PLAN-ANSWER-REFINE-ON-GRAPH: STRUCTURED PLANNING AND SELF-REFINEMENT FOR LARGE LANGUAGE MODEL REASONING ON KNOWLEDGE GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Incorporating knowledge graphs (KGs) into large language model (LLM) reasoning has shown promise in alleviating hallucinations and factual errors. Although existing paradigms of KG-augmented LLMs have achieved encouraging results, they still exhibit notable limitations when handling multi-hop reasoning and complex logical queries: (1) search space truncation bias: current methods generate linear entity-relation reasoning paths, which can prune correct candidates prematurely during iterative exploration; and (2) entity error amplification: existing methods typically follow the retrieve-and-answer paradigm which causes LLMs to over-rely on retrieved evidence, exacerbating the impact of incorrect entities during reasoning. To alleviate the existing challenges, we propose Plan-Answer-Refine-on-Graph (PARoG), a novel framework for LLM reasoning on knowledge graphs. First, PARoG leverages SPARQL queries from KG data as references, decomposing them into structured step-by-step plans. We further train LLMs to construct such structured plans, which improves the logical consistency of reasoning, ensures uniform step granularity, and facilitates effective execution on the graph. Second, during reasoning over KGs, PARoG adopts a plan-answer-refine paradigm: the model first attempts to answer each sub-query independently, and then refines its prediction by integrating evidence retrieved from the KG. This process mitigates knowledge conflicts between LLM and KG, substantially reducing hallucinations. Experimental results on multiple KG reasoning benchmarks demonstrate that PARoG significantly outperforms state-of-the-art approaches, achieving especially superior accuracy on multi-hop and logically complex queries. Our code is available at `https://anonymous.4open.science/r/prog-D8CD`

1 INTRODUCTION

Large Language Models (LLMs) (Brown et al., 2020; Ouyang et al., 2022; OpenAI et al., 2023; Dubey et al., 2024; Guo et al., 2025) have demonstrated remarkable reasoning capabilities in a wide range of complex natural language processing tasks (Bang et al., 2023; Zhao et al., 2023; Huang & Chang, 2023; Qiao et al., 2023). However, LLMs remain prone to hallucinations and factual errors in real-world applications due to their reliance on implicit parametric knowledge (Hu et al., 2023; Wang et al., 2023a; Huang et al., 2024). Knowledge graphs (KGs), as large-scale structured external source of factual knowledge, offer explicit, interpertable relational structures which can ground LLM reasoning, providing a natural complement to limitations of LLMs (Pan et al., 2024).

Recent LLM$\otimes$KG approaches can be categorized into two paradigms. The first leverages step-wise graph exploration, where LLMs iteratively perform entity–relation walks to progressively construct reasoning paths (Sun et al., 2024; Ma et al., 2025). The second generates global reasoning plans where questions are decomposed into sub-objectives and the KG is queried along the planned path to obtain external information (Luo et al., 2024; Chen et al., 2024b). Though demonstrating notable improvements, these methods often struggle with complex logical queries that involve conjunctions or multiple constraints. Our systematic analysis of existing approaches (as described in Appendix B) identifies the following two fundamental limitations.

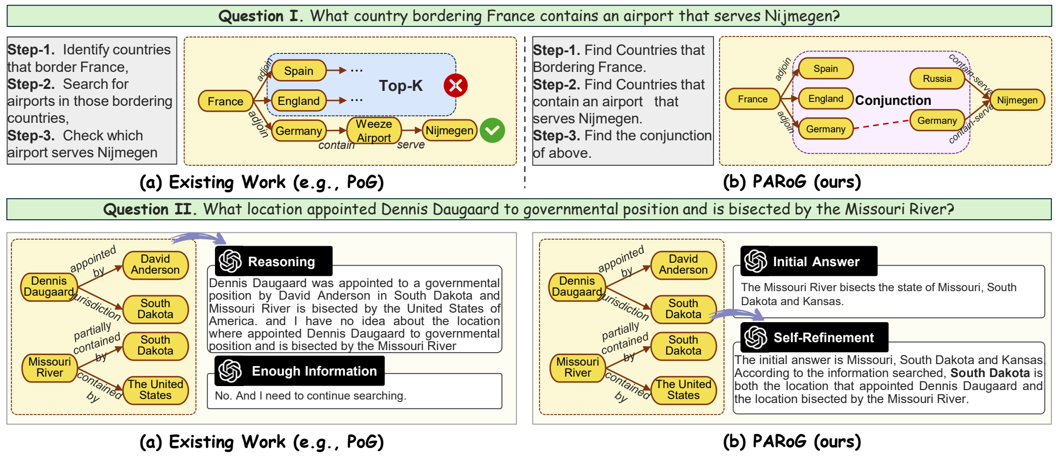
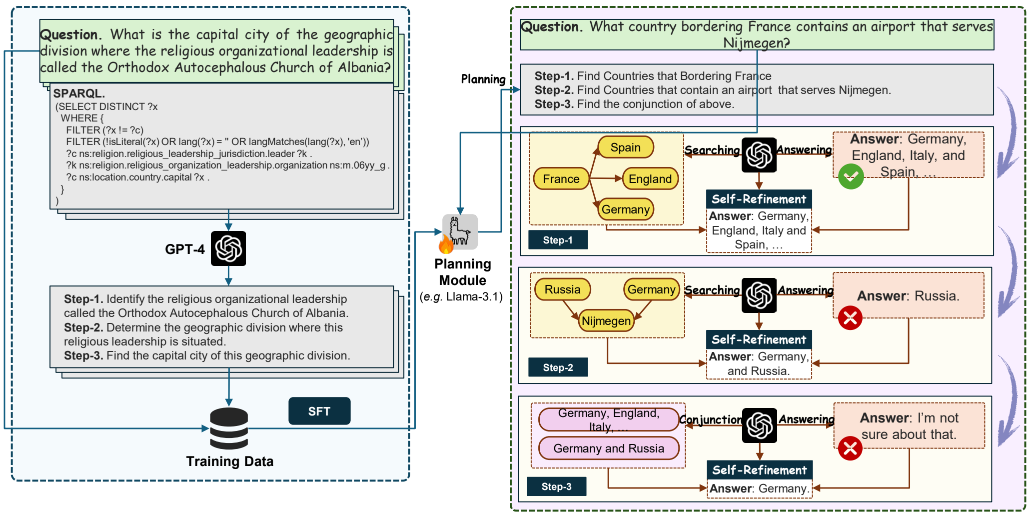


Figure 1: Illustration of the challenges in the existing methods and how our proposed PARoG addresses these issues: I. Search Space Truncation Bias and II. Error Amplification.

Search Space Truncation Bias due to Linear Reasoning Paths. Current methods construct reasoning paths primarily along linear entity–relation steps, iteratively expanding from one entity to its neighbors. To control the combinatorial explosion of graph exploration, they prune candidate entities at each step (*e.g.*, using top-k selection). While efficiency, this strategy often eliminates correct entities prematurely. For instance in Figure 1 (I-a), the correct answer *Germany* is eliminated early due to pruning, leading to an incorrect prediction. A more reasonable planning strategy would first decompose the question into two sub-problems: (i) identify countries bordering France, and (ii) identify countries with airports serving Nijmegen, and then compute the conjunction of these results. The limited planning capability of existing methods fundamentally biases the search space and limits reasoning performance.

Error Amplification from Faulty Entities and Relations. LLM-generated reasoning paths may introduce spurious or weakly related entities and relations during KG exploration. Existing methods typically follow a retrieve-and-answer paradigm, where the LLM heavily relies on the retrieved evidence to produce the final answer. This reliance amplifies errors. For example in Figure 1 (II-a), during graph-based reasoning, the system retrieves facts such as ”*Dennis Daugaard was appointed by David Anderson in South Dakota”* and ”*Missouri River is partially constrained by South Dakota, USA”*. Though individually correct, the knowledge are not sufficiently directly connected to answer the question. Existing methods typically make the LLM to over-rely on the retrieved information, attempt further reasoning steps, and ultimately fail to produce the correct answer.

To alleviate theses challenges, we propose a Plan–Answer–Refine framework (PARoG), a hybrid reasoning paradigm that tightly integrates structured explicit guidance with parametric LLM reasoning. As shown in Figure 2, our method introduces two key technical contributions. First, we leverage SPARQL queries as the structured references to supervise planning and train the planning module using a relatively smaller model (*e.g.* Llama-3.1-8B) to generate flexible, compositional reasoning paths that allow complex logical operations over sub-queries (*e.g.* conjunctions, compositions, superlatives and comparatives). For example in Figure 1 (I-b), instead of searching sequentially from “France” to its neighboring countries and then their airports, the model can generate conjunctive sub-objectives such as “find countries bordering France” and “find countries with airports serving Nijmegen,” then reason over the combination of the sub-objectives, which mitigates search space truncation bias by moving beyond linear expansions. Second, rather than committing to retrieved entities in a one-shot retrieve-and-answer paradigm, PARoG first produces a tentative answer using its parametric knowledge and then explicitly refines it by referring to the retrieved KG entities as shown in Figure 1 (II-b). This refinement step overrides earlier faulty evidences, preventing error amplification due to spurious entities or weakly related relations. Our main contributions are as follows:



(a) SPARQL-Guided Structured Planning (b) Plan-Answer-Refine Paradigm

Figure 2: Overall framework of the proposed PARoG. Unlike prior methods that sequentially expand entity–relation paths with pruning, and follow the one-shot retrieve-and-answer paradigm, PARoG combines (a) structured planning with (b) iterative self-refinement, enabling robust handling of complex logical queries with conjunctions, compositions, comparisons, superlatives.

- We propose leveraging SPARQL as structured references to supervise planning to train the model to generate compositional reasoning paths, which enables the model to handle complex logical reasoning including conjunction, composition, superlative and comparative queries.
- We propose a plan-answer-refine framework, where the agent first attempts to answer then explicitly refines the results using retrieved evidence. This step reduces error propagation caused by faulty entities or relations involved in the reasoning paths.
- We introduce a novel framework PARoG by combining the proposed techniques and evaluate the performance on multiple real-world KGQA datasets. The experimental results demonstrate significant improvements over state-of-the-art baselines.
- We provide further analysis demonstrating that PARoG uses a relatively small model (*e.g.* Llama-3.1-8B (Dubey et al., 2024)) to generate reasoning paths, yet its performance can surpass larger planning LLMs (*e.g.* ChatGPT OpenAI et al. (2023) or Deepseek-R1 (Guo et al., 2025)). We also discuss the broader impact of using structured symbolic guidance with LLM reasoning beyond KGQA.

2 Preliminaries

Knowledge Graph. A *knowledge graph* (KG) is composed of a large set of fact triples, represented as a graph $\mathcal{G} = \{\langle e, r, e' \rangle \mid e, e' \in \mathcal{E}, r \in \mathcal{R}\}$, where $\mathcal{E}$ and $\mathcal{R}$ denote the sets of entities and relations respectively. For KGQA tasks in this paper, we assume the availability of a KG that contains the entities relevant to answering the given natural language question.

SPARQL Query Language. *SPARQL* (SPARQL Protocol and RDF Query Language) is a formal query language which allows users to query structured knowledge bases. Given a question q, the SPARQL query $\mathcal{S}$ specifies a pattern of triples to match against the knowledge graph G. A general SPARQL query consists of a `SELECT` clause that specifies the variables to retrieve, and a `WHERE` clause that defines the graph pattern to match. In our method, SPARQL queries are used to supervise the generation of reasoning paths by decomposing complex queries into smaller sub-queries. Figure 2 provides an example of SPARQL query.

3 METHODOLOGY

In this section, we present PARoG, a novel hybrid reasoning framework which integrates structured guidance planning with parametric reasoning and refinement over knowledge graphs. As shown in Figure 2, PARoG comprises 2 major stages:SPARQL-Guided Structured Planning-training LLMs to generate compositional planning of sub-objective paths based on SPARQL-guided supervision for knowledge graph exploration, and Plan-Answer-Refine Paradigm-iteratively completing a sub-objective with parametric knowledge and then correcting the answer using external KG evidence to mitigate errors and inconsistencies.

3.1 SPARQL-GUIDED STRUCTURED PLANNING

The current LLM⊗KG paradigm typically generates linear entity-relation reasoning paths. In this process, the model starts from an entity and iteratively explores its neighbors by following predefined ⟨*entity-relationship-entity*⟩ paths . While effective for simple queries, this linear path generation often fails to capture complex multi-step reasoning required for more sophisticated queries. Specifically, when reasoning over complex questions involving compositionality, conjunctions, comparatives, or superlatives, LLMs inherent behavior of linear exploration can lead to *Search Space Truncation Bias*, where pruning intermediate candidates prematurely eliminates correct answers.

SPARQL-Guided Supervision. To address this issue, we propose to use SPARQL queries as a structured guide for reasoning. SPARQL inherently supports complex queries that involve logical operations. In this paper, we consider the following operation types:

- **Conjunctions**: finding entities that satisfy multiple constraints simultaneously. *e.g. Find countries that border France and have airports serving Nijmegen.*
- **Compositions**: expressing queries where the output of one relation serves as the input to another. *e.g. Find the capital city of the country that has airports serving Nijmegen.*
- **Comparatives**: retrieving entities based on relative attributes. *e.g. Find countries larger than France in area and have airports serving Nijmegen.*
- **Superlatives**: selecting the best entity according to a ranking predicate. *e.g. Find the largest city bordering France.*

When the intermediate candidate set is large, these query types cannot be properly handled using linear entity-relation paths but are essential for real-world KGQA tasks.

SPARQL-to-Planning. To transfer this expressiveness into model training, we leverage SPARQL queries as guidance signals and use state-of-the-art LLMs to automatically generate planning data from complex questions. For each input question, GPT-4o produces a set of decomposed sub-questions that reflect the logical structure of the underlying SPARQL query. Specifically, we design a systematic pipeline to automatically construct a large-scale dataset tailored for KGQA tasks. The graph-matching process in SPARQL naturally decomposes a complex query into a sequence of consecutive search operations and constraints, thereby providing a precise planning path for identifying intermediate sub-objectives. Building upon this observation, the SPARQL-to-Planning pipeline consists of the following two steps.

- **Source Data Collection.** We first select diverse ⟨*Question*, *SPARQL*⟩ pairs of multi-hop queries from public KGQA training datasets including WebQSP(Yih et al., 2016), CWQ (Talmor & Berant, 2018), and GrailQA (Gu et al., 2021). These pairs serve as the foundation for aligning natural language with structured reasoning.
- **Semantic Consistency Mapping.** With the collected data, we decompose the SPARQL queries into sub-operations and then translate each atomic operation into a fluent natural language question as single sub-objective of the reasoning plan. Following that, we also rephrase the decomposed sub-objective sequence back to natural language queries to maintain the semantic consistency. Instead of the original questions, we use the rephrased natural language queries and the generated sub-objectives as the training data

During dataset construction, the SPARQL queries are decomposed into atomic operations to maintain the consistency across plan steps. In this paper, we use GPT-4o Hurst et al. (2024) to automate the overall process. Finally, the pipeline produces 74,802 high-quality decomposition examples covering a wide range of query types and reasoning depths. The statistics of different query types are summarized in Figure 3.

Model Training. We employ a relatively small but powerful open-source model Llama-3.1-8B Dubey et al. (2024) as the foundation backbone. The training objective follows the standard autoregressive language modeling loss. Specifically, the input template is formatted as following.

```
Instruction: Decompose the following complex question into a logical
sequence of simpler sub-questions.
Input: Question: [The original complex question]
Response: 1. [First sub-question] 2. [Second sub-question] 3. ···
```

Specifically, given the input tokens $\mathbf{x}$, the model parameters θ are optimized by minimizing the negative log-likelihood:

$$\arg\min_{\theta} \mathcal{L}(\theta) = -\sum_{i}^{H}\sum_{j}^{T_h} \log P_\theta(o_{i,j}|\mathbf{o}^h_{i,<j}, \mathbf{x}) \tag{1}$$

where H and T_h deba the total number of sub-objectives and the token number of a single sub-objective respectively, and $\mathbf{o}_i = \{o_{i,1}, \cdots, o_{i,T_h}\}$ is the h-th sub-objective. With supervised training, the model learns to map complex natural language questions into sequences of structured sub-questions which mirror SPARQL compositional logic. This training equips the planning module with the ability to produce complex logical reasoning paths (*e.g.* conjunctions or comparatives), ensuring correct entities are preserved during exploration and mitigating the Search Space Truncation Bias of existing approaches.

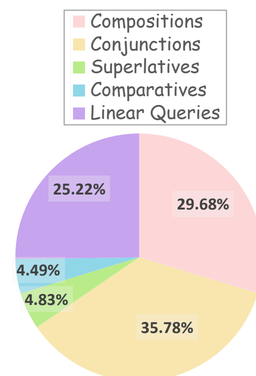


Figure 3: Statistics of different query types of the generated planning data.

3.2 Plan-Answer-Refine Paradigm

Another fundamental challenge for LLM⊗KG reasoning is *Error Amplification from Faulty Entities and Relations*. Existing methods typically adopt the one-shot retrieve-and-answer paradigm, where LLM generates a reasoning path, retrieves corresponding entities and relations from the KG, and then directly uses these retrieved facts to finalize an answer. Though intuitive, this paradigm suffers from two issues:

- **Error Propagation.** When a spurious or weakly related entity is introduced by the reasoning path, the subsequent steps will propagate and accumulate this error.
- **Over-Reliance on Retrieval.** LLMs often assumes the retrieved evidences from KG to be correct and sufficient, even when the external information only partially address the query. This over-reliance prevents the model from self-correcting, leading to faulty answers.

To mitigate this challenge, we introduce the plan-answer-refine mechanism. We employ the LLMs to generate a tentative answer using the parametric reasoning ability and then leverage the KG reasoning agent to iteratively explore the knowledge graph to obtain external information and refine the answer by adjusting entities or relations. The algorithmic procedure of this mechanism is summarized in Appendix C.

Answering. Let $\mathcal{O}$ denotes the reasoning plan generated by the planning module. For each sub-objective $o_i \in \mathcal{O}$ generated by the planning module, the initial step is leveraging a LLM $\mathcal{M}$ to generate a tentative answer as:

$$\hat{a}_i = \mathcal{M}(Q, o_i, I_A) \tag{2}$$

where I_A denotes a predefined instruction template and Q is the input question.

Exploration. The KG exploration process of PARoG is similar to existing work (Sun et al., 2024; Chen et al., 2024b). To be specific for each sub-objective, PARoG starts from an initial entity and iteratively exploring the knowledge graph. Following previous work (Sun et al., 2024; Jiang et al., 2023a), the iterations begins with a set of n_0 topic entities $\mathcal{E}_0$ =

$\{e_1^0, e_2^0, \ldots, e_{n_0}^0\}$. For the i-th iteration ($i > 1$), we first obtain the current set of K reasoning paths $\mathcal{P}_{i-1} = p_1^{i-1}, \cdots, p_K^{i-1}$ after previous $i-1$ iterations. Here, each reasoning path $p_k^{i-1} = [(e_{s,k}^{i-1,1}, r_k^{i-1,1}, e_{o,k}^{i-1,1}), \cdots, (e_{s,k}^{t}, r_k^{i-1,t}, e_{o,k}^{i-1,t}), \cdots, (e_{s,k}^{T_k}, r_k^{i-1,T_k}, e_{o,k}^{T_k})]$ is a sequence of T_k triples ($T_k < i$) where t indexes the elements, $e_{s,k}^{i-1,t}$ and $e_{o,k}^{i-1,t}$ denote the subject and object entities respectively, and $r_k^{i-1,t}$ is a relation linking them. Then, we continue to extend the reasoning paths forward based on the current triples. Concretely, the set of tail nodes in the current reasoning paths is denoted by $\mathcal{E}_{i-1} = \{e_1^{i-1}, e_2^{i-1}, \cdots e_{n_k}^{i-1}\}$ and the relation set is represented as $\mathcal{R}_{i-1} = \{r_1^{i-1}, r_2^{i-1}, \cdots r_{n_k}^{i-1}\}$. We then expand the reasoning path through searching over relations entities. With the original question Q and sub-objectives $\mathcal{O}$, we leverage the LLMs to select the most relevant relations and entities. Specifically, during the relation searching stage, we begin with all relations connected to the tail entities in E^{i-1}, which are denoted by $\mathcal{R}_{init}^K = \{r_{init,1}^i, r_{init,2}^i, \ldots, r_{init,n}^i\}$. and employ the LLM to filter out irrelevant relations. In this step, the entire reasoning plan O is also provided to the LLM so that the model maintains awareness of the global reasoning objective, thereby preventing it from over-focusing on local. Given the tail entities and filtered relations, the missing entities are obtained using predefined SPARQL query templates such as $(e, r, ?)$ or $(?, r, e)$. When all the entities are obtained, we leverage the model to further calculate the relevance between the retrieved entities and the current sub-objective o_i and the question Q. The most relevant entities from a large set of candidates are reserved to update the reasoning path set, which is denoted by $\mathcal{P}_i$.

Self-Refinement. After each KG exploration iteration, PARoG explicitly re-evaluate the tentative answer against the retrieved evidences. If inconsistencies or supplementary are detected, PARoG refines the result by adjusting entities or relations, effectively correcting errors from earlier steps. Specifically we use the LLM $\mathcal{M}$ to correct answer a_i as:

$$a_i = \mathcal{M}(\mathcal{P}_i, o_i, \hat{a}_i, I_R) \tag{3}$$

where $\mathcal{P}_i$ denotes the set of retrieved triples in the current iteration, and I_R is the instruction prompt. It is also worth noting that we also explicitly ask the LLM to judge whether the retrieved knowledge aligns with the question; if it does not, the generated tentative answer is directly used instead.

After each round of self-refinement, PARoG is leveraged to determine whether the current result a_i is sufficient to answer the overall question Q. If the answer is "yes", we stop searching and use a_i as the final answer to avoid over-exploration. Otherwise, PARoG continues iterative searches until PARoG finds enough knowledge or reach the maximum number of iterations. Unlike existing methods, PARoG introduces a mechanism that explicitly integrates the parametric knowledge of LLMs with external knowledge, reducing reliance on any single retrieval and providing resilience against misleading entities or relations.

4 EXPERIMENTS

Datasets. We conduct comprehensive experiments on multiple Knowledge Graph Question Answering (KGQA) benchmark datasets to evaluate the effectiveness of our proposed approach. Specifically, we utilize three widely-adopted datasets: WebQSP (Web Questions Semantic Parses) (Yih et al., 2016), GrailQA (Strongly Generalizable Question Answering) (Gu et al., 2021), and CWQ (ComplexWebQuestions) (Talmor & Berant, 2018). All three datasets are grounded on the Freebase knowledge graph, which contains 88 million entities, 20K relations and 126 million triplets, making it one of the most comprehensive knowledge bases for KGQA evaluation.

Metrics. For evaluation, we adopt the Exact Match accuracy (Hits@1) as our primary metric, which measures the percentage for which the predicted answer exactly matches the ground truth. This ensures that our evaluation strictly reflects the capability to provide precise answers rather than partially correct responses. The results are averaged over three seeds and reported as mean ± 95% confidence interval.

Compared Methods. We compare PARoG with 17 LLM-based approaches from 3 categories: (1) LLM prompting methods, (2) LLM reasoning over KGs (LLM $\otimes$ KG), (3) end-to-end fine-tuned KG-augmented LLMs, and (4) graph-retrieval methods. The details of the compared approaches are described in Appendix E.

Table 1: Performance comparison of different methods on two KGQA benchmarks.

Methods	**WebQSP**	**CWQ**
LLM Prompting		
IO (Brown et al., 2020)	63.3	37.6
CoT (Wei et al., 2022)	62.2	38.8
SC (Wang et al., 2023c)	61.1	45.4
Graph-Retrieval Methods		
GNN-Rag (Mavromatis & Karypis, 2025)	82.8	62.8
SubgraphRag + GPT4o (Li et al., 2025)	87.1	54.9
LLM $\otimes$ KG with GPT-3.5		
ToG (Sun et al., 2024)	76.2	57.1
RoG (Luo et al., 2024)	81.5	52.6
KG-Agent (Jiang et al., 2025)	79.2	56.1
StructGPT (Jiang et al., 2023a)	75.2	55.2
PoG (Chen et al., 2024b)	82.0	63.2
ReKnowS (Wang et al., 2025)	81.1	58.5
PARoG	**89.0** ($\pm$ 1.3)	**73.1** ($\pm$ 0.9)
LLM $\otimes$ KG with GPT-4		
ToG (Sun et al., 2024)	80.7	65.4
KG-Agent (Jiang et al., 2025)	81.2	67.0
StructGPT (Jiang et al., 2023a)	79.5	64.7
PoG (Chen et al., 2024b)	87.3	75.0
ReKnowS (Wang et al., 2025)	83.8	66.8
PARoG	**91.2** ($\pm$ 0.9)	**79.3** ($\pm$ 1.1)

Implementations. For SPARQL-Guided Supervision, we use the training split of WebQSP, GrailQA, and CWQ as the source and employ GPT-4 to generate the training data, and the statistics is summarized in Appendix D. We use Llama-3.1-8B as the backbone to train the planning module with learning rate 2e-5 on 4 Nvidia A800 GPUs. We use GPT-3.5 or GPT-4 to serve as the underlying LLMs and report the results on both, thereby analyzing our method across diverse settings.

4.1 Performance Comparison

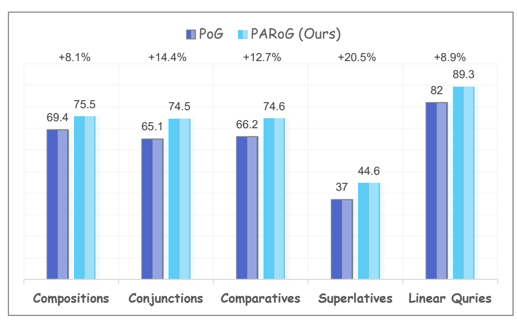


Figure 4: Performance comparison over different query types.

Main Results. The comparison results on WebQSP and CWQ in Table 2. Across both benchmarks, our propose method PARoG consistently outperforms existing approaches. Compared with the state-of-the-art baseline Planning-on-Graph (PoG), PARoG gains substantial improvements of 3.9 and 4.3 points on WebQSP and CWQ respectively with GPT-4. Under the more challenging setting with GPT-3.5, more significant improvements can be observed: PARoG surpasses the baseline by 7.3 on WebQSP and 10.1 on CWQ. It is worth noting that the improvements are particularly pronounced on CWQ, which contains more complex multi-hop and compositional queries, underscoring the advantage of our structured planning and self-refinement mechanism. On the GrailQA benchmark (Table 2), PARoG also achieves consistent state-of-the-art performance across all evaluation settings. Using GPT-3.5, PARoG reaches an overall accuracy of 82.7, surpassing the state-of-the-art Debate-on-Graph (DoG) by a large margin of 4.9 points. Under the stronger GPT-4 setting, our method further improves to an overall 87.2 points, exceeding the compared methods by 2.5 points. It can be observed that the improvements of PARoG are particularly significant on compositional and zero-shot queries, demon-

Table 2: Performance comparison of different methods on GrailQA.

Method	Overall	I.I.D.	Compositional	Zero-shot
LLM Prompting				
IO Prompt (Brown et al., 2020)	29.4	–	–	–
CoT (Wei et al., 2022)	28.1	–	–	–
Self-Consistency (Wang et al., 2023c)	29.6	–	–	–
End-to-End Fine-Tuned KG-Augmented LLMs				
RnG-KBQA (Ye et al., 2022)	68.8	86.2	63.8	63.0
TIARA (Shu et al., 2022)	73.0	87.8	69.2	68.0
FC-KBQA (Zhang et al., 2023)	73.2	88.5	70.0	67.6
Pangu (Gu et al., 2023)	75.4	84.4	74.6	71.6
FlexKBQA (Li et al., 2024b)	62.8	71.3	59.1	60.6
GAIN (Shu & Yu, 2024)	76.3	88.5	73.7	71.8
KG-Agent (Jiang et al., 2025)	86.1	92.0	80.0	86.3
LLM $\otimes$ KG with GPT-3.5				
KB-BINDER (Li et al., 2023a)	53.2	72.5	51.8	45.0
ToG (Sun et al., 2024)	68.7	70.1	56.1	72.7
PoG (Chen et al., 2024b)	76.5	76.3	62.1	81.7
DoG (Ma et al., 2025)	77.8	–	–	–
PARoG	**82.7** (± 1.5)	**85.4** (± 0.9)	**66.7** (± 2.0)	**87.1** (± 2.2)
LLM $\otimes$ KG with GPT-4				
ToG (Sun et al., 2024)	81.4	79.4	67.3	86.5
PoG (Chen et al., 2024b)	84.7	87.9	69.7	88.6
DoG (Ma et al., 2025)	80.0	–	–	–
PARoG	**87.1** (± 1.3)	**89.5** (± 2.1)	**73.2** (± 1.9)	**91.1** (± 2.3)

strating robustness in both complex reasoning and out-of-distribution scenarios. These results highlight that PARoG not only advances the overall accuracy but also generalizes better to complex and zero-shot queries, demonstrating the effectiveness of the proposed methodology.

Analysis on Different Query Types. We summarize the comparison between our method PARoG and PoG on different query types in Figure 4. Overall, our method PARoG consistently outperforms PoG across all types. Compared with simple categories such as Linear Queries and Compositions, the gains become substantially larger on structurally more complex queries. In particular, PARoG achieves significant improvements of 12.7% on Comparatives, 14.4% on Conjunctions, and 20.5 % on Superlatives. These results highlight that PARoG is especially effective in handling queries with multi-step reasoning and complex logical structures.

4.2 Generalization Study

To analyze the robustness across different schema organizations, we evaluate PARoG on CWQ and WebQSP with different source KGs (Freebase and WikiData), as shown in Table 3. Notably, the absolute performance on Wikidata is lower than that on Freebase because the datasets are originally annotated for Freebase. Moreover, Wikidata is substantially larger and more heterogeneous, which increases the difficulty for KG exploration and relation filtering. PARoG consistently produces substantial improvements over compared approaches under the WikiData setting. The results demonstrate that the proposed SPARQL-Guided Structured Planning and Plan-Answer-Refine are not tied to specific relations but generalize well under different schema organizations, relation granularities, and naming conventions. The improvements are especially pronounced on CWQ, where the queries are relatively more complicated.

Table 3: Generalization Study: performance of PARoG using different source KGs on WebQSP and CWQ.

Method	WebQSP	CWQ
with Freebase		
ToG	76.2	57.1
PoG	82.0	63.2
PARoG	**89.3**	**73.3**
with WikiData		
ToG	68.6	54.9
PoG	73.8	60.7
PARoG	**79.1**	**69.5**

4.3 Ablation Study

We conduct ablation studies to examine the contributions of the two core components in our framework SPARQL-Guided Structured Planning and the Plan-Answer-Refine paradigm. The results are shown in Table 5 and 4. First, remove self-refinement consistently reduces performance across all datasets and settings. It can be observed the decline in performance when using GPT-3.5. This result demonstrates the Answer–Refine paradigm effectively mitigates error amplification, especially particularly in scenarios where the underlying LLM is relatively weak. Second, compare our SPARQL-supervised planning module (trained with Llama-3.1-8B) using much larger LLMs directly as planners. Despite having few parameters (8B), our model consistently outperforms GPT-3.5 (∼20B) and DeepSeek-R1 (671B) by large margins (up to 8.1 points on complex CWQ). This result demonstrates that SPARQL-guided supervision provides strong compositional reasoning signals, enabling smaller models to surpass much larger LLMs on reasoning path planning. These ablations prove that both self-refinement and SPARQL-supervised planning are essential to the effectiveness and efficiency of our framework.

To better understand the behavioral contribution of the Plan-Answer-Refine component, we further examine its ability to correct initially incorrect predictions. Specifically, for each dataset, we isolate all examples where the tentative answer produced by the vanilla LLM is incorrect, and compute the proportion of these errors that are successfully fixed after refinement. As shown in Table 6, PARoG corrects around 70% of the initial wrong answers and the correction rate is particularly high on GrailQA, which is consistent with its greater logical compositionality and schema diversity. This result highlights the effectiveness of the answer-refinement paradigm. The correctness rate of the initial answers is listed in Appendix F.

Table 4: Ablation Study: w/ or w/o Self-Refinement (*SR*).

Method	WebQSP	GrailQA	CWQ
	GPT-3.5		
w/o *SR*	88.0	78.9	69.2
w/ *SR*	**89.3**	**82.7**	**73.3**
	GPT-4		
w/o *SR*	89.7	85.5	77.2
w/ *SR*	**91.2**	**87.2**	**79.3**

4.4 Efficiency Analysis

We further analyze the efficiency of different methods in terms of LLM calls and token usage, as shown in Table 7. Our method consistently achieves not only higher accuracy but also greater efficiency across all datasets. These results demonstrate that PARoG not only advances the SOTA performance but also significantly reduces computational overhead, making it more efficient and cost-effective for real-world application.

Table 5: Ablation Study: Comparison our SPARQL-supervised planing module to LLMs.

Method	# Para	WebQSP	GrailQA	CWQ
Ours	8B	**89.3**	**82.7**	**73.3**
GPT-3.5	$\sim$ 20B	83.2	76.9	65.2
Deepseek-R1	671B	88.5	80.2	68.7

4.5 Case Study

We also provide case studies to discuss the effectiveness and limitations of PARoG in Appendix I.

5 Related Work

LLMs with Knowledge Graphs. Large Language Models (LLMs) have shown remarkable reasoning capabilities (Brown et al., 2020; Wei et al., 2022; Zhou et al., 2023) but often prone to hallucinate when answering knowledge-intensive queries (Ji et al., 2023). To address this, combining LLMs reasoning with external knowledge graphs (KGs) have been introduced (Logan et al., 2019; Luo et al., 2023; Jiang et al., 2023b; Pan et al., 2024). Approaches such as KG-based representation

Table 6: Error correction rate (CR Rate) of the Plan-Answer-Refine paradigm.

Dataset	WebQSP	CWQ	GrailQA
CR Rate	73.4	62.4	77.1

Table 7: Efficiency Analysis: Performance vs. token cost across different methods and datasets.

Dataset	Method	LLM Call	Input Token	Output Token	Total Token	Hits@1
WebQSP	ToG	15.9	6,031.2	987.7	7,018.9	76.2
	PoG	9.0	5,234.8	282.9	5,517.7	82.0
	PARoG	**8.3**	**5,012.3**	**241.4**	**5,253.8**	**89.3**
CWQ	ToG	22.6	8,182.9	1,486.4	9,669.4	57.1
	PoG	13.3	7,803.0	353.2	8,156.2	63.2
	PARoG	**10.2**	**7,110.7**	**288.5**	**7,398.8**	**73.3**
GrailQA	ToG	11.1	4,066.0	774.6	4,840.6	68.7
	PoG	6.5	3,372.8	202.8	3,575.6	76.5
	PARoG	**6.0**	**3,180.9**	**178.1**	**3,358.2**	**82.7**

learning (Guu et al., 2020; Li et al., 2023b; Dehghan et al., 2024), knowledge-based instruction-tuning (Zhang et al., 2023; Chen et al., 2024a; Luo et al., 2024), retrieval-augmented generation with KG facts (Wang et al., 2024; Wen et al., 2024; Zhang et al., 2024; Wang et al., 2023b), graph-constrained generation (Guan et al., 2024; Luo et al., 2025), and semantic parsing on KGs (Ye et al., 2022; Yu et al., 2022) demonstrate the benefit of grounding LLM outputs in structured knowledge.

Interactive LLM Reasoning over Knowledge Graphs. Inspired by strong capability of deep reasoning on structured data (Jiang et al., 2023a; Edge et al., 2024; Jin et al., 2024), recent methods introduce explicit reasoning paths to guide LLM interactively inference over KGs and have achieved significant improvements (Yao et al., 2023; Li et al., 2024a; Mavromatis & Karypis, 2024; Sun et al., 2024; Tan et al., 2025; Chen et al., 2024b). Think-on-Graph (Sun et al., 2024) treats reasoning as agent-based exploration where LLMs iteratively search paths with traceability and correction. Generate-on-Graph (Xu et al., 2024) extends to incomplete KGs by enabling LLMs to generate missing triples. Plan-on-graph (Chen et al., 2024b) applies adaptive planning by decomposing questions into sub-goals and refining paths via guidance and reflection. KG-Agent (Jiang et al., 2025) formalizes multi-hop reasoning as program execution with tool use, KG execution, and memory updates. Debate-on-Graph (Ma et al., 2025) models reasoning as a multi-agent debate, where agents generate, and critique reasoning paths to enhance reliability. ReKnoS (Wang et al., 2025) introduces super-relations to connect relational paths, enabling bidirectional reasoning and improving retrieval efficiency. There are also more recent efforts such as (Shen et al., 2025) and (Zhu et al., 2025) introducing alignment and reflection-based strategies to regulate LLM reasoning over KGs.

Our work also belongs to this line of work but differs from prior methods by introducing SPARQL-guided structured planning and answer-refine mechanism. Compared with existing work, the proposed method enables reasoning over complex logical operations beyond linear paths, and explicitly mitigate the inconsistency between the parametric knowledge of LLM and external KG evidence.

6 Discussion and Conclusion

In this paper, we present Plan-Answer-Refine-on-Graph (PARoG) a novel framework for LLM reasoning over knowledge graphs. PARoG introduces two innovations SPARQL-guided structured planning and the Answer-Refinement paradigm, effectively mitigating search space truncation bias and error amplification issues. Extensive experiments on WebQSP, CWQ, and GrailQA demonstrate that PARoG achieves new state-of-the-art results while also being more efficient and cost-effective.

Limitation. Despite the improvements, PARoG still relies on the coverage and correctness of available KGs. Besides, while SPARQL-guided training reduces dependence on large models, generating high-quality planning data still requires strong teacher models (e.g., GPT-4o), which may limit accessibility. Moreover, our refinement process is static and offline without dynamic feedback loops during reasoning, we leave the exploration of online refinement for future work due to its complexity.

Broader Impact and Future Work. PARoG that structured symbolic guidance can enhance LLM reasoning, which can be applied wherever external structured signals are available beyond KGQA. In the future the study direct include dynamic refinement, multi-modal knowledge graphs, and bootstrapped self-improvement, which could make PARoG more scalable, general and accessible.

REFERENCES

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Liyi Chen, Chuan Qin, Ying Sun, Xin Song, Tong Xu, Hengshu Zhu, and Hui Xiong. Collaboration-aware hybrid learning for knowledge development prediction. In *Proceedings of the ACM Web Conference 2024*, pp. 3976–3985, 2024a.

Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. In *Proceedings of the 38th Conference on Neural Information Processing Systems*, 2024b.

Mohammad Dehghan, Mohammad Alomrani, Sunyam Bagga, David Alfonso-Hermelo, Khalil Bibi, Abbas Ghaddar, Yingxue Zhang, Xiaoguang Li, Jianye Hao, Qun Liu, et al. Ewek-qa: Enhanced web and efficient knowledge graph retrieval for citation-based question answering systems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14169–14187, 2024.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. Beyond IID: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference*, pp. 3477–3488, 2021.

Yu Gu, Xiang Deng, and Yu Su. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023.

Xinyan Guan, Yanjiang Liu, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18126–18134, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pp. 3929–3938. PMLR, 2020.

Xuming Hu, Junzhe Chen, Xiaochuan Li, Yufei Guo, Lijie Wen, Philip S Yu, and Zhijiang Guo. Do large language models know about facts? *arXiv preprint arXiv:2310.05177*, 2023.

Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 1049–1065, 2023.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38, 2023.

Jinhao Jiang, Kun Zhou, KeMing Ye, Xin Zhao, Ji-Rong Wen, et al. Structgpt: A general framework for large language model to reason over structured data. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023a.

Jinhao Jiang, Kun Zhou, Xin Zhao, and Ji-Rong Wen. Unikgqa: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. In *The Eleventh International Conference on Learning Representations*, 2023b.

Jinhao Jiang, Kun Zhou, Xin Zhao, Yang Song, Chen Zhu, Hengshu Zhu, and Ji-Rong Wen. KG-agent: An efficient autonomous agent framework for complex reasoning over knowledge graph. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9505–9523, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.468. URL https://aclanthology.org/2025.acl-long.468/.

Bowen Jin, Chulin Xie, Jiawei Zhang, Kashob Kumar Roy, Yu Zhang, Zheng Li, Ruirui Li, Xianfeng Tang, Suhang Wang, Yu Meng, et al. Graph chain-of-thought: Augmenting large language models by reasoning on graphs. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 163–184, 2024.

Mufei Li, Siqi Miao, and Pan Li. Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation. In *The Thirteenth International Conference on Learning Representations*, 2025.

Tianle Li, Xueguang Ma, Alex Zhuang, Yu Gu, Yu Su, and Wenhu Chen. Few-shot in-context learning on knowledge base question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6966–6980, 2023a.

Wendi Li, Wei Wei, Xiaoye Qu, Xian-Ling Mao, Ye Yuan, Wenfeng Xie, and Dangyang Chen. Trea: Tree-structure reasoning schema for conversational recommendation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2970–2982, 2023b.

Xingxuan Li, Ruochen Zhao, Yew Ken Chia, Bosheng Ding, Shafiq Joty, Soujanya Poria, and Lidong Bing. Chain-of-knowledge: Grounding large language models via dynamic knowledge adapting over heterogeneous sources. In *The Twelfth International Conference on Learning Representations*, 2024a.

Zhenyu Li, Sunqi Fan, Yu Gu, Xiuxing Li, Zhichao Duan, Bowen Dong, Ning Liu, and Jianyong Wang. Flexkbqa: A flexible llm-powered framework for few-shot knowledge base question answering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 18608–18616, 2024b.

Robert Logan, Nelson F Liu, Matthew E Peters, Matt Gardner, and Sameer Singh. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5962–5971, 2019.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *The Twelfth International Conference on Learning Representations*, 2024.

Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Yuan-Fang Li, Chen Gong, and Shirui Pan. Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models. In *Forty-second International Conference on Machine Learning*, 2025.

Ziyang Luo, Can Xu, Pu Zhao, Xiubo Geng, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. Augmented large language models with parametric knowledge guiding. *arXiv preprint arXiv:2305.04757*, 2023.

Jie Ma, Zhitao Gao, Qi Chai, Wangchun Sun, Pinghui Wang, Hongbin Pei, Jing Tao, Lingyun Song, Jun Liu, Chen Zhang, et al. Debate on graph: a flexible and reliable reasoning framework for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 24768–24776, 2025.

Costas Mavromatis and George Karypis. Gnn-rag: Graph neural retrieval for large language model reasoning. *arXiv preprint arXiv:2405.20139*, 2024.

Costas Mavromatis and George Karypis. Gnn-rag: Graph neural retrieval for efficient large language model reasoning on knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 16682–16699, 2025.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, and et al. Gpt-4 technical report, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599, 2024.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. Reasoning with language model prompting: A survey. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5368–5393, 2023.

Xiangqing Shen, Fanfan Wang, and Rui Xia. Reason-align-respond: Aligning llm reasoning with knowledge graphs for kgqa. 2025.

Yiheng Shu and Zhiwei Yu. Distribution shifts are bottlenecks: Extensive evaluation for grounding language models to knowledge bases. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 71–88, 2024.

Yiheng Shu, Zhiwei Yu, Yuhan Li, Börje Karlsson, Tingting Ma, Yuzhong Qu, and Chin-Yew Lin. Tiara: Multi-grained retrieval for robust question answering over large knowledge base. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 8108–8121, 2022.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations*, 2024.

Alon Talmor and Jonathan Berant. The Web as a knowledge-base for answering complex questions. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 641–651, 2018.

Xingyu Tan, Xiaoyang Wang, Qing Liu, Xiwei Xu, Xin Yuan, and Wenjie Zhang. Paths-over-graph: Knowledge graph empowered large language model reasoning. In *Proceedings of the ACM on Web Conference 2025*, pp. 3505–3522, 2025.

Jiapu Wang, Sun Kai, Linhao Luo, Wei Wei, Yongli Hu, Alan Wee-Chung Liew, Shirui Pan, and Baocai Yin. Large language models-guided dynamic adaptation for temporal knowledge graph reasoning. *Advances in Neural Information Processing Systems*, 37:8384–8410, 2024.

Keheng Wang, Feiyu Duan, Sirui Wang, Peiguang Li, Yunsen Xian, Chuantao Yin, Wenge Rong, and Zhang Xiong. Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. *arXiv preprint arXiv:2308.13259*, 2023a.

Song Wang, Junhong Lin, Xiaojie Guo, Julian Shun, Jundong Li, and Yada Zhu. Reasoning of large language models over knowledge graphs with super-relations. In *The Thirteenth International Conference on Learning Representations*, 2025.

Xintao Wang, Qianwen Yang, Yongting Qiu, Jiaqing Liang, Qianyu He, Zhouhong Gu, Yanghua Xiao, and Wei Wang. Knowledgpt: Enhancing large language models with retrieval and storage access on knowledge bases. *arXiv preprint arXiv:2308.11761*, 2023b.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023c.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Yilin Wen, Zifeng Wang, and Jimeng Sun. Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10370–10388, 2024.

Yao Xu, Shizhu He, Jiabei Chen, Zihao Wang, Yangqiu Song, Hanghang Tong, Guang Liu, Jun Zhao, and Kang Liu. Generate-on-graph: Treat llm as both agent and kg for incomplete knowledge graph question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 18410–18430, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6032–6043, 2022.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 201–206, 2016.

Donghan Yu, Sheng Zhang, Patrick Ng, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Yiqun Hu, William Yang Wang, Zhiguo Wang, and Bing Xiang. Decaf: Joint decoding of answers and logical forms for question answering over knowledge bases. In *The Eleventh International Conference on Learning Representations*, 2022.

Lingxi Zhang, Jing Zhang, Yanling Wang, Shulin Cao, Xinmei Huang, Cuiping Li, Hong Chen, and Juanzi Li. Fc-kbqa: A fine-to-coarse composition framework for knowledge base question answering. In *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023.

Qinggang Zhang, Junnan Dong, Hao Chen, Daochen Zha, Zailiang Yu, and Xiao Huang. Knowgpt: Knowledge graph based prompting for large language models. *Advances in Neural Information Processing Systems*, 37:6052–6080, 2024.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2), 2023.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*, 2023.

Jiajun Zhu, Ye Liu, Meikai Bao, Kai Zhang, Yanghai Zhang, and Qi Liu. Self-reflective planning with knowledge graphs: Enhancing llm reasoning reliability for question answering. 2025.

A The Use of Large Language Models (LLMs)

We used OpenAI ChatGPT for writings refinement and correction of typos during the preparation of this manuscript.

B Error Analysis of Existing Methods.

We conduct an error analysis of existing methods (ToG and PoG). Among the failed examples:83.39% of WebQSP questions,85.34% of GrailQA questions, and 90.14% of CWQ queries fail due to missing answer entities in the retrieval phase. Figure 5 shows the distribution of different query types of the failed examples.

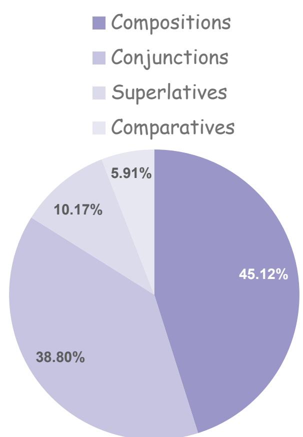


Figure 5: Error Distribution of PoG.

C Algorithm for the Plan-Answer-Refine Paradigm

We summarize the comprehensive algorithmic procedure of Plan-Answer-Refine as shown in Algorithm 1.

D Details of Datasets

WebQSP consists of 4,737 natural language questions that require single or multi-hop reasoning over Freebase. GrailQA presents a more challenging scenario with 64,331 questions designed to test compositional generalization capabilities. CWQ contains 34,689 complex questions requiring multi-hop reasoning and constraint handling. These datasets collectively provide a robust testbed for evaluating various aspects of KGQA performance, including reasoning complexity, generalization ability, and scalability. The statistics of the datasets are summarized in Table 8.

Table 8: Statistics of datasets.

Datasets	#Train	#Test	Max #hop
WebQSP	2,826	1,628	2
CWQ	27,639	3,531	4
GrailQA	44,337	13,231	4

Algorithm 1 PARoG.

Require: Question Q, Knowledge Graph $\mathcal{G}$, LLM $\mathcal{M}$, Planning module $\textsc{Plan}(\cdot)$, instruction templates I_A, I_R, initial topic entity set $\mathcal{E}_0$ (size n_0), max iterations $T_{\max}$
Ensure: Final answer a

$\mathcal{O} \leftarrow \textsc{Plan}(Q)$ ▷ Generate sub-objectives with the SPARQL-supervised planner
$\mathcal{A} \leftarrow \emptyset$
for each sub-objective $o_i \in \mathcal{O}$ **do**
  $\hat{a}_i \leftarrow \mathcal{M}(Q, o_i, I_A)$ ▷ Tentative answer by parametric reasoning
  $\mathcal{P}_0 \leftarrow \{[\,]\}$ ▷ Initialize reasoning paths
  $\mathcal{E}_0 \leftarrow \{e_1^0, \ldots, e_{n_0}^0\}$
  for $t = 1$ **to** $T_{\max}$ **do**
    $\mathcal{E}_{t-1} \leftarrow \textsc{TailEntities}(\mathcal{P}_{t-1})$
    $\mathcal{R}_t^{init} \leftarrow \textsc{NeighborRelations}(\mathcal{G}, \mathcal{E}_{t-1})$
    $\mathcal{R}_t \leftarrow \mathcal{M}(Q, o_i, \mathcal{O}, \mathcal{R}_t^{init})$ ▷ Filter relations with global plan awareness
    $\mathcal{E}_t^{cand} \leftarrow \textsc{SPARQLQuery}(\mathcal{G}, \mathcal{E}_{t-1}, \mathcal{R}_t)$
    $\textsc{score}(e) \leftarrow \mathcal{M}(Q, o_i, e),\ \forall e \in \mathcal{E}_t^{cand}$
    $\mathcal{E}_t \leftarrow \textsc{SelectRelevant}(\mathcal{E}_t^{cand}, \textsc{score})$ ▷ Select a variable number of most relevant entities
    $\mathcal{P}_t \leftarrow \textsc{ExtendPaths}(\mathcal{P}_{t-1}, \mathcal{E}_t, \mathcal{R}_t)$
    $a_i \leftarrow \mathcal{M}(\mathcal{P}_t, o_i, \hat{a}_i, I_R)$ ▷ Self-refine tentative answer
    if $\textsc{Align}(\mathcal{P}_t, Q)$ = **false then**
      $a_i \leftarrow \hat{a}_i$ ▷ Fallback if retrieved knowledge is irrelevant
    end if
    if $\textsc{Sufficient}(a_i, Q)$ = **true then**
      break
    end if
  end for
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{a_i\}$
end for
$a \leftarrow \textsc{Aggregate}(\mathcal{A}, \mathcal{O})$ ▷ Combine refined sub-answers according to $\mathcal{O}$ to form the final answer.
return a

E Details of Compared Baselines

We compare PARoG with 17 LLM-based approaches from 3 categories: (1) LLM prompting methods, (2) LLM reasoning over KGs (LLM $\otimes$ KG), and (3) end-to-end fine-tuned KG-augmented LLMs. The details of the compared approaches are described as follows.

E.1 LLM Prompting

- **Input-Output Prompting** (Brown et al., 2020):A standard few-shot prompting approach without explicit reasoning guidance, serving as a basic LLM QA baseline.
- **Chain-of-Thought** (Wei et al., 2022): Chain-of-Thought prompting encourages the LLM to explicitly generate intermediate reasoning steps, improving logical consistency on complex queries.
- **Self-Consistency** (Wang et al., 2023c): Self-consistency prompting samples multiple reasoning chains and aggregates their results, reducing random errors and improving answer stability.

E.2 LLM $\otimes$ KGs

- **Think-on-Graph** (Sun et al., 2024): Think-on-Graph models reasoning as an agent-based exploration, where the LLM iteratively traverses the knowledge graph to build interpretable paths.
- **Reasoning-on-Graph** (Luo et al., 2024): Reason-on-Graph constrains LLM outputs to faithful graph-grounded reasoning paths, improving interpretability and correctness of answers.
- **KG-Agent** (Jiang et al., 2025): An autonomous agent framework that formalizes multi-hop reasoning as program execution with KG queries, external tool use, and memory updates.

- **StructGPT** (Jiang et al., 2023a): A structured generation framework where LLMs are guided by schema-constrained prompts to produce reasoning paths over structured data.
- **Plan-on-Graph** (Chen et al., 2024b): Plan-on-Graph decomposes complex queries into structured sub-goals and adaptively plans reasoning paths on the KG, enabling better compositional reasoning.
- **ReKnowS** (Wang et al., 2025): ReKnowS introduces the concept of super-relations to connect multiple relational paths, allowing bidirectional reasoning and improving retrieval efficiency.
- **KB-BINDER** (Li et al., 2023a): KB-BINDER bridges LLM reasoning with KG facts using a binding mechanism that grounds parametric knowledge in structured evidence.
- **Debate-on-Graph** (Ma et al., 2025): Debate-on-Graph models reasoning as a multi-agent debate, where different agents generate and critique reasoning paths to improve reliability.

E.3 End-to-End Fine-Tuned KG-Augmented LLMs

- **RnG-KBQA** (Ye et al., 2022): A generation-augmented KBQA model that iteratively ranks candidate answers, combining generative reasoning with retrieval.
- **TIARA** (Shu et al., 2022): A multi-grained retrieval framework designed to strengthen robustness of KBQA systems against noisy or incomplete evidence.
- **FC-KBQA** (Zhang et al., 2023): Fine-to-Coarse composition framework that first retrieves broad candidates and then refines answers hierarchically for complex KBQA.
- **Pangu** (Gu et al., 2023): An end-to-end KBQA model that emphasizes compositional generalization, allowing it to handle more complex query structures.
- **FlexKBQA** (Li et al., 2024b): A flexible, LLM-powered KBQA framework designed for few-shot learning and adaptable to low-resource settings.
- **GAIN** (Shu & Yu, 2024): A KBQA method optimized for distribution shifts, making reasoning more robust across different domains and data splits.

E.4 Graph-Retrieval Methods

- **GNN-RAG** (Mavromatis & Karypis, 2025): A deep learning method uses graph neural networks to retrieve the most relevant nodes and subgraphs for LLM reasoning.
- **SubgraphRag** (Li et al., 2025): A method builds and retrieves localized subgraphs to provide structured graph context for LLMs.

F Statistical Analysis of Major Components

We conduct analysis to measure the proportion of questions for which the search process prematurely prunes the correct path (reported as truncation rate). The results are as follows.

Table 9: Truncation Rate.

Method	Truncation Rate (%)
PoG	38.82
PARoG	19.48

Moreover, to better understand how often the tentative answers produced by the LLM are incorrect, we list the correctness rate (%) in the initial answers generated by LLM as shown in Table 10.

Table 10: Correctness rate of initial answers (%).

Dataset	WebQSP	CWQ	GrailQA
Correctness rate of initial answers (%)	61.7	28.9	43.4

These numbers verify that LLM parametric knowledge is often incomplete or unreliable, highlighting the necessity of a refinement stage grounded in KG execution.

G Stability and Sensitivity Analysis

To analyze the stability of the proposed method, we run the experiments under three seeds and report the mean ± 95% CI for different settings. The results are shown in Table 11, 12 and 13.

Table 11: Stability Analysis on main datasets.

Method	WebQSP	GrailQA	CWQ
	GPT-3.5		
ToG	76.4 ± 1.9	68.9 ± 1.5	57.2 ± 1.8
PoG	82.1 ± 2.2	76.5 ± 2.1	63.2 ± 1.2
PARoG	**89.0** ± 1.3	**82.7** ± 1.5	**73.1** ± 0.9
	GPT-4		
ToG	80.7 ± 1.7	80.9 ± 1.4	65.6 ± 2.2
PoG	87.3 ± 1.5	84.3 ± 1.8	75.0 ± 1.4
PARoG	**91.2** ± 0.9	**87.1** ± 1.3	**79.3** ± 1.1

Table 12: Stability analysis on w/ or w/o Self-Refinement (SR).

Method	WebQSP	GrailQA	CWQ
	GPT-3.5		
w/o *SR*	88.0 ± 1.6	78.9 ± 1.4	69.2 ± 1.2
w/ *SR*	**89.0** ± 1.3	**82.7** ± 1.5	**73.1** ± 0.9
	GPT-4		
w/o *SR*	89.7 ± 1.1	85.5 ± 1.4	77.2 ± 1.0
w/ *SR*	**91.2** ± 0.9	**87.1** ± 1.3	**79.3** ± 1.1

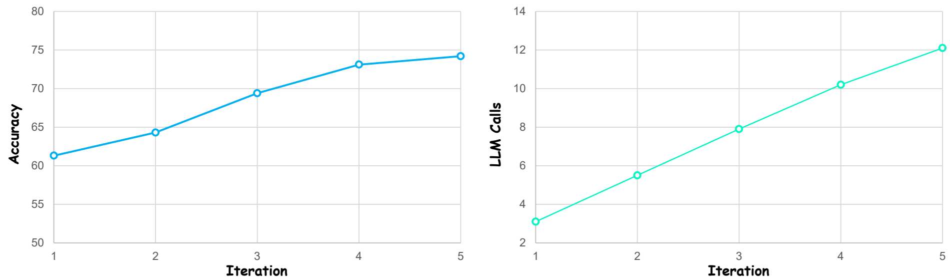
Table 13: Stability and Sensitivity analysis on different planners.

Method	# Para	WebQSP	GrailQA	CWQ
		General LLMs		
GPT-3.5	$\sim$ 20B	83.1 ± 1.4	76.7 ± 1.6	65.4 ± 1.2
Deepseek-R1	671B	88.2 ± 1.5	80.2 ± 1.3	68.7 ± 1.1
		Ours		
llama3.1-8B	8B	**89.0** ± 1.3	**82.7** ± 1.5	**73.3** ± 0.9
llama3.2-1B	1B	87.2 ± 1.2	79.1 ± 1.6	68.5 ± 1.1
llama2-13B	13B	88.4 ± 1.4	81.9 ± 1.2	70.3 ± 1.0

Notably, PARoG shows higher accuracy and consistently smaller confidence intervals than all baselines. Moreover, the smaller planner outperforms larger general LLMs. Even the smallest 1B planner still achieves comparable performance compared to Deepseek-R1, confirming that SPARQL-guided training provides a stronger planning signal than scaling model size alone. These results prove the stability and reproducibility of the SPARQL-guided planner and the Answer-Refine mechanism.

H Efficiency Trade-offs.

To better characterize the efficiency–accuracy trade-off, we conducted an additional study varying the maximum planning iterations in PARoG as shown in Figure 6. Unlike existing methods, PARoG does not use a fixed beam width. Instead, it maintains a dynamic beam, while the overall search budget is controlled by the maximum number of planning iterations.



(a) Accuracy vs. Iteration (b) LLM Calls vs. Iteration

Figure 6: Efficiency Trade-offs.

I Case Study

To better understand the reasoning improvements brought by our proposed method, we analyze two representative failure cases from the baseline method Planning-on-Graph (PoG) and contrast them with the corresponding inference paths of our model (PARoG), as shown in Table 14.

Table 14: Examples of reasoning conducted by PARoG. Red denotes incorrect reasoning paths and answers, while green denotes correct ones.

Case 1: Composition plan error	
Question	What is considered modern in the country where Bilady, Bilady, Bilady language is the national anthem?
Answer	Modern Standard Arabic
PoG	# Plan: [Identify the country where ”Bilady, Bilady, Bilady” is the national anthem, Research the definition of ”modern” in the context of countries] # Reasoning Path: m.0478lbx → government.national_anthem_of_a_country.anthem → Bilady, Bilady, Bilady m.0478lbx → government.national_anthem_of_a_country.country → Egypt # Answer: Egypt
PARoG	# Plan: [Identify the country where ”Bilady, Bilady, Bilady” is the national anthem, Research modern cultural, social, or technological aspects in that country] # Reasoning Path: m.0478lbx → government.national_anthem_of_a_country.anthem → Bilady, Bilady, Bilady m.0478lbx → government.national_anthem_of_a_country.country → Egypt Egypt → location.country.languages_spoken → Modern Standard Arabic # Answer: Modern Standard Arabic
Case 2: Parametric Knowledge Bridging KG Gaps	
Question	What movies did Adam Sandler play in and is about Christmas?
Answer	Eight Crazy Nights
PoG	# Reasoning Path: Adam Sandler → film.actor.film → {Eight Crazy Nights,The Chanukah Song, Reign Over Me, Funny People, The Meyerowitz Stories, The Week Of} # Answer: The Chanukah Song
PARoG	# Reasoning Path: Adam Sandler → film.actor.film → {Eight Crazy Nights, Funny People, Reign Over Me, etc.} LLM answers:Eight Crazy Nights is about Christmas # Answer: Eight Crazy Nights

I.1 Case 1: Composition Plan Error

This example investigates the question:

> What is considered modern in the country where Bilady, Bilady, Bilady language is the national anthem?

The question requires a two-hop reasoning process: (1) identify the country given the national anthem, and (2) determine the language that is considered "modern" in that country. The baseline PoG fails in the planning stage by misinterpreting the intent of "modern" as referring to the country itself rather than a language. As a result, it outputs "Egypt" as the final answer, which is incomplete and incorrect.

In contrast, PARoG correctly preserves the linguistic intent of the original question during the planning phase. Unlike PoG, which misinterprets "modern" as referring to a modern country, PARoG correctly interprets it as referring to a modern language spoken in the identified country.

It first identifies Egypt as the country where "Bilady, Bilady, Bilady" is the national anthem—same as PoG—but goes a step further by reasoning that Modern Standard Arabic is the relevant modern language spoken in Egypt. This is captured in the following reasoning path:

Egypt $\rightarrow$ languages spoken $\rightarrow$ Modern Standard Arabic

By grounding the abstract query term "modern" into a specific linguistic attribute, PARoG successfully answers the question with Modern Standard Arabic, demonstrating its ability to disambiguate vague terms and construct semantically aligned plans that lead to correct, complete answers.

This case highlights the strength of the Plan-Answer-Refine framework in maintaining semantic consistency and avoiding reasoning drift during multi-hop KGQA.

I.2 Case 2: Parametric Knowledge Bridging KG Gaps

We also examine the following question:

> What movies did Adam Sandler play in and is about Christmas?

The PoG baseline retrieves several films involving Adam Sandler—such as The Chanukah Song, Reign Over Me, and Funny People—but it fails to search whether these movies are related to Christmas, ultimately yielding the incorrect result The Chanukah Song. In contrast, PARoG follows a similar retrieval process and also identifies several films involving Adam Sandler through its agent module. However, despite failing to retrieve any explicit evidence from the knowledge graph regarding the "Christmas" constraint, PARoG demonstrates a more robust and semantically grounded refinement process. By leveraging the parametric knowledge encoded within the LLM, it successfully infers that Eight Crazy Nights is a Christmas-themed film. This case illustrates the strength of the plan-answer-refine paradigm in producing correct answers even when symbolic evidence from the KG is absent, showcasing the complementary power of LLM-based reasoning.

J Failure Case Analysis

Table 15 presents representative failure cases of PARoG and highlights the underlying causes across three major error categories. Case 1 illustrates a relation-selection error, where the planner or exploration stage selects a relation chain which is semantically plausible but ultimately incorrect(e.g., directed_by instead of actor-performance relations), diverting the search away from the gold path. Case 2 shows an instance of incomplete multi-branch retrieval, where the system successfully recovers the film-related path but misses an additional TV-series branch required to produce the full multi-answer set. This demonstrates that although PARoG effectively handles compositional constraints, multi-source answer aggregation remains challenging when evidence spans heterogeneous subgraphs. Case 3 shows a failure case caused by KG incompleteness: when numerical attributes or relational paths needed to instantiate a constraint are absent from the underlying KG, PARoG is unable to retrieve the gold entity and must instead rely on parametric knowledge, often leading to incorrect final predictions. Overall, these cases reveal that remaining errors primarily stem from (i) subtle relation disambiguation, (ii) multi-branch retrieval coverage, and (iii) missing or incom-

Table 15: Failure cases of PARoG and corresponding error causes.

Case 1: Relation selection error	
Question	What movie did Ron Howard do with cinematography was by Mark Irwin?
Ground truth	Osmosis Jones
error cause	# PARoG Reasoning Path: Mark Irwin → film.cinematographer.film →Scream Ron Howard" → film.film.directed_by→Apollo 13 Answer:Scream # Ground truth path: Ron Howard → film.actor.film → m.03g24 m.03g24 → film.performance.film → Osmosis Jones Osmosis Jones → film.film.cinematography → Mark Irwin # Cause: Incorrect relation selection (choosing film.film.directed_by and film.cinematographer.film instead of the actor–performance relations) diverts exploration to irrelevant paths, thereby missing the gold answer.
Case 2: Incomplete multi-answer retrieval (partial answer set)	
Question	who plays blaine in batman
Ground truth	Matthew Wagner,Danny Trejo,Carlos Alazraqui,Tom Hardy
error cause	# PARoG reasoning path (only one branch retrieved): # Film path retrieved: Blaine → film.film_character.portrayed_in_films → film.performance.actor → Tom Hardy, Carlos Alazraqui, Matthew Wagner Missing actor from the TV branch: Danny Trejo. # Ground truth path (union of two branches): # Film path: Blaine → film.film_character.portrayed_in_films → film.performance.actor → Tom Hardy, Carlos Alazraqui, Matthew Wagner # TV path: Blaine → tv.tv_character.appeared_in_tv_program → tv.regular_tv_appearance.actor → Danny Trejo # Cause: The ground-truth answer set is formed by the union of a film path and a TV path, but PARoG identifies only one branch (film) and overlooks the tv_program branch, resulting in an incomplete multi-answer retrieval.
Case 3: Path absence due to KG incompleteness.	
Question	What countries in Oceania, that have the emissions per capita in dated metric ton of 0.091351?
Ground truth	Papua New Guinea
error cause	PARoG Reasoning: Cook Islands is a country in Oceania that has emissions per capita of 0.091351 metric tons Path absence due to KG incompleteness. The KG does not contain a valid relational path that instantiates the numeric condition 0.091351 (i.e., this value is missing/not represented), so PARoG cannot retrieve the exact matching entity and is forced to Use the LLM internal Knowledge.

plete KG facts—providing actionable directions for future improvements such as stronger relation grounding, expanded query branching, and KG-aware confidence estimation.

K SEARCH SPARQL

we define several SPARQL queries for Freebase queries, which can be executed to search the relation and entity in the Knowledge graph

```
PREFIX ns: <http://rdf.freebase.com/ns/>
SELECT ?relation
WHERE {
    ns:mid ?relation ?x .
}
```

```
PREFIX ns: <http://rdf.freebase.com/ns/>
SELECT ?relation
WHERE {
    ?x ?relation ns:mid .
}
```

K.1 Entity Search

```
    PREFIX ns: <http://rdf.freebase.com/ns/>
    SELECT ?tailEntity
    WHERE {
        ns:mid ns:relation ?tailEntity .
    }
```

```
PREFIX ns: <http://rdf.freebase.com/ns/>
SELECT ?tailEntity
WHERE {
    ?tailEntity ns:mid ns:relation  .
}
```

K.2 Entity Name

```
PREFIX ns: <http://rdf.freebase.com/ns/>
SELECT DISTINCT ?tailEntity
WHERE {
    {
        ?entity ns:type.object.name ?tailEntity .
        FILTER(?entity = ns:mid)
    }
    UNION
    {
        ?entity <http://www.w3.org/2002/07/owlsameAs> ?tailEntity .
        FILTER(?entity = ns:mid)
    }
}
```

L Prompt Templates for LLM Agents

We introduce the full prompting strategy used in our framework, which can be divided into two main stages:

- **Data Generation Stage:** Generating training data from SPARQL queries.
- **Agent Reasoning Stage:** Guiding the LLM through the full Plan-Answer-Refine reasoning procedure based on decomposed subgoals and retrieved knowledge.

L.1 Data Generation Stage:

For each input question, GPT-4o produces a set of decomposed sub-questions that reflect the logical structure of the underlying SPARQL query.Here we display the prompt we use to generate the sub-question.

L.1.1 DATA GENERATE

```
Please break down the process of answering the question into as few
    subobjectives as possible based on semantic analysis and sparql

Now you need to directly output subobjectives of the following question
    in list format like the example above. The output format should be [
    subobjective1, subobjective2,...]

Q: \{Query\}

Sparql: \{Sparql Query\}

Output:
```

L.2 AGENT REASONING STAGE:

We detail the complete prompt templates used in our iterative reasoning framework, including answer initialization, relation/entity pruning, state updating, and self-refinement.

L.3 INIT ANSWERING

```
ased on your own knowledge, output the current known information required
     to achieve the subobjectives.

\texttt{In-Context Few-shot}

Q: \{Query\}

Subobjectives:\{list of sub questions\}

Now you need to directly output the results of the following question in
    JSON format without other information or notes.

Output:
```

L.4 RELATION PRUNE

```
Please provide as few highly relevant relations as possible to the
    question and its subobjectives from the following relations.

\texttt{In-Context Few-shot}

Q: \{Query\}

Subobjectives:\{list of sub questions\}

Topic Entity: \{Topic Entity\}

Relations: \{list of relations\}

output:
```

The LLM is instructed to directly select a subset of candidate relations(No thresholds), we do not set a fixed thresholds.and this discrete selection is used as the pruned relation set; we do not apply any additional numeric thresholds on LLM scores.

L.5 ENTITY PRUNE

For each triple pattern (e,r,?)(e, r, ?)(e,r,?), we construct a prompt of the form.

```
Which entities in the following list ([] in Triples) can be used to
    answer question? Please provide the minimum possible number of
    entities, and strictly adhering to the constraints mentioned in the
    question.
Now you need to directly output the entities from [] in Triplets for the
    following question in list format without other information or notes.

\texttt{In-Context Few-shot}

Q: \{Query\}

Subobjectives:\{list of sub questions\}

Relation: \{Current Relation\}

Entites:  \{list of entities\}

output:
```

The LLM is prompted to directly choose a subset of the provided candidate entities that are most relevant for answering the question(no thresholds).similarly,we do not set a fixed threshold.

L.6 UPDATE THE KG REASONING STATE

```
Based on the provided information to revise the memory,if the memory has
    conflict with the provided information,use the provided information
    to revise the memory.If the provided information is not enough to
    revise the memory, keep the memory unchanged.

\texttt{In-Context Few-shot}

Now you need to directly output the results of the following question in
    JSON format without other information or notes.

Q: \{Query\}

Memory: \{the status of the sub-questions\}

Knowledge triples: \{Explored Paths\}

output:

Q: \{Query\}

Knowledge triples: \{Explored Paths\}

Output:
```

L.7 SELF-REFINE

```
Given the question and the associated retrieved knowledge graph triples (
    entity, relation, entity), you are asked to
self-refine the initial answers based on them. if the initial answers
    have conflict with the provided information,use the provided
    information to refine them.If the provided information is not enough
    to refine the answers, keep the answers unchanged.
```

```
\texttt{In-Context Few-shot}

Q: \{Query\}

Knowledge triples: \{Explored Paths\}

Inital answers:\{The inital answers generated by the llm\}

Output:
```

L.8 ANSWER

PARoG runs an iterative plan–answer–refine loop with two complementary stopping conditions. First, we cap the planning horizon by a fixed maximum number of iterations. Second, at each iteration we call a final-answer head that takes the original question, the currently explored KG paths, and the refined result of resolved sub-questions as input.

```
Please answer the question based on the memory, related knowledge
    triplets and your knowledge.

Now you need to directly output the results of the following question in
    JSON format (must include "A" and "R") without other information or
    notes.

\texttt{In-Context Few-shot}

Q: \{Query\}

Knowledge triples: \{Explored Paths\}

Memory: \{the status of the sub-questions\}

Output:
```

The model is required to output a JSON object with keys ”A” (final answer) and ”R” (short rationale). If the predicted answer field ”A” is non-empty and not an “unknown” placeholder, we accept it and terminate the whole process early, without further KG exploration. Otherwise, we continue to the next iteration until either a satisfactory answer is produced or the maximum iteration is reached, in which case we fall back to answering based on the LLM’s own parametric knowledge plus the accumulated triples in the context.