Cognition on Graph: Navigating Massive Knowledge Space via Cognitive Cycles and Bidirectional Graph-Text Synergy

Anonymous ACL submission

Abstract

Retrieval-Augmented Generation (RAG) has empowered Large Language Models (LLMs) to tackle knowledge-intensive tasks. However, navigating global, heterogeneous knowledge bases (large-scale knowledge graphs and text corpora) for complex reasoning remains a challenge. Existing methods typically employ reactive, graph-driven exploration strategies, which blindly follow graph topology without adapting to the question context or evolving exploration progress, and lack deep bidirectional synergy between graph and text. To address these limitations, we propose CoG (Cognition on Graph), a cognitive-inspired, training-free framework for adaptive knowledge exploration. Drawing from human problem-solving, CoG performs a continuous *plan-explore-reflect* cycle, where it proactively formulates investigation plans, performs dual-source retrieval, and dynamically reflects on progress to adjust strategies. Crucially, it establishes deep bidirectional synergy between structured graph and unstructured text, where entities extracted from text dynamically guide graph exploration to bridge knowledge gaps. Extensive experiments on seven multi-hop QA benchmarks demonstrate that CoG significantly outperforms state-of-the-art methods while achieving superior exploration efficiency. Our code and datasets are available at `https://anonymous.4open.science/r/CoG-6ED6`.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities across various natural language processing tasks (Achiam et al., 2023; Yang et al., 2025a; Guo et al., 2025). However, their efficacy in knowledge-intensive tasks is hindered by their reliance on static parametric knowledge, which can lead to factual hallucinations and opaque reasoning processes (Huang et al., 2025b; Li et al., 2025; Wang et al., 2025b). To address these challenges, Retrieval-Augmented Generation (RAG) has emerged as a prevailing paradigm, enhancing LLMs by retrieving relevant information from external knowledge sources to improve the timeliness, accuracy, and traceability of their responses (Zhang et al., 2025b; Song et al., 2025b).

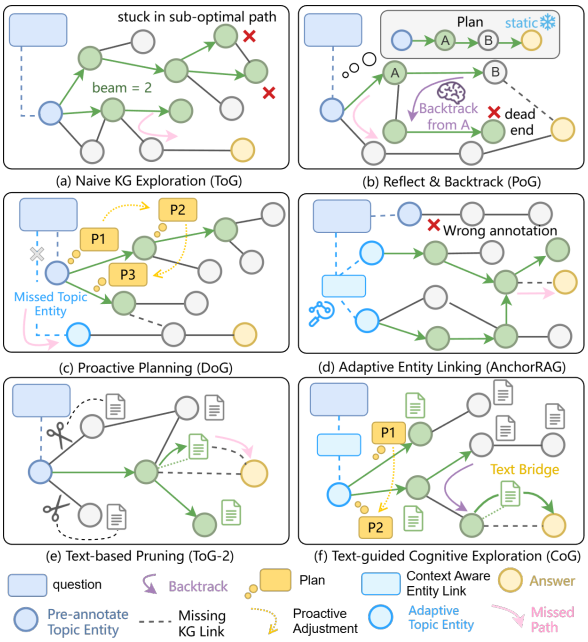

Figure 1: Comparison of exploration strategies. CoG integrates all prior capabilities in a cognitive cycle, uniquely establishing a bidirectional synergy where text actively bridges graph gaps, enabling robust navigation in global knowledge bases.

The core of RAG lies in effective retrieval. Precise extraction of query-relevant information from knowledge bases determines the quality of generation. To achieve this, VectorRAG (Gao et al., 2024; Lewis et al., 2020) leverages vector similarity to retrieve text chunks from a corpus, while GraphRAG (Gutiérrez et al., 2025; Luo et al., 2025a; Gutiérrez et al., 2024; Yang et al., 2025b; Wu et al., 2025; Dong et al., 2025) constructs knowledge graphs (KGs) to guide the retrieval process. However, these methods predominately operate within simplified, local settings. In this setup, the knowledge scope is often restricted to pre-selected oracle para-

graphs known to contain the answers, rather than raw, complete entity documents. This simplified setting masks the inherent noise and distractors present in retrieval, failing to reflect the complexity of real-world scenarios (Luo et al., 2025b).

When transitioning to the global setting, the core challenge escalates from simple information retrieval to complex knowledge exploration. It requires systems to progressively construct evidence chains by synthesizing structural associations from KGs and semantic information from text corpora. Throughout this process, the systems must effectively navigate the noise, ambiguity and distractions inherent in large-scale knowledge bases.

Existing works attempting to address this challenge, however, fall short in two critical aspects (illustrated in Figure 1). First, exploration remains heavily **KG-dependent** with **insufficient text utilization**. Most methods rely solely on KG traversal for exploration (Sun et al., 2024; Chen et al., 2024b; Ma et al., 2025a; Wang et al., 2025a). Even hybrid methods (Ma et al., 2025b) merely use text as auxiliary context, failing to leverage entities within text to actively guide the exploration, leaving the inherent sparsity and incompleteness of KGs unaddressed. Second, the exploration strategies are **fragile and rigid**. Relying on pre-annotated entities as rigid starting points (Xu et al., 2025) and lacking mechanisms for backtracking (Chen et al., 2024b) or proactive adjustment (Ma et al., 2025a), these systems are prone to getting lost in irrelevant subgraphs or stranded by linking failures or knowledge gaps.

To bridge these gaps, we draw inspiration from the cognitive process of human problem-solving. Facing complex questions in open domains, humans do not blindly traverse information; instead, they engage in a dynamic cycle of planning, exploration, and reflection. They proactively formulate investigation plans, flexibly synthesize information from diverse sources, and continuously reflect on accumulated evidence to adjust their strategy.

Guided by this mechanism, we propose CoG (Cognition on Graph), a framework that instantiates this human-like reasoning process into a continuous cognitive loop comprising Planning, Exploration, Synthesis, and Reflection. In the **Planning** phase, instead of relying on pre-annotated entities, CoG proactively analyzes the question to formulate an investigation strategy, from which it derives exploratory sub-queries and identifies relevant entities to anchor the search. During the **Exploration** phase, CoG performs dual-source knowledge gathering. It retrieves 1-hop relevant subgraphs from the KG while simultaneously conducting hierarchical reading of entity-related documents in the text corpus, capturing both structural relations and rich contextual details. The **Synthesis** module then aggregates evidence from both sources into a unified retrieval summary, distilling key insights that advance the reasoning process and achieving deep bidirectional synergy between structured and unstructured knowledge. Finally, the **Reflection** module evaluates the utility of the retrieved information: for productive explorations, it updates the strategy and generates the next query based on the progress made; for unproductive ones, it diagnoses the failure using the global interaction history and adjusts the course, ensuring robust navigation through vast knowledge spaces.

Our main contributions are threefold:

- We propose CoG, a cognitive-inspired, closed-loop, training-free GraphRAG framework. It leverages continuous cognitive cycles of plan-explore-reflect for **adaptive knowledge exploration** in noisy global knowledge bases.
- We establish deep **bidirectional synergy** between structured and unstructured knowledge, where entities dynamically extracted from text actively guide graph exploration to effectively address KG incompleteness.
- Extensive experiments on seven knowledge-intensive, multi-hop QA benchmarks demonstrate that CoG significantly outperforms state-of-the-art methods, showcasing superior reasoning capability and robustness in navigating large-scale knowledge bases.

2 Related Work

RAG enhances LLMs by dynamically retrieving information from external knowledge bases, mitigating hallucinations and domain-specific knowledge gaps (Song et al., 2025b). VectorRAG (Gao et al., 2024; Lewis et al., 2020), the traditional approach, primarily relys on dense vector retrieval to fetch relevant text chunks based on semantic similarity. To mitigate the knowledge fragmentation introduced by text chunking, GraphRAG methods construct graph-structured indices, such as knowledge graphs (Gutiérrez et al., 2024; Guo et al., 2024; Gutiérrez et al., 2025; Zhu et al., 2025), hierarchical community (Edge et al., 2024; Huang et al., 2025a), or

Method	Scope	Source	Text Entity Utilization	Reflection & Backtracking	Proactive Planning	Adaptive Entity Linking	Exploration Strategy
VectorRAG	Local	Text	–	–	–	–	Dense Retrieval
GraphRAG Series†	Local	KG+Text	✓	–	–	–	Graph-Guided Retrieval
StructGPT, KG-Agent	Global	KG	✗	✗	✗	✗	Interaction History
ToG, ReKnoS	Global	KG	✗	✗	✗	✗	Beam Search
PoG	Global	KG	✗	✓	✗	✗	Beam Search w/ Backtrack
DoG	Global	KG	✗	✗	✓	✗	Multi-Agent Debate
AnchorRAG	Global	KG	✗	✗	✗	✓	Parallel Anchor Exploration
KERAG	Global	KG	✗	✗	✗	✓	Schema-Guided Expansion
ToG-2	Global	KG+Text	✗	✗	✗	✗	Beam Search
CoG (Ours)	**Global**	**KG+Text**	✓	✓	✓	✓	**Cognitive Cycle (Plan-Explore-Reflect)**

Table 1: Comparison of different retrieval and exploration methods. "Scope" denotes whether the method operates on a local subset or global knowledge bases. "Source" indicates the type of knowledge used. "†" includes HippoRAG, HippoRAG2, GFM-RAG, et. al. ✓ indicates the feature is present, ✗ indicates it is absent, and "–" means not applicable or not explicitly discussed.

document graphs (Wang et al., 2024; Chen et al., 2025; Li et al., 2024a), to guide retrieval. However, these pioneering works primarily operate in local settings, where the knowledge base consists of oracle paragraphs highly relevant to the answers. This simplified setup fails to assess a model's robustness and generalization in noisy, ambiguous global environments. In contrast, our work targets the more realistic and challenging global knowledge base setting, where systems must explore both large-scale KGs (Wikidata) and comprehensive text corpora (Wikipedia) to construct evidence chains for complex, multi-hop questions.

To address global knowledge exploration, a line of research focuses on iterative reasoning over large-scale KGs. Path-based exploration iteratively traverses KGs from seed entities to gather evidence. StructGPT (Jiang et al., 2023) and KG-Agent (Jiang et al., 2025) perform multi-turn interactions with KGs, while ToG (Sun et al., 2024) employs beam search for parallel path exploration. To improve coverage, ReKnoS (Wang et al., 2025a) expands relations via super-relations, and KERAG (Sun et al., 2025) leverages schema to guide expansion and pruning. Strategy-based methods employ advanced mechanisms for proactive exploration. PoG (Chen et al., 2024b) incorporates backtracking to recover from dead ends, DoG (Ma et al., 2025a) uses multi-agent debate to evolve queries, and AnchorRAG (Xu et al., 2025) employs multi-agent parallel exploration. Learning-based exploration trains specialized modules for autonomous navigation. GRAIL (Chang et al., 2025) and KG-R1 (Song et al., 2025a) adopt imitation or reinforcement learning to train retrieval agents, and KGFR (Cui et al., 2025) integrates a trainable GNN retriever for iterative interaction.

However, these KG-centric methods overlook the rich context and implicit relations within unstructured text, leaving them vulnerable to KG sparsity and incompleteness. To address this, hybrid methods attempt to integrate KGs with text corpora (Lee et al., 2025; Sarmah et al., 2024). CoK (Li et al., 2024b) dynamically retrieves from multi-source knowledge bases to verify and refine intermediate reasoning steps. ToG-2 (Ma et al., 2025b) further synchronizes graph exploration with document retrieval, using graph entities to anchor context acquisition. However, these approaches still exhibit superficial text utilization and lack robust retrieval mechanisms. In contrast, our work draws inspiration from human cognitive processes to achieve deep bidirectional synergy between structured and unstructured knowledge, enabling robust and adaptive multi-turn exploration in noisy, large-scale knowledge spaces.

Table 1 provides a systematic comparison of these methods across key capabilities.

3 Method

3.1 Task Definition

We focus on the task of multi-hop Question Answering (QA) over a global, hybrid knowledge base. Given a complex question q, the system must construct an evidence chain by iteratively retrieving and reasoning over heterogeneous knowledge sources to generate the final answer a.

The knowledge bases $\mathcal{K} = \{\mathcal{G}, \mathcal{D}\}$ comprising two complementary components:

Structured Knowledge Graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, where $\mathcal{E}$ and $\mathcal{R}$ denote the sets of entities and relations. The KG provides structured information in two forms: 1) relational fact triples $\mathcal{T} = \{(h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$ that connect pairs of entities, and

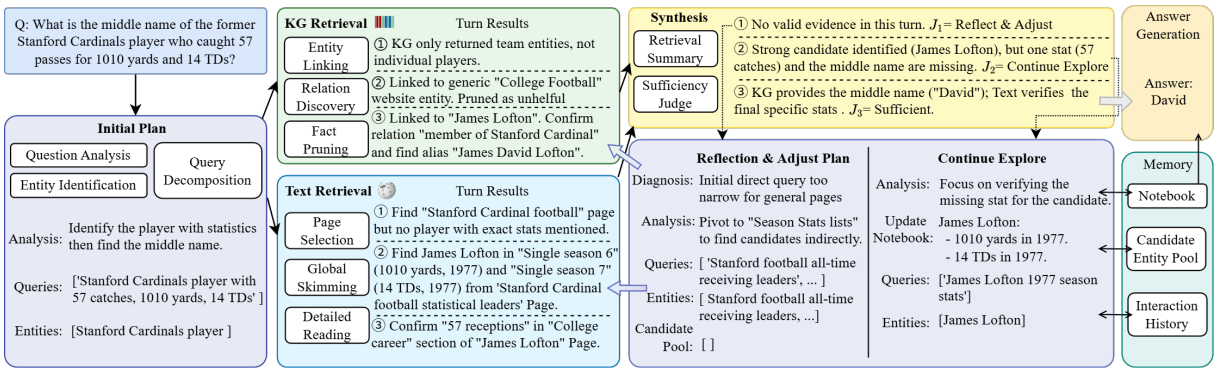

Figure 2: Overview of the CoG framework. It employs a continuous cognitive cycle of plan-explore-reflect to adaptively navigate structured and unstructured knowledge for complex QA.

2) entity attributes (e.g., date of birth, gender) that describe the properties of individual entities.

Unstructured Text Corpus $\mathcal{D}$, a large-scale collection of documents. Each document $d \in \mathcal{D}$ corresponds to an entity and is hierarchically structured, containing a summary paragraph, a table of contents, and the full text organized into sections. It provides rich, fine-grained semantic descriptions that complement the factual statements in $\mathcal{G}$.

3.2 CoG Framework

Figure 2 illustrates the overall architecture of CoG. Given a question q, CoG iteratively executes a cognitive cycle until sufficient evidence is gathered to generate an answer. Each cycle consists of four tightly integrated phases: (1) Planning formulates an investigation strategy and generates a sub-query; (2) Exploration retrieves evidence from both KG and text; (3) Synthesis aggregates and distills the retrieved information; (4) Reflection evaluates progress and decides the next action.

Throughout this process, CoG maintains three long-term memory components to enable coherent multi-turn reasoning: a *Notebook* $\mathcal{N}$ to accumulate verified facts, a *Candidate Entity Pool* $\mathcal{P}$ to manage promising but deferred exploration leads, and an *Interaction History* $\mathcal{H}$ to enable global reflection and strategic adjustment.

3.2.1 Planning

The Planning Module proactively analyzes the question and formulates an initial exploration strategy. Unlike methods that rely on pre-annotated entities, CoG dynamically derives a strategic plan directly from the question q. This process comprises three key components:

Question Analysis. It analyzes the question q to identify information needs and reasoning dependencies, determining what knowledge must be gathered first. This analysis A_0 serves as a high-level guide for subsequent actions.

Query Decomposition. Based on the analysis, the question is decomposed into a set of focused sub-queries $Q_0 = \{q_1^{(0)}, \ldots, q_B^{(0)}\}$, each targeting a specific information need that can be explored independently as the immediate next step.

Entity Identification. It identifies a list of anchor entities $E_0 = \{e_1^{(0)}, \ldots, e_B^{(0)}\}$ corresponding one-to-one with Q_0. Each $e_i^{(0)}$ serves as the core anchor point for initiating the exploration of query $q_i^{(0)}$ in both the KG and text corpus.

Formally, the initial plan P_0 is generated by:

$$P_0 = (A_0, Q_0, E_0) = \text{Prompt}_{\text{init}}(q, \text{LLM}). \quad (1)$$

The prompt is provided in Appendix G, Table 11.

3.2.2 Exploration

Given a sub-query $q_i^{(t)}$ and its anchor entity $e_i^{(t)}$ from the Planning phase, the Exploration module performs **dual-source retrieval** to gather complementary evidence from both the structured KG $\mathcal{G}$ and the unstructured text corpus $\mathcal{D}$.

KG Exploration. To navigate massive global KGs with inherent ambiguity and noise, we design a three-stage pipeline that progressively narrows the search scope from entity disambiguation to relation filtering and fine-grained fact pruning.

(1) Entity Linking. We design a context-aware coarse-to-fine recall strategy. First, we retrieve top-k candidate entities $\mathcal{C}$ via vector similarity between the query context $C_i^{(t)} = (q, A_t, q_i^{(t)}, e_i^{(t)})$ and entity profiles (aliases, description and neighbors). Then, the LLM performs precise disambiguation by evaluating each candidate's profile against the

query context to determine the optimal match:

$$\mathcal{C} = \text{VectorRecall}(C_i^{(t)}, \mathcal{E}, k),$$
$$e^* = \text{Prompt}_{\text{link}}(C_i^{(t)}, \mathcal{C}, \text{LLM}). \quad (2)$$

(2) Relation Discovery. First, we retrieve all relations R_e connected to the linked entity e^*. Then, the LLM selects a subset $R_{\text{sel}} \subset R_e$ relevant to the query context $C_i^{(t)}$ by considering relation labels, connectivity and potential to lead toward the answer, thereby pruning the search space at the schema level:

$$R_{e^*} = \text{GetRelations}(e^*, \mathcal{G}),$$
$$R_{\text{sel}} = \text{Prompt}_{\text{rel}}(C_i^{(t)}, e^*, R_{e^*}, \text{LLM}). \quad (3)$$

(3) Fact Pruning. We retrieve all facts F_{raw} (triples and attributes) associated with the selected relations R_{sel}. The LLM performs fine-grained pruning to retain only facts that either directly answer the query $q_i^{(t)}$ or serve as promising stepping stones for further exploration, filtering out noise while preserving critical evidence:

$$F_{raw} = \{(e^*, r, t) \in \mathcal{T} \mid r \in R_{\text{sel}}, t \in \mathcal{E}\}$$
$$\cup \{(h, r, e^*) \in \mathcal{T} \mid r \in R_{\text{sel}}, h \in \mathcal{E}\},$$
$$F_i^{KG} = \text{Prompt}_{\text{fact}}(C_i^{(t)}, F_{raw}, \text{LLM}). \quad (4)$$

Text Exploration. We propose a hierarchical reading strategy that mimics human reading behavior, narrowing the focus from document retrieval to global skimming and fine-grained section analysis, balancing retrieval efficiency with depth.

(1) Adaptive Page Selection. We employ an adaptive mechanism to retrieve the optimal entity document d_i for $e_i^{(t)}$. When encountering disambiguation or missing pages, the LLM progressively refines the search query by analyzing query context $C_i^{(t)}$ and Wikipedia API feedback, ensuring robust page localization despite real-world noise.

(2) Global Skimming. The LLM skims the high-level structure of the retrieved page d_i (summary, table of contents and infobox) to assess the page's relevance to the query context $C_i^{(t)}$ and extract useful evidence I_{skim}. If deeper investigation is needed, it selects a focused subset of promising sections S_{sel} likely to contain missing details:

$$S_{sel}, I_{skim} = \text{Prompt}_{\text{skim}}(C_i^{(t)}, d_i, \text{LLM}). \quad (5)$$

(3) Detailed Reading. For each selected section $s \in S_{sel}$, we retrieve its full content, split it into chunks, and identify the top-k relevant chunks via dense retrieval. Concurrently, tables within the section are extracted and serialized. Finally, the LLM performs joint analysis of text chunks and tables to synthesize fine-grained evidence I_{detail}^s:

$$I_{detail}^{(s)} = \text{Prompt}_{\text{detail}}(C_i^{(t)}, s, \text{LLM}). \quad (6)$$

Combining skimmed and detailed extractions, the final text evidence for entity $e_i^{(t)}$ is $F_i^{text} = I_{skim} \cup \{I_{detail}^{(s)} \mid s \in S_{sel}\}$.

Dual-Source Evidence. The final retrieved evidence for sub-query $q_i^{(t)}$ is the aggregation of dual-source knowledge: $K_i^{(t)} = F_i^{KG} \cup F_i^{text}$.

Prompts for KG (Tables 12–15) and text (Tables 16–19) exploration are provided in Appendix G.

3.2.3 Synthesis & Reflection

The **Synthesis** module aggregates dual-source evidence and evaluates progress. We design structured formats to present KG and text retrieval $K_t = \{K_1^{(t)}, \ldots, K_B^{(t)}\}$. The LLM first distills key facts and promising leads into a unified retrieval summary M_t, achieving deep synergy between structured KG and unstructured text. Simultaneously, it makes a judgment J_t based on the notebook $\mathcal{N}$ and the current plan $P_t = (A_t, Q_t, E_t)$ to determine whether to generate the final answer, continue exploration, or adjust the strategy:

$$J_t, M_t = \text{Prompt}_{\text{syn}}(q, P_t, K_t, \mathcal{N}, \text{LLM}). \quad (7)$$

Depending on J_t, the **Reflection** module dynamically directs exploration through two pathways:

(1) Continue Exploration. When progress is made but evidence remains incomplete, it refines the analysis to reflect new findings, formulates a new plan P_{t+1} targeting remaining knowledge gaps. Simultaneously, it updates two long-term memory structures: Notebook $\mathcal{N}$, which accumulates verified key facts directly relevant to q, and Candidate Entity Pool $\mathcal{P}$, which stores promising but deferred entities due to exploration width constraints:

$$P_{t+1}, \mathcal{N}, \mathcal{P} = \text{Prompt}_{\text{cont}}(q, P_t, M_t, \mathcal{N}, \mathcal{P}, \text{LLM}). \quad (8)$$

(2) Strategy Adjustment. When facing dead ends, it performs critical reflection by analyzing the global interaction history $\mathcal{H}$ to diagnose strategic errors. Then it pivots the strategy by backtracking to deferred entities in $\mathcal{P}$ or reformulating queries

from entirely new angles based on the q and current progress, thereby ensuring resilient exploration:

$$P_{t+1}, \mathcal{P} = \text{Prompt}_{\text{recov}}(q, P_t, M_t, \mathcal{H}, \mathcal{N}, \mathcal{P}, \text{LLM}). \quad (9)$$

Interaction History Update. After each turn, the system appends the current retrieval summary M_t and plan P_t to the interaction history: $\mathcal{H} \leftarrow \mathcal{H} \cup \{(P_t, M_t)\}$. This cumulative memory enables global retrospection, allowing the system to avoid repeating failed strategies and to make globally informed decisions across multiple turns.

Prompts for synthesis (Table 20) and reflection (Tables 23, 24) are provided in Appendix G.

3.2.4 Overall Workflow & Answer Generation

CoG executes the aforementioned cognitive cycle iteratively, until the Reflection module deems the evidence sufficient or a maximum turn limit is reached. The complete algorithmic workflow is detailed in Algorithm 1 in Appendix A.

Upon sufficiency, the system synthesizes accumulated facts in Notebook $\mathcal{N}$ and final retrieval result M_t. The LLM constructs a comprehensive reasoning chain connecting these facts to directly address the question, producing the final answer a.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluates on seven knowledge-intensive multi-hop QA benchmarks, categorized into two types: (1) KG-based QA: KGQA-Gen (Zhang et al., 2025a), CWQ (Talmor and Berant, 2018), QALD10-en (Usbeck et al., 2023), and WebQSP (Yih et al., 2016), which primarily require reasoning over structured facts. (2) Text-based QA: 2WikiMQA (Ho et al., 2020), AdvHotpotQA (Ye and Durrett, 2022), and MusiQue (Trivedi et al., 2022), which demand deep semantic understanding and multi-document synthesis. Detailed descriptions and statistics are provided in Appendix B. We use Exact Match (EM) as the evaluation metric for all datasets following previous work (Ma et al., 2025b). All reported results are the median of three independent runs to ensure robustness.

Baselines. We compare CoG against representative methods across four categories: (1) LLM-only: Direct few-shot prompting, Chain-of-Thought (Wei et al., 2022), and Self-Consistency (Wang et al., 2023), relying solely on the LLM's internal parametric knowledge; (2) Text-based RAG: VectorRAG, which retrieves top-3 text chunks via dense vector similarity over full-text entity documents; (3) KG-based RAG: ToG (Sun et al., 2024), a representative method performing iterative exploration on KGs; (4) Hybrid RAG: ToG-2 (Ma et al., 2025b), the current state-of-the-art framework combining KG traversal with text retrieval in a global setting.

Implementation Details. Following previous works (Ma et al., 2025b; Li et al., 2024b), we adopt the global knowledge setting, utilizing the full Wikidata and Wikipedia as structured and unstructured knowledge bases, respectively (details in Appendix C). This setting presents a realistic and challenging environment compared to simplified local settings. We employ Qwen3-32B (non-thinking mode) as the backbone LLM for CoG and baselines to ensure fair comparison. For retrieval, we utilize bge-m3 (Chen et al., 2024a) as the embedding model for text chunk retrieval, and Qwen3-Embedding-4B (Zhang et al., 2025c) for entity linking in the KG. We set the maximum exploration depth to 4 and width (sub-queries per turn) to 5. For entity linking, we retrieve the top-20 candidate entities. In text exploration, we extract up to 3 relevant chunks per section.

4.2 Main Results

Table 2 presents the main results across seven benchmarks. CoG demonstrates substantial improvements over existing methods on the majority of benchmarks. Notably, its advantage becomes more pronounced on challenging datasets. Specifically, on highly challenging text-based MusiQue, CoG achieves a remarkable 71.6% relative improvement over the best baseline. Similarly, on the high-quality KG-based KGQAGen, CoG achieves a 34.3% improvement. Even on simpler WebQSP (1-2 hops), CoG maintains comparable performance, suggesting that while excelling in complex scenarios, it remains robust on simple queries. These results empirically validate the effectiveness of our cognitive-inspired framework in unifying structured and unstructured knowledge exploration.

Analysis of Baselines VectorRAG shows limited improvement over LLM-only methods, but struggles with KG-based tasks, as its flat vector retrieval lacks the structural guidance to construct multi-hop reasoning chains. KG-based method ToG excels on KG-based datasets but degrades severely on text-based benchmarks, unable to leverage the rich semantic context to bridge knowledge gaps when KG links are sparse or missing. ToG-2, the previous

Baseline Type	Method	Multi-hop KG-based QA				Multi-hop text-based QA		
		KGQAGen	CWQ	QALD10-en	WebQSP	2WikiMQA	AdvHotpotQA	MusiQue
LLM-only	Direct	36.70	32.42	47.45	72.45	47.20	20.13	8.40
	CoT	38.28	33.59	45.35	70.84	47.40	23.70	9.60
	CoT-SC	40.50	33.40	46.55	71.40	49.60	24.03	10.60
Text-based RAG	Vector RAG	48.66	30.76	39.04	70.49	52.60	32.14	12.40
KG-based RAG	ToG	46.90	28.42	48.65	73.28	50.40	18.83	9.20
Hybrid RAG	ToG-2	55.61	41.41	53.45	**81.89**	73.00	35.71	16.20
Proposed	CoG	**74.70**	**42.87**	**55.26**	81.75	**85.20**	**51.95**	**27.80**

Table 2: Main results (Exact Match, %) of different methods on seven multi-hop QA datasets. All methods use Qwen3-32B (non-thinking mode) as the backbone LLM. The best results are highlighted in **bold**.

Variant	KGQA	CWQ	QALD	2Wiki	AdvHot	MusiQ	Avg.
CoG (Full)	**74.70**	**42.87**	**55.26**	**85.20**	**51.95**	**27.80**	**56.30**
Impact of Knowledge Sources							
w/o Text	75.90	39.06	51.65	76.20	27.27	20.20	48.38
w/o KG	56.44	40.92	48.65	79.80	49.68	20.80	49.38
Impact of Bidirectional Synergy							
w/o T-guide	74.14	42.58	52.85	76.20	39.61	22.80	51.36
w/o Synthesis	73.12	41.11	50.15	83.60	43.83	24.20	52.67
Impact of Cognitive Strategy							
w/o Reflect	72.57	42.38	52.85	85.00	48.05	25.80	54.44
w/o Plan	78.78	43.85	50.75	81.00	47.73	25.40	54.59
w/o Adap.EL	70.81	44.14	53.15	86.00	47.73	23.40	54.21

Table 3: Ablation results (EM, %) of CoG across three groups: (1) **Knowledge Sources**: removing either KG (*w/o KG*) or Text (*w/o Text*); (2) **Bidirectional Synergy**: disabling text-guided KG exploration (*w/o T-guide*), or removing summary synthesis module (*w/o Synthesis*); (3) **Cognitive Strategy**: excluding reflection-based adjustment (*w/o Reflect*), proactive planning (*w/o Plan*), or adaptive entity linking (*w/o Adap.EL*).

state-of-the-art, effectively combines KG and text to outperform single-source baselines. However, its reactive exploration strategy lacks long-term planning and bidirectional synergy, causing it to falter on complex tasks requiring deep reasoning.

4.3 Ablation Study

To dissect the contribution of each component in CoG, we conduct comprehensive ablation studies across three dimensions: knowledge sources, bidirectional synergy, and cognitive strategies. The results are presented in Table 3.

Knowledge Sources. Removing either source leads to significant performance drops: 6.92% for *w/o KG* and 7.92% for *w/o Text* on average. Notably, removing the text corpus (*w/o Text*) causes a more severe degradation than removing the KG (*w/o KG*), especially on text-heavy datasets like AdvHotpotQA and MusiQue. This confirms that while the KG provides structural guidance, the rich semantic information in text is indispensable for bridging knowledge gaps in global exploration.

Bidirectional Synergy. Merely accessing both sources is insufficient; their deep interaction is key. Disabling text-to-KG feedback (*w/o T-guide*) results in a notable decline of 4.94%. This validates our hypothesis that text should not be a passive context but an active driver for graph exploration, where entities extracted from text are crucial for discovering reasoning paths that are disconnected in KGs. Furthermore, removing the synthesis module (*w/o Synthesis*) degrades performance, indicating that explicitly synthesizing heterogeneous evidence from KG and text is crucial. It not only filters out noise but also constructs a unified, coherent summary to enable more informed decisions for subsequent reasoning.

Cognitive Strategy. Cognitive mechanisms enhance robustness in knowledge exploration. Removing reflection (*w/o Reflect*) consistently degrades performance, proving that the ability to diagnose failures and backtrack is vital for navigating noisy search spaces. Disabling proactive planning (*w/o Plan*) leads to suboptimal performance, confirming that goal-oriented decomposition is more effective than aimless, blind exploration. The decline in *w/o Adap.EL* highlights the importance of context-aware entity linking over fixed pre-annotation, ensuring accurate starting points for navigating ambiguous knowledge bases.

4.4 In-depth Analysis

Generalizability across Backbone LLM. Table 4 demonstrates the effectiveness of CoG across diverse backbone models, including open-source and proprietary models (see Appendix D for full

dataset breakdown and setup details). CoG consistently boosts performance for all models. Notably, open-source models exhibit substantial gains, indicating that CoG effectively bridges the capability gap between mid-sized models and top-tier proprietary LLMs. Even for powerful closed-source models like GPT-5.1, CoG yields a +32.0% improvement, confirming its value in supplementing parametric knowledge with up-to-date external evidence. Improvements are particularly pronounced on text-based tasks. This implies that LLMs, while capable of structural KGQA, depend on CoG for complex document-grounded reasoning.

Backbone	Method	KG Avg.	Text Avg.	Overall Avg.
Qwen3-32B	Direct	41.48	25.24	33.36
	CoG	**59.31**	**54.98**	**57.15** (+71.3%)
DeepSeek-V3.2	Direct	47.91	28.06	37.99
	CoG	**64.79**	**60.50**	**62.64** (+64.9%)
GPT-5.1	Direct	45.67	38.67	42.17
	CoG	**56.00**	**55.33**	**55.67** (+32.0%)
Gemini-2.5-Flash	Direct	45.00	44.33	44.67
	CoG	**49.33**	**53.00**	**51.17** (+14.6%)

Table 4: Performance comparison (EM, %) of CoG versus Direct Prompting across different backbone LLMs. Proprietary models are evaluated on a sampled subset (100 samples per dataset) due to cost constraints.

Impact of Exploration Depth. Figure 3 illustrates the performance trends across varying depths (see Appendix E for full results). On KG tasks, CoG consistently outperforms VectorRAG across all depths, confirming the superiority of structured cognitive exploration over flat retrieval. Similarly, text tasks show steady gains as depth increases, indicating that deeper exploration effectively gathers scattered semantic evidence. Notably, the optimal depth correlates strongly with dataset difficulty. Simpler datasets (WebQSP, 2WikiMQA) achieve optimal performance at shallower depths and then stabilize, whereas challenging ones (KGQAGen, AdvHotpotQA) continue to benefit from deeper exploration. It demonstrates CoG's adaptability to varying task complexity.

Exploration Efficiency. We analyze the exploration efficiency by measuring the average number of unique documents accessed per question (Table 5). Remarkably, CoG achieves superior performance while accessing 12.4× fewer documents on average (2.6 vs. 32.5). This disparity stems from their distinct paradigms. While ToG-2 blindly retrieves documents for every involved entity in beam search, CoG's cognitive mechanism enables goal-oriented retrieval, accessing documents only when necessary. This precise localization significantly reduces redundancy and computational overhead, achieving a superior balance between efficiency and effectiveness.

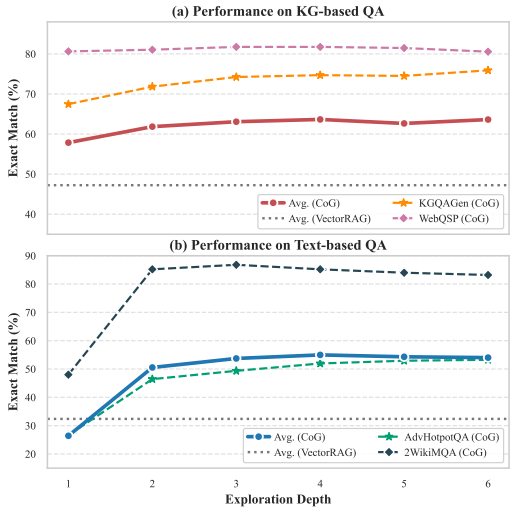

Figure 3: Impact of exploration depth on KG-based and Text-based tasks and representative datasets.

Method	KGQA	CWQ	QALD	WebQ	2Wiki	Hotpot	MusiQ	Avg.
ToG-2	37.1	40.4	31.2	22.9	17.7	36.1	42.2	32.5
CoG	**3.1**	**3.1**	**2.3**	**1.5**	**2.4**	**2.5**	**3.5**	**2.6**
Gain	×11.9	×13.2	×13.3	×15.5	×7.3	×14.8	×12.1	**×12.4**

Table 5: Comparison of exploration efficiency (Average number of unique documents accessed per question).

Qualitative Analysis. We provide a detailed case study in Appendix F, demonstrating how CoG's cognitive cycle and bidirectional synergy enable it to recover from dead ends and discover critical text evidence to solve complex multi-hop queries.

5 Conclusion

In this paper, we introduced CoG, a cognitive-inspired framework for autonomous knowledge exploration in global, hybrid knowledge bases. Under the continuous "plan-explore-reflect" human cognitive cycle, CoG shifts the paradigm from passive, reactive retrieval to proactive knowledge discovery, enabling adaptive strategy adjustment and robust recovery from exploration failures. Crucially, it establishes a deep bidirectional synergy between structured KGs and unstructured text, leveraging their complementary strengths to bridge knowledge gaps. Extensive experiments demonstrate that CoG significantly outperforms state-of-the-art baselines on multiple multi-hop QA tasks while achieving superior exploration efficiency.

6 Limitations

While CoG demonstrates superior performance and exploration efficiency in global knowledge exploration, several limitations remain.

Latency and Cost. The iterative nature of the cognitive loop (plan-explore-reflect) inevitably incurs higher inference latency and computational cost compared to passive exploration methods. Although CoG achieves high exploration efficiency in terms of document access, the multiple LLM calls required for planning and reflection may limit real-time applicability.

Dependency on Backbone LLM. The framework's effectiveness relies on the reasoning capabilities of the backbone LLM. While effective across mid-to-large scale open and proprietary models, the framework may experience performance degradation on smaller models with weaker instruction-following abilities.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ge Chang, Jinbo Su, Jiacheng Liu, Pengfei Yang, Yuhao Shang, Huiwen Zheng, Hongli Ma, Yan Liang, Yuanchun Li, and Yunxin Liu. 2025. GRAIL:learning to interact with large knowledge graphs for retrieval augmented reasoning. *Preprint*, arxiv:2508.05498.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024a. M3-embedding: Multilinguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arxiv:2402.03216.

Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. 2024b. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. In *NeurIPS*, volume 37, pages 37665–37691. Curran Associates, Inc.

Weijie Chen, Ting Bai, Jinbo Su, Jian Luan, Wei Liu, and Chuan Shi. 2025. KG-retriever: Efficient knowledge indexing for retrieval-augmented large language models. *Preprint*, arxiv:2412.05547.

Yuanning Cui, Zequn Sun, Wei Hu, and Zhangjie Fu. 2025. KGFR: A foundation retriever for generalized knowledge graph question answering. *Preprint*, arxiv:2511.04093.

Junnan Dong, Siyu An, Yifei Yu, Qian-Wen Zhang, Linhao Luo, Xiao Huang, Yunsheng Wu, Di Yin, and Xing Sun. 2025. Youtu-GraphRAG: Vertically unified agents for graph retrieval-augmented complex reasoning. *Preprint*, arxiv:2508.19855.

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

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. Lightrag: Simple and fast retrieval-augmented generation. *arXiv preprint arXiv:2410.05779*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in neural information processing systems*, volume 37, pages 59532–59569. Curran Associates, Inc.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *ICML*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Haoyu Huang, Yongfeng Huang, Junjie Yang, Zhenyu Pan, Yongqiang Chen, Kaili Ma, Hongzhi Chen, and James Cheng. 2025a. Retrieval-augmented generation with hierarchical knowledge. *Preprint*, arxiv:2503.10150.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and 1 others. 2025b. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Xin Zhao, and Ji-Rong Wen. 2023. StructGPT: A general framework for large language model to reason over structured data. In *EMNLP*, pages 9237–9251. Association for Computational Linguistics.

Jinhao Jiang, Kun Zhou, Xin Zhao, Yang Song, Chen Zhu, Hengshu Zhu, and Ji-Rong Wen. 2025. KG-agent: An efficient autonomous agent framework for complex reasoning over knowledge graph. In *Proceedings of the 63rd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 9505–9523. Association for Computational Linguistics.

Meng-Chieh Lee, Qi Zhu, Costas Mavromatis, Zhen Han, Soji Adeshina, Vassilis N. Ioannidis, Huzefa Rangwala, and Christos Faloutsos. 2025. HybGRAG: Hybrid retrieval-augmented generation on textual and relational knowledge bases. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 879–893. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *NeurIPS*, volume 33, pages 9459–9474. Curran Associates, Inc.

Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, Wenbo Su, and Bo Zheng. 2024a. GraphReader: Building graph-based agent to enhance long-context abilities of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12758–12786. Association for Computational Linguistics.

Xingxuan Li, Ruochen Zhao, Yew Ken Chia, Bosheng Ding, Shafiq Joty, Soujanya Poria, and Lidong Bing. 2024b. Chain-of-knowledge: Grounding large language models via dynamic knowledge adapting over heterogeneous sources. In *ICLR*.

Yangning Li, Weizhi Zhang, Yuyao Yang, Wei-Chieh Huang, Yaozu Wu, Junyu Luo, Yuanchen Bei, Henry Peng Zou, Xiao Luo, Yusheng Zhao, and 1 others. 2025. Towards agentic rag with deep reasoning: A survey of rag-reasoning systems in llms. *arXiv preprint arXiv:2507.09477*.

Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Dinh Phung, Chen Gong, and Shirui Pan. 2025a. GFM-RAG: Graph foundation model for retrieval augmented generation. *Preprint*, arxiv:2502.01113.

Qi Luo, Xiaonan Li, Tingshuo Fan, Xinchi Chen, and Xipeng Qiu. 2025b. Towards global retrieval augmented generation: A benchmark for corpus-level reasoning. *Preprint*, arxiv:2510.26205.

Jie Ma, Zhitao Gao, Qi Chai, Wangchun Sun, Pinghui Wang, Hongbin Pei, Jing Tao, Lingyun Song, Jun Liu, Chen Zhang, and Lizhen Cui. 2025a. Debate on graph: A flexible and reliable reasoning framework for large language models. In *AAAI*, volume 39, pages 24768–24776.

Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Li, Huaren Qu, Cehao Yang, Jiaxin Mao, and Jian Guo. 2025b. Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation. In *ICLR*.

Bhaskarjit Sarmah, Dhagash Mehta, Benika Hall, Rohan Rao, Sunil Patel, and Stefano Pasquali. 2024. HybridRAG: Integrating knowledge graphs and vector retrieval augmented generation for efficient information extraction. In *Proceedings of the 5th ACM International Conference on AI in Finance*, ICAIF '24, pages 608–616. Association for Computing Machinery.

Jinyeop Song, Song Wang, Julian Shun, and Yada Zhu. 2025a. Efficient and transferable agentic knowledge graph RAG via reinforcement learning. *Preprint*, arxiv:2509.26383.

Zirui Song, Bin Yan, Yuhan Liu, Miao Fang, Mingzhe Li, Rui Yan, and Xiuying Chen. 2025b. Injecting domain-specific knowledge into large language models: a comprehensive survey. *arXiv preprint arXiv:2502.10708*.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *ICLR*, Vienna, Austria.

Yushi Sun, Kai Sun, Yifan Ethan Xu, Xiao Yang, Xin Luna Dong, Nan Tang, and Lei Chen. 2025. KERAG: Knowledge-enhanced retrieval-augmented generation for advanced question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 6194–6216. Association for Computational Linguistics.

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

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. 10:539–554.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages

10014–10037. Association for Computational Linguistics.

Ricardo Usbeck, Xi Yan, Aleksandr Perevalov, Longquan Jiang, Julius Schulz, Angelie Kraft, Cedric Möller, Junbo Huang, Jan Reineke, Axel-Cyrille Ngonga Ngomo, and 1 others. 2023. Qald-10–the 10th challenge on question answering over linked data. *Semantic Web*, (Preprint):1–15.

Song Wang, Junhong Lin, Xiaojie Guo, Julian Shun, Jundong Li, and Yada Zhu. 2025a. Reasoning of large language models over knowledge graphs with super-relations. In *ICLR*, Singapore.

Xinming Wang, Jian Xu, Aslan H Feng, Yi Chen, Haiyang Guo, Fei Zhu, Yuanqi Shao, Minsi Ren, Hongzhu Yi, Sheng Lian, and 1 others. 2025b. The hitchhiker's guide to autonomous research: A survey of scientific agents. *TechRxiv.August 07, 2025.*

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Yu Wang, Nedim Lipka, Ryan A. Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. 2024. Knowledge graph prompting for multi-document question answering. In *AAAI*, volume 38, pages 19206–19214.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Xiaojun Wu, Cehao Yang, Xueyuan Lin, Chengjin Xu, Xuhui Jiang, Yuanliang Sun, Hui Xiong, Jia Li, and Jian Guo. 2025. Think-on-graph 3.0: Efficient and adaptive LLM reasoning on heterogeneous graphs via multi-agent dual-evolving context retrieval. *Preprint*, arxiv:2509.21710.

Jiasheng Xu, Mingda Li, Yongqiang Tang, Peijie Wang, and Wensheng Zhang. 2025. Towards open-world retrieval-augmented generation on knowledge graph: A multi-agent collaboration framework. *Preprint*, arxiv:2509.01238.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Cehao Yang, Xiaojun Wu, Xueyuan Lin, Chengjin Xu, Xuhui Jiang, Yuanliang Sun, Jia Li, Hui Xiong, and Jian Guo. 2025b. GraphSearch: An agentic deep searching workflow for graph retrieval-augmented generation. *Preprint*, arxiv:2509.22009.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380. Association for Computational Linguistics.

Xi Ye and Greg Durrett. 2022. The unreliability of explanations in few-shot prompting for textual reasoning. In *NeurIPS*.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. 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)*, pages 201–206.

Liangliang Zhang, Zhuorui Jiang, Hongliang Chi, Haoyang Chen, Mohammed Elkoumy, Fali Wang, Qiong Wu, Zhengyi Zhou, Shirui Pan, Suhang Wang, and Yao Ma. 2025a. Diagnosing and addressing pitfalls in KG-RAG datasets: Toward more reliable benchmarking. In *NeurIPS*.

Qinggang Zhang, Shengyuan Chen, Yuanchen Bei, Zheng Yuan, Huachi Zhou, Zijin Hong, Hao Chen, Yilin Xiao, Chuang Zhou, Yi Chang, and 1 others. 2025b. A survey of graph retrieval-augmented generation for customized large language models. *arXiv preprint arXiv:2501.13958*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025c. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arxiv:2506.05176.

Xiangrong Zhu, Yuexiang Xie, Yi Liu, Yaliang Li, and Wei Hu. 2025. Knowledge graph-guided retrieval augmented generation. In *NAACL*, pages 8912–8924.

A CoG Algorithm

We provide the pseudo-code of the CoG framework in Algorithm 1. It outlines the iterative workflow, illustrating how the Planning, Exploration, Synthesis, and Reflection modules coordinate to achieve autonomous knowledge exploration in large-scale knowledge bases.

B Evaluation Dataset

In this section, we provide detailed descriptions of the seven multi-hop question answering datasets used in our experiments, covering both KG-based and text-based multi-hop reasoning tasks.

Algorithm 1 The CoG Algorithm

Input: Question q, KG $\mathcal{G}$, Text Corpus $\mathcal{D}$, Max Turns M
Output: Answer a

Initialize: $\mathcal{N} \leftarrow \emptyset, \mathcal{P} \leftarrow \emptyset, \mathcal{H} \leftarrow \emptyset$
// **Phase 1: Initial Planning** (§3.2.1)
$P_0 = (A_0, Q_0, E_0) \leftarrow \text{INITPLAN}(q)$
for $t = 0$ to $M - 1$ **do**
  // **Phase 2: Exploration Dual-Source** (§3.2.2)
  $K_t \leftarrow \emptyset$
  for each $(q_i^{(t)}, e_i^{(t)}) \in (Q_t, E_t)$ **do**
    $C_i^{(t)} \leftarrow (q, A_t, q_i^{(t)}, e_i^{(t)})$ // Query context
    $F_i^{KG} \leftarrow \text{EXPLOREKG}(C_i^{(t)}, \mathcal{G})$
    // Entity Link → Relation Discovery → Fact Prune
    $F_i^{text} \leftarrow \text{EXPLORETEXT}(C_i^{(t)}, \mathcal{D})$
    // Page Selection → Skimming → Detailed Reading
    $K_t \leftarrow K_t \cup \{F_i^{KG} \cup F_i^{text}\}$ // Aggregate dual-source evidence
  end for
  // **Phase 3: Synthesis** (§3.2.3)
  $J_t, M_t \leftarrow \text{SYNTHESIZE}(q, P_t, K_t, \mathcal{N})$
  // **Phase 4: Reflection & Re-Planning** (§3.2.3)
  if $J_t =$ `SUFFICIENT` **then**
    break // Proceed to Answer Generation
  else if $J_t =$ `INSUFFICIENT_USEFUL` **then**
    // Pathway 1: Continue Exploration
    $P_{t+1}, \mathcal{N}, \mathcal{P} \leftarrow \text{CONTINUEPLAN}(q, P_t, M_t, \mathcal{N}, \mathcal{P})$
  else
    // Pathway 2: Strategy Adjustment
    $P_{t+1}, \mathcal{P} \leftarrow \text{ADJUSTPLAN}(q, P_t, M_t, \mathcal{H}, \mathcal{N}, \mathcal{P})$
  end if
  $\mathcal{H} \leftarrow \mathcal{H} \cup \{(P_t, M_t)\}$ // Update Interaction History
end for
// **Answer Generation** (§3.2.4)
$a \leftarrow \text{GENERATEANSWER}(q, \mathcal{N}, P_t, M_t)$
return a

KG-based QA Datasets. These benchmarks primarily require reasoning over structured facts in the knowledge graph.

WebQSP (Yih et al., 2016) is a widely used knowledge base QA benchmark derived from WebQuestions, containing questions paired with SPARQL queries over Freebase (mapped to Wikidata in our setting). It primarily focuses on questions that require 1-2 hop reasoning over structured KG facts to retrieve target entities.

CWQ (Talmor and Berant, 2018) extends the WebQSP by generating questions with higher complexity. It necessitates multi-hop reasoning (up to 4 hops) and challenges models to handle intricate logical constraints, such as composition, conjunctions, and superlatives over the graph structure.

QALD10-en (Usbeck et al., 2023) is the English subset of the 10th Question Answering over Linked Data challenge, specifically adapted for Wikidata. It is characterized by highly complex natural language questions that map to intricate SPARQL queries. The dataset specifically tests a system's ability to handle advanced structural operations, such as aggregations, filters, and multi-hop relationships.

KGQAGen (Zhang et al., 2025a) is a highly reliable, Wikidata-grounded benchmark generated via an LLM-in-the-loop framework with symbolic verification, featuring a wide range of topics and structural complexities. It systematically addresses the inaccurate annotations and ambiguity in existing evaluation datasets, making it a superior standard for evaluating multi-hop reasoning capabilities in KG-based RAG systems.

Text-based QA Datasets. These benchmarks demand semantic understanding and multi-document synthesis.

2WikiMQA (Ho et al., 2020) is a large-scale dataset synthesized from Wikidata and Wikipedia, specifically designed to test multi-hop reasoning across multiple documents. It focuses on questions that require comparison, aggregation, and bridging reasoning types. The questions typically require reasoning chains of 2 to 4 hops, challenging models to perform cross-document integration and logical inference over unstructured text.

MusiQue (Trivedi et al., 2022) is a highly challenging multi-hop QA dataset constructed by compositing single-hop questions in a bottom-up manner. It is explicitly designed to minimize reasoning shortcuts and enforce connected reasoning, ensuring that answering the final question strictly requires resolving intermediate sub-questions. The dataset features 2-4 hop questions that are significantly harder to solve via disconnected reasoning compared to HotpotQA or 2WikiMQA.

AdvHotpotQA (Ye and Durrett, 2022) is an adversarial version of the HotpotQA (Yang et al., 2018) dataset. It introduces distractor paragraphs that share high lexical overlap with the question but do not contain the answer, rigorously testing the system's robustness against retrieval noise and its ability to identify supporting evidence accurately.

Table 6 summarizes the statistics of the evaluation datasets. For WebQSP, AdvHotpotQA, and QALD10-en, we use the same test splits as in ToG-2 (Ma et al., 2025b) for fair comparison. For CWQ, we adopt the first 1,024 questions from the test set used in ToG (Sun et al., 2024). For 2WikiMQA and MusiQue, we employ the evaluation sets provided by IRCoT (Trivedi et al., 2023). For KGQAGen, we use the full test set from the KGQAGen-10k benchmark.

Category	Dataset	# Questions
KG-based QA	WebQSP	1,430
	CWQ	1,024
	QALD10-en	333
	KGQAGen-10k	1,079
Text-based QA	2WikiMQA	500
	MusiQue	500
	AdvHotpotQA	308

Table 6: Statistics of the evaluation datasets.

C Knowledge Bases

We construct our global knowledge environment using massive-scale data from Wikidata and Wikipedia to simulate real-world open-domain QA scenarios. The detailed statistics and configurations are as follows:

Structured Knowledge Graph (Wikidata). We utilize the full Wikidata dump[1] (May 2024) as our structured knowledge source. It contains approximately 89.3 million entities and 1.38 billion edges. Specifically, the edges are categorized into two types:

- **Relational Edges:** 823,749,591 triples representing relationships between two entities (e.g., *(Steve Jobs, founded, Apple)*).
- **Attribute Edges:** 562,016,470 triples representing literal values or properties of entities (e.g., *(Steve Jobs, date of birth, 1955-02-24)*).

This massive scale introduces significant challenges, including entity ambiguity and high fan-out nodes, necessitating robust context-aware entity linking and relation pruning mechanisms.

Unstructured Text Corpus (Wikipedia). We employ the real-time English Wikipedia[2] as our unstructured text corpus. It comprises approximately 7.1 million entity-centric articles, totaling over 5 billion words. The rich semantic descriptions in Wikipedia serve as a critical complement to the sparse structured facts in Wikidata. Distinct from traditional RAG that chunks text flatly, we process each page to retain its hierarchical structure, including the summary paragraph, table of contents, and sectioned full text. This structure supports CoG's hierarchical reading strategy (global skimming followed by detailed reading), enabling efficient navigation through long documents.

[1] https://dumps.wikimedia.org/wikidatawiki/entities/

[2] https://en.wikipedia.org/w/api.php

D Detailed Results on Backbone LLMs

This section supplements the backbone LLM generalizability analysis in Section 4.4. Table 7 presents the detailed performance breakdown across different backbone LLMs, corresponding to the summarized results in Table 4.

Evaluation Setup. To balance computational cost with evaluation rigor, we adopted the following dataset selection and sampling strategies:

Dataset Selection. We excluded the WebQSP dataset from this analysis, as modern LLMs already achieve near-saturated performance on its simple 1-2 hop queries in the Direct Prompting setting, making it less discriminative for measuring improvement.

Open-source Models. For large-scale KG-based datasets (KGQAGen and CWQ), we sampled 300 instances each to align with the scale of QALD10-en. For the remaining datasets (QALD10-en, 2WikiMQA, AdvHotpotQA, and MusiQue), we evaluated on the full test sets.

Proprietary Models. We evaluated on a sampled subset of 100 examples for each dataset.

E Detailed Results on Exploration Depth

Table 8 provides the full results corresponding to the exploration depth analysis discussed in Section 4.4 and visualized in Figure 3. The table lists the Exact Match (EM) scores for CoG across exploration depths from 1 to 6 on all seven benchmark datasets, alongside the average performance on KG-based and Text-based tasks.

F Case Studies

Table 9 and Table 10 compare the reasoning trajectories of ToG-2 and CoG on a complex multi-hop question from AdvHotpotQA.

ToG-2 Analysis. It fails due to its passive, graph-driven exploration strategy. The system starts by retrieving general information about "Stanford Cardinal" but fails to locate the specific player due to the absence of statistical attributes in the KG. Lacking a reflection mechanism, ToG-2 cannot diagnose this failure. Instead, it blindly continues its beam

Backbone	Method	KG-based QA			Text-based QA			Average		
		KGQAGen	CWQ	QALD	2Wiki	Hotpot	MusiQ	**KG**	**Text**	**Total**
Open-Source Models										
Qwen3-32B	Direct	40.00	37.00	47.45	47.20	20.13	8.40	41.48	25.24	33.36
	CoG	**76.33**	**46.33**	**55.26**	**85.20**	**51.95**	**27.80**	**59.31**	**54.98**	**57.15**
DeepSeek-V3.2	Direct	53.67	45.33	44.74	39.20	31.17	13.80	47.91	28.06	37.99
	CoG	**85.33**	**51.67**	**57.36**	**88.40**	**59.09**	**34.00**	**64.79**	**60.50**	**62.64**
Proprietary Models (Sampled Set, N=100)										
GPT-5.1	Direct	54.00	47.00	36.00	59.00	40.00	17.00	45.67	38.67	42.17
	CoG	**81.00**	**49.00**	**38.00**	**82.00**	**55.00**	**29.00**	**56.00**	**55.33**	**55.67**
Gemini-2.5-Flash	Direct	62.00	**46.00**	27.00	65.00	46.00	22.00	45.00	44.33	44.67
	CoG	**73.00**	44.00	**31.00**	**78.00**	**58.00**	**23.00**	**49.33**	**53.00**	**51.17**

Table 7: Detailed performance comparison (EM, %) of different backbone LLMs with and without the CoG framework across six datasets (WebQSP excluded).

Method	KG-based QA					Text-based QA				Avg.
	KGQAGen	CWQ	QALD10	WebQSP	**Avg.**	2Wiki	AdvHotpot	MusiQue	**Avg.**	
VectorRAG	48.66	30.76	39.04	70.49	47.24	52.60	32.14	12.40	32.38	40.87
CoG (Depth=1)	67.47	37.99	45.35	80.63	57.86	48.00	26.62	4.60	26.41	44.38
CoG (Depth=2)	71.83	42.58	51.95	81.05	61.85	85.20	46.43	20.00	50.54	57.01
CoG (Depth=3)	74.24	42.58	53.75	**81.75**	63.08	**86.80**	49.35	25.00	53.72	59.07
CoG (Depth=4)	74.70	42.87	**55.26**	**81.75**	**63.65**	85.20	51.95	**27.80**	**54.98**	**59.93**
CoG (Depth=5)	74.51	42.97	51.65	81.47	62.65	84.00	52.92	26.00	54.31	59.07
CoG (Depth=6)	**75.90**	**43.65**	54.35	80.56	63.62	83.20	**53.25**	25.60	54.02	59.50

Table 8: Full results of exploration depth analysis. We report Exact Match (EM) scores for CoG with maximum exploration depths ranging from 1 to 6 on all seven datasets, alongside the VectorRAG baseline. The best result for each dataset is highlighted in **bold**.

search, pivoting to high-ranking but irrelevant entities like "quarterback" in Turn 3, ultimately leading to an irreversible dead end.

CoG Analysis. In contrast, CoG demonstrates robust navigation through its cognitive cycle and bidirectional synergy. CoG initially encounters the same dead end as ToG-2 in Turn 1. However, Reflection diagnoses the strategy as too narrow. Proactive Planning then pivots to broader queries like "football receiving leaders" instead of the player directly (Turn 2). Crucially, Text Entity Utilization bridges KG gaps by identifying the entity "James Lofton", which aligns with the statistical constraints in the question, from Wikipedia related sections. This bidirectional synergy enables CoG to dynamically identify the target player and verify the specific statistics in Turn 3, successfully deriving the answer "David".

G Prompts

This section provides the main prompt used in CoG to guide LLM reasoning at each stage of the cognitive cycle.

Table 11 presents the prompt used for the **initial planning phase** (Section 3.2.1). The prompt includes four examples to guide the LLM in handling diverse multi-hop reasoning scenarios.

Tables 12–15 presents the prompts for the three-stage **KG Exploration** (Section 3.2.2): Entity Linking, Relation Discovery, and Fact Pruning, respectively. These prompts demonstrate how CoG leverages in-context learning to achieve precise, schema-aware navigation over large-scale KGs.

Table 16–19 details the prompts used in the **Text Exploration module** (Section 3.2.2), covering Adaptive Page Selection (Tables 16 and 17) and Hierarchical Reading (Tables 18 and 19).

Table 20 presents the prompt for the **Synthesis module**. Tables 21 and 22 provide examples of structured evidence formats for KG and text, respectively, that populate the `${evidence_blocks}`

Question: *What is the middle name of the former Stanford Cardinals player who caught 57 passes for 1010 yards and 14 TDs?*
Pre-annotated Entities: Stanford Cardinal (Q2938060), Touchdown (Q650807)
Ground Truth: David

Turn	ToG-2 Reasoning Process
turn 1	**Starting Entities:** Stanford Cardinal (Q2938060), Touchdown (Q650807) **Relations Explored:** member of sports team, sport, educated at, facet of, named after (6 relations) **Retrieved Entities:** 57 entities (49 from "member of sports team", 4 from "sport", 1 from "educated at", 2 from "Touchdown" relations) Notable: 49 Stanford athletes including Greg Camarillo, John Elway, Mark Marquess, etc. **Document Retrieval:** Retrieved Wikipedia pages for 55 entities (2 pages failed: sports terminology, Margaret Hayden) **Issue:** No specific statistics matching 57/1010/14 found in documents **Reasoning:** Information insufficient - cannot identify the specific player **Entity Ranking (for next turn):** [0.28] Greg Camarillo | [0.12] John Elway | [0.05] Mark Marquess
turn 2	**Starting Entities:** Greg Camarillo (Q1248387), John Elway (Q432931), Mark Marquess (Q6768732) **Relations Explored:** given name, educated at, member of sports team, position played, occupation (9 relations) **Retrieved Entities:** 11 entities (Greg, Menlo-Atherton HS, Stanford Univ., 5 NFL teams, John, quarterback, Mark, baseball player, Stanford Cardinal) **Document Retrieval:** Retrieved Wikipedia pages for all 11 entities **Issue:** Found Greg Camarillo's given name but no middle name; no confirmation of statistics **Reasoning:** Cannot confirm Greg Camarillo is the target player or find his middle name **Entity Ranking (for next turn):** [0.45] Stanford Cardinal | [0.00] quarterback
turn 3	**Starting Entities:** Stanford Cardinal (Q2938060), quarterback (Q622747) **Relations Explored:** member of sports team, position played, sport (6 relations) **Retrieved Entities:** 106 entities (50 from "member of sports team", 4 from "sport", 1 from "educated at", 50 from "position played", 1 from "part of") Notable: 50 quarterbacks (Logan Thomas, John Brantley, Charlie Whitehurst, etc.), 50 Stanford athletes **Document Retrieval:** Retrieved Wikipedia pages for 103 entities (3 pages failed: Cameron Dantley, Dai Minagawa, Margaret Hayden) **Critical Failure:** Explored wrong position (quarterback vs. wide receiver) **No Reflection:** System continued beam search without recognizing strategic misalignment **Entity Ranking:** [0.31] Logan Thomas | [0.12] John Brantley | [0.02] Charlie Whitehurst **Final Reasoning:** Information insufficient to identify the player or middle name **Outcome: FAILURE** - Generated answer without evidence

Table 9: Reasoning trace of ToG-2. The method employs a rigid beam search where the next exploration steps are determined by ranking score based on entity document relevance. Without reflection, the pruning mechanism blindly discards potentially correct paths and misguides the search towards irrelevant high-ranking entities, leading to an irreversible failure. Red highlights indicate critical failure analysis for the system.

placeholder in the Synthesis prompt.

Tables 23 and 24 detail the prompts for the **Reflection module** (Section 3.2.3), enabling the system to either deepen exploration based on new findings or pivot its strategy by diagnosing failures from the interaction history.

Question: *What is the middle name of the former Stanford Cardinals player who caught 57 passes for 1010 yards and 14 TDs?*
Ground Truth: David

Turn	CoG Reasoning Process
Turn 1	**Planning Phase:** Analysis: "To determine the middle name, I first need to identify the specific player who caught 57 passes for 1010 yards and 14 touchdowns." Query: "Stanford Cardinals player with 57 catches, 1010 yards, 14 TDs" Entity: "Stanford Cardinals player" **Exploration Phase:** *KG:* Entity linking failed - 9 candidates all team-level entities (e.g., Stanford Cardinal football, Stanford Cardinal men's soccer), no individual player entities *Wikipedia:* Searched "Stanford Cardinal football" page, extracted team context (Jim Plunkett, Andrew Luck) but no player matching the statistics **Synthesis:** No player-specific information found **Reflection:** Judgment = INSUFFICIENT_USELESS "The initial approach was too narrow... The core strategic error was not breaking the problem into smaller, more general steps."
Turn 2	**Recovery Planning:** Strategy Shift Reflection: "Instead of starting with the player's name, we should first identify the season or year when these stats might have been achieved, or search for a list of Stanford receivers with notable seasons." Updated Analysis: "Shift focus to general sources of Stanford football statistics or notable players." Next Queries: ["Stanford football all-time receiving leaders", "Stanford football players with 14 TDs in a season"] **Exploration (Query: "Stanford football all-time receiving leaders"):** *KG:* Linked to "Stanford Cardinal football statistical leaders" (Q19878473) Relation Discovery: Found 1 relation - "sport" (P641) → American football Fact Pruning: Kept 0 facts (too general, not useful for identifying specific players) *Wikipedia:* Found "Stanford Cardinal football statistical leaders" page with "Receiving" section "Receiving" section containing 9 statistical tables, extracted the following information from table: "Single season 6" (receiving yards): James Lofton - 1010 yards (1977) "Single season 7" (receiving TDs): James Lofton - 14 TDs (1977); J.J. Arcega-Whiteside - 14 TDs (2018) **Key Discovery:** Identified "James Lofton" as candidate with 2/3 statistics matched (1010 yards + 14 TDs in 1977) **Synthesis:** Strong candidate found but 57 receptions unconfirmed **Reflection:** Judgment = INSUFFICIENT_USEFUL - Continue with James Lofton
Turn 3	**Planning Phase:** Query: "James Lofton 1977 season stats" Entities: "James Lofton" **Exploration Phase:** *KG:* Linked to James Lofton (Q1680714) **Identity Check:** `educated at` → Stanford; `member of team` → Stanford Cardinal. **Attribute Retrieval:** Found alias "**James David Lofton**" (Directly reveals middle name) *Wikipedia:* Found "James Lofton" page **From "College career" section:** Confirmed "57 receptions for 1, 010 yards and 14 touchdowns during the 1977 season" **Synthesis:** KG confirmed identity & name; Text verified the missing statistical constraint. Middle name derivation, complete. **Reflection:** Judgment = SUFFICIENT **Final Answer:** "The middle name of the former Stanford Cardinals player is David." **Outcome: SUCCESS** in 3 turns

Table 10: Reasoning trace of CoG. Reflection and Proactive Planning diagnose the initial failure and recover from dead ends. Adaptive Entity Linking identifies implicitly useful entities ("1977 Stanford Cardinals football team") not directly mentioned in the question. Finally, Text Entity Utilization effectively extracts the target entity ("James Lofton") from text to guide the final answer derivation. Purple highlights show successful reflection and strategic adjustment; blue highlights show key information extraction from dual sources (KG structure + Wikipedia text).

ROLE

You are a highly strategic retrieval planner. Your function is to analyze a complex question and identify the immediate, parallelizable search actions required to proceed.

TASK

Given a user's question, do NOT provide the final answer. Instead, determine the next logical search queries. A single question might require multiple parallel queries to gather the necessary initial information. Your plan must include your reasoning, a list of search queries, and a corresponding list of entities.

CONSTRAINTS

- The number of queries in the 'Query' list must not exceed 5.
- **[CRITICAL]** The 'Query' and 'Entities' lists must have the exact same number of items. For each query, the corresponding item in the 'Entities' list must be the **single, most specific, and central named entity** being investigated in that query.
- Only output the query/queries for the immediate next step. Do not plan, describe, or allude to any subsequent steps.

OUTPUT FORMAT

Your output must follow this exact structure, with no additional commentary:

`Analysis:` Your thought process explaining why these next steps are necessary and what information is being sought.
`Query:` [A Python-style list of concise and effective search query strings.]
`Entities:` [A Python-style list of the core entities in the Query, corresponding one-to-one with the Query list.]

EXAMPLES

Example 1:
Question: What is the nationality of one of the men for whom the Schumann–Runge bands are named?
Output:
`Analysis:` The question requires the nationality of a person. Before I can find their nationality, I must first identify the individuals the "Schumann–Runge bands" are named after.
`Query:` ["who are the Schumann-Runge bands named after"]
`Entities:` ["Schumann–Runge bands"]

... (Other examples omitted for brevity) ...

Example 4:
Question: Where did the leader of the largest European country after the collapse of the country that denied anything more than an advisory role in the Korean war die?
Output:
`Analysis:` The question is complex and requires multiple pieces of information. First, I need to identify which country denied anything more than an advisory role in the Korean War. [...] I will begin by identifying the country that denied a combat role in the Korean War.
`Query:` ["country that denied anything more than an advisory role in Korean War"]
`Entities:` ["Korean War"]

YOUR TASK

Question: `${question}`
Output:

Table 11: Prompt template for the Initial Planning module. The LLM is instructed to analyze the question, generate parallelizable sub-queries, and identify corresponding anchor entities. This structured output initializes the exploration cycle.

ROLE

You are an expert in Knowledge Graph entity linking. Your task is to disambiguate an entity mention from a user's question by matching it to the correct entity in a knowledge graph, using the surrounding context.

CONTEXT

- **Original Question:** `${question}`
- **Overall Plan (Analysis):** `${analysis}`
- **Current Sub-Query:** `${query}`
- **Entity Mention to Link:** `${entity}`

CANDIDATE ENTITIES

Here are the top candidate entities in the Knowledge Graph, sorted by a preliminary relevance score. Each candidate includes its Wikidata QID, label, description, aliases, popularity, some of its neighbors, and relevance score.

`${formatted_candidates}`

YOUR TASK

Your task is to critically evaluate the candidate entities based on the provided CONTEXT. Your goal is to either identify the single correct entity OR determine that no suitable match exists.

- **Analyze the Context:** Carefully review the 'Original Question' and 'Overall Plan'. What are the key details about the entity `${entity}` (e.g., their time period, relationships, role)?
- **Evaluate Each Candidate:** For each candidate, compare its `Description`, `Aliases`, and `Neighborhood` against the context. A correct match should be consistent with the context.
- **Make a Decision:**
  - **If, and only if,** you find one candidate that is a confident and accurate match for `${entity}` based on all available information, your output should be its QID on a single line.
  - **If none of the candidates are a confident match,** or if their key details contradict the context, you MUST output the single word: `NO_MATCH`.

OUTPUT FORMAT

- If a confident match is found, your entire output MUST be only the QID of that entity (e.g., Q12345).
- If no confident match is found, your entire output MUST be the single word: `NO_MATCH`.
- Your output must not contain any other text, explanation, or reasoning.

YOUR RESPONSE:

Table 12: Prompt template for the Entity Linking module. The LLM is tasked with precise disambiguation by comparing candidate profiles against the question context. The placeholder `${formatted_candidates}` is populated with structured candidate data, as illustrated in Table 13.

Example of Formatted Candidates

```
—
[Entity]
- QID: Q716680
- Label: Steve McQueen
- Description: American actor (1930-1980)
- Aliases: Terrence Stephen McQueen, King of Cool
- Popularity (Degree): In: 154, Out: 45, Attr: 22
[Neighborhood]
- Outgoing Relations (as Head):
- spouse: Neile Adams, Ali MacGraw, Barbara Minty
- child: Chad McQueen, Terry Leslie McQueen
[Scores]
- Final: 0.952 (Name similarity=1.000, Popularity=0.850, Description=0.920, Neighbor=0.880)
—
... (Other candidates omitted) ...
```

Table 13: An example of the formatted candidate data populated into the `${formatted_candidates}` slot in Table 12. This structured format provides the LLM with comprehensive entity details (profile, neighborhood, and retrieval scores) for disambiguation.

ROLE

You are an expert Knowledge Graph analyzer. Your task is to select the most relevant relations (properties) of a given entity that will help answer a specific question.

CONTEXT

- **Original Question:** `${question}`
- **Overall Plan (Analysis):** `${analysis}`
- **Current Sub-Query:** `${query}`
- **Entity in Focus:** `${entity_label}` (`${entity_qid}`): `${entity_description}`

AVAILABLE RELATIONS

The entity `${entity_label}` is connected to `${total_relations_count}` relations (Outgoing: `${outgoing_count}`, Incoming: `${incoming_count}`) in the Knowledge Graph. Below is a list of all relations connected to the entity. Each relation includes its label, PID, and a frequency indicating how many other entities in the graph are linked via this relation.

Outgoing Relations:
`${outgoing_relations}`

Incoming Relations:
`${incoming_relations}`

YOUR TASK

Based on the **CONTEXT**, select relations that will help find the answer using a two-pronged approach:
1. **Direct Relations**: Select relations that seem to directly contain the answer.
2. **Exploratory Relations**: If no direct relations exist, select relations that could lead to intermediate entities, which might then contain the answer.

OUTPUT FORMAT

Your output must follow this exact structure, with no additional commentary:

`Reasoning:` Your brief thought process explaining why you chose these specific relations based on the question and the entity.
`Selected PIDs:` [A Python-style list of strings, where each string is the PID of a selected relation (e.g., ["P31", "P17"]). If no relations seem relevant, output []]

YOUR RESPONSE:

Table 14: Prompt template for the Relation Discovery module. The LLM selects a subset of relations relevant to the sub-query to prune the search space at the schema level.

ROLE

You are an expert assistant specializing in knowledge graph analysis for question answering. Your mission is to intelligently prune a list of facts, guiding a multi-step exploration process.

CONTEXT

- **Original Question:** `${question}`
- **Overall Plan (Analysis):** `${analysis}`
- **Current Sub-Query:** `${query}`
- **Entity in Focus:** `${entity_label}`

AVAILABLE FACTS

Below is a structured list of all facts retrieved from the Knowledge Graph for the entity `${entity_label}`. The facts are categorized by relation type (Outgoing or Incoming).

`${formatted_facts}`

YOUR TASK

Your goal is to decide which facts to keep. Evaluate the facts using a two-pronged approach:

1. **Directly Relevant Facts**: Keep any facts that directly help answer the **Current Sub-Query**.
2. **Promising Intermediate Entities**: Keep facts that represent entities which are promising stepping stones. A fact is "promising" if exploring it further is highly likely to lead to the answer or provide crucial context.

Discard facts that are clearly irrelevant, noisy, or are just general information.

OUTPUT FORMAT

Your output **MUST** be a single, valid JSON object with the following structure:

```
{
  "reasoning": "Structured thought process...",
  "pruned_facts": {
    "outgoing": { "P17": ["kept fact labels"], ... },
    "incoming": { "P802": ["kept fact labels"], ... }
  }
}
```

YOUR RESPONSE:

Table 15: Prompt template for the Fact Pruning module. The LLM performs fine-grained filtering of facts to retain only critical evidence or promising exploration leads.

ROLE

You are a research agent resolving an ambiguous search query on Wikipedia.

TASK

Your search for "`${entity}`" led to a disambiguation page with multiple possible meanings. You must carefully analyze the full context provided below to choose the single most relevant link from the options.

CONTEXT

- **Original Question:** `${question}`
- **Overall Plan (Analysis):** `${analysis}`
- **Current Sub-Task (Query):** `${query}`
- **Ambiguous Search Query:** "`${entity}`"

OPTIONS

`${options_list}`

YOUR DECISION

Analyze the options based on the context to select the best fit. Your output must be a single line containing only the exact title of the chosen page from the list. If none of the options seem relevant for answering the original question, output the string "NO_MATCH".

YOUR RESPONSE:

Table 16: Prompt template for the **Disambiguation** step in Adaptive Page Selection. The LLM resolves ambiguous entities by selecting the correct page from disambiguation options based on context.

ROLE

You are a research agent trying to find the right information on Wikipedia after a search query failed.

TASK

Your search for "`${entity}`" did not find a matching Wikipedia page. Use the contextual information and the suggestions provided to decide on the best query for your next attempt.

CONTEXT

- **Original Question:** `${question}`
- **Overall Plan (Analysis):** `${analysis}`
- **Current Sub-Task (Query):** `${query}`
- **Failed Search Query:** "`${entity}`"

SUGGESTIONS

Below are suggestions from the Wikipedia search API to help you refine your query.

"Did you mean?": This is the API's top recommendation, often correcting a typo or suggesting a more standard page title.
- Suggestion: `${suggestion}`

Similar Pages Found: These are pages with titles that are textually similar to your failed query. One of them might be the correct page under a slightly different name.
- Similar Pages: `${search_results_list}`

YOUR DECISION

Based on the context and the suggestions, choose the best query for your next search attempt. This could be the "Did you mean?" suggestion, one of the similar pages, or a completely new query you formulate based on the feedback from this failed search. Your output must be a single line containing only the new search query. If none of the suggestions seem useful for answering the original question, output the string "NO_MATCH".

YOUR RESPONSE:

Table 17: Prompt template for the **Search Refinement** step in Adaptive Page Selection. The LLM uses API feedback (corrections and similar pages) to recover from search failures.

ROLE

You are a research agent tasked with answering a complex question by navigating Wikipedia.

TASK

You have been provided with the summary, an infobox table, and the section list of a Wikipedia page relevant to your current query. Your goal is to extract all useful information from the summary and infobox, identify promising sections for deeper investigation, or determine that the page is irrelevant.

CONTEXT

- **Original Question:** `${question}`
- **Overall Plan (Analysis):** `${analysis}`
- **Current Search Query:** "`${query}`"
- **Retrieved Wikipedia Page Title:** "`${page_title}`"

AVAILABLE INFORMATION

Page Summary:
`${page_summary}`

Summary Infobox:
`${infobox_table}`

Page Sections (Top-Level):
`${page_sections}`

YOUR DECISION

Based on the information above, what is your next best action? Choose one of the following two options.

1. **If the page seems relevant:**
   - Extract **Direct Information** and **Promising Clues** from the summary/infobox.
   - Identify sections for deeper investigation.
2. **If the page is clearly irrelevant:** Explain why and suggest a better search query.

YOUR RESPONSE:

Table 18: Prompt template for the Global Skimming module. The LLM assesses page relevance, extracts high-level evidence from the summary and infobox, and selects promising sections for detailed reading.

ROLE

You are a meticulous research agent. Your task is to analyze a section of a Wikipedia page that contains both text and tables to extract relevant information for a given query.

TASK

You have been given several retrieved text chunks and a preview of all tables found within a specific Wikipedia section. Your goal is to:
1. Extract **Direct Information** and **Promising Clues** from BOTH the **Retrieved Text Chunks** and the **Table Previews**.
2. Analyze table previews. If a preview is insufficient but promising, select the table for a full read.
3. Provide a single, unified rationale that explains both your information extraction findings and your table selections.

CONTEXT

- **Original Question:** `${question}`
- **Current Search Query:** `"${query}"`
- **Section Being Investigated:** `"${section_title}"`
- **Section Exploration Rationale (during summary reading):** `${exploration_rationale}`

AVAILABLE INFORMATION

1. Retrieved Text Chunks: (Top `${k}` relevant chunks)
`${context_chunks}`

2. Table Previews: (Previews of tables: `${table_names_in_section}`)
`${tables_preview}`

OUTPUT FORMAT

Rationale: A consolidated explanation covering both text and tables.
Extracted Info: Key info from text chunks and table previews. State "None" if nothing relevant is found.
Selected Tables: List of table names requiring a full read (e.g., ["Awards"]). Empty list [] if none.

YOUR RESPONSE:

Table 19: Prompt template for the **Detailed Reading** phase. The LLM performs joint analysis of retrieved text chunks and table previews to synthesize fine-grained evidence.

ROLE

You are a master AI strategist leading a multi-hop question-answering mission. Your task is to synthesize retrieved information from various sources, evaluate progress against the overall plan, and decide the most logical next step.

CONTEXT

- **Original Question:** `${question}`
- **Overall Plan (Analysis):** `${analysis}`
- **Notebook (Summary of Known Facts):**
`${notebook}`
- **Current Sub-Queries:** `${queries}`
- **Core Entities in Sub-Queries:** `${entities}`

EVIDENCE

This section contains the information retrieved from different sources for all sub-queries executed in this turn.
`${evidence_blocks}`

YOUR TASK

Carefully review all evidence, and in conjunction with the **Original Question** and your **Overall Plan**, complete the following three steps:

1. **Verbatim Information Extraction:** Meticulously extract all useful information. Your goal is to faithfully transfer potentially relevant information to your 'Extracted Content'. Include:
    - **Direct Facts:** Core facts that directly contribute to answering the original question.
    - **Promising Leads:** New entities or critical factual clues essential for guiding the next step.
    - **[CRITICAL]** Do NOT summarize or rephrase. This is for extraction only.
2. **Think Step-by-Step:** Document your thought process.
    - Explain how the direct facts help answer the question.
    - Discuss the potential value of promising leads and how they might be explored.
    - Identify what key information is still missing.
    - If evidence is useless, explain *why* the current sub-queries failed.
3. **Make a Judgment:** Choose one of the following options:
    - `SUFFICIENT`: Information is adequate to generate a final, complete answer.
    - `INSUFFICIENT_USEFUL`: Valuable clues found, but more information is needed. Continue investigation based on the current findings.
    - `INSUFFICIENT_USELESS`: Information is irrelevant or has led to a dead end. A new strategy is needed.

OUTPUT FORMAT

`Thought Process:` Your step-by-step thinking process...
`Extracted Content:` A structured collection of key facts and promising new leads...
`Judgment:` SUFFICIENT, INSUFFICIENT_USEFUL, or INSUFFICIENT_USELESS.

YOUR RESPONSE:

Table 20: Prompt template for the **Synthesis** module. The LLM aggregates dual-source evidence, extracts key insights, and judges the current progress to determine the next action (answer, continue, or recover). Examples of the content for `${evidence_blocks}` are shown in Tables 21 and 22.

Example of Formatted Candidates

```
KG Exploration for Mention: "John Cage"

[Entity Linking]
- Linked Entity: John Cage (Q180727)
- Description: American composer and music theorist
- Candidates (top-4 preview):
- John Cage (Q180727): American composer and music theorist
- John Cage (Q5347597): Fictional character from Ally McBeal
- Linking Rationale: The query context discusses "4'33" and "prepared piano", which are signature
works of the composer John Cage.

[Retrieved Facts for "John Cage"]
[Attributes]
- date of birth: ['1912-09-05']
- occupation: ['composer', 'philosopher', 'artist']
[Outgoing]
- notable work: ["4'33"", 'Music of Changes', 'Imaginary Landscape No. 4']
- student of: ['Arnold Schoenberg', 'Henry Cowell']
[Incoming]
- influenced by: ['Erik Satie', 'Marcel Duchamp', 'D. T. Suzuki']

[Filtering Summary]
- Relations explored: 5 of 21 total
- Facts kept: 12 of 43 total

[Reasoning]
- Relation Selection: Selected relations regarding works, influences, and personal life relevant
to his artistic development.
- Fact Pruning: Kept only major works and key figures like Schoenberg and Cunningham; discarded
minor administrative categories.
```

Table 21: An example of structured KG evidence formatted for the ${evidence_blocks} in the Synthesis module. It includes the linked entity, verified facts (attributes and relations), and a reasoning trace for the pruning process.

Example of Formatted Candidates

```
Wikipedia Retrieval Results for entity: "John Cage"

Information Extraction:
- Page Title: John Cage
- All Page Sections: ['Life', 'Music', 'Visual art...', ...]

- A. Skimming Summary and Section Selection:
  - Rationale: The summary highlights Cage as a pioneer of indeterminacy and the prepared piano.
    To understand his specific methods, we need to explore the "Music" section.
  - Selected Sections for Deeper Analysis: ['Music']

- B. Extracted Content:
  - From Summary:
    John Cage was a pioneer of indeterminacy in music. Best known for "4'33"", a silent composition.
    He developed the "prepared piano" and used the "I Ching" as a standard composition tool.
  - From Sections:
    - Section: "Music"
      - Rationale for section processing: Contains detailed descriptions of his composition
        techniques like chance operations and rhythmic structures.
      - Extracted from Text:
          - In 1951, Cage started using the I Ching to compose using chance, imitating nature's
        manner of operation.
          - Music of Changes (1951) was the first major work created using this method.
          - Cheap Imitation (1969) is a chance-controlled reworking of Erik Satie's Socrate.
```

Table 22: An example of Text evidence formatted for the ${evidence_blocks} in the Synthesis module. It summarizes the high-level page content and provides fine-grained details extracted from specific sections relevant to the query.

ROLE

You are a master AI strategist leading a multi-hop question-answering mission. Your task is to plan the next step of the investigation after an information-gathering turn that was useful but insufficient.

CONTEXT

- **Original Question:** `${question}`
- **Current Notebook:** `${notebook}`
- **Previous Overall Plan:** `${analysis}`
- **Previous Sub-Queries:** `${queries}`
- **Candidate Entities Pool:** `${candidate_entities_pool}`

NEW FINDINGS

- **Thought Process:** `${thought_process}`
- **Extracted Content:** `${extracted_content}`

YOUR TASK

Plan the next round of investigation to dig deeper and bridge information gaps.

1. **Update Notebook:** Combine current notebook with new findings into a single, coherent summary, retaining all unique and relevant details.
2. **Update Analysis:** Revise the overall plan to reflect new understanding. Explain what the immediate next step should focus on and why.
3. **Plan Next Queries:** Define a new set of sub-queries and corresponding core entities based on the updated analysis.
4. **Manage Candidate Pool:** Review existing leads and identify new promising entities from the new findings. Update the pool by adding new leads and removing promoted ones.

OUTPUT FORMAT

Your output must follow this exact structure:

`Thought Process:` Your step-by-step reasoning.
`Updated Notebook:` The comprehensive summary of all known facts.
`Updated Analysis:` Your revised analysis and plan.
`Next Queries:` [List of concise and effective search query.]
`Next Entities:` [List of core entities corresponding to queries]
`Updated Candidate Pool:` [List of dictionaries, e.g., {"entity": "Name", "reason": "..."}]

YOUR RESPONSE:

Table 23: Prompt template for **Continue Exploration** in the Reflection module. The LLM updates the notebook and analysis based on new findings and plans the next steps to deepen the investigation.

ROLE

You are a master AI strategist leading a multi-hop question-answering mission. Your task is to recover from a failed information-gathering turn where the retrieved evidence was useless.

CONTEXT

- **Original Question:** `${question}`
- **Current Notebook:** `${notebook}`
- **Candidate Entities Pool:** `${candidate_entities_pool}`
- **Interaction History:** `${interaction_history}`

FAILED TURN DETAILS

The last exploration turn was deemed unproductive. Here's what went wrong:
- **Previous Overall Plan:** `${analysis}`
- **Failed Sub-Queries:** `${queries}`
- **Failed Core Entities:** `${entities}`
- **Extracted Content:** `${extracted_content}`
- **Reasoning for Failure:** `${thought_process}`

YOUR TASK

The previous approach hit a dead end. Critically reflect on the history, diagnose the error, and pivot the strategy.
1. **Critical Reflection:** Analyze why the previous approach failed (e.g., flawed plan, wrong entities). Summarize the core strategic error and derive actionable **guiding principles** for the next attempt.
2. **Update Analysis:** Propose a fundamentally new plan that leverages insights from reflection. Consider promoting leads from the Candidate Pool, re-examining the original question for missed keywords, or formulating queries from an entirely different angle.
3. **Plan Next Queries:** Define a new set of queries that represent a clear change in direction from the failed ones.
4. **Manage Candidate Pool:** Review the pool. Decide if existing leads are now high-priority, add any new promising leads discovered incidentally, and generate the updated pool by carrying over unused leads while removing promoted ones.

OUTPUT FORMAT

Your output must follow this exact structure:

`Thought Process:` Critical reflection on failure and reasoning for the new plan.
`Updated Analysis:` The revised analysis and new strategic direction.
`Next Queries:` [List of concise and effective search query.]
`Next Entities:` [List of core entities corresponding to queries.]
`Updated Candidate Pool:` [List of updated candidate dictionaries, e.g., {"entity": "Name", "reason": "..."}]

YOUR RESPONSE:

Table 24: Prompt template for **Strategy Adjustment** in the Reflection module. The LLM diagnoses the failure using interaction history and pivots the strategy by generating entirely new queries or backtracking to candidate entities.