GUESS FIRST, CORRECT THE REST: GRAPH SPECULATIVE RETRIEVAL FOR MULTI-HOP REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Retrieval-augmented generation (RAG) has been widely employed to reduce hallucinations in large language models (LLMs) by integrating external knowledge. However, traditional RAG systems struggle with fragmented knowledge in large-scale corpora, rendering them inadequate for multi-hop reasoning tasks. Although graph-enhanced RAG (GraphRAG) systems leverage graph structures to capture more knowledge associations, they still suffer from low-quality graphs, leading to two key limitations: (i) *Redundant retrieval results*: Excessive noise in the graph makes relevant evidence hard to reach. (ii) *Biased retrieval paths*: The rigid graph structure fails to dynamically match the reasoning process. This is essentially because existing GraphRAG systems struggle to overcome the quality bottleneck of graph construction, thus requiring a flexible retrieval strategy to reduce blind graph traversal. Motivated by this, we propose **G-Speculator**, which tentatively speculates clues on the graph to guide multi-hop retrieval. Specifically, it consists of two components: (i) Hypothetical Knowledge Graph Construction: We use a lightweight model to serialize passages into entity-relation sequences and construct a graph (HypoKG), which preserves contextual word order as a linguistic prior for speculative retrieval; (ii) Speculation-driven Passage Retrieval: We continuously locate clues and speculate bridge entities on HypoKG to obtain multi-hop evidence, while using chain-of-thought (CoT) to assess evidence sufficiency and adjust retrieval direction. Extensive experiments on four datasets demonstrate that G-Speculator outperforms state-of-the-art baseline methods, validating its practicality for real-world applications. Our code and datasets are available at `https://anonymous.4open.science/r/G-Speculator`.

1 INTRODUCTION

Retrieval-augmented generation (RAG) has been demonstrated as an effective approach for mitigating hallucinations and outdated information in large language models (LLMs) by introducing external knowledge (Lewis et al., 2020; Gao et al., 2023; Song et al., 2025b), enhancing their real-world applicability in domains such as medical diagnosis (Li et al., 2026) and legal judgment (Chen et al., 2026b). However, traditional RAG systems struggle with scattered knowledge in large-scale corpora due to the independent encoding and retrieval of each passage, which leads to incomplete or redundant context. Although recent works attempt to improve retrieval and reranking models via LLM-based post-training techniques (Chen et al., 2024; Lee et al., 2025), this challenge remains fundamentally unresolved, limiting performance on knowledge-intensive tasks (Zhang et al., 2025a).

Graph-enhanced retrieval-augmented generation (GraphRAG) has been recognized as a promising solution that builds graph structures to explicitly capture multi-hop associations (Peng et al., 2025; Zhang et al., 2025a; Xiang et al., 2026). Early efforts in GraphRAG methods, including KGP (Wang et al., 2024), RAPTOR (Sarthi et al., 2024), and MS-GraphRAG (Edge et al., 2024), construct document graphs or hierarchical structures to model cross-document connections, enabling multi-hop reasoning through structured traversal. Subsequent works such as HippoRAG (Gutiérrez et al., 2024), HippoRAG2 (Gutiérrez et al., 2025), and GFM-RAG (Luo et al., 2025) construct knowledge graphs (KGs) via LLM-based open information extraction (OpenIE), and adopt message propagation techniques, e.g., personalized PageRank (PPR) and graph neural networks (GNNs), to enhance multi-hop associations between passages. Recently, the community has shifted toward lightweight graph construction for retrieval, which uses spaCy model for entity extraction. For instance, Lin-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

earRAG (Zhuang et al., 2026) builds graphs via entity co-occurrence within sentences, while KG-Translator (Liu et al., 2026) constructs hypergraphs with entity-masked sentences as hyperedges.

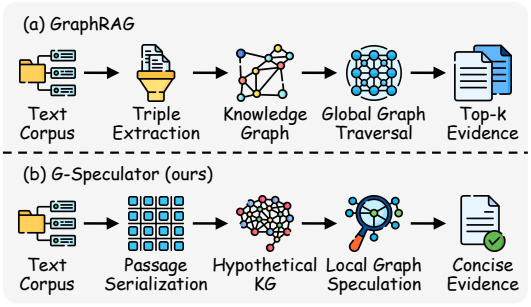


Figure 1: **Two paradigms of GraphRAG.**

Despite recent advances, we identify that existing GraphRAG methods still suffer from low-quality graphs, resulting in two key limitations: ❶ *Redundant retrieval results*: The graph contains substantial noise, such as missing relations or erroneous connections, which makes relevant evidence hard to find, yielding redundant passages in retrieval results. ❷ *Biased retrieval paths*: The fixed graph structure fails to align with the dynamic reasoning process, because real-world queries vary in type and complexity, causing the retrieval direction to deviate from ideal paths. These limitations stem from the same tension: graphs provide extensive associations but are simultaneously bound by their structure, requiring a more flexible GraphRAG paradigm. Therefore, we argue that graph-based retrieval should collect passages via local associations rather than exhaustive traversal to avoid noise interference. In addition, we should monitor the retrieval process and realign retrieval with user intent when evidence is insufficient, correcting retrieval directions that have deviated from target evidence.

To tackle these challenges, we propose **G-Speculator**, a novel framework that tentatively speculates clues on the graph to guide multi-hop retrieval. Specifically, (i) *Hypothetical KG Construction*: A lightweight model is first employed for named entity recognition (NER), and passages are split into entity-relation sequences with entities as delimiters to construct a graph (HypoKG), which retains sentence-level word order as a linguistic basis enabling speculative retrieval. (ii) *Speculation-driven Passage Retrieval*: It integrates graph speculative retrieval with retrieval direction correction. The former parses queries into triples and matches clue trajectories on HypoKG to collect multi-hop passages, where the core is speculating bridge entities from clue positions under a linguistic assumption. The latter uses chain-of-thought (CoT) to assess evidence sufficiency and generates new queries to adjust retrieval direction when insufficient. Our main contributions are summarized as follows:

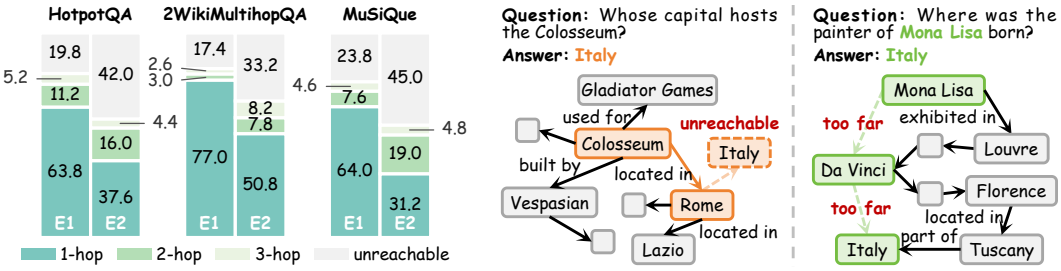
- We identify key limitations in existing GraphRAG systems, highlighting that these systems are experiencing *redundant retrieval results* and *biased retrieval paths*, which motivates the design of G-Speculator, a novel speculative framework for precise passage retrieval.
- G-Speculator uses a lightweight model to serialize passages into entity-relation sequences with entities as delimiters, thereby constructing a hypothetical knowledge graph (HypoKG) that preserves contextual word order as a linguistic prior to support speculative retrieval.
- On top of HypoKG, G-Speculator continuously matches clues on the graph and speculates bridge entities based on a linguistic assumption to collect multi-hop evidence. The evidence is analyzed by an LLM, which adjusts retrieval direction when evidence is insufficient.
- We conduct extensive experiments on four datasets covering diverse question types, demonstrating that G-Speculator achieves superior performance over state-of-the-art baselines and validating its effectiveness in generation accuracy, retrieval quality, and scalability.

2 Preliminary Study

Before diving into technical details, we conduct a series of preliminary experiments to investigate how low-quality graphs affect GraphRAG systems. Our findings reveal critical flaws in graph-based retrieval that explain the retrieval performance bottlenecks of existing GraphRAG systems.

2.1 Substantial Unreachable Evidence in GraphRAG Systems

To identify the underlying cause of graph structure limitations in existing GraphRAG systems, we investigate the reachability from queries to evidence on the KGs and conduct error analysis, as shown in Figure 2a, revealing core flaws in such systems. We provide details of graph construction and evidence reachability calculation in Appendix E. Specifically, we make the following observations:



(a) Evidence reachability (%) on KGs across HotpotQA, 2WikiMultihopQA, and MuSiQue.

(b) Two error cases arise from KG structural flaws: redundant retrieval result (left) and biased retrieval path (right).

Figure 2: **(a) Evidence reachability (%) on graphs.** E1 and E2 represent multi-hop evidence, where E2 depends on E1. n-hop indicates that the evidence is reachable at the n-th hop. **(b) Case study of two error types.** Structural flaws in KGs often lead to failed (left) or costly (right) retrieval.

Obs. 1. Evidence reachability limits the performance of GraphRAG systems. For a 2-hop question, it tends to be unanswerable if its second evidence is not retrieved. Across the three multi-hop datasets, first-evidence reachability on the graph is 80.2%, 82.6%, and 76.2%, respectively. However, for second-evidence retrieval, this rate drops significantly to 58.0%, 66.8%, and 55.0%. This indicates that multi-hop evidence reachability directly determines the retrieval performance upper bound, a systemic limit that even more complex GraphRAG methods struggle to overcome.

Obs. 2. Graph complexity increases the retrieval difficulty of GraphRAG systems. Besides the 33.2% to 45.0% unreachable evidence across the three datasets, reachable evidence also incurs high retrieval cost. For example, on HotpotQA, 11.2% and 5.2% of first evidence take 2 and 3 hops, respectively, while 20.4% of second evidence takes more than 2 hops, meaning that the ideal 2-hop evidence actually requires at least 6 hops to be reached. This indicates that GraphRAG are not only limited by substantial unreachable evidence but also face high retrieval costs for reachable evidence.

2.2 Retrieval Quality and Error Analysis

Case 1: Redundant retrieval result. Graph construction loses substantial information, rendering evidence structurally unreachable, resulting the system to return excessive irrelevant passages. For example, as shown in Figure 2b (left), for the question "Whose capital hosts the Colosseum?", OpenIE fails to extract the relation between "Rome" and "Italy", rendering this evidence unreachable during retrieval. However, the retriever's global graph traversal still scores all passages, forcing it to return numerous similar yet irrelevant passages, e.g., "The Colosseum is built by Vespasian".

Case 2: Biased retrieval path. Graph retrieval follows pre-defined structures only, leaving deviated paths uncorrectable, thereby incurring retrieval cost without guaranteeing target passages. For example, as shown in Figure 2b (right), for the question "Where was the painter of Mona Lisa born?", the target evidence should be reachable within 2-hops ("Mona Lisa"→"Da Vinci"→"Italy"). However, missing relations force the retriever to traverse a 7-hop path for complete evidence. Notably, besides such paths, substantial unreachable evidence still obstructs paths, as described in Case 1.

2.3 Discussion

Existing GraphRAG systems are almost inevitably plagued by structural flaws that degrade retrieval quality. While graph structures capture rich associations to benefit multi-hop retrieval, disconnected knowledge simultaneously undermines it, leading to two fundamental issues: **First, full-graph traversal is unnecessary.** In general, evidence involves only a few triples, while graph traversal scans the entire graph and forcibly returns top-k passages. Given excessive unreachable evidence in the graph, we argue that retrieval should ensure both evidence effectiveness and conciseness to avoid extra cost. **Second, retrieval paths should be continuously corrected rather than relying solely on the static graph.** Even stronger models struggle to extract comprehensive knowledge, leading to inherent graph flaws. To avoid the performance trap from such systemic flaws, GraphRAG systems should timely adjust retrieval direction to align graph structures with dynamic reasoning process.

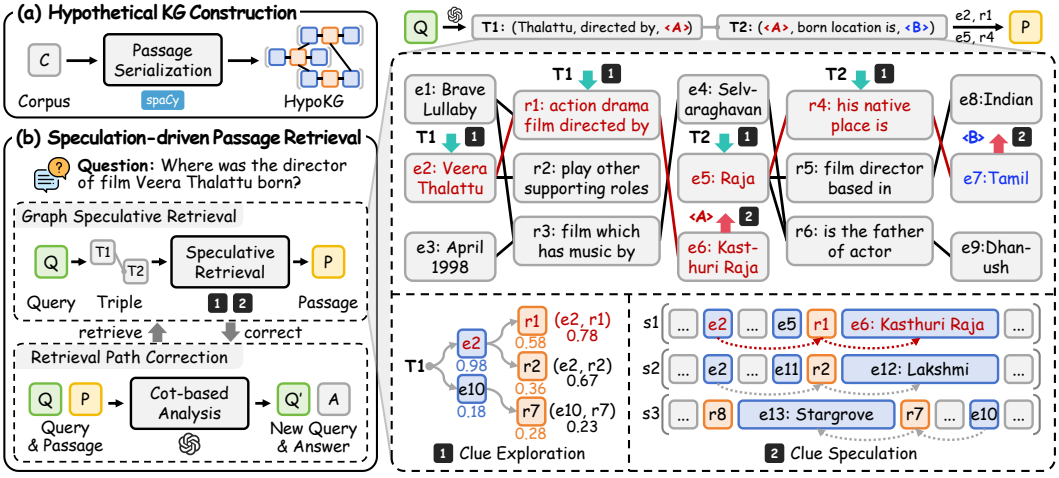


Figure 3: **The overall pipeline of G-Speculator. (a) Hypothetical KG Construction:** We first construct a graph (HypoKG) by leveraging a lightweight model to serialize passages into entity-relation sequences. **(b) Speculation-driven Passage Retrieval:** We continuously match clue and speculate bridge entities to collect multi-hop passages on HypoKG, while adjusting retrieval direction through CoT-based analysis. Our speculation mechanism follows a linguistic assumption: within a sentence, the *subject* (entity) and *object* (entity) typically lie on opposite sides of the *predicate* (relation).

3 Methodology

In the preliminaries, we analyze how structure flaws impair retrieval in GraphRAG systems. Motivated by this, we propose **G-Speculator**, a novel framework that enables flexible adaptation from static graphs to dynamic multi-hop retrieval. The overall pipeline is shown in Figure 3. Specifically,

G-Speculator consists of two components: (i) **Hypothetical KG Construction**, where a lightweight model serializes passages into entity-relation sequences to construct a graph named HypoKG, which preserves contextual word order as a linguistic prior for graph speculative retrieval; (ii) **Speculation-driven Passage Retrieval** comprises *graph speculative retrieval* and *retrieval path correction*. The former parses queries into triples and matches trajectories on HypoKG to collect passages, centered on guessing the bridge entity based on the clue's position within the sentence under a linguistic assumption. The latter analyzes evidence sufficiency via CoT and adjusts retrieval direction with new queries when necessary. This process delivers accurate retrieval and question answering (QA).

3.1 Hypothetical Knowledge Graph Construction

Graph construction almost inevitably incurs substantial knowledge loss. To mitigate this problem, we use a lightweight model (i.e., spaCy) to split passages into alternating entity-relation sequences, preserving contextual constraints and constructing a hypergraph named HypoKG. Specifically,

Passage Serialization. Given a passage set $\mathcal{P}$, we split passages into sentences $\mathcal{S}$ by punctuation, and extract named entities from each sentence $s \in \mathcal{S}$ as $\mathcal{E}_s$. With entities as delimiters, we split each sentence s into an entity-relation sequence, defining non-entity text spans as relations $\mathcal{R}_s$, i.e.,

$$s = [\ldots, x_{m-1}, x_m, x_{m+1}, \ldots], \quad \text{where } x \in \mathcal{E}_s \cup \mathcal{R}_s. \tag{1}$$

This sequence preserves passage context and describes the relative positions of entities and relations.

HypoKG Construction. We define HypoKG as $\mathcal{H} = (\mathcal{E}, \mathcal{R}, \phi)$, where the node set $\mathcal{E} = \bigcup_{s\in\mathcal{S}} \mathcal{E}_s$ and the hyperedge set $\mathcal{R} = \bigcup_{s\in\mathcal{S}} \mathcal{R}_s$. ϕ specifies the entities covered by each hyperedge, with each hyperedge $r \in \mathcal{R}_s$ satisfying $\phi(r) = \mathcal{E}_s$. Notably, since we use spaCy rather than generative models for graph construction, all entities and relations are traceable to the original passage via look-up.

Unlike traditional KGs where each triple is a definite fact (Luo et al., 2025), HypoKG stores hypothetical knowledge and contextual constraints. The former allows each entity to reach another in the

same sentence via hyperedges, building rich associations across passages. The latter define the relative positions of entities and relations in each sentence, preserving complete word-order information within sentences. Both provide the structural foundation for subsequent graph speculative retrieval.

3.2 SPECULATION-DRIVEN PASSAGE RETRIEVAL

Although GraphRAG systems excel at multi-hop retrieval, systemic structure flaws force them to over-retrieve unreachable evidence via global graph traversal (Gutiérrez et al., 2025; Zhuang et al., 2026). Unlike these retrieval strategies, we introduce graph speculative retrieval to collect valid and concise evidence, and duly adjust the retrieval direction by analyzing the necessity of evidence.

3.2.1 GRAPH SPECULATIVE RETRIEVAL

Since multi-hop evidence usually involves only 2–4 local triples with bridge entities as key hubs, precisely identifying them is essential. To resolve this issue, we propose *graph speculative retrieval*, which leverages linguistic regularities to speculate the bridge entity and performs local exploration.

Query Graph Construction. Given a question q, we use an LLM to generate a query graph in triple format, i.e., $\mathcal{G} = \{\tau_0, \tau_1, \ldots, \tau_{n-1}\}$, where each triple is denoted as $\tau = (u, \rho, v)$. The head entity u and tail entity v can be either known entities or placeholders. For example, for the question "Where was the director of film Veera Thalattu born?", the query graph contains two triples:

$$\tau_0 = (\texttt{"Veera Thalattu"}, \texttt{"directed by"}, \texttt{<A>}), \quad \tau_1 = (\texttt{<A>}, \texttt{"born location is"}, \texttt{<B>}). \tag{2}$$

In τ_0, "Veera Thalattu" is a known entity, and **<A>** is a placeholder for the director. Serving as the bridge for multi-hop reasoning, **<A>** provides crucial information for the subsequent retrieval of τ_1.

Clue Exploration. Given a triple τ, we first obtain the candidate entities $\mathcal{E}'$ in HypoKG via generative entity linking (Cao et al., 2021; Liu et al., 2026), and then retrieve hyperedges connected to each entity $e' \in \mathcal{E}'$, denoted as $\mathcal{R}(e') = \{r \in \mathcal{R} \mid e' \in \phi(r)\}$. Each entity e' paired with a relation $r' \in \mathcal{R}(e')$ forms a clue $c = (e', r')$. For each clue c, we define its score $\omega(c \mid \tau)$ as follows:

$$\omega(c \mid \tau) = \frac{1}{2}\Big[\exp\big(\mathcal{M}(e' \mid \tau, \mathcal{E})\big) \cdot \mathrm{idf}_{\text{norm}}(e') + \mathrm{sim}(\rho, r')\Big], \tag{3}$$

where $\mathcal{M}(e' \mid \tau, \mathcal{E})$ is the exponentiated logit from the entity linking model, $\mathrm{idf}_{\text{norm}}(\cdot)$ is the normalized inverse document frequency (IDF) score, and $\mathrm{sim}(\cdot)$ denotes the similarity between r' and ρ. Each clue can determine the corresponding passage according to sentence sequence s (Equation 1).

Clue Speculation. For the query graph $\mathcal{G} = \{\tau_0, \tau_1\}$ in Equation 2, the placeholder **<A>** of triple τ_1 must be filled before the next-hop clue exploration. We infer the placeholder entity (bridge entity) e^* based on the following linguistic assumption: In a *subject-predicate-object* sentence, the *subject* and *object* typically lie on opposite sides of the *predicate* and are directly adjacent to the *predicate*. Thus, given a clue $c = (e', r')$ existing in sentence s, based on the position (pos), we directly infer the nearest entity on the side of r' opposite to e' as the placeholder entity e^*, formalized as follows:

$$e^* = \begin{cases} \arg\min\{\mathrm{pos}(e, s) : e \in \mathcal{E}_s,\ \mathrm{pos}(e, s) > \mathrm{pos}(r', s)\}, & \mathrm{pos}(e', s) < \mathrm{pos}(r', s), \\ \arg\max\{\mathrm{pos}(e, s) : e \in \mathcal{E}_s,\ \mathrm{pos}(e, s) < \mathrm{pos}(r', s)\}, & \mathrm{pos}(e', s) > \mathrm{pos}(r', s). \end{cases} \tag{4}$$

We define τ^σ as the placeholder-filled variant of a triple, where σ aggregates all placeholder entities from preceding triples and fills them into the current triple as needed. Clue exploration proceeds sequentially in topological order across triples, yielding a complete clue trajectory, denoted as:

$$\pi = \{c_0, c_1, \ldots, c_{n-1}\} = \{\mathrm{clue_exploration}(\tau_i^\sigma)\}_{i=0}^{n-1}, \tag{5}$$

$$\omega(\pi) = \frac{1}{n}\sum\nolimits_{i=0}^{n-1} \omega(c_i \mid \tau_i^\sigma). \tag{6}$$

Notably, each triple returns multiple clues, causing exponential branch growth as retrieval deepens. Therefore, we adopt two-stage pruning: each hop retains the top-α clues by $\omega(c \mid \tau)$ to control exploration width; the final stage returns the top-β ($\beta \leq \alpha$) trajectories selected from α candidates by $\omega(\pi)$. Moreover, since speculative retrieval builds on linguistic assumption while sentences are diverse, deep speculation tends to propagate errors. We thus introduce δ to control the speculative depth, defined as the maximum depth allowed from zero-in-degree nodes in the query graph.

3.2.2 RETRIEVAL PATH CORRECTION

Graph speculative retrieval yields concise multi-hop evidence but remains limited by excessive unreachable evidence, requiring dynamic alignment of graph structure with user intent to avoid retrieval stalling. To tackle this problem, we use CoT (Wei et al., 2022) to analyze evidence sufficiency and then correct retrieval direction. For the passages $\mathcal{P}'$ obtained by graph speculative retrieval:

$$q', a^* \xrightarrow{\text{CoT}} \texttt{LLM}(q, \mathcal{P}'; \texttt{prompt}), \tag{7}$$

where q' and a^* denote the next-step query and the final answer, respectively, with exactly one non-empty. An empty a^* indicates insufficient evidence, requiring continued retrieval via the new query; a non-empty a^* indicates sufficient evidence and a completed answer. We define graph speculative retrieval as **SPEC** and retrieval path correction as **CORR**, and the process is formalized as follows:

$$\begin{gathered} q'_0 = q, \quad \mathcal{P}'_{\leq 0} = \varnothing, \\ \mathcal{P}'_{\leq t} = \mathcal{P}'_{\leq t-1} \cup \textbf{SPEC}(q'_{t-1}), \quad (a^*_t, q'_t) = \textbf{CORR}(q, \mathcal{P}'_{\leq t}), \quad t \in \{1, 2, \ldots, T\}, \end{gathered} \tag{8}$$

where T is the maximum number of steps. This process stops when a^*_t is non-empty or T is reached. If a^*_t is non-empty, it serves as the final answer; otherwise, the final answer is generated by an LLM in the zero-shot setting. In addition, when $q'_t = q'_{t-1}$, i.e., retrieval stagnates, we perform two operations: (i) Discard $\mathcal{P}'_t$ to ensure concise evidence; (ii) Expand 3 pieces of evidence via dense retrieval, complementary to graph speculative retrieval, to alleviate evidence-acquisition stagnation. See Appendix H for the detailed algorithm description and Appendix K for the prompt at each stage.

4 EXPERIMENTS

In this section, we conduct comprehensive experiments, aiming to answer the following questions: **Q1 (Effectiveness)**: How does G-Speculator perform compared to state-of-the-art methods in terms of performance? **Q2 (Ablation Study)**: How does each component of G-Speculator contribute to its overall performance? **Q3 (Efficiency)**: How efficient is G-Speculator in end-to-end time and cost? (Note that for a comprehensive evaluation, we conduct additional experiments, including **parameter sensitivity, different model backbones, case studies, etc.**, which are presented in Appendix D.)

4.1 EXPERIMENTAL SETUP

Datasets. We evaluate the effectiveness on four multi-hop QA datasets: HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and ScaleQA (Liu et al., 2026), where ScaleQA uses a large-scale corpus (35k). Following KG-Translator's setting (Liu et al., 2026), we select 1,000, 2,125, 1,313, and 1,500 as the test sets for these datasets, respectively, and report detailed cross-type performance. More details about datasets are provided in Appendix B.

Baselines. We compare G-Speculator against four categories of baselines: (i) *Zero-shot LLM inference:* Llama-3.1-8B (Grattafiori et al., 2024), GPT-4o-mini (OpenAI, 2024), and GPT-5.4-mini (OpenAI, 2026); (ii) *Vanilla RAG methods:* a basic RAG system (selecting top-1/3/5 passages) (Gao et al., 2023); (iii) *GraphRAG methods:* KGP (Wang et al., 2024), RAPTOR (Sarthi et al., 2024), MS-GraphRAG (Edge et al., 2024), LightRAG (Guo et al., 2025b), HippoRAG (Gutiérrez et al., 2024), HippoRAG2 (Gutiérrez et al., 2025), PER-QA (Liu et al., 2025), and LinearRAG (Zhuang et al., 2026). (iv) *Multi-step methods:* IRCoT (Trivedi et al., 2023), Search-R1 (Jin et al., 2025), and R1-Searcher (Song et al., 2025a) for multi-step RAG; HippoRAG+IRCoT (Gutiérrez et al., 2024), GFM-RAG+IRCoT (Luo et al., 2025), LogicRAG (Chen et al., 2026a), and KG-Translator (Liu et al., 2026) for multi-step GraphRAG. Descriptions of baselines can be found in Appendix C.

Evaluation Metrics. For QA performance, we adopt F1 score as the metric following prior work (Luo et al., 2025; Liu et al., 2026). For retrieval performance, we use Precision, defined as the ratio of supporting passages to all retrieved passages, to jointly assess retrieval accuracy and conciseness.

Implementation Details. For G-Speculator, we set the clue exploration width (α) to 3, the number of returned trajectories (β) to 1, the clue speculation depth (δ) to 2, and the number of additional passages (k) to 3. For all methods, we use the same embedding model (all-MiniLM-L6-v2) and LLM (GPT-4o-mini). All baselines adopt top-5 retrieval. More details can be found in Appendix F.

Table 1: **QA performance (%).** We present the F1 scores across different question types, including Bridge (Br.), Comparison (Cp.), Inference (In.), Bridge-comparison (Bc.), covering 2-hop (2H), 3-hop (3H), and 4-hop (4H). On ScaleQA, we report results on both 10k and 35k corpora to show scalability. Since ScaleQA Bri.$^{2H}_{10k}$ is sourced from the Bri.^{2H} of the other three datasets, we exclude it when computing the overall average performance. We highlight the **best** and second-best results.

Method	HotpotQA			2WikiMultihopQA						MuSiQue				ScaleQA		Avg.
	Br.^{2H}	Cp.^{2H}	Avg.	Br.^{2H}	Cp.^{2H}	In.^{2H}	Cp.^{4H}	Bc.^{4H}	Avg.	Br.^{2H}	Br.^{3H}	Br.^{4H}	Avg.	Br.$^{2H}_{10k}$	Br.$^{2H}_{35k}$	
ZERO-SHOT LLM INFERENCE																
LLaMA-3.1-8B	26.09	62.11	44.10	11.82	54.73	24.97	50.12	49.72	36.18	14.05	12.05	9.24	12.14	17.32	17.32	27.43
GPT-4o-mini	35.29	73.69	54.49	17.98	56.37	31.29	52.80	50.38	39.82	20.52	16.22	14.13	17.36	24.60	24.60	33.48
GPT-5.4-mini	42.99	79.46	61.23	21.00	67.20	36.35	65.60	49.26	44.76	24.50	18.61	18.02	20.71	29.50	29.50	38.36
VANILLA RAG METHODS																
Vanilla RAG (Top-1)	41.41	75.10	58.26	15.35	63.08	29.11	49.60	52.59	40.59	20.60	15.71	17.25	17.94	25.79	24.91	34.60
Vanilla RAG (Top-3)	55.26	84.73	70.00	22.49	78.35	38.63	61.60	65.03	51.74	31.51	25.25	25.68	27.74	36.42	34.37	45.12
Vanilla RAG (Top-5)	59.40	84.59	72.00	25.27	79.80	40.24	60.80	64.31	52.90	35.37	28.13	25.62	30.29	40.01	38.24	47.41
GRAPHRAG METHODS																
KGP	72.59	79.29	75.94	54.50	74.52	55.73	72.80	54.51	60.58	41.70	32.88	27.27	34.90	56.26	54.81	56.03
RAPTOR	58.39	82.61	70.50	24.61	74.18	41.40	55.20	58.33	49.96	35.08	26.68	25.66	29.64	39.36	36.18	45.44
MS-GraphRAG	40.10	74.05	57.08	16.78	62.59	26.78	59.20	50.26	40.28	24.45	21.33	13.64	20.68	27.11	24.85	34.88
LightRAG	64.82	81.51	73.17	56.77	78.73	49.29	60.00	64.65	62.22	41.64	22.96	21.15	29.64	54.41	50.42	53.88
HippoRAG	66.04	87.55	76.80	66.67	93.58	57.46	80.80	70.37	72.54	40.93	21.09	21.99	28.86	57.88	55.73	59.35
HippoRAG2	69.94	85.66	77.80	60.22	85.40	53.58	75.20	60.38	65.50	48.20	29.83	27.57	36.29	59.45	54.85	58.42
PER-QA	69.84	85.10	77.47	58.47	89.77	60.98	76.00	68.86	69.90	52.24	37.62	25.40	40.27	60.18	57.18	61.41
LinearRAG	68.52	86.00	77.26	60.37	89.26	52.48	80.00	66.08	67.81	44.26	21.23	25.18	30.94	57.72	48.86	56.46
MULTI-STEP METHODS																
IRCoT	68.30	86.00	77.15	49.42	84.58	53.59	69.60	62.28	62.89	43.92	33.37	33.56	37.43	53.88	51.48	56.78
Search-R1	74.17	86.26	80.21	60.10	84.19	51.85	71.20	62.50	65.04	53.23	37.30	23.74	40.13	62.50	60.40	60.92
R1-Searcher	74.26	87.24	80.75	63.44	88.06	58.57	76.80	63.57	68.90	56.81	38.43	26.84	42.66	64.84	62.54	63.49
HippoRAG + IRCoT	74.62	87.08	80.85	77.76	93.18	64.26	81.60	71.49	76.96	51.20	30.35	28.31	37.81	67.86	66.71	66.37
GFM-RAG + IRCoT	72.19	82.25	77.22	73.95	84.30	56.49	74.40	61.43	69.36	46.43	28.33	29.30	35.45	64.19	64.04	61.84
LogicRAG	69.76	88.21	78.99	65.39	93.07	58.25	92.46	71.52	73.26	50.47	38.53	30.78	41.23	61.87	60.35	63.88
KG-Translator	72.00	88.53	80.27	80.45	90.90	70.71	81.24	71.32	78.52	55.45	39.44	30.13	43.32	69.30	68.33	68.45
G-Speculator (ours)	**74.83**	**89.42**	**82.13**	**81.44**	**94.28**	**81.18**	**97.53**	**90.77**	**87.54**	**60.99**	**46.04**	**34.26**	**48.92**	**72.42**	**72.23**	**74.22**

4.2 MAIN RESULTS (Q1)

To address **Q1**, we comprehensively evaluate G-Speculator against state-of-the-art baselines on four datasets, with detailed results shown in Table 1 and 2. We summarize the observations as follows:

Obs. 1. G-Speculator consistently outperforms baselines across four datasets. Table 1 shows that G-Speculator is superior to the state-of-the-art baselines, with up to 9.02% average improvement on 2Wiki. We argue that GraphRAG such as KG-Translator still suffers from blind graph search, introducing graph noise into retrieval. Notably, even with only speculative retrieval, G-Speculator achieves surprising performance, contributing 54.53%–88.26% of the total performance across the four datasets. This suggests that speculative retrieval effectively uses linguistic assumptions to recall multi-hop evidence, and that the correction of retrieval direction guides better speculative retrieval.

Obs. 2. G-Speculator achieves superior performance while maintaining evidence conciseness. Existing RAG systems rely on top-5 retrieved passages for QA, as the scoring reflects relative relevance and is difficult to be thresholded. As shown in Table 2, with the top-5 setting, the best baseline (KG-Translator) reaches 63.17% average Precision on 2Wiki, i.e., 36.83% of retrieved passages are irrelevant. In contrast, G-Speculator cuts passages from 5 to 2.9 on average and raises Precision by 19.37%, while still maintaining a 9.02% absolute performance gain. This benefits from speculative retrieval, which only mines the most relevant clue trajectory without traversing the entire graph.

4.3 ABLATION STUDY (Q2)

To examine module necessity in G-Speculator, we ablate four modules grouped into two categories: **(i) Graph Speculative Retrieval**: speculation direction (w *opposite speculation*) and trajectory scoring (w/o *idf*). The former selects the relation-adjacent entity on the same side of the clue entity

Table 2: **Retrieval performance (%).** We evaluate retrieval performance using Precision, which measures the effectiveness and conciseness of evidence to achieve end-to-end QA performance.

Method	HotpotQA			2WikiMultihopQA						MuSiQue				ScaleQA		Avg.
	$Br.^{2H}$	$Cp.^{2H}$	Avg.	$Br.^{2H}$	$Cp.^{2H}$	$In.^{2H}$	$Cp.^{4H}$	$Bc.^{4H}$	Avg.	$Br.^{2H}$	$Br.^{3H}$	$Br.^{4H}$	Avg.	$Br._{10k}^{2H}$	$Br._{35k}^{2H}$	
VANILLA RAG & GRAPHRAG METHODS																
Vanilla RAG (Top-5)	27.00	34.92	30.96	22.60	31.32	23.72	23.68	34.48	27.77	22.96	26.68	22.30	24.22	24.19	23.03	26.33
HippoRAG	30.24	38.48	34.36	34.24	39.16	29.56	39.84	68.92	42.79	23.60	27.16	25.88	25.50	29.36	28.03	33.82
HippoRAG2	32.20	34.88	33.54	32.52	34.80	30.12	30.08	48.08	36.01	27.64	26.80	23.07	26.23	30.79	28.95	31.65
LinearRAG	30.60	34.24	32.42	32.52	37.48	29.96	39.52	58.68	39.65	25.36	25.52	28.31	26.12	29.49	25.52	31.87
MULTI-STEP METHODS																
IRCoT	31.88	35.52	33.70	29.60	32.72	28.52	23.84	47.00	33.84	27.08	31.48	27.80	28.93	29.52	28.40	31.35
HippoRAG + IRCoT	34.20	38.44	36.32	37.72	39.24	33.32	39.84	75.44	46.04	27.48	31.64	**31.31**	29.98	33.13	32.23	37.36
GFM-RAG + IRCoT	30.92	32.00	31.46	35.44	36.00	32.12	36.64	62.44	41.21	21.56	23.52	25.69	23.29	29.31	28.76	32.46
KG-Translator	45.95	62.60	54.28	58.61	68.09	50.53	72.24	73.18	63.17	31.63	29.81	26.48	29.71	45.40	37.44	47.77
G-Speculator (ours)	**59.60**	**73.86**	**66.73**	**81.45**	**93.72**	**74.88**	**90.85**	**78.02**	**82.54**	**46.29**	**34.81**	30.27	**38.10**	**62.45**	**60.22**	**64.41**

as the placeholder (bridge) entity, while the latter removes IDF correction in clue scoring, letting high-frequency entities bias retrieval. **(ii) Retrieval Path Correction**: removing it (*w/o correction*) eliminates multi-step setting and CoT analysis, leaving QA to be completed by the zero-shot LLM. We also ablate supplementary evidence (*w/o evidence expansion*) from dense retrieval, triggered by retrieval stagnation. Detailed results are shown in Figure 4, and we obtain the following findings:

Obs. 3. Each module in G-Speculator is critical for performance. As shown in Figure 4, under single-step settings: using opposite speculation and removing IDF cause F1 drops of up to 11.0% and 8.4% on ScaleQA, respectively. Without retrieval path correction, F1 drops by 4.2% on 2Wiki. In addition, evidence expansion is only active under the multi-step setting, and its removal causes an 8.8% F1 drop on MuSiQue. This highlights that each component contributes complementary value, enabling G-Speculator to achieve strong performance.

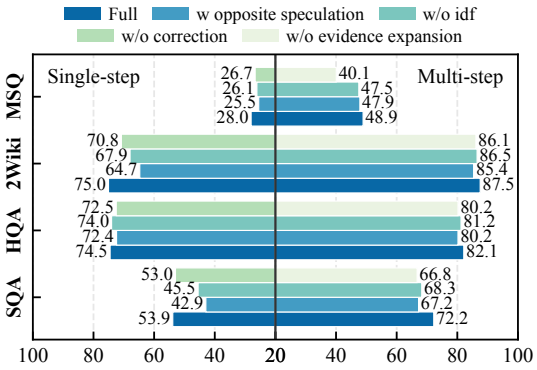


Figure 4: **Ablation study.** We report F1 (x-axis, %) under single-step (left) and multi-step (right).

Obs. 4. Graph speculative retrieval and retrieval path correction are complementary under the multi-step setting. Figure 4 shows that under the multi-step setting, ablating speculative retrieval modules causes smaller performance drops than under single-step. The ablation of opposite speculation and IDF yields maximum drops of 5.0% and 3.9% on ScaleQA, respectively. This is because the path correction adjusts retrieval directions during multi-step retrieval, compensating for retrieval errors of the graph speculative retrieval. This complementary mechanism indicates that G-Speculator possesses stronger robustness and adaptability in complex reasoning.

4.4 EFFICIENCY ANALYSIS (Q3)

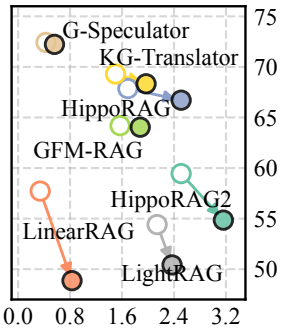
To address **Q3**, we present an end-to-end efficiency analysis across various RAG systems in Table 3, covering graph construction, retrieval, and API usage. We draw the following key observations:

Obs. 5. Costly graph construction does not guarantee higher performance gains. As shown in Table 3, HippoRAG and GFM-RAG use LLMs for triple extraction, with input and output token overhead reaching up to 4.37×10^6 and 1.28×10^6, respectively. By comparison, G-Speculator builds graphs without tokens and still outperforms HippoRAG+IRCoT and GFM-RAG+IRCoT by 5.52% and 8.19% in F1, respectively. Moreover, despite low-cost graph indexing, lightweight GraphRAG methods like LinearRAG and KG-Translator lack contextual word order, hindering bridge entity inference under the linguistic assumption, failing to fully leverage graph structural advantages.

Obs. 6. G-Speculator offers a competitive trade-off between efficiency and performance. As shown in Table 3, G-Speculator is efficient in both graph construction and retrieval. Although RAP-

Table 3: **End-to-end efficiency (left).** For graph indexing, we report the LLM's input/output tokens ($\times 10^6$) and time (s) per 5k passages. For retrieval, we report retriever runtime (s) and context count. Additionally, we measure the average API cost of each stage for each query, including: retrieval (**R**), intermediate reasoning (**I**) and QA (**A**). **Retriever runtime and QA performance with expanding corpus (right).** We plot single-step retriever runtime (x-axis) against F1 (y-axis). The arrow denotes corpus size scaling from 10k to 35k. All experimental data are derived from evaluations on ScaleQA.

Method	Indexing			Retrieval		API Call			F1
	Tok-i	Tok-o	Time	Time	Ctx	**R**	**I**	**A**	
RAPTOR	0.61	0.09	1528	0.02	5.00	0	0	1	36.18
LightRAG ▶	4.78	0.73	5396	2.37	-	1	0	1	50.42
HippoRAG2 ▶	4.30	1.28	1062	3.16	5.00	1	0	1	54.85
LinearRAG ▶	0.00	0.00	225	0.83	5.00	0	0	1	48.86
HippoRAG + IRCoT ▶	4.30	1.28	1064	2.51	5.00	2.67	1.67	1	66.71
GFM-RAG + IRCoT ▶	4.10	1.18	7797	1.88	5.00	2.67	1.67	1	64.04
KG-Translator ▶	0.00	0.00	225	1.97	4.51	0	1.45	1	68.33
G-Speculator (ours) ▶	0.00	0.00	226	0.56	3.28	1.72	1.98	0.21	72.23



TOR is more efficient at retrieval, it trades most of its performance for speed. G-Speculator builds graphs with a lightweight model and retrieves evidence without graph traversal, making it more efficient than other GraphRAG systems. A 3.5× corpus expansion increases retriever runtime by only 0.13s, and the retrieved context remains concise. In addition, although G-Speculator introduces additional API calls, the performance gains are significant, keeping the overall pipeline competitive. Notably, we also provide a component-wise efficiency analysis of G-Speculator in Appendix D.3.

5 Related Work

Retrieval-Augmented Generation (RAG) reduces LLM hallucinations by guiding generation with retrieved external evidence (Gao et al., 2023; Song et al., 2025b). Classic RAG follows a "retrieve-and-read" paradigm, retrieving passages from unstructured corpora and feeding them to an LLM for generation (Gao et al., 2023). Later works extend this to a multi-step setting (Press et al., 2023; Trivedi et al., 2023), e.g., IRCoT (Trivedi et al., 2023) interleaves retrieval with single-step CoT for progressive retrieval. Recently, such multi-step retrieval has been internalized into LLMs (Jin et al., 2025; Song et al., 2025a), enabling autonomous retrieval decisions. However, existing RAG overlooks fine-grained associations in corpora, limiting adaptability to complex reasoning tasks.

Graph-enhanced RAG (GraphRAG) leverages graph structures to capture cross-passage associations. Early works organize corpora into document graphs or hierarchical structures (Wang et al., 2024; Sarthi et al., 2024; Edge et al., 2024). Subsequent works construct KGs via LLM-based OpenIE, then applying graph algorithms such as PPR and GNN to score nodes, relations, or subgraphs (Gutiérrez et al., 2024; 2025; Luo et al., 2025). More recently, the community has shifted toward lightweight graph construction for passage retrieval: LinearRAG (Zhuang et al., 2026) and KG-Translator (Liu et al., 2026) use spaCy for passage parsing and NER. The former acquires multi-hop evidence via PPR, while the latter translates queries into graph clues with a specially trained model. Despite these advances, existing GraphRAG remains bounded by low-quality graphs, leading to ineffective traversal and unhelpful contexts. We provide more detailed related work in Appendix G.

6 Conclusion

In this paper, we identify that existing GraphRAG systems suffer from redundant retrieval results and biased retrieval paths, caused by unreachable evidence and noise in the graph. Motivated by this, we propose a novel framework, named G-Speculator, which enables precise passage retrieval through graph speculation. Specifically, G-Speculator first constructs a hypothetical knowledge graph (HypoKG) by leveraging a lightweight model to serialize passages into entity-relation sequences. Based on HypoKG, it continuously matches clues and speculates bridge entities to retrieve multi-hop passages, while duly correcting retrieval direction through CoT-based analysis. Extensive experiments and in-depth analysis demonstrate that G-Speculator outperforms state-of-the-art baseline methods.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript writing and polishing, and to generate synthetic datasets for the preliminary study. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, or interpreting results. We have reviewed all AI-assisted work: We manually validated content and confirmed the polished text retains its meaning. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our research addresses only scientific questions and involves no human subjects, animals, or environmentally sensitive materials. The datasets used in our experiments, including HotpotQA, 2WikiMultihopQA, MuSiQue, and GraphRAG-bench, are either widely used public datasets or constructed from publicly available resources. We strictly adhere to the ICLR Code of Ethics, particularly regarding data privacy, transparency, and responsible computing practices. We anticipate no ethical risks, conflicts of interest, or concerns related to bias, fairness, or security.

REPRODUCIBILITY STATEMENT

We provide comprehensive materials to ensure the reproducibility of our work. Dataset details are introduced in Section 4.1 and further detailed in Appendix B, baseline descriptions in Section 4.1 and Appendix C, and implementation details for both our method and baselines in Section 4.1 and Appendix F. Key hyperparameter configurations are provided in Appendix J, and all prompts used in experiments are listed in Appendix K. Additionally, detailed performance reports are available in Appendix I. All baseline models are sourced from public repositories, and our code is available in a GitHub repository: `https://anonymous.4open.science/r/G-Speculator`.

REFERENCES

Aisha Alansari and Hamzah Luqman. Large language models hallucination: A comprehensive survey. *Computer Science Review*, 61:100970, 2026.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International conference on learning representations*, volume 2024, pp. 9112–9141, 2024.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego De Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2206–2240, 2022.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. Autoregressive entity retrieval. In *International Conference on Learning Representations*, 2021.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics, 2024.

Shengyuan Chen, Chuang Zhou, Zheng Yuan, Qinggang Zhang, Zeyang Cui, Hao Chen, Yilin Xiao, Jiannong Cao, and Xiao Huang. You don't need pre-built graphs for RAG: Retrieval augmented generation with adaptive reasoning structures. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30270–30278, 2026a.

Zerui Chen, Qinggang Zhang, Zhishang Xiang, Zhimin Wei, Linfeng Gao, Xiao Huang, Zhihong Zhang, and Jinsong Su. LegalGraphRAG: Multi-agent graph retrieval-augmented generation for reliable legal reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 37455–37484. Association for Computational Linguistics, 2026b.

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

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Kai Guo, Xinnan Dai, Shenglai Zeng, Harry Shomer, Haoyu Han, Yu Wang, and Jiliang Tang. Beyond static retrieval: Opportunities and pitfalls of iterative retrieval in graphRAG. *arXiv preprint arXiv:2509.25530*, 2025a.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. LightRAG: Simple and fast retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 10746–10761, 2025b.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 59532–59569. Curran Associates, Inc., 2024.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. From RAG to memory: Non-parametric continual learning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 21497–21515, 2025.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-Retriever: Retrieval-augmented generation for textual graph understanding and question answering. In *Advances in Neural Information Processing Systems*, volume 37, pp. 132876–132907. Curran Associates, Inc., 2024.

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

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

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling*, 2025.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-Embed: Improved techniques for training llms as generalist embedding models. In *International Conference on Learning Representations*, volume 2025, pp. 79310–79333, 2025.

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. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020.

Lei Li, Xiao Zhou, Yingying Zhang, and Xian Wu. From retrieval to generation: Unifying external and parametric knowledge for medical question answering. In *Proceedings of the ACM Web Conference 2026*, pp. 2419–2430. Association for Computing Machinery, 2026.

Mufei Li, Siqi Miao, and Pan Li. Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation. In *International Conference on Learning Representations*, volume 2025, pp. 6061–6089, 2025.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*, 2023.

Lei Liang, Zhongpu Bo, Zhengke Gui, Zhongshu Zhu, Ling Zhong, Peilong Zhao, Mengshu Sun, Zhiqiang Zhang, Jun Zhou, Wenguang Chen, et al. KAG: Boosting LLMs in professional domains via knowledge augmented generation. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 334–343, 2025.

Qichuan Liu, Chentao Zhang, Chenfeng Zheng, Guosheng Hu, Xiaodong Li, and Zhihong Zhang. Beyond the answer: Advancing multi-hop QA with fine-grained graph reasoning and evaluation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23433–23456. Association for Computational Linguistics, 2025.

Qichuan Liu, Qinggang Zhang, Yuxuan Hu, Chenfeng Zheng, Zerui Chen, Chentao Zhang, and Zhihong Zhang. From retrieval to translation: Translating query into graph-level clues for retrieval-augmented generation. In *Forty-third International Conference on Machine Learning*, 2026.

Linhao Luo, Zicheng Zhao, Reza Haffari, Dinh Phung, Chen Gong, and Shirui Pan. GFM-RAG: Graph foundation model for retrieval augmented generation. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 36371–36405. Curran Associates, Inc., 2025.

Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Li, Huaren Qu, Cehao Yang, Jiaxin Mao, and Jian Guo. Think-on-Graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation. In *International Conference on Learning Representations*, volume 2025, pp. 52782–52806, 2025.

Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

Agada Joseph Oche, Ademola Glory Folashade, Tirthankar Ghosal, and Arpan Biswas. A systematic review of key retrieval-augmented generation (RAG) systems: Progress, gaps, and future directions. *arXiv preprint arXiv:2507.18910*, 2025.

OpenAI. Hello GPT-4o, 2024. URL `https://openai.com/index/hello-gpt-4o/`.

OpenAI. Introducing gpt-5.4 mini and nano, 2026. URL `https://openai.com/index/introducing-gpt-5-4-mini-and-nano/`.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. Graph retrieval-augmented generation: A survey. *ACM Transactions on Information Systems*, 44(2):1–52, 2025.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711. Association for Computational Linguistics, 2023.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher Manning. RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations*, volume 2024, pp. 32628–32649, 2024.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-Searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025a.

Zirui Song, Bin Yan, Yuhan Liu, Miao Fang, Mingzhe Li, Rui Yan, and Xiuying Chen. Injecting domain-specific knowledge into large language models: A comprehensive survey. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 25297–25311. Association for Computational Linguistics, 2025b.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. Think-on-Graph: Deep and responsible reasoning of large language model on knowledge graph. In *International Conference on Learning Representations*, volume 2024, pp. 3868–3898, 2024.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 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)*, pp. 10014–10037. Association for Computational Linguistics, 2023.

Yu Wang, Nedim Lipka, Ryan A Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. Knowledge graph prompting for multi-document question answering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 19206–19214, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022.

Yuhe Wu, Guangyu Wang, Yuran Chen, Jiatong Zhang, Yutong Zhang, Yujie Chen, Jiaming Shang, Guang Zhang, and Zhuang Liu. PRISM: Probing reasoning, instruction, and source memory in LLM hallucinations. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33619–33656. Association for Computational Linguistics, 2026.

Zhishang Xiang, Chuanjie Wu, Qinggang Zhang, Shengyuan Chen, Zijin Hong, Xiao Huang, and Jinsong Su. When to use graphs in RAG: A comprehensive analysis for graph retrieval-augmented generation. In *International Conference on Learning Representations*, volume 2026, pp. 66145–66178, 2026.

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

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

Qinggang Zhang, Shengyuan Chen, Yuanchen Bei, Zheng Yuan, Huachi Zhou, Zijin Hong, Hao Chen, Yilin Xiao, Chuang Zhou, Junnan Dong, et al. A survey of graph retrieval-augmented generation for customized large language models. *arXiv preprint arXiv:2501.13958*, 2025a.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. Siren's song in the AI ocean: A survey on hallucination in large language models. *Computational Linguistics*, 51(4):1373–1418, 2025b.

Luyao Zhuang, Shengyuan Chen, Yilin Xiao, Huachi Zhou, Yujing Zhang, Hao Chen, Qinggang Zhang, and Xiao Huang. LinearRAG: Linear graph retrieval augmented generation on large-scale corpora. In *International Conference on Learning Representations*, volume 2026, pp. 147053–147075, 2026.

APPENDIX

A Frequently Asked Questions (FAQs) 15

A.1 What are the advantages of G-Speculator? 15

A.2 How is evidence reachability evaluated in preliminary study? 15

A.3 Why is HypoKG essential to support graph speculative retrieval? 15

A.4 Why evaluate GraphRAG across diverse queries and corpus scales? 15

A.5 Why use Precision to evaluate retrieval performance? 16

B Dataset Statistics 16

C Baseline Details 17

D Additional Experiments 18

D.1 Hyperparameter Sensitivity Analysis 18

D.2 Cross-domain Generalization 19

D.3 Component-wise Efficiency Analysis 20

D.4 Detailed Retrieval Performance 20

D.5 Effectiveness of Different Step Counts 21

D.6 Effectiveness of Different LLM Backbones 22

D.7 Effectiveness of Different Entity Linking Models 23

D.8 Effectiveness of Different Sentence Embeddings 23

D.9 Case Study 23

D.10 Error Analysis 24

E Setup of Preliminary Study 24

F Implementation Details 27

F.1 Implementation Details of G-Speculator 27

F.2 Implementation Details of Baselines 27

G Detailed Related Work 27

G.1 Retrieval-Augmented Generation 27

G.2 Graph-enhanced Retrieval-Augmented Generation 28

H Algorithm of G-Speculator 29

I Detailed Performance Reports 30

J Detailed Baseline Configuration 32

K Prompts 33

A Frequently Asked Questions (FAQs)

A.1 What are the advantages of G-Speculator?

Existing GraphRAG generally adopts a traversal paradigm, where retrieval proceeds strictly along graph topology and paths are entirely determined by graph quality. This paradigm has two limitations: (i) Undifferentiated traversal wastes computation on irrelevant nodes and introduces redundant evidence, and (ii) Missing relations or noise can divert traversal from correct reasoning paths.

Different from the traversal paradigm, we propose the speculative paradigm with the core principle of "speculate first, correct the rest". We argue that humans do not exhaustively review all materials before concluding in complex reasoning, but first form a tentative judgment and then seek targeted evidence to verify it. This is precisely the idea behind speculative retrieval: instead of strictly probing step by step along the graph structure, it first infers the most likely reasoning path from existing evidence, then verifies this speculation and corrects the direction with actual retrieval results.

Specifically, rather than traversing node by node, graph speculative retrieval first locally locates relevant clues on HypoKG based on query intent, then for each clue directly infers bridge entities based on *subject-predicate-object* positional regularities in sentences. This enables precisely targeting the most relevant nodes and skipping irrelevant paths, avoiding redundant evidence and scaling well to large corpora. In addition, the LLM's active analysis of evidence sufficiency provides flexible correction for graph speculative retrieval, adapting to dynamic and complex reasoning processes.

A.2 How is evidence reachability evaluated in preliminary study?

GraphRAG relies on graph structures to capture cross-document knowledge associations for multi-hop reasoning, and thus graph quality directly determines whether multi-hop evidence is retrievable on the graph. To this end, we first define and quantify *evidence reachability* in the preliminary study: for a multi-hop question and its gold evidence chain, the evidence is reachable if there exists a path within a limited hop count in the constructed KG that covers all entities in the chain; otherwise it is unreachable. We set the reachability hop limit to 3, as paths longer than 3 hops are difficult to capture in practical graph retrieval. Detailed experimental settings are provided in Appendix E.

We select 2-hop bridge questions as test cases in the preliminary study due to their clear hop structure. For example, the question "When was Juan Manuel Morales's father born?" corresponds to the evidence chain (`"Juan Manuel Morales"`, `"Carlos María Morales"`, `"1 March 1970"`). Notably, the evidence chain has a strict dependency: the answer entity (bridge entity) of the first evidence serves as the query starting point for the second evidence, whose retrieval depends on the first evidence.

A.3 Why is HypoKG essential to support graph speculative retrieval?

Existing graph construction adopts different strategies: HippoRAG extracts triples via OpenIE, LinearRAG relies on sentence-level entity co-occurrence, and KG-Translator uses entity-masked sentences. However, these methods fail to preserve the relative positions of entities and relations within sentences, causing to provide only semantic supervision, without the inherent word-order structure.

In contrast, G-Speculator converts passages into sentence-level entity-relation sequences and constructs a HypoKG. For example, the sentence "William D. Russell (April 30, 1908 - April 1, 1968) was an American director." is serialized as [`"William D. Russell"`, `"April 30, 1908"`, `"April 1, 1968"`, `"was an"`, `"American director"`]. When the head entity "William D. Russell" and the relation "was an" are identified, the contextual constraints within this sequence implies that the tail entity is most likely "American director". This provides linguistic grounds for bridge entity speculation, avoiding blind traversal of irrelevant paths. In addition, all entities within the same sentence are fully connected via hyperedges, maintaining connectivity even without explicit relation words.

A.4 Why evaluate GraphRAG across diverse queries and corpus scales?

Existing GraphRAG evaluations are often limited to a single query type (2-hop bridge), and a fixed corpus scale, which fails to reflect true capability. First, retrieval demands vary across query types. Bridge questions require intermediate entity localization, where errors propagate across hops, es-

pecially for 3-hop or 4-hop bridges; comparison questions require comparative evidence. Consequently, gains on one type rarely transfer to others, and single-type evaluation overestimates generalization. Second, real-world corpora vary in scale. Expanding corpus size exposes precision-latency trade-offs hidden on small corpora, necessitating cross-scale evaluation to assess scalability.

A.5 Why use Precision to evaluate retrieval performance?

Unlike end-to-end QA that reflects overall performance of the system, retrieval metrics should quantify properties of the retrieval process and offer a distinct assessment perspective. Specifically, Recall quantifies the fraction of gold supporting passages that are successfully retrieved, reflecting evidence coverage. This dimension is implicitly reflected in end-to-end QA scores, since the absence of supporting passages directly impairs answer correctness. In contrast, Precision measures the proportion of supporting passages among all retrieved results, jointly capturing retrieval accuracy and conciseness, with higher values indicating more precise and less redundant retrieval by the system.

B Dataset Statistics

For the experiments, we include four multi-hop QA datasets: **HotpotQA** (Yang et al., 2018), **2WikiMultihopQA** (Ho et al., 2020), and **MuSiQue** (Trivedi et al., 2022), together with **ScaleQA** (Liu et al., 2026), a multi-hop QA dataset for evaluating model performance under corpus expansion.

- **HotpotQA** (Yang et al., 2018) is a benchmark for multi-hop reasoning under noisy evidence, derived from a large English Wikipedia corpus. Each question is paired with two gold supporting passages and multiple distractor passages, requiring models to identify relevant evidence and integrate information across documents while disregarding noise.
- **2WikiMultihopQA (2Wiki)** (Ho et al., 2020) is a cross-document benchmark built from structured Wikidata and Wikipedia templates. Each question requires combining information from either 2 or 4 documents, and the dataset provides fine-grained evidence labels and reasoning chains, making it suitable for evaluating structured cross-document inference.
- **MuSiQue** (Trivedi et al., 2022) is a challenging benchmark, in which each question constructed by composing 2–4 sub-questions into a coherent reasoning chain across compositional hops. The dataset requires models to integrate information scattered across multiple distinct passages while maintaining contextual consistency, thereby discouraging shallow shortcut-based inference and emphasizing stepwise logical reasoning across documents.
- **ScaleQA** (Liu et al., 2026) is a multi-hop QA benchmark developed for corpus expansion settings. It targets 2-hop bridge questions drawn from HotpotQA, 2WikiMultihopQA, and MuSiQue, constituting a primary task for GraphRAG systems. The benchmark combines the corpora of these three datasets into a corpus of 35k passages, aligning with practical expansion use cases and assessing how well methods adapt to large-scale corpora.

Notably, to fully demonstrate G-Speculator's effectiveness across diverse query types, we evaluate four query types, including Bridge (Br.), Comparison (Cp.), Inference (In.), and Bridge-comparison (Bc.), involving 2-hop (2H), 3-hop (3H), or 4-hop (4H) reasoning. We introduce these types below:

- **Bridge (Br.)** includes 2-hop (Br.^{2H}), 3-hop (Br.^{3H}), and 4-hop (Br.^{4H}) questions, where a bridge entity connects key sequential reasoning steps. A 2-hop case asks for the spouse of the British Virgin Islands's owner, requiring the model to first identify the owner.
- **Comparison (Cp.)** comprises 2-hop (Cp.^{2H}) and 4-hop (Cp.^{4H}) that require retrieving independent facts and then comparing them. As an illustration, determining whether Ransom Riggs and John Berry are from the same country requires identifying both countries.
- **Inference (In.)** is a special 2-hop bridge type (In.^{2H}) unique to 2Wiki, involving rule-based relational inference. This is exemplified by asking for Elizabeth of Pomerania's father-in-law, which requires first identifying her spouse and then the spouse's own father.
- **Bridge-comparison (Bc.)** is a 4-hop question type (Bc.^{4H}) unique to 2Wiki, and it combines bridge reasoning with subsequent comparison. An instance compares the birth dates of the directors of The FBI Story and Mayrig, to determine which director was born later.

Table 4: **Dataset and graph statistics.** #Test is the total number of test samples per dataset. #Query Type gives the count for each type. #Psg. and #Sent. refer to the numbers of passages and sentences in the corpus, while #Ent. and #Rel. refer to the numbers of entities and hyperrelations in HypoKG.

Dataset	#Test	#Query Type							#Psg.	#Sent.	#Ent.	#Rel.
		Br.^{2H}	Br.^{3H}	Br.^{4H}	In.^{2H}	Cp.^{2H}	Cp.^{4H}	Bc.^{4H}				
HotpotQA	1,000	500	/	/	/	500	/	/	9,633	38,625	69,506	109,914
2Wiki	2,125	500	/	/	500	500	125	500	12,406	44,270	81,701	133,745
MuSiQue	1,313	500	500	313	/	/	/	/	12,718	46,815	80,085	137,029
ScaleQA	1,500	1,500	/	/	/	/	/	/	34,733	129,649	208,307	380,492

Data Statistics. We select diverse query types for each dataset, as summarized in Table 4. These types are derived from the original dataset categorizations. For each dataset, we combine supporting and distractor passages into a unified corpus for graph construction and retrieval. ScaleQA contains approximately 35k passages, 2.73 to 3.61 times larger than other datasets, and is used to evaluate GraphRAG performance on large-scale corpora. Its data consists of 2-hop bridge questions from the other datasets, a type widely adopted in GraphRAG evaluation. To demonstrate scalability from 10k to 35k, we additionally report results on the original 10k corpus, denoted as Bri.$^{2H}_{10k}$ and Bri.$^{2H}_{35k}$.

Graph Statistics. As shown in Table 4, we construct one HypoKG per dataset, and all query types within a dataset are evaluated on its corresponding HypoKG. For ScaleQA benchmark, the entity count is 2.54 to 3.00 times that of other datasets, and the hyperrelation count is 2.78 to 3.46 times.

C Baseline Details

In our experiments, we evaluate G-Speculator against 21 baselines. These baselines can be classified into four categories: (i) Zero-shot LLM inference, (ii) Vanilla RAG method, (iii) GraphRAG method, and (iv) Multi-step method (multi-step RAG & multi-step GraphRAG). Details are as follows:

(i) **Zero-shot LLM inference** evaluates the reasoning capabilities of LLMs based exclusively on their parametric knowledge, without incorporating external knowledge sources. The evaluated models span diverse model families and parameter scales to ensure broad coverage. Our evaluation includes three models, selected from both open-source and commercial model families: LLaMA-3.1-8B (Grattafiori et al., 2024), GPT-4o-mini (OpenAI, 2024), and GPT-5.4-mini (OpenAI, 2026).

(ii) **Vanilla RAG method** follows a standard "retrieve-and-read" paradigm (Gao et al., 2023). For each query, it identifies relevant passages from an unstructured corpus and uses them as contextual evidence for answer generation. This grounds responses in retrieved information during generation.

(iii) **GraphRAG method** incorporates graph- or tree-based structures to model relationships among distinct passages and facilitate the integration of global contextual information to support multi-hop evidence retrieval and to inform answer generation. In this setting, the compared methods include:

- **KGP** (Wang et al., 2024) constructs a knowledge graph in which passages serve as nodes and semantic relations serve as edges. An LLM-guided traversal agent incrementally traverses this graph to retrieve relevant evidence for multi-hop question answering tasks.
- **RAPTOR** (Sarthi et al., 2024) recursively embeds, clusters, and summarizes text segments in a bottom-up manner to construct a hierarchical tree structure, where nodes across different levels capture information with varying degrees of abstraction. This design enables retrieval across multiple granularities and abstraction levels for the question answering tasks.
- **MS-GraphRAG** (Edge et al., 2024) constructs an entity-centric knowledge graph and hierarchical summaries for entity communities, thereby retrieving both summaries and source documents to provide global and local information for complex multi-hop reasoning tasks.
- **LightRAG** (Guo et al., 2025b) combines graph-based representations with vector embeddings and further employs a dual-level retrieval to capture fine-grained details and high-level semantic information for more accurate and highly reliable results in complex tasks.

- **HippoRAG** (Gutiérrez et al., 2024) uses OpenIE to effectively construct an entity-centric knowledge graph that connects extracted entities and facts through relational and synonym links, and then applies PPR for efficient cross-passage retrieval to support multi-hop QA.
- **HippoRAG2** (Gutiérrez et al., 2025) extends HippoRAG with passage nodes and context edges, connects queries to the knowledge graph through triples, and applies LLM-based triple filtering, enabling evidence propagation across entities and passages during retrieval.
- **PER-QA** (Liu et al., 2025) employs a "Planner-Executor-Reasoner" architecture to decompose questions into sub-queries and conduct retrieval according to a directed acyclic graph (DAG), thereby coordinating sub-query dependencies across retrieval and reasoning steps.
- **LinearRAG** (Zhuang et al., 2026) builds a relation-free hierarchical graph by linking entities, sentences, and passages through named-entity-based literal matching and then performs two-stage retrieval via local semantic bridging and global importance aggregation.

(iv) **Multi-step method** interleaves retrieval and reasoning process, using intermediate reasoning states to generate next-step queries and gather supporting evidence for complex reasoning. The compared methods include both multi-step RAG (IRCoT, Search-R1, and R1-Searcher) and multi-step GraphRAG (HippoRAG+IRCoT, GFM-RAG+IRCoT, LogicRAG, and KG-Translator) approaches:

- **IRCoT** (Trivedi et al., 2023) interleaves chain-of-thought generation with retrieval, where each intermediate reasoning step guides the next retrieval to collect supporting passages as contextual evidence for multi-step question answering in knowledge-intensive scenarios.
- **Search-R1** (Jin et al., 2025) trains LLMs with Group Relative Policy Optimization (GRPO) to enable multi-step search and evidence integration in complex reasoning, using final-answer correctness as the reward function while selectively masking retrieved-token loss.
- **R1-Searcher** (Song et al., 2025a) optimizes with modified REINFORCE++ using two-stage rewards, unlike Search-R1 with final-answer correctness: Stage 1 rewards retrieval and format. Stage 2 replaces retrieval with answer-F1 rewards for evidence utilization.
- **HippoRAG+IRCoT** (Gutiérrez et al., 2024) equips HippoRAG with IRCoT for interleaved multi-step retrieval and reasoning, progressively improving multi-hop evidence acquisition.
- **GFM-RAG+IRCoT** (Luo et al., 2025) adopts HippoRAG's graph construction, employing a query-dependent Graph Foundation Model (GFM) to rank evidence via query-guided message passing. It can leverage IRCoT for more effective multi-step passage retrieval.
- **LogicRAG** (Chen et al., 2026a) decomposes queries into sub-queries and creates a DAG. It topologically sorts them, merges same-rank sub-queries into a unified query, and uses rolling memory with pruning. As needed, it iteratively adds sub-queries and dependencies.
- **KG-Translator** (Liu et al., 2026) constructs a reliable parsing-based hypergraph through lightweight NER and syntactic parsing, translates queries into graph-level clues through constrained decoding, and thereby ranks the corresponding original passages for retrieval.

D Additional Experiments

In this section, we provide additional experiments and analyses to comprehensively demonstrate the effectiveness and efficiency of G-Speculator. We report more detailed evaluation results of ablation experiments, hyperparameter experiments and experiments with different models in Appendix I.

D.1 Hyperparameter Sensitivity Analysis

We conduct four hyperparameter ablation experiments on four datasets to comprehensively evaluate the sensitivity of G-Speculator. As shown in Figure 5, we evaluate three hyperparameters (α, β, δ) for graph speculative retrieval stage and one hyperparameter (k) for retrieval path correction stage.

Obs. 7. G-Speculator maintains stable QA performance across most hyperparameter settings. Except for δ=1, where F1 drops by up to 5% (ScaleQA), fluctuations under all other settings remain around 1%, with a minimum of 0. Specifically, (i) Clue exploration width (α): increasing exploration does not yield consistent gains, as local exploration on HypoKG is constrained by query-graph relation paths, making exploration precise. (ii) Number of returned trajectories (β): similarly, precise

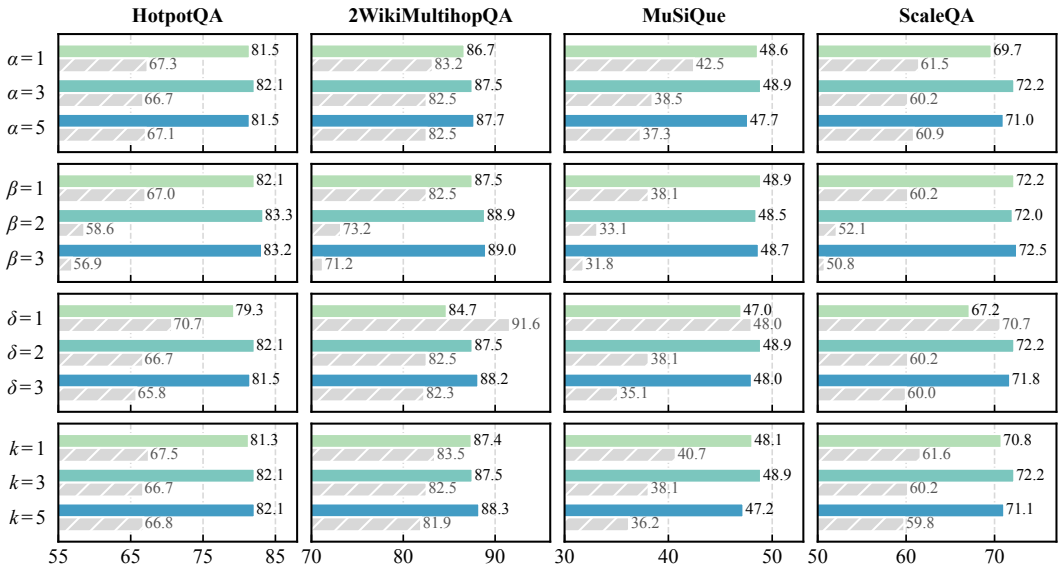


Figure 5: **Hyperparameter ablation of G-Speculator across four datasets.** Each row corresponds to one hyperparameter (α: clue exploration width, β: number of returned trajectories, δ: clue speculation depth, k: number of additional passages). The x-axis denotes F1 (blue) and Precision (gray) score (%), and the y-axis lists the tested values, where $\alpha, k \in \{1, 3, 5\}$ and $\beta, \delta \in \{1, 2, 3\}$.

exploration ensures that the most relevant trajectories are ranked first. (iii) Clue speculation depth (δ): excessively deep speculation suffers from bridge-entity error propagation, which necessitates retrieval path correction to alleviate the limitations of graph-only retrieval. (iv) Number of additional passages (k): graph speculative retrieval and path correction align query intent well with HypoKG. Consequently, additional passages yield limited gains. This indicates that G-Speculator captures multi-hop associations primarily via HypoKG, rather than relying on external dense retrieval.

Obs. 8. G-Speculator effectively captures retrieval-beneficial associations in large-scale corpora via graph speculative retrieval. Compared to the 10k corpus, G-Speculator exhibits higher sensitivity on the 35k corpus. For clue speculation depth (δ), setting it to 1 disables speculation, leaving only one initial clue exploration and forcing multi-hop retrieval to rely entirely on path correction. Even so, performance still drops by 5.0% on the large-scale corpus (δ: 2→1), indicating that graph speculative retrieval captures multi-hop associations more effectively rather than relying excessively on continuous path correction. Additionally, reducing α from 3 to 1 yields a 2.5% drop on ScaleQA, but only 0.3%–0.8% on other datasets. This suggests that larger corpora contain more complex associations, requiring broader exploration to precisely capture retrieval-relevant clues.

Furthermore, we find that G-Speculator excels at capturing the most relevant clue trajectories within 2 hops, while excessive speculation and returned clues introduce redundant evidence. For example, on HotpotQA, increasing β from 1 to 2 reduces Precision by 8.4%. For MuSiQue, deeper speculation (δ from 2 to 3) lowers Precision by 3.0%, as this dataset contains harder 3/4-hop bridge questions. Similarly, excessive additional evidence does not benefit G-Speculator, since it can already mine the most relevant clues on HypoKG. On MuSiQue, increasing k from 1 to 5 drops Precision by 4.5%.

D.2 Cross-domain Generalization

To further demonstrate the generalization of G-Speculator beyond multi-hop reasoning, we conduct experiments on GraphRAG-Bench (Xiang et al., 2026), which comprises two domain-specific datasets: G-Bench (Novel) and G-Bench (Medical). The former collects novels from the Gutenberg library to simulate real-world documents with implicit, non-linear narratives, while the latter uses NCCN medical guidelines to provide conceptual relationships such as treatment protocols linking symptoms, drugs, and outcomes. Both datasets cover four task types of increasing difficulty: fact retrieval, complex reasoning, contextual summarization, and creative generation. We evaluate on the two challenging tasks, contextual summarization and creative generation, as shown in Figure 6. Fol-

lowing GraphRAG-Bench, we report accuracy (Acc) and coverage (Cov). The former scores both semantic similarity and factual consistency against the reference answer, while the latter evaluates whether the response fully covers all knowledge relevant to the question. Observe the following:

Obs. 9. G-Speculator shows strong cross-domain generalization ability on two challenging tasks across two domains. We compare G-Speculator with HippoRAG+IRCoT, LinearRAG, and KG-Translator on contextual summarization (Sum.) and creative generation (Cre.) tasks across the Novel (Nov.) and Medical (Med.) domains, representing multi-step GraphRAG, lightweight GraphRAG, and multi-step lightweight GraphRAG, respectively. We find that our G-Speculator outperforms baselines across all metrics, achieving the largest gains of 3.13% on Acc (Nov. Sum.) and 1.82% on Cov (Med. Cre.), demonstrating its adaptability and robustness across diverse complex reasoning tasks. We attribute this to G-Speculator's ability to precisely locate relevant evidence, which provides supporting evidence while reducing noise introduced by redundant context.

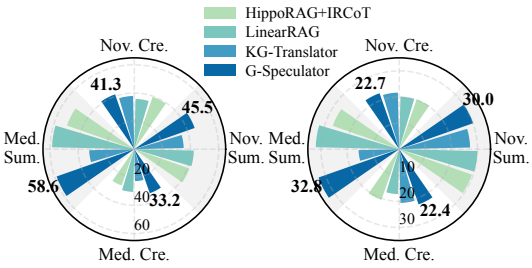


Figure 6: **QA performance (%) on the Novel and Medical domains.** Acc (left) and Cov (right).

D.3 Component-wise Efficiency Analysis

We report the detailed efficiency of each component in G-Speculator, as shown in Table 5. We measure the single-run time of each component across two corpora with different scales (10k and 35k).

Table 5: **Component-wise Efficiency (s) of G-Speculator.** → denote sub-components.

Component of Each Stage	10K	35K
Graph Speculative Retrieval		
Query Graph Construction	1.4644	1.4184
Clue Exploration	0.1104	0.1436
→ Entity Linking	0.0891	0.0949
→ Relation Selection	0.0212	0.0484
Clue Speculation	0.0001	0.0003
Retrieval Path Correction & QA		
CoT-based Analysis	2.7224	2.7894
Evidence Expansion	0.0012	0.0383
Zero-shot QA	1.2499	1.2292

Obs. 10. The components of G-Speculator exhibit stable latency across corpus scales. As shown in Table 5, (i) Graph speculative retrieval stage: Query graph construction is dominated by LLM latency and remains stable across corpus scales. Clue exploration grows slightly with scale, due to trie-based entity linking and similarity-based relation selection. The latter increases by 1.28× and can be further reduced using Approximate Nearest Neighbor (ANN) search (Malkov & Yashunin, 2018). Clue speculation has negligible latency as it only involves sequence indexing. (ii) Retrieval path correction and QA stage: latency is dominated by LLM calls, while evidence expansion can also be accelerated via ANN. Overall, despite LLM inference costs, G-Speculator remains competitive given its outstanding performance. Furthermore, with a 3.5× increase in corpus size, its retrieval latency remains stable within millisecond-level variation, demonstrating its strong scalability to large-scale corpora.

D.4 Detailed Retrieval Performance

To verify the retrieval effectiveness of G-Speculator, we report the number of contexts contained in its retrieval results and present the Recall score of its supporting evidence. The experimental results are shown in Table 6 and Figure 7, respectively. We obtain the following key findings:

Obs. 11. Without sacrificing QA performance, G-Speculator retrieves nearly the number of contexts required by the dataset. G-Speculator retrieves far more concise contexts than traditional top-k setting methods. For example, on HotpotQA, which requires only 2 contexts on average, our method retrieves 2.58, a 48.4% reduction over the top-5 setting. Notably, on the more challenging MuSiQue, G-Speculator uses 0.59 more contexts than top-5, but this is worthwhile given its strong QA performance. In the single-step setting, contexts are even fewer, near or below the requirement, yet still contribute most of the QA performance (discussed in Obs. 1). Moreover, scaling the corpus from 10k to 35k raises the ScaleQA context count by only 0.04, versus 0.54 for KG-Translator. This shows that G-Speculator retrieves on demand instead of blind traversal and forced context return.

Table 6: **Retrieved Context.** We report retrieved contexts across four datasets. ♢ is the gold context count per dataset. ♣ covers standard top-k methods (e.g., HippoRAG, LinearRAG). For summary-based or context-refined methods (e.g., MS-GraphRAG, LogicRAG), this metric is unavailable.

Method	HotpotQA			2WikiMultihopQA						MuSiQue				ScaleQA		Avg.
	$Br.^{2H}$	$Cp.^{2H}$	Avg.	$Br.^{2H}$	$Cp.^{2H}$	$In.^{2H}$	$Cp.^{4H}$	$Bc.^{4H}$	Avg.	$Br.^{2H}$	$Br.^{3H}$	$Br.^{4H}$	Avg.	$Br._{10k}^{2H}$	$Br._{35k}^{2H}$	
For Reference (gold)♢	2.00	2.00	2.00	2.00	2.00	2.00	2.00	4.00	2.47	2.00	3.00	4.00	2.86	2.00	2.00	2.36
Top-k Setting Methods♣	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
KG-Translator	3.76	3.50	3.63	3.58	3.05	3.93	2.86	4.79	3.78	4.58	4.84	4.78	4.73	3.97	4.51	4.15
G-Speculator (single-step)	**2.07**	**2.03**	**2.05**	**1.97**	**2.00**	**1.86**	**2.02**	**3.98**	**2.43**	**1.99**	**2.49**	**2.81**	**2.37**	**2.01**	**2.02**	**2.25**
G-Speculator (multi-step)	2.88	2.29	2.58	2.45	2.07	2.49	2.20	4.77	2.90	4.35	6.06	6.80	5.59	3.23	3.27	3.54

Besides evidence Precision, we also report Recall rate on MuSiQue, which is a widely recognized challenging dataset. As shown in Figure 7, G-Speculator achieves an average Recall of 56.0% on MuSiQue, outperforming multi-step methods. Combined with the QA performance in Table 1, G-Speculator further widens the QA gap, surpassing the SOTA baseline by 5.6%. This indicates that G-Speculator benefits both from the retrieval advantage of graph speculative retrieval and from careful evidence analysis during retrieval path correction. Below we conduct a detailed analysis of the retrieval stage.

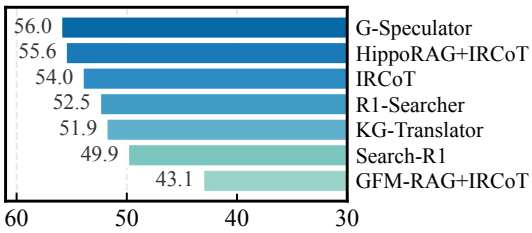


Figure 7: **Recall scores (%) of Multi-step Methods.** Experimental results from MuSiQue.

We further analyze the supporting evidence retrieved by G-Speculator to reveal the source of its retrieval performance. Specifically, supporting evidence comes from graph speculative retrieval or evidence expansion, where the latter is only triggered when retrieval stagnates. As shown in Table 7, speculative retrieval accounts for over 90% of supporting evidence across all datasets, peaking at 98.43% on 2Wiki. Moreover, even on the large-scale ScaleQA, 95.84% of supporting evidence comes from speculative retrieval. This indicates that G-Speculator achieves its QA gains through precise speculative retrieval and effective direction correction, rather than over-relying on evidence expansion.

Table 7: **Proportion and count of evidence from each retrieval stage.** We count graph speculative retrieval (GSR) and evidence expansion (EE).

Dataset	GSR	EE	Total
HotpotQA	97.01 (1461)	2.99 (45)	100.00 (1506)
2Wiki	98.43 (4695)	1.57 (75)	100.00 (4770)
MuSiQue	92.56 (1592)	7.44 (128)	100.00 (1720)
ScaleQA	95.84 (2094)	4.16 (91)	100.00 (2185)

D.5 Effectiveness of Different Step Counts

To further demonstrate the effectiveness of G-Speculator, we compare agents specially trained for multi-step retrieval and set their actions to 4/10 following the default configuration. As shown in Figure 8, we report the performance of G-Speculator on four datasets under 1 to 5 maximum steps.

Obs. 12. G-Speculator can gain continuous benefits from multi-step retrieval path correction and significantly outperforms retrieval agents with more steps. At T=1, performance is entirely from a single-step passage retrieval. At T=2, retrieval path correction takes effect, realigning query intent with HypoKG and yielding F1 gains of 7.6%, 12.5%, 12.0%, and 15.5% on the four datasets. By the fifth step, F1 stabilizes and surpasses R1-Searcher under the 10-step setting, with the largest gain of 11.8% on 2Wiki. This is because these agents still adopt vector retrieval and struggle to capture multi-hop associations in a single pass. This shows that G-Speculator benefits from complementarity between graph speculative retrieval and retrieval path correction: the former precisely locates multi-hop evidence, which enables the latter to stop at fewer steps and yield correct answers.

Moreover, corpora with more distractor passages benefit more from retrieval path correction. While HotpotQA and 2Wiki achieve relative F1 improvements of 10.2% and 16.7% from the first to the second step, MuSiQue exhibits a much larger gain of 42.9%. Similarly, from T=2 to T=3, absolute gains on HotpotQA and 2Wiki drop to merely 1.1% and 1.3%, whereas MuSiQue still maintains a

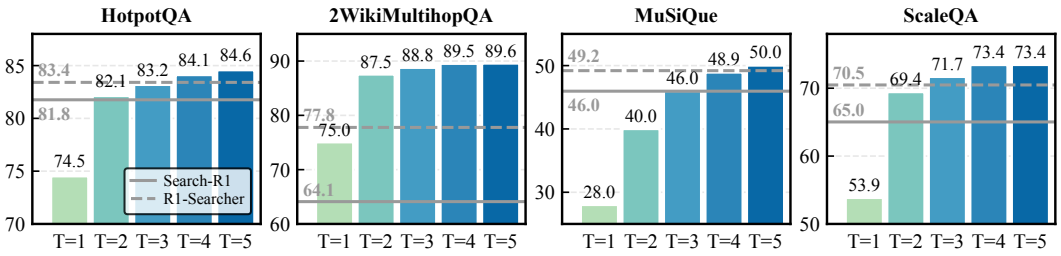


Figure 8: **QA performance (%) against retrieval agents.** For G-Speculator, we evaluate its performance across maximum steps 1–5. For Search-R1 and R1-Searcher, we use their default settings, i.e., 4 and 10 steps, respectively. The x-axis indicates the step (T), and the y-axis indicates the F1.

substantial gain of 6.0%. We find that MuSiQue contains more same-topic distractor passages, giving topic entities denser neighborhoods in the graph. This distracts graph-only retrieval methods: for example, PPR propagates importance scores across numerous neighbors, diluting the signal of useful evidence. This shows that beyond noise such as missing relations, graph structural characteristics also constrain retrieval, highlighting the importance of retrieval path correction.

Obs. 13. G-Speculator consistently delivers concise evidence under different step counts. As shown in Table 8, we report Precision across five steps on four benchmarks. On HotpotQA, 2Wiki, and ScaleQA, Precision variance remains low (0.214–0.459), indicating that the proportion of useful evidence among retrievals stays stable as steps increase. This proves that G-Speculator achieves strong QA performance while maintaining evidence conciseness, even on large-scale corpora. Differently, MuSiQue exhibits larger variance (10.551) due to challenging questions and corpus, which introduces additional noise into retrieved passages. Nevertheless, compared with baselines using fixed top-k retrieval, G-Speculator still shows advantages in robustness against such noise, indicating that its retrieval always centers on the most relevant clues.

Table 8: **Precision scores (%) across five steps and the corresponding variance (Var.).**

Dataset	T=1	T=2	T=3	T=4	T=5	Var.
HotpotQA	65.27	66.73	66.49	67.06	66.47	0.459
2Wiki	81.44	82.54	82.03	81.59	81.51	0.214
MuSiQue	45.80	40.57	39.38	38.10	37.81	10.551
ScaleQA	59.16	60.67	60.36	59.90	59.39	0.403

D.6 Effectiveness of Different LLM Backbones

To demonstrate the adaptability of G-Speculator to different-type LLMs, we equip it with LLMs from different families, including GPT-4o-mini (OpenAI, 2024), Qwen-3.6-flash (Yang et al., 2025) and Gemini-3.5-flash-lite[1], and test the QA performance on four datasets, as shown in Figure 9.

Obs. 14. G-Speculator adapts to diverse LLMs and benefits from newer versions. We evaluate G-Speculator on three LLMs of different release dates: GPT-4o-mini (July 2024), Qwen-3.6-Flash (April 2026), and Gemini-3.5-Flash-Lite (July 2026). G-Speculator delivers stable performance across all four datasets, with newer models bringing further gains. For example, Gemini-3.5-Flash-Lite improves F1 by 2.6%, 3.4%, 5.4%, and 2.5% over GPT-4o-mini on the four datasets, respectively. This demonstrates that G-Speculator adapts well to different LLMs, and its graph speculative retrieval and retrieval path correction do not depend on a specific model. Notably, as newer models lack careful evaluation on GraphRAG systems, we keep GPT-4o-mini as the default setting.

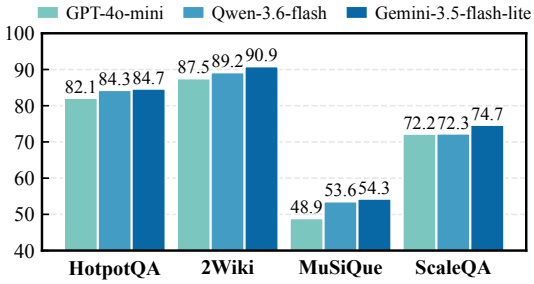


Figure 9: **F1 scores (%) of G-Speculator with three LLM backbones across four datasets.**

[1] https://deepmind.google/models/model-cards/gemini-3-5-flash-lite/

D.7 Effectiveness of Different Entity Linking Models

Generative entity linking has been shown to outperform similarity-based approaches (Cao et al., 2021) and works well in GraphRAG systems (Liu et al., 2026). It leverages constrained decoding to autoregressively generate token-level entity sequences for faithful entity linking. To demonstrate performance differences across entity linking models, we compare (i) Embedding-based model: all-MiniLM-L6-v2, which links entities via semantic similarity, and (ii) Generative model: Qwen-2.5-0.5B and Llama-3.2-1B, trained on data from KG-Translator. To clearly show the impact of these models, we conduct experiments under the single-step setting. Results are shown in Table 9.

Table 9: **F1 and Precision (%) of G-Speculator with different entity linking model across four datasets.**

Model	HQA		2Wiki		MSQ		SQA	
	F1	P	F1	P	F1	P	F1	P
Embedding-based Entity Linking								
all-MiniLM-L6-v2	73.6	65.0	72.1	79.2	24.8	42.7	49.6	57.3
Generative Entity Linking								
Qwen-2.5-0.5B	74.3	64.4	74.9	80.6	26.1	43.8	52.1	57.6
Llama-3.2-1B	**74.5**	**65.3**	**75.0**	**81.4**	**28.0**	**45.8**	**53.9**	**59.2**

Obs. 15. G-Speculator leverages generative entity linking models for better evidence localization. During clue exploration, G-Speculator uses triples in the query graph for entity linking without explicit NER. Generative entity linking performs linking via end-to-end decoding, allowing it to leverage parametric knowledge to resolve ambiguity and achieve better localization. For example, on MuSiQue, the embedding-based model underperforms the generative model (Llama) by 3.2% F1 and 3.1% Precision. Moreover, when switching to the smaller, different-family Qwen model, performance drops range from 0.1% to 2%. This indicates that G-Speculator is insensitive to the choice of generative entity linking model, and such models are better suited for graph-level clue localization.

D.8 Effectiveness of Different Sentence Embeddings

To investigate how sentence embedding model representational capacity affects G-Speculator, we experiment with different models, including all-MiniLM-L6-v2, gte-Qwen2.5-1.5B-Instruct (Li et al., 2023) and NV-Embed-v2 (Lee et al., 2025). Results are reported in Figure 10.

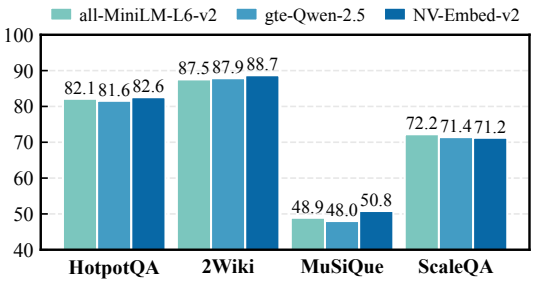


Figure 10: **F1 scores (%) of G-Speculator with three embedding models across four datasets.**

Obs. 16. G-Speculator is insensitive to the choice of sentence embedding model. When replacing all-MiniLM-L6-v2 (22.7M) with gte-Qwen-2.5-Instruct (1.5B), we observe changes within 1.0%, and even decreases on HotpotQA, MuSiQue, and ScaleQA. Switching to the more capable NV-Embed-v2 (7B) raises performance slightly on HotpotQA, 2Wiki, and MuSiQue by 0.5%, 1.2%, and 1.9%, respectively. This indicates that the overall performance of G-Speculator mainly comes from the speculative retrieval rather than the representation capability. We adopt all-MiniLM-L6-v2 as the default due to its efficiency.

D.9 Case Study

To illustrate how G-Speculator operates, we present an in-depth qualitative analysis of two representative examples, shown in Tables 10 and 11. The first shows single-step graph speculative retrieval with no correction, while the second illustrates how path correction realigns intent with HypoKG.

Case 1: Direct graph speculative retrieval for multi-hop reasoning. As shown in Table 10, we present a case where single-step graph speculative retrieval achieves multi-hop retrieval. For the question "Aerodynamik is a Kraftwerk song on an album recorded for what anniversary of the Tour de France bicycle race?", two sub-questions must be resolved: (i) Which Kraftwerk album contains the song "Aerodynamik"? (ii) For which anniversary of the Tour de France was this album recorded? During graph speculative retrieval, G-Speculator finds clue c_1 based on τ_1 and speculates the correct bridge entity "Tour de France Soundtracks". Then τ_2 is filled by this entity, and clue c_2 is found.

Both clues trace back to passages p_5 and p_6, exactly the two supporting evidence required. Thus, CoT-based evidence analysis directly yields the correct final answer "100th anniversary".

Case 2: Corrected graph speculative retrieval for multi-hop reasoning. Table 11 shows a multi-step case. For "Where was the wife of Otto I, Count of Guelders born?", two sub-questions arise: (i) The identity of his wife, and (ii) Her birthplace. G-Speculator first obtains clues c_1 and c_2 via graph speculative retrieval, tracing to passages p_7 and p_8. Although the correct bridge entity "Richardis" is found, this entity is ambiguous and appears with different meanings across passages, making c_2 a wrong clue (p_8 is irrelevant). During CoT-based analysis, the LLM accurately identifies insufficient evidence and generates a follow-up query "Where was Richardis of Bavaria born?" to adjust direction. This allows re-searching for clues: Using the unambiguous entity "Richardis of Bavaria", the second supporting evidence p_9 is retrieved, yielding the correct answer "Kelheim".

D.10 Error Analysis

We analyze two error cases of each module in G-Speculator, as shown in Tables 12 and 13, illustrating its potential limitations: (i) Biased clue localization, and (ii) Evidence sufficiency misjudgment.

Case 3: Biased clue localization in graph speculative retrieval. Table 12 shows successive biased evidence localization for a multi-hop question. In the first speculative retrieval, the query mistakenly enters a song named "Butterfly" rather than the insect, causing two failed clue explorations. After LLM analysis, although "the seventh-largest country is India" is identified, the second speculative retrieval still erroneously enters stamp-related passages, preventing G-Speculator from correct retrieval under the two-step setting. Notably, dense-retrieval-based evidence expansion during retrieval stagnation alleviates such issues, and more steps can further mitigate this problem.

Case 4: Evidence sufficiency misjudgment in retrieval path correction. As shown in Table 13, for the question "A relative of the emu is native to which continent?", two sub-questions should be resolved: (i) What is a relative of the emu? (ii) Which continent is that relative native to? During speculative retrieval, the question is not correctly decomposed into two triples, only evidence p_4 about the emu is returned. As p_4 explicitly identifies the ostrich as a relative of the emu, the system should then look for evidence on "where the ostrich is native". However, the LLM incorrectly concludes during analysis that "since the ostrich is a relative of the emu, their native continents are identical", causing insufficient retrieval and a wrong answer. We argue that such errors can be mitigated in the future by leveraging LLMs for careful logical reasoning and fact-checking.

E Setup of Preliminary Study

In Section 2, we investigate evidence reachability across three KGs. In addition, we provide an explanation of evidence reachability in Appendix A.2. We present the experimental setup below:

Dataset & Corpus Selection. We select HotpotQA, 2Wiki, and MuSiQue datasets along with their corresponding corpora listed in Table 4. The measurement of evidence reachability is conducted on the 2-hop bridge questions from these datasets, as these questions exhibit a clear hop structure.

KG & Evidence Chain Construction. We first extract triples from the corpora of three multi-hop datasets and construct KGs. The prompt for triple extraction follows HippoRAG (Gutiérrez et al., 2024). Then, we use QA pairs from PER-QA (Liu et al., 2025) and select 2-hop bridge questions as samples. For each QA pair, PER-QA provides steps consisting of sub-questions and sub-answers. We further use an LLM to extract query entities from each sub-question and combine them with sub-answers into (entity, entity) pairs, forming a coherent evidence chain. For example, for the question "When was Juan Manuel Morales's father born?", the evidence chain is (`"Juan Manuel Morales"`, `"Carlos María Morales"`, `"1 March 1970"`). Each question corresponds to one evidence chain.

Evidence Reachability Calculation. We match the shortest path on the KG that fully covers this chain. We only count paths reachable within 3 hops for each pair. All other cases are treated as unreachable. For a 2-hop bridge question, the second evidence depends on the first. Our statistical assumption is as follows: in a single retrieval run, if the first evidence is unreachable on the KG, the second evidence is deemed unreachable. Notably, while direct retrieval of the second evidence is possible in practice, we adhere to this assumption in experiments to ensure evidence completeness.

Table 10: **Case 1: Direct graph speculative retrieval for multi-hop reasoning (single-step).**

QUESTION	Aerodynamik is on an album recorded for which Tour de France anniversary?
ANSWER	100th anniversary
▶ **SINGLE-STEP RETRIEVAL PROCESS** (Olive: Clue; Teal: Speculated bridge entity)	
↪ QUERY GRAPH	τ_1: ("Aerodynamik", "is song on album", "<A>") τ_2: ("<A>", "recorded for anniversary of Tour de France", "<B>")
EXPLORATION	c_1: ("Aerodynamik", "from their tenth studio album") $\Rightarrow \omega(c_1 \mid \tau_1)$: 0.76
SPECULATION	s_{230}: ["e_{38744}: Aerodynamik", "r_{230_0}: was released on", "e_{24342}: 15 Mar 2004", "r_{230_1}: from their tenth studio album", "e_{64733}: Tour de France Soundtracks"] $\Rightarrow$ <A>: "Tour de France Soundtracks"
EXPLORATION	c_2: ("Tour de France Soundtracks", "was recorded for the 100th anniversary of") $\Rightarrow \omega(c_2 \mid \tau_2^\sigma)$: 0.74
SPECULATION	s_{238}: ["e_{22743}: Tour de France Soundtracks", "r_{238_0}: was recorded for the 100th anniversary of", "e_{58131}: Tour de France bicycle race", "r_{238_1}: although it missed its intended release date for the actual tour"] $\Rightarrow$ <B>: "Tour de France bicycle race"
RETRIEVALS	$s_{230} \rightarrow p_5$ **(Aerodynamik)**: ... their tenth album, Tour de France Soundtracks ... $s_{238} \rightarrow p_6$ **(Tour de France Soundtracks)**: ... Tour de France Soundtracks was recorded for the 100th anniversary of Tour de France bicycle race ...
COT ANALYSIS	... So the response is: {"final_answer": "100th anniversary", "next_query": None}
PREDICTION	100th anniversary (✔)

Table 11: **Case 2: Corrected graph speculative retrieval for multi-hop reasoning (multi-step).**

QUESTION	Where was the wife of Otto I, Count Of Guelders born?
ANSWER	Kelheim
▶ **MULTI-STEP RETRIEVAL PROCESS** (Olive: Clue; Teal: Speculated bridge entity)	
↪ QUERY GRAPH	τ_1: ("Otto I, Count Of Guelders", "wife is", "<A>") τ_2: ("<A>", "born location is", "<B>")
EXPLORATION	c_1: ("Otto I, Count of Guelders", "He married") $\Rightarrow \omega(c_1 \mid \tau_1)$: 0.78
SPECULATION	s_{108}: ["e_{26789}: Otto I", "r_{108_0}: He married", "e_{5292}: Richardis", "r_{108_1}: of", "e_{59976}: Bavaria", "r_{108_2}: in", "e_{49811}: 1184"] $\Rightarrow$ <A>: "Richardis"
EXPLORATION	c_2: ("Richardis", "The city of") $\Rightarrow \omega(c_2 \mid \tau_2^\sigma)$: 0.62
SPECULATION	s_{6408}: ["e_{41724}: Jülich", "r_{6408_0}: The city of", "e_{46867}: Aachen", "r_{6408_1}: was later ordered to pay a compensation to", "e_{5292}: Richardis"] $\Rightarrow$ <B>: "Jülich"
RETRIEVALS	$s_{108} \rightarrow p_7$ **(Otto I, Count of Guelders)**: ... married Richardis of Bavaria in 1184 ... $s_{6408} \rightarrow p_8$ **(William IV, Count of Jülich)**: ... The city of Aachen was later ...
COT ANALYSIS	... not provide any information about Richardis's birthplace ... So the response is: {"final_answer": None, "next_query": "Where was Richardis of Bavaria born?"}
NEXT QUERY	Where was Richardis of Bavaria born?
↪ QUERY GRAPH	τ_3: ("Richardis of Bavaria", "born in place is", "<C>")
EXPLORATION	c_3: ("Richardis of Bavaria", "in") $\Rightarrow \omega(c_3 \mid \tau_3)$: 0.64
SPECULATION	s_{106}: ["e_{52201}: Richardis of Bavaria", "e_{33889}: 1173", "r_{106_0}: in", "e_{19029}: Kelheim", "r_{106_1}: was a", "e_{10767}: German noblewoman"] $\Rightarrow$ <C>: "Kelheim"
RETRIEVALS	(p_7, p_8 from the first graph speculative retrieval) $s_{106} \rightarrow p_9$ **(Richardis of Bavaria)**: Richardis of Bavaria (1173 in Kelheim) was a German noblewoman. She was a daughter of Count Palatine Otto I of Bavaria ...
COT ANALYSIS	... So the response is: {"final_answer": "Kelheim", "next_query": None}
PREDICTION	Kelheim (✔)

Table 12: **Case 3: Biased clue localization in graph speculative retrieval.**

QUESTION	What is the scientific name of the butterfly widespread in the 7th-largest country?
ANSWER	I. helicon
▶ **RETRIEVAL PROCESS** (Olive: Clue; Teal: Speculated bridge entity)	
↪ QUERY GRAPH	τ_1: ("butterfly", "widespread in country", "<A>") τ_2: ("<A>", "seventh-largest country is", "<B>")
EXPLORATION	c_1: ("Butterfly", "is a song by") $\Rightarrow \omega(c_1 \mid \tau_1)$: 0.56
SPECULATION	s_{1195}: ["e_{67715}: Butterfly", "r_{1195_0}: is a song by", "e_{42893}: American", "r_{1195_1}: singer and songwriter", "e_{43715}: Mariah Carey"] $\Rightarrow$ <A>: "American"
EXPLORATION	c_2: ("United States of America", "also known as") $\Rightarrow \omega(c_2 \mid \tau_2^g)$: 0.56
SPECULATION	s_{532}: ["e_{11124}: Procter & Gamble Co.", "r_{532_0}: also known as", "e_{39877}: P&G", "r_{532_1}: is an", "e_{42893}: American"] $\Rightarrow$ <B>: "Procter & Gamble Co."
RETRIEVALS	$s_{1195} \rightarrow p_1$ **(Butterfly)**: Butterfly is a song by American singer and songwriter Mariah Carey from her sixth studio album of the same name ... $s_{532} \rightarrow p_2$ **(Procter & Gamble Co.)**: Procter & Gamble Co., also known as P&G, is an American consumer goods corporation headquartered in downtown ...
COT ANALYSIS	... seventh-largest country is India ... not contain any information about butterflies or their scientific names. So the response is: {"final_answer": None, "next_query": "What is the scientific name of a butterfly that is widespread in India?"}
NEXT QUERY	What is the scientific name of a butterfly that is widespread in India?
↪ QUERY GRAPH	τ_3: ("butterfly that is widespread in India", "scientific name is", "<C>")
EXPLORATION	c_3: ("The Indian Postal Department", "includes") $\Rightarrow \omega(c_3 \mid \tau_3)$: 0.52
SPECULATION	s_{2627}: ["e_{26183}: The Indian Postal Department", "r_{2627_0}: includes", "e_{26183}: Azad Hind Stamps", "r_{2627_1}: in its book"] $\Rightarrow$ <C>: "Azad Hind Stamps"
RETRIEVALS	(p_1, p_2 from the first graph speculative retrieval) $s_{2627} \rightarrow p_3$ **(Azad Hind stamps)**: The Azad Hind Stamps are a set of ... The Indian Postal Department includes Azad Hind Stamps in its commemorative book ...
COT ANALYSIS	... seventh-largest country is India ... not contain any information about butterflies or their scientific names. So the response is: {"final_answer": None, "next_query": "What is the scientific name of a butterfly that is widespread in India?"}
PREDICTION	No evidence found (✗)

Table 13: **Case 4: Evidence sufficiency misjudgment in retrieval path correction.**

QUESTION	A relative of the emu is native to which continent?
ANSWER	Africa
▶ **RETRIEVAL PROCESS** (Olive: Clue; Teal: Speculated bridge entity)	
↪ QUERY GRAPH	τ_1: ("emu", "relative is native to continent", "<A>")
EXPLORATION	c_1: ("Emu", "It is endemic to") $\Rightarrow \omega(c_1 \mid \tau_1)$: 0.68
SPECULATION	s_{1314}: ["e_{65600}: Emu", "r_{1314_0}: It is endemic to", "e_{14780}: Australia", "r_{1314_1}: where it is the largest native bird"] $\Rightarrow$ <A>: "Australia"
RETRIEVALS	$s_{1314} \rightarrow p_4$ **(Emu)**: The emu ("Dromaius novaehollandiae") is the second-largest living bird by height, after its ratite relative, the ostrich. It is endemic to Australia where it is the largest native bird. The emu's range covers most of mainland ...
COT ANALYSIS	... The evidence states that the emu is endemic to Australia. It also mentions that the emu is a ratite relative of the ostrich. Since the emu is native to Australia, we can conclude that its relatives, including the ostrich, are also associated with Australia ... So the response is: {"final_answer": "Australia", "next_query": None}
PREDICTION	Australia (✗)

F Implementation Details

In this section, we introduce the implementation details of G-Speculator and all baselines.

F.1 Implementation Details of G-Speculator

Graph Construction. We adopt the spaCy model (en_core_web_trf[2]) to perform corpus preprocessing, whose native components handle sentence segmentation and NER in a single pass. Specifically, in the NER stage, we exclude `ORDINAL` and `CARDINAL` entities given their limited semantic significance. In the passage serialization stage, passages are serialized by splitting sentences at named entities, producing alternating entity-relation sequences, where relations are text spans without entities, effectively capturing key semantics. Notably, empty elements are removed afterward, yielding non-strict alternation between entities and relations. Additionally, as generative entity linking depends on token-level entity sequences, we use MarisaTrie to store them efficiently. MarisaTrie is a compressed, memory-efficient prefix tree (trie) optimized for fast string storage and querying.

Retrieval & QA Settings. For all datasets, we use the same hyperparameters: α=3, β=1, δ=2, and k=3. For entity linking, we adopt a trained Llama-3.2-1B-Instruct following the training data of KG-Translator (Liu et al., 2026), which autoregressively generates target entities. We employ all-MiniLM-L6-v2[3] to obtain semantic embeddings. Additionally, gpt-4o-mini-2024-07-18 is used for query graph construction, retrieval path correction, and QA. Since some questions are answered during path correction, only unanswered samples proceed to QA. We conduct the experiments on a single RTX 5090 32G GPU. The prompts used in the experiments are shown in Appendix K.

Evaluation Metrics. For the four multi-hop datasets (HotpotQA, 2Wiki, MuSiQue, and ScaleQA), we adopt F1 score to evaluate QA performance and Precision to evaluate retrieval performance. For GraphRAG-Bench (Xiang et al., 2026), following its setup, we use Accuracy (Acc) and Coverage (Cov) to evaluate generation quality on contextual summarization and creative generation tasks. For metric calculation, we adopt the default weight parameter of 0.75, with bge-large-en-v1.5[4] as the embedding model and GPT-4o-mini as the generative judge model. Acc scores both semantic similarity and factual consistency against the reference answer, whereas Cov assesses whether all necessary knowledge appear in the final answer rather than measuring simple overlap.

F.2 Implementation Details of Baselines

All baselines share the same embedding model (all-MiniLM-L6-v2) and LLM (gpt-4o-mini-2024-07-18). For GFM-RAG, we retain its open-source all-mpnet-base-v2 embedding model, as it is aligned with its graph foundation model during training. All methods use a common top-5 retrieval and share a unified QA prompt for fair comparison. For multi-step methods, we allow retrieving the top-5 passages at each step and collect all retrieved passages before QA to obtain the final top-5 passages. The descriptions and settings of key baseline hyperparameters are provided in Appendix J.

G Detailed Related Work

In this section, we provide a detailed review of related work, covering (i) Retrieval-augmented generation (RAG) and (ii) Graph-enhanced retrieval-augmented generation (GraphRAG). The latter is discussed along two research lines: graph construction and graph-enhanced reasoning.

G.1 Retrieval-Augmented Generation

Large language models (LLMs) are prone to hallucination when required knowledge is missing from parametric memory (Huang et al., 2025; Zhang et al., 2025b; Wu et al., 2026; Alansari & Luqman, 2026). Retrieval-augmented generation (RAG) mitigates this issue by grounding generation in relevant external evidence (Borgeaud et al., 2022; Gao et al., 2023; Oche et al., 2025; Song et al., 2025b). However, traditional RAG systems overlook connections among evidence distributed

[2] `https://huggingface.co/spacy/en_core_web_trf`

[3] `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`

[4] `https://huggingface.co/BAAI/bge-large-en-v1.5`

across documents. Some works interleave retrieval with reasoning for multi-hop knowledge acquisition to mitigate this issue. For instance, Self-Ask (Press et al., 2023) decomposes questions into follow-up queries, IRCoT (Trivedi et al., 2023) retrieves evidence through CoT steps, and Self-RAG (Asai et al., 2024) reflects on retrieval needs and evaluates passages. Recently, this capability has been internalized into LLMs. Search-R1 (Jin et al., 2025) extends reasoning LLMs with reinforcement learning, enabling the model to autonomously generate multi-turn search queries during step-by-step reasoning. It adopts Group Relative Policy Optimization (GRPO) to optimize search-augmented reasoning trajectories and employs retrieved token masking to stabilize training, relying solely on a simple outcome-based reward. R1-Searcher (Song et al., 2025a) proposes a two-stage outcome-based RL framework: the first stage uses retrieval and format rewards to teach the model to invoke external retrieval, while the second stage replaces retrieval rewards with answer-F1 rewards to incentivize effective evidence utilization. Despite these advances, existing RAG still struggles to capture fine-grained associations in corpora, which hinders precise retrieval and question answering.

G.2 Graph-enhanced Retrieval-Augmented Generation

Graph-enhanced RAG (GraphRAG) systems address this limitation by organizing corpora into graph structures (Peng et al., 2025; Zhang et al., 2025a), as graph structures are better at capturing the associative relationships between passages within corpora, thereby providing navigation for multi-hop reasoning. In the following, we discuss two major research directions related to our work, which focus on graph construction and graph-enhanced reasoning. Specifically, we summarize as follows:

Graph Construction. Existing studies propose diverse structures adapted for reasoning, considering granularity, efficiency, and other perspectives. Early works organize corpora into passage-level graphs or hierarchical structures. For example, KGP (Wang et al., 2024) builds document graphs using hyperlinks and semantic similarity, while RAPTOR (Sarthi et al., 2024) recursively clusters and summarizes passages. MS-GraphRAG (Edge et al., 2024) and LightRAG (Guo et al., 2025b) construct entity-centric graphs with community-level and dual-level structures, respectively. Subsequent works, including HippoRAG (Gutiérrez et al., 2024), HippoRAG2 (Gutiérrez et al., 2025), and GFM-RAG (Luo et al., 2025), use open information extraction (OpenIE) to extract triples and construct knowledge graphs (KGs). More recently, research has shifted toward lightweight graph construction. LinearRAG (Zhuang et al., 2026) and KG-Translator (Liu et al., 2026) use lightweight models to extract named entities, avoiding explicit relation extraction. The former constructs graphs based on entity co-occurrence in sentences, while the latter builds directed hypergraphs from syntax and entity-masked sentences. Despite these advances, graph construction remains prone to missing or erroneous relations and ambiguous entities, making it challenging to obtain high-quality graphs.

Graph-enhanced Reasoning. Another line of research focuses on mining knowledge associations over constructed graphs. These methods can be mainly categorized into two groups: (i) *Score-driven ranking*: These methods score nodes, relations, or subgraphs via graph algorithms such as personalized PageRank (PPR) or graph neural networks (GNNs) to capture multi-hop evidence (He et al., 2024; Gutiérrez et al., 2024; Li et al., 2025; Gutiérrez et al., 2025; Luo et al., 2025; Zhuang et al., 2026). For example, HippoRAG2 (Gutiérrez et al., 2025) and LinearRAG (Zhuang et al., 2026) apply PPR over graphs to propagate node weights and identify key passages. GFM-RAG (Luo et al., 2025) employs a query-dependent graph foundation model for multi-hop retrieval. (ii) *Agent-driven exploration*: This paradigm leverages LLMs to navigate graphs and identify reasoning paths (Sun et al., 2024; Wang et al., 2024; Ma et al., 2025; Liang et al., 2025; Liu et al., 2026). For instance, KGP (Wang et al., 2024) employs an LLM to incrementally traverse a KG and collect multi-hop evidence. KG-Translator (Liu et al., 2026) leverages an LLM to translate queries into clues via constrained decoding, guiding exploration over graphs. In addition, GraphRAG is also often equipped with multi-step settings, exhibiting complementary benefits (Gutiérrez et al., 2024; Luo et al., 2025; Guo et al., 2025a; Liu et al., 2026; Chen et al., 2026a). For instance, HippoRAG (Gutiérrez et al., 2024) and GFM-RAG (Luo et al., 2025) integrate IRCoT for better performance, whereas KG-Translator (Liu et al., 2026) adopts a Self-Ask-like strategy. However, existing GraphRAG methods are bounded by low-quality, static graphs, making them difficult to adapt to diverse queries. Recent works attempt to decompose queries into graphs to guide complex reasoning (Liu et al., 2025; Chen et al., 2026a), but this introduces error propagation across interdependent sub-queries and overlooks valuable knowledge associations within the corpus. This motivates G-Speculator, which does not probe the graph step by step but first infers the most likely reasoning path from existing evidence, then verifies and corrects it with retrieval results, enabling accurate passage retrieval and QA.

H ALGORITHM OF G-SPECULATOR

In this section, we present algorithmic descriptions of G-Speculator's two components.

Algorithm 1 HypoKG Construction

Require: Passage set $\mathcal{P}$, spaCy model `spaCy`
Ensure: HypoKG $\mathcal{H} = (\mathcal{E}, \mathcal{R}, \phi)$

$\mathcal{E} \leftarrow \varnothing,\ \mathcal{R} \leftarrow \varnothing$ ▷ Initialize global sets
$\mathcal{S} \leftarrow$ `spaCy`.split_by_punctuation($\mathcal{P}$) ▷ Split passages into sentences by punctuation
for $s \in \mathcal{S}$ **do**
  $\mathcal{E}_s \leftarrow$ `spaCy`.ner(s) ▷ Extract entities from sentence s
  $\mathcal{R}_s \leftarrow s$.split_by_entity($\mathcal{E}_s$) ▷ Split s by entities; output fragments are hyperedges
  $s \leftarrow s$.to_sequence($\mathcal{E}_s, \mathcal{R}_s$) ▷ Serialize sentence
  $\mathcal{E} \leftarrow \mathcal{E} \cup \mathcal{E}_s$ ▷ Update global entity node set
  $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{R}_s$ ▷ Update global hyper-edge set
  $\phi(r) = \mathcal{E}_s,\ \forall r \in \mathcal{R}_s$ ▷ Each hyperedge maps to all entities in sentence s
end for
return $\mathcal{H} = (\mathcal{E}, \mathcal{R}, \phi)$

Algorithm 2 Speculation-Driven Passage Retrieval & QA

Require: Query q, HypoKG $\mathcal{H}$, Clue exploration width α, Number of returned trajectories β, Clue speculation depth δ, Number of additional passages k, Max steps T, General LLM `LLM`
Ensure: Final answer a^*

$q'_0 \leftarrow q,\ \mathcal{P}'_{\leq 0} \leftarrow \varnothing$ ▷ Initialize query and accumulated evidence
for $t = 1, 2, \ldots, T$ **do**
  /* Graph Speculative Retrieval: `SPEC`(q'_{t-1}) */
  $\mathcal{G} \leftarrow \{\tau_0, \tau_1, \ldots, \tau_{n-1}\} \leftarrow$ `LLM`.parse(q'_{t-1}) ▷ Construct query graph $\mathcal{G}$
  $\mathcal{E}^* \leftarrow \varnothing$ ▷ Accumulated speculated bridge entities
  for $\tau_i \in \mathcal{G}$, $\mathcal{G}$.get_depth(τ_i) $\leq \delta$ **do**
    $\mathcal{C}_i \leftarrow \mathrm{Top}_\alpha$(explore_clue($\tau_i, \mathcal{E}^*; \mathcal{H}$)) ▷ Select top-$\alpha$ clues
    for $c \in \mathcal{C}_i$ **do**
      $\mathcal{E}^* \leftarrow \mathcal{E}^* \cup \{$speculate_bridge($c; \mathcal{H}$)$\}$ ▷ Speculate bridge entity
    end for
  end for
  $\Pi \leftarrow$ build_trajectory($\mathcal{C}_0, \ldots, \mathcal{C}_{n-1}$) ▷ Assemble clue trajectories by topology
  $\Pi \leftarrow \mathrm{Top}_\beta(\Pi, \omega(\pi))$ ▷ Select top-β trajectories
  $\mathcal{P}'_t \leftarrow$ map_passage(Π) ▷ Map clue trajectories to passages
  /* Retrieval Path Correction: `CORR`$(q, \mathcal{P}'_{\leq t})$ */
  $\mathcal{P}'_{\leq t} \leftarrow \mathcal{P}'_{\leq t-1} \cup \mathcal{P}'_t$ ▷ Accumulate retrieved passages
  $(a^*_t,\ q'_t) \leftarrow$ `LLM`.think($q, \mathcal{P}'_{\leq t}$) ▷ Exactly one of a^*_t, q'_t non-empty
  if $a^*_t \neq \varnothing$ **then**
    return a^*_t
  end if
  if $q'_t = q'_{t-1}$ **then** ▷ Retrieval stagnation
    $\mathcal{P}'_{\leq t} \leftarrow \mathcal{P}'_{\leq t-1}$ ▷ Discard $\mathcal{P}'_t$, rollback
    $\mathcal{P}_{\text{dense}} \leftarrow$ expand_evidence(q'_t, k) ▷ Complementary top-k dense retrieval
    $\mathcal{P}'_{\leq t} \leftarrow \mathcal{P}'_{\leq t} \cup \mathcal{P}_{\text{dense}}$
  end if
end for
$a^* \leftarrow$ `LLM`.zero_shot_qa($q, \mathcal{P}'_{\leq T}$) ▷ Generate answer if T reached
return a^*

I Detailed Performance Reports

We further report detailed QA (F1) and retrieval (Precision) performance across all question types to facilitate future research, including: (i) Ablation studies (Tables 14 and 17); (ii) Hyperparameter experiments (Tables 15 and 18); (iii) Experiments with different models (Tables 16 and 19).

Table 14: **Detailed QA performance (F1, %) of ablation study.**

Variant	HotpotQA			2WikiMultihopQA						MuSiQue				ScaleQA		Avg.
	Br.^{2H}	Cp.^{2H}	Avg.	Br.^{2H}	Cp.^{2H}	In.^{2H}	Cp.^{4H}	Bc.^{4H}	Avg.	Br.^{2H}	Br.^{3H}	Br.^{4H}	Avg.	Br.$^{2H}_{10k}$	Br.$^{2H}_{35k}$	
Single-step Setting																
G-Speculator (Full)	61.96	87.08	74.52	63.18	93.89	58.49	92.06	80.37	75.05	40.13	20.18	21.14	28.01	55.09	53.85	59.20
w opposite spec.	57.50	87.28	72.39	40.98	93.39	46.15	92.06	71.57	64.73	35.16	20.35	18.28	25.50	44.55	42.94	51.84
w/o idf	60.66	87.39	74.03	51.70	93.99	48.60	93.06	71.18	67.94	34.79	20.27	21.55	26.10	49.05	45.49	54.04
w/o correction	59.67	85.30	72.49	63.15	91.93	55.97	77.60	70.47	70.80	37.41	19.28	21.35	26.68	53.41	52.96	56.82
Multi-step Setting																
G-Speculator (Full)	74.83	89.42	82.13	81.44	94.28	81.18	97.53	90.77	87.54	60.99	46.04	34.26	48.92	72.42	72.23	74.22
w opposite spec.	71.14	89.30	80.22	74.49	95.29	80.65	97.60	88.04	85.38	59.51	44.42	34.74	47.86	68.38	67.22	71.63
w/o idf	73.13	89.32	81.23	79.90	95.93	78.84	97.40	88.69	86.52	58.33	43.38	36.84	47.51	70.45	68.27	72.39
w/o evidence expa.	71.70	88.77	80.24	79.44	94.69	77.57	97.26	90.03	86.13	54.58	33.13	28.00	40.08	68.57	66.75	70.06

Table 15: **Detailed QA performance (F1, %) of hyperparameter experiment.**

Setting	HotpotQA			2WikiMultihopQA						MuSiQue				ScaleQA		Avg.
	Br.^{2H}	Cp.^{2H}	Avg.	Br.^{2H}	Cp.^{2H}	In.^{2H}	Cp.^{4H}	Bc.^{4H}	Avg.	Br.^{2H}	Br.^{3H}	Br.^{4H}	Avg.	Br.$^{2H}_{10k}$	Br.$^{2H}_{35k}$	
Clue Exploration Width (α)																
α=1	73.55	89.37	81.46	81.66	94.69	80.99	96.66	86.84	86.67	62.57	45.16	31.86	48.62	72.59	69.67	73.08
α=3	74.83	89.42	82.13	81.44	94.28	81.18	97.53	90.77	87.54	60.99	46.04	34.26	48.92	72.42	72.23	74.22
α=5	72.32	90.58	81.45	80.79	95.15	82.14	97.53	90.44	87.74	59.40	44.30	34.31	47.67	70.84	71.03	73.60
Number of Returned Trajectories (β)																
β=1	74.83	89.42	82.13	81.44	94.28	81.18	97.53	90.77	87.54	60.99	46.04	34.26	48.92	72.42	72.23	74.22
β=2	76.24	90.41	83.33	81.88	95.99	82.88	97.53	92.64	88.89	59.98	46.66	33.00	48.48	72.70	72.05	74.76
β=3	75.52	90.78	83.15	82.98	95.99	82.42	97.20	92.60	89.01	60.72	45.54	34.60	48.71	73.07	72.54	74.95
Clue Speculation Depth (δ)																
δ=1	68.94	89.65	79.30	75.93	95.03	79.53	97.13	85.34	84.73	58.75	43.06	34.67	47.03	67.87	67.18	71.05
δ=2	74.83	89.42	82.13	81.44	94.28	81.18	97.53	90.77	87.54	60.99	46.04	34.26	48.92	72.42	72.23	74.22
δ=3	74.02	89.03	81.53	81.34	95.99	81.79	97.64	91.20	88.17	59.59	44.41	35.39	48.04	71.65	71.76	74.03
Number of Additional Passages (k)																
k=1	72.90	89.76	81.33	80.64	95.08	81.36	97.33	90.20	87.44	60.84	43.45	35.14	48.09	71.46	70.79	73.50
k=3	74.83	89.42	82.13	81.44	94.28	81.18	97.53	90.77	87.54	60.99	46.04	34.26	48.92	72.42	72.23	74.22
k=5	74.29	89.98	82.14	81.19	95.49	82.46	98.46	91.40	88.27	58.36	44.79	33.34	47.23	71.28	71.09	73.82

Table 16: **Detailed QA performance (F1, %) of different models.**

Model	HotpotQA			2WikiMultihopQA						MuSiQue				ScaleQA		Avg.
	Br.^{2H}	Cp.^{2H}	Avg.	Br.^{2H}	Cp.^{2H}	In.^{2H}	Cp.^{4H}	Bc.^{4H}	Avg.	Br.^{2H}	Br.^{3H}	Br.^{4H}	Avg.	Br.$^{2H}_{10k}$	Br.$^{2H}_{35k}$	
Equipped with Different LLMs																
GPT-4o-mini	74.83	89.42	82.13	81.44	94.28	81.18	97.53	90.77	87.54	60.99	46.04	34.26	48.92	72.42	72.23	74.22
Gemini-3.5-flash-lite	77.71	91.67	84.69	82.62	96.58	88.91	98.40	93.45	90.86	66.92	50.95	39.49	54.30	75.75	74.70	77.66
Qwen-3.6-flash	77.55	91.09	84.32	81.31	95.66	86.97	99.20	90.40	89.21	64.88	52.36	37.39	53.56	74.58	73.39	76.51
Equipped with Different Entity Linking Models																
all-MiniLM-L6-v2	60.93	86.32	73.63	59.10	95.58	51.77	96.20	76.12	72.15	35.44	16.52	21.03	24.80	51.82	49.64	56.24
Qwen-2.5-0.5B	60.89	87.77	74.33	63.25	93.39	57.98	91.46	81.04	74.95	37.48	19.07	19.07	26.08	53.87	52.10	58.27
Llama-3.2-1B	61.96	87.08	74.52	63.18	93.89	58.49	92.06	80.37	75.05	40.13	20.18	21.14	28.01	55.09	53.85	59.20
Equipped with Different Embedding Models																
all-MiniLM-L6-v2	74.83	89.42	82.13	81.44	94.28	81.18	97.53	90.77	87.54	60.99	46.04	34.26	48.92	72.42	72.23	74.22
GTE-Qwen2-1.5B	72.75	90.38	81.57	81.17	95.05	81.64	97.39	91.24	87.87	59.12	45.09	34.93	48.01	71.01	71.43	73.84
NV-Embed-v2	75.35	89.78	82.57	81.88	94.79	82.94	98.19	92.83	88.70	62.02	48.68	36.32	50.81	73.08	71.25	74.88

Table 17: **Detailed Retrieval performance (Precision, %) of ablation study.**

Variant	HotpotQA			2WikiMultihopQA						MuSiQue				ScaleQA		Avg.
	Br.^{2H}	Cp.^{2H}	Avg.	Br.^{2H}	Cp.^{2H}	In.^{2H}	Cp.^{4H}	Bc.^{4H}	Avg.	Br.^{2H}	Br.^{3H}	Br.^{4H}	Avg.	Br.$^{2H}_{10k}$	Br.$^{2H}_{35k}$	
SINGLE-STEP SETTING																
G-Speculator (Full)	56.91	73.62	65.27	80.40	93.43	73.10	89.07	76.93	81.44	48.01	45.74	42.36	45.80	61.77	59.16	65.21
w opposite spec.	50.95	73.60	62.28	63.90	93.43	62.30	89.73	64.02	72.02	42.97	40.48	38.18	40.88	52.61	50.93	58.17
w/o idf	54.92	74.78	64.85	71.60	94.13	64.60	90.93	62.13	74.16	42.23	45.36	43.05	43.62	56.25	52.54	60.38
w/o correction	57.28	73.75	65.52	80.30	93.43	73.40	89.07	76.95	81.49	47.06	45.27	41.51	45.06	61.55	59.42	65.17
MULTI-STEP SETTING																
G-Speculator (Full)	59.60	73.86	66.73	81.45	93.72	74.88	90.85	78.02	82.54	46.29	34.81	30.27	38.10	62.45	60.22	64.41
w opposite spec.	53.28	73.91	63.60	69.12	93.74	67.34	91.25	67.73	75.47	44.04	31.97	29.30	35.93	55.48	53.29	59.12
w/o idf	56.36	75.90	66.13	74.84	94.18	67.87	92.72	67.30	77.03	44.04	34.40	30.83	37.22	58.41	55.41	60.93
w/o evidence expa.	59.19	74.77	66.98	82.73	93.78	75.56	91.20	79.32	83.34	46.99	35.77	29.99	38.66	62.97	60.72	64.99

Table 18: **Detailed Retrieval performance (Precision, %) of hyperparameter experiment.**

Setting	HotpotQA			2WikiMultihopQA						MuSiQue				ScaleQA		Avg.
	Br.^{2H}	Cp.^{2H}	Avg.	Br.^{2H}	Cp.^{2H}	In.^{2H}	Cp.^{4H}	Bc.^{4H}	Avg.	Br.^{2H}	Br.^{3H}	Br.^{4H}	Avg.	Br.$^{2H}_{10k}$	Br.$^{2H}_{35k}$	
CLUE EXPLORATION WIDTH (α)																
α=1	60.37	74.28	67.33	84.40	93.72	76.24	91.57	76.34	83.20	50.75	39.87	33.41	42.47	65.17	61.47	66.03
α=3	59.60	73.86	66.73	81.45	93.72	74.88	90.85	78.02	82.54	46.29	34.81	30.27	38.10	62.45	60.22	64.41
α=5	59.98	74.20	67.09	81.67	93.42	74.75	90.85	78.26	82.54	46.18	33.98	28.54	37.33	62.61	60.88	64.47
NUMBER OF RETURNED TRAJECTORIES (β)																
β=1	59.60	73.86	66.73	81.45	93.72	74.88	90.85	78.02	82.54	46.29	34.81	30.27	38.10	62.45	60.22	64.41
β=2	51.68	65.50	58.59	71.40	86.44	65.54	82.69	67.14	73.22	39.60	31.12	26.07	33.15	54.23	52.15	56.57
β=3	51.00	62.82	56.91	70.18	86.01	64.44	82.24	61.58	71.24	38.73	29.09	25.15	31.82	53.30	50.79	54.94
CLUE SPECULATION DEPTH (δ)																
δ=1	67.43	74.00	70.72	93.93	93.47	87.52	91.25	91.62	91.61	55.78	43.58	42.67	48.01	72.38	70.67	73.16
δ=2	59.60	73.86	66.73	81.45	93.72	74.88	90.85	78.02	82.54	46.29	34.81	30.27	38.10	62.45	60.22	64.41
δ=3	57.41	74.16	65.79	81.49	93.64	73.81	91.39	77.91	82.28	46.00	31.12	24.23	35.14	61.63	59.97	63.45
NUMBER OF ADDITIONAL PASSAGES (k)																
k=1	60.01	75.00	67.51	82.95	93.88	75.61	91.73	79.27	83.45	48.84	37.22	33.29	40.71	63.93	61.64	65.80
k=3	59.60	73.86	66.73	81.45	93.72	74.88	90.85	78.02	82.54	46.29	34.81	30.27	38.10	62.45	60.22	64.41
k=5	59.60	73.92	66.76	81.25	93.53	73.31	91.03	77.41	81.94	44.58	32.80	28.28	36.21	61.81	59.80	63.68

Table 19: **Detailed Retrieval performance (Precision, %) of different models.**

Model	HotpotQA			2WikiMultihopQA						MuSiQue				ScaleQA		Avg.
	Br.^{2H}	Cp.^{2H}	Avg.	Br.^{2H}	Cp.^{2H}	In.^{2H}	Cp.^{4H}	Bc.^{4H}	Avg.	Br.^{2H}	Br.^{3H}	Br.^{4H}	Avg.	Br.$^{2H}_{10k}$	Br.$^{2H}_{35k}$	
EQUIPPED WITH DIFFERENT LLMS																
GPT-4o-mini	59.60	73.86	66.73	81.45	93.72	74.88	90.85	78.02	82.54	46.29	34.81	30.27	38.10	62.45	60.22	64.41
Gemini-3.5-flash-lite	58.75	73.31	66.03	80.02	93.32	76.04	62.90	77.30	80.57	49.87	36.98	28.58	39.89	62.88	61.45	64.29
Qwen-3.6-flash	59.92	74.39	67.16	80.09	93.66	76.05	74.36	74.26	80.62	51.59	38.14	35.02	42.52	63.87	61.75	65.16
EQUIPPED WITH DIFFERENT ENTITY LINKING MODELS																
all-MiniLM-L6-v2	56.42	73.62	65.02	78.30	95.03	67.10	92.27	73.22	79.23	45.71	43.21	37.12	42.71	60.14	57.27	63.21
Qwen-2.5-0.5B	54.88	73.87	64.38	79.00	93.03	71.20	89.73	76.82	80.58	45.52	44.61	39.57	43.76	59.80	57.60	63.90
Llama-3.2-1B	56.91	73.62	65.27	80.40	93.43	73.10	89.07	76.93	81.44	48.01	45.74	42.36	45.80	61.77	59.16	65.21
EQUIPPED WITH DIFFERENT EMBEDDING MODELS																
all-MiniLM-L6-v2	59.60	73.86	66.73	81.45	93.72	74.88	90.85	78.02	82.54	46.29	34.81	30.27	38.10	62.45	60.22	64.41
GTE-Qwen2-1.5B	57.93	73.78	65.86	82.00	93.55	75.27	91.60	78.25	82.82	45.10	35.37	30.65	37.95	61.68	59.98	64.27
NV-Embed-v2	60.19	74.19	67.19	81.81	93.74	75.01	91.97	78.14	82.75	47.10	35.13	31.86	38.91	63.03	60.56	64.83

J Detailed Baseline Configuration

We present the key hyperparameters of all baselines, as summarized in Table 20. The general settings are as follows: (i) **Retrieval**: All baselines are treated as retrievers and evaluated with a unified zero-shot LLM for QA to ensure fair comparison. Notably, Search-R1 and R1-Searcher are optimized for end-to-end QA. However, since their retrieval and generation stages are difficult to separate at the hyperparameter level, we still evaluate them under the above setting. Their end-to-end performance compared with G-Speculator is provided in Appendix D.4. (ii) **Context size**: For baselines using raw passages as context, we adopt the widely-used top-5 setting. For multi-step methods such as IRCoT and Search-R1, each step retrieve 5 passages, and top-5 from all retrievals are selected for QA. (iii) **Step**: All multi-step methods use the same maximum number of steps, i.e., 2 to 4. (iv) **Generation**: Although newer models have been released, their impact on OpenIE and QA stages has not been widely validated. We use the most common GPT-4o-mini as the default configuration.

Table 20: **Hyperparameter settings of baseline methods.**

Method	Key Hyperparameters	Method	Key Hyperparameters
KGP	"initial_retrieval_top_k": 5 "max_neighbors": 5 "linking_threshold": 0.8 "summary_max_tokens": 500	PER-QA	"type_classification": true "planner_mode": "universal" "executor_mode": "rag" "reasoner_mode": "direct"
RAPTOR	"num_layers": 5 "threshold": 0.1 "threshold_cluster_num": 5000 "max_length_in_cluster": 3500	LinearRAG	"spacy": "en_core_web_trf" "threshold": 0.4 "passage_node_weight": 0.05 "damping": 0.5
MS-GraphRAG	"retrieval_mode": "local" "max_cluster_size": 10 "summary_max_length": 500 "reports_max_length": 2000	IRCoT	"top_k_each_retrieval": 5 "few_shot_num": 3 "max_token": 512 "stop": "So the answer is:"
LightRAG	"retrieval_mode": "local" "max_entity_tokens": 6000 "max_relation_tokens": 8000 "max_total_tokens": 30000	Search-R1	"max_prompt_length": 8192 "max_response_length": 1024 "max_start_length": 2048 "max_obs_length": 2048
HippoRAG	"graph_type": "facts_and_sim" "sim_threshold": 0.8 "graph_algorithm": "ppr" "damping": 0.5	R1-Searcher	"prompt_type": "v0" "max_tokens": 1024 "max_num_seqs": 512 "retrieval_batch_size": 64
GFM-RAG	"threshold": 0.8 "max_sim_neighbors": 100 "doc_ranker": "idf_topk_ranker" "doc_ranker_top_k": 70	LogicRAG	"top_k_each_retrieval": 5 "embedding_batch_size": 32 "max_tokens": 250 "filter_repeats": False
HippoRAG2	"graph_type": "facts_and_sim" "sim_threshold": 0.8 "graph_size_alpha": 0.5 "damping": 0.5	KG-Translator	"spacy": "en_core_web_trf" "backbone": "llama_3.2_1B" "beam_size_alpha": 3 "beam_size_beta": 1/3/5

K PROMPTS

All prompts used in G-Speculator are provided in Figure 11. For retrieval stage, prompts are designed for query graph construction and retrieval path correction, along with that for the generative entity linking model. For QA stage, a JSON-formatted prompt is used to facilitate programmatic result parsing. This QA prompt is shared across all baselines to ensure fair comparison.

(i) Query Graph Construction Prompt:
Your task is to parse a natural language question into a collection of structured triples. Each triple strictly follows the format: (Subject, Relation Predicate, Object).

Core Rules:
1. Do not omit any information from the question. Keep all proper nouns and named entities complete and unchanged, no abbreviation.
2. Relation predicates must be plain lowercase phrases separated by spaces. The predicate must precisely describe the semantic link between subject and object.
3. Unknown entities that the question asks to retrieve are marked with sequential placeholders `<A>`, `<B>`, `<C>`, etc. Reuse the same placeholder if the identical unknown entity appears multiple times. Every triple must contain at least one placeholder.

(ii) Generative Entity Linking Prompt:
Given a query triple, your task is to faithfully translate the triple into an entity.

(iii) Retrieval Path Correction Prompt:
Given evidence and a question, your task is to first think and then determine whether the evidence is sufficient to answer the question.

Output Format:
1. First show your reasoning under "Think:"
- *Step 1*. Decompose the question intent, and extract answers from evidence step-by-step. During evidence reading, perform coreference resolution and verify the real subject of every statement to avoid mistakes from superficial literal matching.
- *Step 2*. If evidence is sufficient, list all answers completely. Answer requires a brief phrase or entity or "yes/no", make it as concise as possible.
- *Step 3*. If evidence is insufficient, generate a concise next_query. Substitute identified entities directly into the sub-query and do not repeat facts already resolved by evidence.
2. Then write exactly: "So the response is:"
3. Output a JSON object: {"final_answer": "ANSWER", "next_query": "QUERY"}.

JSON Fields:
- "final_answer": answer string if you found one in Step 2; otherwise None.
- "next_query": None if you found an answer; otherwise a sub-question for retrieval.

(iv) QA Prompt:
Given a question and its evidence, your task is to answer the question based on the evidence. You must give ONLY the direct answer in the most concise way possible.
Generate a JSON with a single key "Response" and a value that is a short phrase or a few words. In JSON, put every value as a string always, not float. Format your response as follows: {"response": "ANSWER_HERE"}

Figure 11: **All prompts used in this paper.**