knowledge graphs. In *AAAI*, pages 5825–5833. AAAI Press.

Miao Peng, Nuo Chen, Zongrui Suo, and Jia Li. 2025. Rewarding graph reasoning process makes llms more generalized reasoners. In *KDD (2)*, pages 2257–2268. ACM.

Xinying Qian, Ying Zhang, Yu Zhao, Baohang Zhou, Xuhui Sui, and Xiaojie Yuan. 2025. Plan of knowledge: Retrieval-augmented large language models for temporal knowledge graph question answering. *CoRR*, abs/2511.04072.

Xinying Qian, Ying Zhang, Yu Zhao, Baohang Zhou, Xuhui Sui, Li Zhang, and Kehui Song. 2024. TimeR^4 : Time-aware retrieval-augmented large language models for temporal knowledge graph question answering. In *EMNLP*, pages 6942–6952. Association for Computational Linguistics.

QianyiHu QianyiHu, Xinhui Tu, Guo Cong, and Shunping Zhang. 2025. Time-aware react agent for temporal knowledge graph question answering. In *NAACL (Findings)*, pages 6013–6024. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Apoorv Saxena, Soumen Chakrabarti, and Partha P. Talukdar. 2021. Question answering over temporal knowledge graphs. In *ACL/IJCNLP (1)*, pages 6663–6676. Association for Computational Linguistics.

Apoorv Saxena, Aditay Tripathi, and Partha P. Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *ACL*, pages 4498–4507. Association for Computational Linguistics.

Chao Shang, Guangtao Wang, Peng Qi, and Jing Huang. 2022. Improving time sensitivity for question answering over temporal knowledge graphs. In *ACL (1)*, pages 8017–8026. Association for Computational Linguistics.

Ning Shang, Yifei Liu, Yi Zhu, Li Lyna Zhang, Weijiang Xu, Xinyu Guan, Buze Zhang, Bingcheng Dong, Xudong Zhou, Bowen Zhang, Ying Xin, Ziming Miao, Scarlett Li, Fan Yang, and Mao Yang. 2025. rstar2-agent: Agentic reasoning technical report. *CoRR*, abs/2508.20722.

Rulin Shao, Akari Asai, Shannon Zejiang Shen, Hamish Ivison, Varsha Kishore, Jingming Zhuo, Xinran Zhao, Molly Park, Samuel G Finlayson, David Sontag, and 1 others. 2025. Dr tulu: Reinforcement learning with evolving rubrics for deep research. *arXiv preprint arXiv:2511.19399*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024a. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024b. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Yaorui Shi, Sihang Li, Chang Wu, Zhiyuan Liu, Junfeng Fang, Hengxing Cai, An Zhang, and Xiang Wang. 2025. Search and refine during think: Autonomous retrieval-augmented reasoning of llms. *arXiv preprint arXiv:2505.11277*.

Qiang Sun, Sirui Li, Du Huynh, Mark Reynolds, and Wei Liu. 2025. Timelinekgqa: A comprehensive question-answer pair generator for temporal knowledge graphs. In *WWW (Companion Volume)*, pages 797–800. ACM.

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, and 1 others. 2025. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *CoRR*, abs/2212.03533.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Kristian Kersting, Jeff Z Pan, Hinrich Schütze, and 1 others. 2025. Memory-r1: Enhancing large language model agents

to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 40 others. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *ICLR*. OpenReview.net.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *CoRR*, abs/2504.13837.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

A Experimental Settings

A.1 Few-shot Prompt for Initial Trajectory Generation

As shown in Figure 5, we adopt a few-shot prompting strategy to elicit structured, tool-augmented reasoning during trajectory generation. This stage provides the initial pool of trajectories before subsequent format and answer-correctness filtering.

A.2 Dataset Details

Type		Train	Valid	Test
Single	Equal	135,890	18,983	17,311
	Before/After	75,340	11,655	11,073
	First/Last	72,252	11,097	10,480
Multiple	Equal Multi	16,893	3,213	3,207
	After First	43,305	6,499	6,266
	Before Last	43,107	6,532	6,247
Total		**386,787**	**57,979**	**54,584**

Table 3: Data statistics of MULTITQ.

MULTITQ. MULTITQ is a large-scale temporal question answering dataset that incorporates multi-granularity temporal information. It provides a comprehensive evaluation protocol across several dimensions: Question Type (Multiple vs. Single), Answer Type (Entity vs. Time), and Time Granularity (year, month, and day). The detailed statistics of MULTITQ are presented in Table 3.

Type	Train	Valid	Test
Simple Entity	90,651	7,745	7,812
Simple Time	61,471	5,197	5,046
Before/After	23,869	1,982	2,151
First/Last	118,556	11,198	11,159
Time Join	55,453	3,878	3,832
Simple Reasoning	152,122	12,942	12,858
Complex Reasoning	197,878	17,058	17,142
Entity Answer	225,672	19,362	19,524
Time Answer	124,328	10,638	10,476
Total	**350,000**	**30,000**	**30,000**

Table 4: Dataset Statistics of CronQuestions.

CronQuestions. CronQuestions is a temporal QA benchmark consisting of 410K unique question–answer pairs. Its questions can be categorized into two major types: Simple temporal reasoning (e.g., Simple Entity and Simple Time) and Complex temporal reasoning (e.g., Before/After, First/Last, and Time-Join queries), depending on the temporal constraints involved. Detailed statistics of CronQuestions are shown in Table 4.

A.3 Statistics of the CoT Dataset for SFT

Table 5: Statistics of the SFT datasets constructed from MULTITQ and CronQuestions.

Dataset	Question Type	Count
MULTITQ	Equal	4017
	Before/After	1668
	First/Last	1062
	Equal Multi	1395
	After First	1074
	Before Last	1055
	Total	**10271**
CronQuestions	Simple Entity	2509
	Simple Time	1726
	Before/After	668
	First/Last	3301
	Time Join	1508
	Total	**9712**

To mitigate the cold-start issue in reinforcement learning, we construct CoT-style supervised fine-tuning datasets from MULTITQ and CronQuestions. As summarized in Table 5, these datasets cover diverse temporal reasoning types and provide

Few-shot Prompt for Generating Trajectories

Your task is to construct multi-turn reasoning trajectories for temporal question answering over a temporal knowledge graph. You can only respond to a given question over a temporal knowledge graph using the following 9 functions: think, plan, search_specific, search_before, search_after, search_between, search_time, observation and answer.

Below are the descriptions of these functions:

1.think: Before using any of the plan, search_specific, search_before, search_after, search_between, search_time or answer functions, you must first use the think function to provide the reasoning, justification, and the procedural steps for the function you intend to use next. In think you should identify the question type (Equal, Before/After, First/Last, Equal Multi, Before Last, After First), the required time granularity (year, month, day), and which search function you will call next. Begin with <think> and end with </think>. 2. plan: Based on the given question, you must break it down into very detailed, fine-grained sub-questions to facilitate execution using the search functions. The plan should describe the temporal constraints that need to be checked, the search steps to perform, and the order of operations. Begin with <plan> and end with </plan>.3. search_specific: Use this function to retrieve events occurring at a specific time t. The argument t may be a year (YYYY), a month (YYYY-MM), or a full date (YYYY-MM-DD). Begin with <search_specific> and end with </search_specific>. Use it like this: <search_specific>("query", "t")</search_specific>. 4. search_before: Use this function to retrieve events occurring strictly before time t. Begin with <search_before> and end with </search_before>. Use it like this: <search_before>("query", "t")</search_before>. 5. search_after: Use this function to retrieve events occurring strictly after time t. Begin with <search_after> and end with </search_after>. Use it like this: <search_after>("query", "t")</search_after>. 6. search_between: Use this function to retrieve events occurring between times t1 and t2 (inclusive). Begin with <search_between> and end with </search_between>. Use it like this: <search_between>("query", "t1", "t2")</search_between>. 7. search_time: Use this function to retrieve events and their associated timestamps related to the queried event. Begin with <search_time> and end with </search_time>. Use it like this: <search_time>("query")</search_time>. 8. observation: This function represents the result returned by the environment after using any search function. You must NOT invent or fabricate the content inside `<observation>...</observation>`. The environment will insert `<observation>...</observation>` blocks after your search calls 9. answer: Your response must include the answer function at the end, indicating that you are confident in the final answer. The answer should be concise and contain no explanation. Begin with <answer> and end with </answer>.

Important Notes:

1. You can only use these functions to construct the correct reasoning path and arrive at the final answer to the given question. 2. Based on the results of the plan function, you may use the search_specific, search_before, search_after, search_between and search_time functions multiple times to gather sufficient temporal evidence before formulating your response. 3. After each search call, you must wait for an <observation>...</observation> block from the environment and then perform a new <think>...</think> based on that observation. 4. Do not give the answer unless you are completely sure. The answer must be concise, such as <answer> Beijing </answer>. 5. You must use the think function before each use of plan, search_specific, search_before, search_after, search_between, search_time or answer. 6. Special Token Restriction: <think>, <plan>, <search_specific>, <search_before>, <search_after>, <search_between>, <search_time>, <observation>, and <answer> are special tokens and must not appear in free text, especially not within the think function. 7. If the answer contains multiple items, they must be separated by commas (,) with no additional text, for example <answer>answer1,answer2,answer3</answer>.8. The required output structure must follow this general pattern:

<think>...</think>
<plan>...</plan>
<think>...</think>
[One or more repetitions of the following block:]
  <search_xxx>(arguments)</search_xxx>
  <observation>...</observation> (inserted by the environment)
  <think>...</think>
<answer> FINAL_ANSWER </answer>
where <search_xxx> is one of <search_specific>, <search_before>, <search_after>, <search_between>, or <search_time>.

Here are some examples:

{examples}

Now, answer the following question:

{question}

Figure 5: Few-shot Prompt for Generating Trajectories.

explicit supervision signals for temporal decomposition and retrieval behaviors.

A.4 Baseline Details

we compare TKG-Thinker against three categories of baselines: (1) **PLM-based methods**, including BERT(Devlin et al., 2019), ALBERT(Lan et al., 2020), and DistilBERT(Sanh et al., 2019); (2) **Embedding-based methods**, such as EmbedKGQA(Saxena et al., 2020), CronKGQA(Saxena et al., 2021), and MultiQA(Chen et al., 2023); and (3) **LLM-based methods**, including Naive RAG (Chen et al., 2024a), ReAct RAG(Yao et al., 2023), ARI(Chen et al., 2024b), RTQA(Gong et al.,

2025), and TempAgent(QianyiHu et al., 2025), TimeR4 (Qian et al., 2024), and PoK (Qian et al., 2025). For consistency with prior work, we adopt the baseline results reported in Qian et al. (2025), Chen et al. (2024b), and QianyiHu et al. (2025) for comparison.

A.5 Implementation Details

Table 6: Key hyperparameters used in model training.

Category	Value
Supervised Fine-tuning (SFT)	
Finetuning Type	Full
Epochs	4
Batch Size	4
Grad. Accumulation	4
Learning Rate	1×10^{-5}
Scheduler	Cosine Decay
Warmup Ratio	0.03
Cutoff Length	8192
Precision	BF16
Reinforcement Learning (RL)	
Algorithm	PPO / GRPO
Training Steps	80
Train Batch Size	256
Max Turns ($B_{\max}$)	8
Temperature	0.7
Retriever Top-k	15
Max Response Length (per turn)	512
Max Observation Length (per turn)	1024
Format Reward Coef. (α)	0.2
Outcome Penalty Coef. (λ)	0.4
Retrieval Reward Coef. (γ)	0.1
Fallback Reward Coef. (δ)	0.1
Inference	
Temperature	0.01
Top-p	0.95
Max New Tokens	512
Max Infer Step	8

As summarized in Table 6, during the SFT stage, we fine-tune the models via the LLaMA-Factory framework (Zheng et al., 2024) with a batch size of 4 for 4 epochs using AdamW with a learning rate of 1e−5 and cosine decay scheduling. In the RL stage, we switch to the Verl framework and train both PPO and GRPO policies with a batch size of 256, a mini-batch size of 32, and 5 rollouts. We set the interaction-turn budget to $B_{\max} = 8$. For rollout collection, we apply sampling with a temperature of 0.7 and conduct retrieval-augmented interactions using the top-15 evidence candidates per query. To stabilize trajectory generation and prevent excessive growth of reasoning tokens, model-generated responses and retrieved observations are truncated to 512 and 1024 tokens per turn, respectively. Regarding the training data, the SFT stage uses rejection sampling to retain trajectories, while the RL stage is trained on TKGQA data not used in SFT (5,001 QA pairs for MULTITQ and 5,230 for CronQuestions). During inference, we disable stochastic sampling and adopt deterministic decoding with temperature 0.01, and top-$p = 0.95$. All experiments are implemented in PyTorch and conducted on 8 NVIDIA A800 (80GB) GPUs.

B Cross-domain Generalization Study

Dataset	Type	Train	Val	Test
Timeline-CronQuestion	Simple	7,200	2,400	2,400
	Medium	8,252	2,751	2,751
	Complex	9,580	3,193	3,193
	Total	**25,032**	**8,344**	**8,344**
Timeline-ICEWS	Simple	17,982	5,994	5,994
	Medium	15,990	5,330	5,330
	Complex	19,652	6,550	6,550
	Total	**53,624**	**17,874**	**17,874**

Table 7: Data statistics of Timeline-CronQuestion and Timeline-ICEWS.

Model	Overall	Simple	Medium	Complex
RAG baseline	0.235	0.704	0.092	0.009
LLaMA2-7B	0.169	0.049	0.143	0.282
GPT-4o	0.206	0.069	0.130	0.376
RTQA	0.298	0.608	0.218	0.135
TKG-Thinker (Ours)				
Qwen2.5-7B-Instruct	0.460	0.567	0.294	0.523
Llama3-8B-Instruct	**0.491**	**0.612**	**0.320**	**0.546**

Table 8: Results on the Timeline-CronQuestion dataset across different reasoning difficulty levels. **Bold** indicates the best performance and underline denotes the second best performance.

Model	Overall	Simple	Medium	Complex
RAG baseline	0.265	0.660	0.128	0.011
LLaMA2-7B	0.111	0.035	0.066	0.322
GPT-4o	0.113	0.051	0.035	0.353
TKG-Thinker (Ours)				
Qwen2.5-7B-Instruct	**0.508**	0.556	**0.409**	**0.583**
Llama3-8B-Instruct	0.462	**0.596**	0.273	0.533

Table 9: Results on the Timeline-ICEWS dataset across different reasoning difficulty levels. **Bold** indicates the best performance, while underline denotes the second-best performance.

We further evaluate the cross-domain generalizability of TKG-Thinker. Specifically, we eval-

Model	Overall	Answer Type		Fine-grained Question Type				
		Entity	Time	Simple Entity	Simple Time	Before/After	First/Last	Time Join
TKG-Thinker	**0.893**	**0.863**	0.949	**0.946**	**0.977**	**0.699**	0.891	**0.787**
w/o SFT Stage	0.638	0.732	0.568	0.861	0.532	0.487	0.564	0.623
w/o Plan Action	0.860	0.811	**0.951**	0.942	**0.977**	0.538	**0.899**	0.603
w/o Temporal Retrievers	0.821	0.756	0.942	0.924	0.963	0.477	0.891	0.414

Table 10: Ablation study on the CronQuestions dataset. **Bold** indicates the best performance, while underline marks the second-best. Simple-type questions include Simple Entity, and Simple Time; Complex-type questions include Before/After, First/Last, and Time Join. "*w/o*" means removing or replacing the corresponding module.

uate TKG-Thinker on the TimelineKGQA (Sun et al., 2025) benchmark, which includes Timeline-CronQuestions and Timeline-ICEWS. These datasets differ in both temporal representations and the complexity of temporal reasoning. In TimelineKGQA, Simple questions require a single contextual fact and typically involve temporally constrained retrieval or timeline position identification; Medium questions require two contextual facts and further involve the combination of retrieval with temporal semantic operations and timeline arithmetic; Complex questions require three contextual facts and cover the full spectrum of temporal reasoning capabilities. The dataset statistics are summarized in Table 7. **For this evaluation, we use Qwen2.5-7B-Instruct and Llama3-8B-Instruct as backbones, both trained on MULTITQ using SFT and GRPO.** Following Gong et al. (2025) and Sun et al. (2025), we compare TKG-Thinker with a RAG baseline, LLaMA2-7B, GPT-4o, and RTQA. For consistency with prior work, baseline results are sourced from Gong et al. (2025).

As shown in Table 8 and Table 9, TKG-Thinker consistently outperforms all baselines in Overall score, and the results reveal three consistent trends. First, TKG-Thinker exhibits clear performance gains as temporal reasoning complexity increases. While improvements on Simple questions are marginal, the gains on Medium and Complex questions are substantially larger, indicating that TKG-Thinker is particularly effective at handling compositional temporal reasoning and timeline arithmetic. Second, the relative advantage of TKG-Thinker is preserved across datasets with distinct temporal characteristics: on Timeline-CronQuestion the improvements are most salient on Medium and Complex queries, whereas on Timeline-ICEWS the model maintains strong performance across all difficulty levels despite its larger event space and more diverse temporal expressions. Third, LLaMA2-7B and GPT-4o exhibit limited ability in settings that require explicit temporal grounding, whereas retrieval-augmented methods that rely on sub-question decomposition, such as RTQA, partially close the performance gap but still struggle with multi-hop temporal composition. These findings suggest that structured tool-augmented trajectories and agentic decision-making are key to improving temporal reasoning capabilities in various TKGQA scenarios.

C Ablation Study on CronQuestion

We further conduct ablation experiments on CronQuestions to analyze the contribution of each component in TKG-Thinker, as summarized in Table 10. Consistent with the findings on MULTITQ, several clear observations can be drawn. First, removing the SFT stage and training the model with reinforcement learning alone leads to a substantial performance drop of 25.50% in overall accuracy. This result indicates that SFT is essential for alleviating the cold-start problem in reinforcement learning by providing a stable initialization and structured reasoning priors. Second, eliminating the Plan action leads to a noticeable performance decline of 3.30% in the overall score, with the most severe degradation observed on Complex-type temporal questions. This suggests that explicit planning is particularly important for handling complex temporal dependencies that require multi-step reasoning. Finally, replacing the temporal retrievers with a purely semantic retriever that disregards temporal constraints leads to the performance drop of 7.2%. This result highlights that explicit temporal alignment between the question and the retrieved evidence is a fundamental prerequisite for accurate TKGQA. Overall, these results demonstrate that TKG-Thinker benefits from the combined effects of supervised initialization, explicit planning, and temporally aware retrieval to achieve robust temporal reasoning.

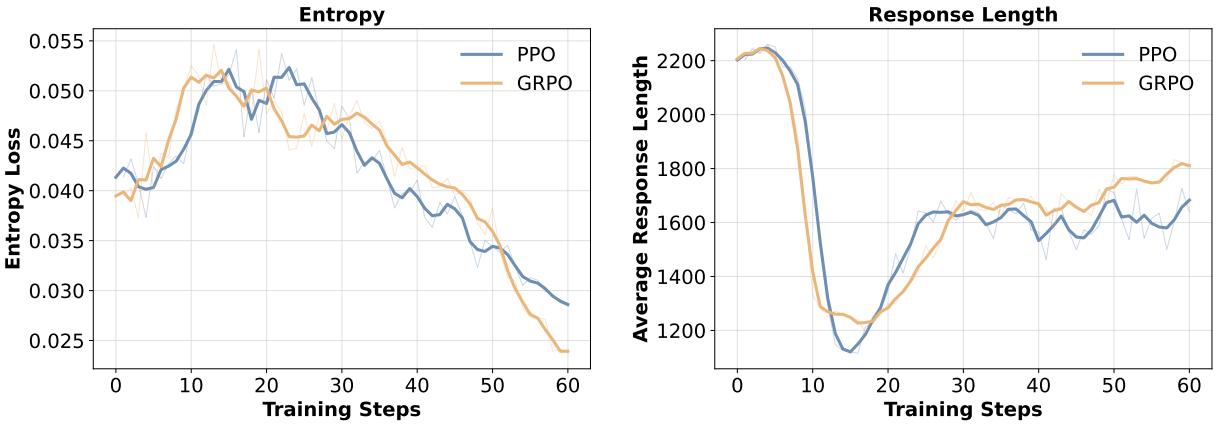

Figure 6: Training dynamics of TKG-Thinker implemented with GRPO and PPO on MULTITQ. **Left**: Generation Entropy; **Right**: Response Length.

D Training Dynamics Details

Figure 6 illustrates the evolution of training entropy and response length, providing deeper insights into the training process of TKG-Thinker.

Training Entropy. The left panel shows that after an initial exploration phase, the policy entropy for both PPO and GRPO consistently declines. This trend indicates that the model is successfully converging from diverse exploration to a stable, optimized policy by exploiting high-reward reasoning paths.

Response Length. The right panel exhibits a distinct "V-shaped" trajectory:

- **Alignment Phase (Steps 0-15):** A sharp decline in length occurs as the model learns to prune redundant tokens and strictly follow the required agentic interaction format.
- **Reasoning Expansion (Steps 15-60):** The length gradually increases and stabilizes, suggesting that once the format is mastered, TKG-Thinker learns to generate more substantial reasoning chains and necessary tool calls to solve complex temporal questions.

E Case Study

To illustrate how TKG-Thinker, equipped with autonomous planning and adaptive retrieval, performs temporal reasoning over TKGs to obtain correct answers, we analyze several representative questions.

MULTITQ. Figures 7 and 8 present case studies of Equal and Equal Multi questions, respectively. These examples demonstrate that TKG-Thinker can accurately identify relevant temporal conditions, retrieves supporting quadruples for each constraint, and integrates the evidence to derive the final answer. Figures 9 and 10 further illustrate After First and Before Last questions. In these cases, TKG-Thinker first plans an anchor event, then performs iterative retrieval under strict temporal constraints, and crucially conducts internal temporal verification to ensure that no earlier or later events violate the query requirements. For example, Figure 10 shows that TKG-Thinker performs bounded verification via repeated *Search_between(query,* t_1, t_2*)* calls, updating the candidate from Association_of_Southeast_Asian_Nations to Qatar and finally to Japan. These examples provide concrete evidence of the model's verification behavior during temporal reasoning.

CronQuestions. Unlike MULTITQ, which represents facts as timestamped quadruples, CronQuestions models knowledge using quintuples with explicit temporal intervals. As illustrated in Figure 11 and Figure 12, both simple and complex queries in CronQuestions require reasoning over interval-level temporal constraints. In this setting, TKG-Thinker is able to decompose a given question into a sequence of executable subtasks via the plan action, perform iterative and temporally aligned retrieval, and ultimately produce the correct answer.

TimelineKGQA. Figure 13 illustrates a TimelineKGQA example. Although TKG-Thinker is not trained on this dataset, it successfully adapts its tool usage strategy, verifies temporal consistency, and produces the correct answer. This result demonstrates that TKG-Thinker generalizes well to previously unseen temporal settings.

Example Trajectory on MULTITQ with Temporal Tool Calls (Equal Question)

Questions: When did Mswati III denounce Swaziland? Gold Answer: 2011-03-19

<think>
The question asks for the specific time when Mswati III denounced Swaziland.This is an Equal-type temporal question, as it requires identifying a singleexact timestamp. The answer type is time, and the required granularity is day. To answer this question, I need to retrieve events where Mswati III criticized or denounced Swaziland and extract the corresponding date.The appropriate next step is to use a time-oriented retrieval function.
</think>
<plan>
1. Use search_time to retrieve events relevant to the question: when did Mswati III denounce Swaziland?
2. From the retrieved events, locate the one that satisfies the query condition.
3. Extract its timestamp in the required granularity and output it as the final answer.
</plan>
<think>
Following the plan, I will now call a time-oriented retrieval function to obtain events related to Mswati III denouncing Swaziland, and then I will infer the full date from the timestamps of the matching events.
</think>

<search_time> ("when did Mswati III denounce Swaziland?") **</search_time>**
<observation>
Doc 1: (Time: 2011-03-19) Mswati_III Criticize_or_denounce Swaziland on 2011-03-19
Doc 2: (Time: 2011-04-12) Swaziland Criticize_or_denounce Mswati_III on 2011-04-12
……
Doc 14: (Time: 2012-08-20) Mswati_III Demand Swaziland on 2012-08-20
Doc 15: (Time: 2013-05-05) News_Editor_(Swaziland) Criticize_or_denounce Swaziland on 2013-05-05
</observation>
<think>
The observation contains an event where Mswati III criticized or denounced Swaziland on 2011-03-19. This matches the query condition of Mswati III denouncing Swaziland. Therefore, the correct date is 2011-03-19.
</think>

<answer> 2011-03-19 **</answer>**

Figure 7: A Case Study on MULTITQ for Equal Questions.

Example Trajectory on MULTITQ with Temporal Tool Calls (Equal Multi Question)

Questions: Who was Kuwait negotiating with in the same month as Christine Lagarde? Gold Answer: UN Security Council

---

<think> The question asks for the entity that Kuwait was negotiating with in the same month as Christine Lagarde. This is an Equal-Multiple type question, where the answer type is entity, and the required time granularity is month. To answer this, I should first identify the relevant month(s) by finding when Christine Lagarde was involved in negotiations. The appropriate next function is search_time.</think>
<plan>
1. Use search_time to retrieve events involving Christine Lagarde to identify the relevant month(s) of negotiation. 2. If the initial search_time query does not yield sufficient or relevant temporal evidence, reformulate the search_time query and repeat the retrieval until a valid month can be identified. 3. Use search_specific with the identified month to retrieve events in which Kuwait was negotiating with an entity. 4. Output all entities that satisfy the condition.</plan>
<think> Following the plan, I will first retrieve events related to Christine Lagarde to determine the relevant month of negotiation.</think>

<search_time> ("Christine Lagarde negotiating") </search_time>
<observation> Doc 1: (Time: 2014-06-05) Christine_Lagarde Engage_in_negotiation Yannis_Stournaras on 2014-06-05 Doc 2: (Time: 2008-11-12) Christine_Lagarde Engage_in_negotiation Agustín_Carstens on 2008-11-12…… **Doc 7: (Time: 2009-06-21) Christine_Lagarde Engage_in_negotiation Kuwait on 2009-06-21**……Doc 15: (Time: 2015-01-17) Christine_Lagarde Express_intent_to_meet_or_negotiate Macky_Sall on 2015-01-17</observation>
<think>The observation shows that Christine Lagarde was negotiating with Kuwait in June 2009. I will now retrieve events in which Kuwait was negotiating with an entity in the same month,June 2009</think>

<search_specific> ("Kuwait negotiating with", "2009-06") </search_specific>
<observation> **Doc 1: Kuwait Engage_in_negotiation UN_Security_Council on 2009-06-03** Doc 2: Kuwait Engage_in_negotiation Christine_Lagarde on 2009-06-21 Doc 3: UN_Security_Council Engage_in_negotiation Kuwait on 2009-06-03……Doc 15: Iraq Consult Kuwait on 2009-06-03
</observation>
<think> The observation shows that in June 2009, Kuwait was negotiating with the UN Security Council. Therefore, the correct answer is UN Security Council.</think>

<answer> UN Security Council </answer>

Figure 8: A Case Study on MULTITQ for Equal Multi Questions.

Example Trajectory on MULTITQ with Temporal Tool Calls (After First Question)

Questions: Who wants to negotiate with the citizens of North Korea first, after China? Gold Answer: Envoy (United States)

<think> The question asks for the entity that wanted to negotiate with the citizens of North Korea first after China. This is an After-First temporal question. The required time granularity is day, and the answer type is entity. To solve this, I must first identify the time when China wanted to negotiate with the citizens of North Korea. The next step is to retrieve this anchor time using search_time. </think>
<plan> 1. Use search_time to find when China wanted to negotiate with the citizens of North Korea and determine the anchor time. 2. Use search_after to retrieve all negotiation intent events with the citizens of North Korea that occurred strictly after time t.3. Among these events, identify the earliest one as the candidate.4. If necessary, use search_between to verify that no earlier negotiation occurred between t and the candidate time.5. Return the entity involved in the confirmed first negotiation after China. </plan>
<think> First, I will retrieve the date of the Eurovision Song Contest 2005. </think>

<search_time> ("China want to negotiate citizens of North Korea") </search_time>
<observation> Doc 1: (Time: 2005-07-23) Citizen_(North_Korea) Express intent to meet or negotiate China on 2005-07-23 ……**Doc 7: (Time: 2006-04-08) China Express intent to meet or negotiate Citizen (North_Korea) on 2006-04-08** …… Doc 15: (Time: 2005-03-23) Wen Jiabao Express intent to meet or negotiate Citizen_(North_Korea) on 2005-03-23</observation>
<think> The observation shows that China expressed an intent to negotiate with the citizens of North Korea on 2006-04-08 (Doc 7). This establishes the anchor time as 2006-04-08. I now need to find who wanted to negotiate with the citizens of North Korea first after this date. Therefore, the next step is to retrieve negotiation events with the citizens of North Korea that occurred after 2006-04-08. </think>

<search_after> ("Express intent to meet or negotiate Citizen_(North_Korea)", "2006-04-08") </search_after>
<observation> Doc 1: Citizen_(North_Korea) Express intent to meet or negotiate International Government Organizations at 2007-05-22…… **Doc 14: Envoy_(United_States) Express intent to meet or negotiate Citizen_(North_Korea) at 2006-04-11** Doc 15: Citizen_(North_Korea) Express intent to meet or negotiate South_Korea at 2006-08-24</observation>
<think> The observation lists multiple negotiation intent events occurring after the reference time. To answer the question, I must consider only events where an entity expressed an intent to negotiate with the citizens of North Korea. Among these events, the earliest one after 2006-04-08 appears to be Envoy_(United_States) Express intent to meet or negotiate Citizen_(North_Korea) at 2006-04-11 (Doc 14). However, to confirm this is truly the earliest, I should check if there are any negotiation intent events between 2006-04-08 and 2006-04-11 using search_between. <think>
<search_between> ("Express intent to meet or negotiate Citizen_(North_Korea)", "2006-04-08", "2006-04-11") </search_between>
<observation> **Doc 1: Envoy_(United_States) Express intent to meet or negotiate Citizen_(North_Korea) on 2006-04-11** Doc 2: Japan Express intent to meet or negotiate Citizen_(North_Korea) on 2006-04-08…... Doc 15: Citizen_(Croatia) Express intent to engage in diplomatic cooperation (such_as_policy_support) Party_Member_(Croatia) at 2006-04-08</observation>
<think> The observation confirms that the earliest negotiation intent event by an entity after 2006-04-08 is Envoy_(United_States) Express intent to meet or negotiate Citizen_(North_Korea) at 2006-04-11 (Doc 1). No other negotiation intent events with the citizens of North Korea occurred between 2006-04-08 and 2006-04-11. Therefore, Envoy_(United_States) is confirmed as the first entity that wanted to negotiate with the citizens of North Korea after China.</think>

<answer> Envoy_(United_States) </answer>

Figure 9: A Case Study on MULTITQ for After First Questions.

Example Trajectory on MULTITQ with Temporal Tool Calls (Before Last Question)

Questions: Before Kuwait, which country was the last to express the intention to participate in diplomatic cooperation South Korea? Gold Answer: Japan

---

<think> The question asks for the country that expressed the intention to participate in diplomatic cooperation with South Korea most recently before Kuwait did. This is a Before-Last temporal question. The required time granularity is day, and the answer type is country…… The next step is to retrieve this anchor time using search_time. </think>
<plan> 1. Use search_time to find when Kuwait expressed the intention to participate in diplomatic cooperation with South Korea and determine the anchor time t. 2. Use search_before to retrieve all events where countries expressed the intention to participate in diplomatic cooperation with South Korea that occurred strictly before time t. 3. Identify the most recent expression of intention among these events as a candidate. 4. Use search_between to verify that no later expression occurred between the candidate time and t. 5. Return the confirmed country as the answer.</plan>
<think> Following the plan, I will first retrieve events where Kuwait expressed the intention to participate in diplomatic cooperation with South Korea to determine the anchor time </think>
<search_time> ("Kuwait express_intention_to_participate_in_diplomatic_cooperation South Korea“)</search_time>
<observation> **Doc 1: (Time: 2007-03-07) Kuwait Express intent to engage in diplomatic cooperation (such_as_policy_support) South_Korea on 2007-03-07**…… Doc 15: (Time: 2007-05-24) Kuwait Express intent to engage in diplomatic cooperation (such_as_policy_support) Vietnam on 2007-05-24 </observation>
<think> The observation shows that Kuwait expressed the intention to engage in diplomatic cooperation with South Korea on 2007-03-07 (Doc 1). This establishes the anchor time as 2007-03-07…... Therefore, the next step is to retrieve events where countries expressed the intention to engage in diplomatic cooperation with South Korea that occurred strictly before 2007-03-07.</think>
<search_before> (" Express intent to engage in diplomatic cooperation South Korea", "2007-03-07") </search_before>
<observation> **Doc 1: South_Korea Express intent to engage in diplomatic cooperation (such_as_policy_support) Association_of_Southeast_Asian_Nations on 2006-05-03**……. </observation>
<think> The observation lists multiple events where countries expressed the intention to engage in diplomatic cooperation with South Korea before the anchor time……However, to confirm that this candidate is truly the last one, I must verify whether any expression occurred after 2006-05-03 but before 2007-03-07. The next step is to perform a bounded verification using search_between.<think>
<search_between>("Express intent to engage in diplomatic cooperation South Korea", "2006-05-03", "2007-03-07")</search_between>
<observation>……. **Doc 7: Qatar Express intent to engage in diplomatic cooperation (such_as_policy_support) South_Korea on 2007-01-31**……Doc 15: South_Korea Express intent to engage in diplomatic cooperation (such_as_policy_support) Japan on 2007-01-12 </observation>
<think> The observation shows several expressions of intention to engage in diplomatic cooperation with South Korea between 2006-05-03 and 2007-03-07. Among these, the most recent expression by a country is on 2007-01-31 by South_Korea to Qatar. This is the current candidate for the last expression of intention before Kuwait on 2007-03-07. The next step is to perform another bounded verification using search_between. </think>
<search_between>("Express intent to engage in diplomatic cooperation South Korea", "2007-01-31", "2007-03-07")</search_between>
<observation>Doc 1: Business_(South_Korea) Express intent to engage in diplomatic cooperation (such_as_policy_support) South_Korea on 2007-02-20……**Doc 9: Japan Express intent to engage in diplomatic cooperation (such_as_policy_support) South_Korea on 2007-02-25**…Doc 15: People_First_Party_Korea Express intent to engage in diplomatic cooperation (such_as_policy_support) Kuomintang on 2007-02-25 </observation>
<think>The observation shows several expressions of intention to engage in diplomatic cooperation with South Korea between 2007-01-31 and 2007-03-07. Among these, the most recent expression by a country is on 2007-02-25 by Japan. This is the current candidate for the last expression of intention before Kuwait on 2007-03-07. To confirm that this candidate is truly the last one, I must verify whether any expression occurred after 2007-02-25 but before 2007-03-07. The next step is to perform another bounded verification using search_between.</think>
<search_between> ("Express intent to engage in diplomatic cooperation South Korea", "2007-02-25", "2007-03-07") </search_between>
<observation> ……**Doc 4: Japan Express intent to engage in diplomatic cooperation (such_as_policy_support) South_Korea on 2007-02-25**...Doc 15: Japan Express intent to cooperate South Korea on 2007-03-05</observation>
<think>The observation shows that Japan expressed the intention to engage in diplomatic cooperation with South Korea on 2007-02-25, and no other country expressed such intention between 2007-02-25 and 2007-03-07. Therefore, Japan is confirmed as the last country to express the intention to participate in diplomatic cooperation with South Korea before Kuwait. </think>

<answer> Japan </answer>

Figure 10: A Case Study on MULTITQ for Before Last Questions.

Example Trajectory on CronQuestions with Temporal Tool Calls (Simple Question)

Questions: Before Athens 2004, who held member of the Swedish Parliament's position? Gold Answer: Alf Svensson, Gunilla Carlsson

---

<think> The question asks who held the position of member of the Swedish Parliament before Athens 2004. To answer this, I need to determine the time period of Athens 2004. Then, I will retrieve members of the Swedish Parliament whose service ended before or at the start of Athens 2004. </think>
<plan>
1. Use search_time to retrieve the time interval for Athens 2004.
2. Identify the starting year of Athens 2004.
3. Use search_before with that starting year to retrieve members of the Swedish Parliament whose service ended before or at that time.
4. Extract their names as the final answer.
</plan>
<think> First, I will retrieve the time interval for Athens 2004. </think>

<search_time> ("Athens 2004") </search_time>
<observation> Doc 1: (Time: 2004-2004) Atromitos Yeroskipou victory CFA Integration Championship from 2004-01-01 to 2004-01-01 **Doc 2: (Time: 2004-2004) 2004 Summer Olympics significant event occurrence from 2004-01-01 to 2004-01-01**……Doc 15: (Time: 2004-2004) 2004 Advanta Championships significant event occurrence from 2004-01-01 to 2004-01-01 </observation>
<think> The observation indicates that Athens 2004 refers to the 2004 Summer Olympics, which occurred on 2004-01-01. Therefore, the start year of Athens 2004 is 2004. Next, I need to find members of the Swedish Parliament whose service ended before or at the year 2004. </think>

<search_before> ("member of the Swedish Parliament", "2004-01-01") </search_before>
<observation>Doc 1: Hans Karlsson i Hallsberg position held member of the European Parliament from 2000-01-01 to 2004-01-01 Doc 2: Cecilia Malmström position held member of the European Parliament from 1999-01-01 to 2004-01-01…… **Doc 12: Alf Svensson position held member of the Riksdag from 1991-01-01 to 2009-01-01**……**Doc 15: Gunilla Carlsson position held member of the Riksdag from 2002-01-01 to 2013-01-01** </observation>
<think>From the observation, many documents describe positions held in the European Parliament, which is not the Swedish Parliament. I need to focus on documents referring to the Swedish Parliament, also known as the "Riksdag". Doc 12: Alf Svensson position held "member of the Riksdag" from 1991-01-01 to 2009-01-01, Doc 15: Gunilla Carlsson position held "member of the Riksdag" from 2002-01-01 to 2013-01-01 Both terms overlap the time point 2004-01-01, which corresponds to "before Athens 2004". Therefore, both individuals were members of the Swedish Parliament before Athens 2004. Thus, the valid answers are: Alf Svensson and Gunilla Carlsson. </think>

<answer>Alf Svensson, Gunilla Carlsson</answer>

Figure 11: A Case Study on CronQuestions for Simple TKGQA Questions.

Example Trajectory on CronQuestions with Temporal Tool Calls (Complex Question)

Questions: What person held the position of Agriculture Minister of Israel in ESC 2005? Gold Answer: Yisrael Katz

---

<think> The question asks for the person who held the position of Agriculture Minister of Israel during the time of the Eurovision Song Contest (ESC) 2005. To answer this, I need to determine the date of ESC 2005 and then find out who was the Agriculture Minister of Israel during that time. </think>
<plan> 1. Use search_time to retrieve the date of the Eurovision Song Contest 2005. 2. Identify the specific date or time interval of ESC 2005. 3. Use search_between with that date or time interval to find the person who held the position of Agriculture Minister of Israel during that period. 4. Extract the name of the person as the final answer. </plan>
<think> First, I will retrieve the date of the Eurovision Song Contest 2005. </think>

<search_time> ("Eurovision Song Contest 2005 date") </search_time>
<observation> **Doc 1: (Time: 2005-2005) Eurovision Song Contest 2005 significant event occurrence from 2005-01-01 to 2005-01-01** Doc 2: (Time: 2013-2013) Eurovision Song Contest 2013 significant event occurrence from 2013-01-01 to 2013-01-01 ……Doc 15: (Time: 2005-2005) 2005 Budapest Grand Prix significant event occurrence from 2005-01-01 to 2005-01-01 </observation>
<think> The observation indicates that the Eurovision Song Contest 2005 occurred on 2005-01-01. Now, I need to find out who held the position of Agriculture Minister of Israel during this time. </think>

<search_between> ("Agriculture Minister of Israel", "2005-01-01", "2005-01-01") </search_between>
<observation> Doc 1: Lars Sponheim position held Minister of Agriculture and Food (Norway) from 2001-01-01 to 2005-01-01 Doc 2: Mary Coughlan position held Minister for Agriculture- Food and the Marine from 2004-01-01 to 2008-01-01 ……Doc 15: Yisrael Katz position held Knesset member from 2003-01-01 to 2006-01 01 </observation>
<think> The search results did not return any information about the Agriculture Minister of Israel during the specified time. I need to refine the search to ensure I retrieve the correct information about the Agriculture Minister of Israel in 2005. </think>

<search_time> ("Agriculture Minister of Israel 2005") </search_time>
<observation> **Doc 1: (Time: 2003-2006) Yisrael Katz position held Minister of Agriculture and Rural Development from 2003-01-01 to 2006-01-01** Doc 2: (Time: 2000-2001) Ehud Barak position held Minister of Agriculture and Rural Development from 2000-01-01 to 2001-01-01…...Doc 15: (Time: 2003-2005) Ehud Olmert position held Minister of Communications from 2003-01-01 to 2005-01-01 </observation>
<think>The search results indicate that Yisrael Katz held the position of Minister of Agriculture from 2003 to 2006. This aligns with the time of the Eurovision Song Contest 2005, which occurred on 2005-01-01.Therefore, Yisrael Katz was the Agriculture Minister of Israel during ESC 2005. </think>

<answer> Yisrael Katz </answer>

Figure 12: A Case Study on CronQuestions for Complex TKGQA Questions.

Example Trajectory on TimelineKGQA with Temporal Tool Calls

Questions: From when to when, Adélaïde Hautval academic degree doctorate, at the same time, German submarine U-96 significant event order, at the same time, Jordi Xuclà position held Member of the Congress of Deputies of Spain?
Gold Answer: No Answer

<think> The question asks for a time range when multiple events occurred simultaneously: Adélaïde Hautval receiving a doctorate, a significant event involving the German submarine U-96, and Jordi Xuclà holding a position as a Member of the Congress of Deputies of Spain. This is an Equal-Multiple type question, requiring the identification of a time window when all these events happened together. The answer type is time, and the required time granularity is day because the question asks "from when to when." To solve this, I must first retrieve events involving each of the individuals and entities mentioned, and then identify the overlapping time period. The appropriate next step is to use search_time to retrieve time-related facts.</think>
<plan>1. Use search_time to retrieve events involving Adélaïde Hautval receiving a doctorate. 2. Use search_time to retrieve events involving a significant event with the German submarine U-96. 3. Use search_time to retrieve events involving Jordi Xuclà holding a position as a Member of the Congress of Deputies of Spain. 4. Identify the time range(s) when all these events occurred simultaneously.5. Provide the identified time range as the final answer.</plan>
<think>Following the plan, I will first retrieve related to Adélaïde Hautval receiving a doctorate.</think>

<search_time> (“Adélaïde Hautval academic degree doctorate ”)</search_time>
<observation>**Doc 1: (Time: 1933-1933) Adélaïde Hautval academic degree doctorate from 1933-01-01 to 1933-01-01** Doc 2: (Time: 1923-1923) Christine Marie Berkhout academic degree Doctor of Philosophy from 1923-01-01 to 1923-01-01……</observation>
<think>The observation shows that Adélaïde Hautval received a doctorate on 1933-01-01. This is the only relevant date for Adélaïde Hautval's doctorate. The next step is to retrieve events related to a significant event involving the German submarine U-96.</think>

<search_time> ("German submarine U-96 significant event order") </search_time>
<observation>Doc 1: (Time: 1941-1941) German submarine U-975 significant event order from 1941-01-01 to 1941-01-01 **Doc 2: (Time: 1938-1938) German submarine U-96 significant event order from 1938-01-01 to 1938-01-01**……</observation>
<think> The observation shows that the German submarine U-96 significant event order occurred on 1938-01-01. This is the only relevant date for the German submarine U-96 event. Now, I need to check if Jordi Xuclà held a position as a Member of the Congress of Deputies of Spain on this date. </think>

<search_time>("Jordi Xuclà position held Member of the Congress of Deputies of Spain") </search_time>
<observation> **Doc 1: (Time: 2004-2008) Jordi Xuclà position held Member of the Congress of Deputies of Spain from 2004-01-01 to 2008-01-01** Doc 2: (Time: 2015-2016) Jordi Xuclà position held Member of the Congress of Deputies of Spain from 2015-01-01 to 2016-01-01 ……</observation>
<think>The observation shows that Jordi Xuclà held a position as a Member of the Congress of Deputies of Spain from 2004-01-01 to 2008-01-01. However, Adélaïde Hautval received a doctorate on 1933-01-01, which does not overlap with Jordi Xuclà's tenure. Therefore, there is no date when all three events occurred simultaneously. Based on the available data, the correct answer is that there is no overlapping time period for all three events.</think>

<answer> No Answer </answer>

Figure 13: A Case Study on TimelineKGQA.