

PropRAG: Guiding Retrieval with Beam Search over Proposition Paths

Jingjin Wang Jiawei Han

Siebel School of Computing and Data Science, University of Illinois Urbana-Champaign
{jingjin9, hanj} @illinois.edu

Abstract

Retrieval Augmented Generation (RAG) has become the standard approach for equipping Large Language Models (LLMs) with up-to-date knowledge. However, standard RAG, relying on independent passage retrieval, often fails to capture the interconnected nature of information required for complex, multi-hop reasoning. While structured RAG methods attempt to address this using knowledge graphs built from triples, we argue that the inherent context loss of triples (context collapse) limits the fidelity of the knowledge representation. We introduce **PropRAG**, a novel RAG framework that shifts from triples to context-rich **propositions** and introduces an efficient, **LLM-free online beam search** over proposition paths to discover multi-step reasoning chains. By coupling a higher-fidelity knowledge representation with explicit path discovery, PropRAG achieves state-of-the-art zero-shot Recall@5 and F1 scores on 2Wiki, HotpotQA, and MuSiQue, advancing non-parametric knowledge integration by improving evidence retrieval through richer representation and efficient reasoning path discovery.

1 Introduction

Large Language Models (LLMs) often struggle with knowledge-intensive tasks, particularly when information is newly emergent or requires complex reasoning over multiple facts. Retrieval Augmented Generation (RAG) (Lewis et al., 2020) addresses this by retrieving external knowledge, offering a non-parametric solution that mitigates issues like catastrophic forgetting in continual learning (Cohen et al., 2024; Gu et al., 2024). However, conventional RAG systems (Karpukhin et al., 2020; Lee et al., 2025) typically retrieve evidence passages independently based on query similarity. This approach struggles with multi-step queries that require synthesizing interconnected information—a process crucial for sense-making (Klein

et al., 2006) and associativity (Suzuki, 2007).

To improve multi-hop retrieval, structured RAG methods have emerged. Approaches like HippoRAG 2 (Gutiérrez et al., 2025) utilize knowledge graphs (KGs) built from (Subject, Predicate, Object) triples and use graph algorithms (e.g., Personalized PageRank) to rank relevant nodes. While effective, we argue these methods suffer from "context collapse." Triples are a lossy compression of natural language, discarding crucial nuances like conditionality, provenance, and n-ary relationships (Section 3). This loss of fidelity limits the system's ability to reconstruct complex reasoning chains. Other advanced RAG strategies rely on online LLM calls during the retrieval process (e.g., generating subsequent queries) (Trivedi et al., 2022a; Jiang et al., 2024), which introduces significant latency, cost, and potential consistency issues.

Our main insight is that effective multi-hop RAG requires two key shifts: (1) moving from lossy triples to a higher-fidelity knowledge representation that preserves context, and (2) moving from simple node ranking to explicit, efficient discovery of reasoning paths.

We introduce **PropRAG**, a novel RAG framework designed for dynamic, interconnected memory retrieval without requiring online LLM inference during the search process. PropRAG features two core innovations:

1. Propositions as High-Fidelity Knowledge Units:

Extracted offline by an LLM, propositions are atomic, self-contained statements that preserve contextual richness (e.g., conditional clauses, multi-entity events) beyond triples (Section 3).

2. LLM-Free Online Beam Search for Path Discovery:

A novel two-stage retrieval process culminating in an algorithmic beam search discoverers and scores paths of interconnected propositions. This search operates on a pre-built graph

using embeddings and graph structure, avoiding the costs and latency of LLM inference during evidence discovery (Section 5).

By coupling context-rich propositions with an efficient, algorithmic path-finding mechanism, PropRAG enhances the system’s ability to perform associative reasoning. Experiments demonstrate substantial improvements, particularly on multi-hop QA tasks. PropRAG sets new state-of-the-art zero-shot RAG scores on several challenging benchmarks, advancing the capabilities of non-parametric knowledge integration in LLMs.

2 Related Work

Retrieval Augmented Generation (RAG) frameworks (Lewis et al., 2020) augment LLMs by retrieving documents. Early methods like DPR (Karpukhin et al., 2020) used embedding similarity. Despite better embeddings (Izacard et al., 2022; Ni et al., 2022; Lee et al., 2025), standard RAG struggles with multi-document synthesis (Asai et al., 2020).

Multi-Hop RAG aims to address this. Iterative methods (Asai et al., 2020; Trivedi et al., 2022a) retrieve sequentially, often using LLMs online. Graph-based RAG (Edge et al., 2024; Sarthi et al., 2024) structures knowledge, often using KGs. HippoRAG 2 (Gutiérrez et al., 2025) used Personalized PageRank over triple-based KGs, focusing on node ranking. PropRAG differs fundamentally by using context-rich propositions and employing an explicit, LLM-free online beam search focused on discovering complete reasoning paths, rather than just ranking nodes.

Beam Search in Retrieval. Recent work has explored leveraging beam search to improve multi-hop retrieval. For instance, (Zhang et al., 2023) proposed Beam Retrieval, an end-to-end trainable framework where beam search is used during both training and inference to find optimal passage sequences. Their approach learns a scoring function optimized across hops using ground-truth passage chains.

In contrast, PropRAG adopts a zero-shot online retrieval strategy. While also employing beam search, PropRAG operates over a graph of contextually rich propositions, extracted offline. Our beam search algorithmically discovers proposition paths based on pre-computed embedding similarity and graph connectivity, crucially avoiding online LLM inference costs or task-specific training during the

retrieval phase. PropRAG thus decouples complex reasoning path discovery from end-to-end training dependencies, focusing on leveraging richer knowledge units (propositions) and algorithmic path exploration.

Propositions as Retrieval Units. Chen et al. (2024) demonstrated that retrieving propositions (atomic factoids) can outperform passage or sentence retrieval in standard RAG. PropRAG builds upon this insight by integrating propositions into a graph structure and utilizing beam search to chain them together for multi-hop reasoning.

3 Propositions: Escaping the Tyranny of the Triple

The efficacy of multi-hop reasoning is constrained by the fidelity of the underlying knowledge representation. Traditional KG-based RAG systems rely on $\langle \text{Subject}, \text{Predicate}, \text{Object} \rangle$ triples. We contend this is a lossy compression that discards crucial nuances, leading to "context collapse." Propositions—atomic, self-contained statements that preserve context—offer a higher-fidelity alternative. Figure 1 illustrates how a proposition graph retains context lost in a standard KG.

Key limitations of triples addressed by propositions include:

1. Omission of Conditional Context: Facts are often contingent upon specific conditions (temporal, spatial, etc.), which triples frequently discard.

- *Example Passage:* The experimental drug showed promise in Phase 2 trials, reducing tumor size significantly, but only in patients under 50 with the specific KRAS mutation.
- *Triples might yield:* (drug, showed, promise), (drug, reduced, tumor size). The critical conditions are lost.
- *Propositions preserve conditions:* E.g., "The significant tumor size reduction by the experimental drug was observed only in patients under 50."

2. Limitations in Representing Higher-Order Relations: Representing complex constructs like provenance, causality, or nested clauses is unnatural in S-P-O format.

- *Triple Limitation:* As shown in Figure 1 (Left), the triple representation fails to capture the provenance ("Archival records in-

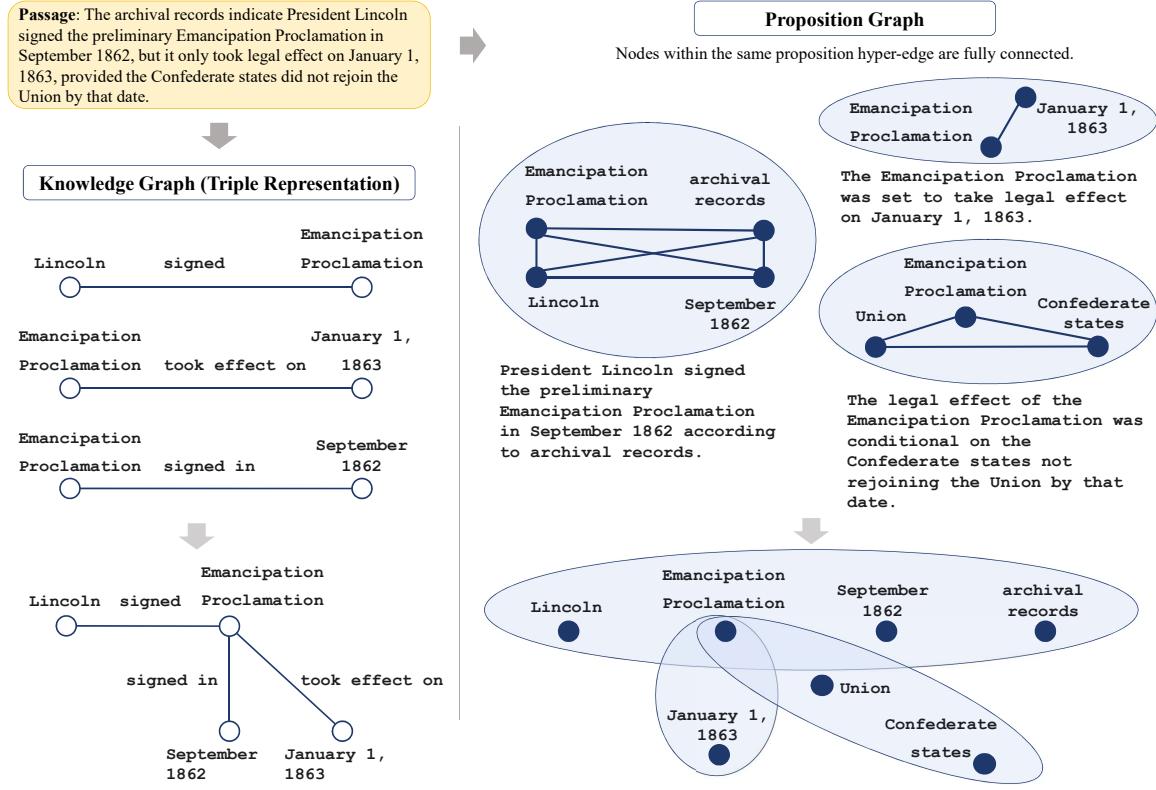


Figure 1: Comparison of a traditional Knowledge Graph (KG) versus a Proposition Graph for a complex passage. **Left:** The triple-based KG struggles to natively represent provenance ("archival records") and conditional clauses. It results in disconnected facts where the crucial context (the conditionality of the Emancipation Proclamation taking effect) is omitted, leading to context loss. **Right:** The PropRAG proposition graph utilizes implicit hyper-edges (fully connected cliques within shaded ovals) to link all entities co-occurring within a single, context-rich proposition. This structure directly preserves nuances like conditionality and provenance.

dicate...") and completely omits the critical condition ("provided the Confederate states...") regarding the Emancipation Proclamation. While KGs can use *reification*, this increases complexity and sacrifices clarity.

- *Proposition Advantage:* Propositions naturally encapsulate these relations. Figure 1 (Right) shows these preserved in distinct propositions, modeled using implicit hyper-edges that maintain the full context.

3. Forced Binarization of Unary Predicates (Attributes): The S-P-O model is fundamentally relational and struggles to represent intrinsic properties (e.g., "The manuscript is fragile").

- *Triple Limitation:* Forcing this into (*manuscript*, *is*, *fragile*) artificially treats the attribute ("fragile") as an independent object, relying on weak predicates.
- *Proposition Advantage:* Propositions han-

dle attributes natively: "The ancient manuscript was fragile."

4. Fragmentation of N-ary Relationships (Events):

(Events): Real-world events often involve multiple participants. The S-P-O structure forces decomposition into multiple binary triples, fragmenting the event's holistic semantics.

- *Example Passage (Collaboration):* "The groundbreaking research paper on quantum entanglement was co-authored by Alice, Bob, and Charlie in 2023."
- *Triples fragment this:* (Alice, co-authored, paper), (Bob, co-authored, paper), etc., losing that they collaborated *jointly*.
- *Proposition Advantage:* A single proposition captures the N-ary relation: "Alice, Bob, and Charlie co-authored the groundbreaking research paper..." This is represented as an implicit hyper-

edge in the proposition graph.

To operationalize this, PropRAG leverages the NLU capabilities of LLMs (Llama-3.3-70B-Instruct) during the **offline** indexing phase to extract these high-fidelity propositions. This ensures the subsequent online retrieval operates on a contextually rich foundation without further LLM inference for knowledge representation.

4 Problem Formulation: Finding the Reasoning Path

Traditional RAG aims to retrieve documents D_{ret} maximizing individual relevance $\text{sim}(\text{emb}(d), \text{emb}(q))$. Multi-hop KG-RAG methods like HippoRAG 2 (Gutiérrez et al., 2025) typically rank nodes (entities/passage) based on proximity to query seeds or graph centrality (e.g., via PPR), but do not explicitly construct or evaluate multi-step reasoning paths. Furthermore, when KGs use context-poor triples, the semantic richness available for ranking is limited.

We reformulate multi-hop retrieval as finding an optimal *path* of interconnected propositions $P = (p_1, p_2, \dots, p_k)$ that collectively answer query q . Propositions p_i, p_{i+1} are linked if they share common or synonymous entities. This connection occurs naturally in the proposition graph (Figure 1, right), where entities shared between proposition hyper-edges act as bridges.

The objective is to find

$$P^* = \underset{P \in \text{ConnectedPaths}(\mathcal{P})}{\text{argmax}} \text{Score}(P, q), \quad (1)$$

where $\text{Score}(P, q)$ measures the relevance of the entire path, potentially via aggregated or concatenated embeddings. As finding the global optimum over all possible paths is intractable, we employ beam search as an efficient heuristic.

5 Methodology: PropRAG

The PropRAG framework operationalizes the path-finding objective (Section 4) through a structured offline indexing phase followed by an efficient, two-stage online retrieval process (Figure 2). We utilize the complex multi-hop query presented in Figure 3 as a running example to illustrate the methodology.

5.1 Offline Indexing (LLM-assisted)

The objective of the offline indexing phase is to construct the high-fidelity proposition graph, a prerequisite for efficient online path discovery.

1. Knowledge Extraction: An LLM (e.g., Llama-3.3-70B-Instruct) is employed to extract propositions and their constituent entities from the corpus \mathcal{D} (Prompts detailed in Appendix A.3).

2. Graph Construction: A proposition graph $G = (V, E)$ is constructed. Nodes V represent entities and passages. Edges E link entities if they co-occur within the same proposition (forming implicit hyper-edges) or if they are determined to be synonymous via embedding similarity (Details in Appendix A.1).

3. Embedding Generation: Embeddings (e.g., using NV-Embed-v2) are computed and stored for all nodes and propositions.

5.2 Online Retrieval (LLM-free)

The online retrieval component implements a two-stage, LLM-free architecture designed to identify salient reasoning paths. This coarse-to-fine optimization strategy balances the need for broad exploration of the graph with focused refinement of candidate paths.

5.2.1 Stage 1: Coarse-grained Subgraph Induction

The objective of Stage 1 is to efficiently prune the search space by inducing a highly relevant, localized subgraph G_{sub} from the global graph G .

1. Initial Candidate Retrieval: The top- N_{prop} propositions most semantically similar to the query q are retrieved. (*Running Example, Fig. 3*): Propositions P1 (Mantua Cathedral) and P3 (Governor’s death) are retrieved due to high initial similarity, while the necessary intermediate P2 is not highly ranked.

2. Seed Set Initialization ($S_{initial}$): Entities $\mathcal{E}(p)$ contained within these top propositions are extracted. The top- N_{entity} unique entities form the initial seed set $S_{initial}$. Uniform weights are assigned to promote diverse graph exploration.

3. Exploratory Graph Traversal (PPR): Personalized PageRank (PPR) is executed on the global graph G , with the reset probability distribution concentrated on $S_{initial}$. A high damping factor (e.g., 0.75) is utilized to facilitate broader traversal originating from these seeds.

4. Subgraph Induction (G_{sub}): The top- K passages identified by the PPR scores, along with their associated entities and propositions in G ,

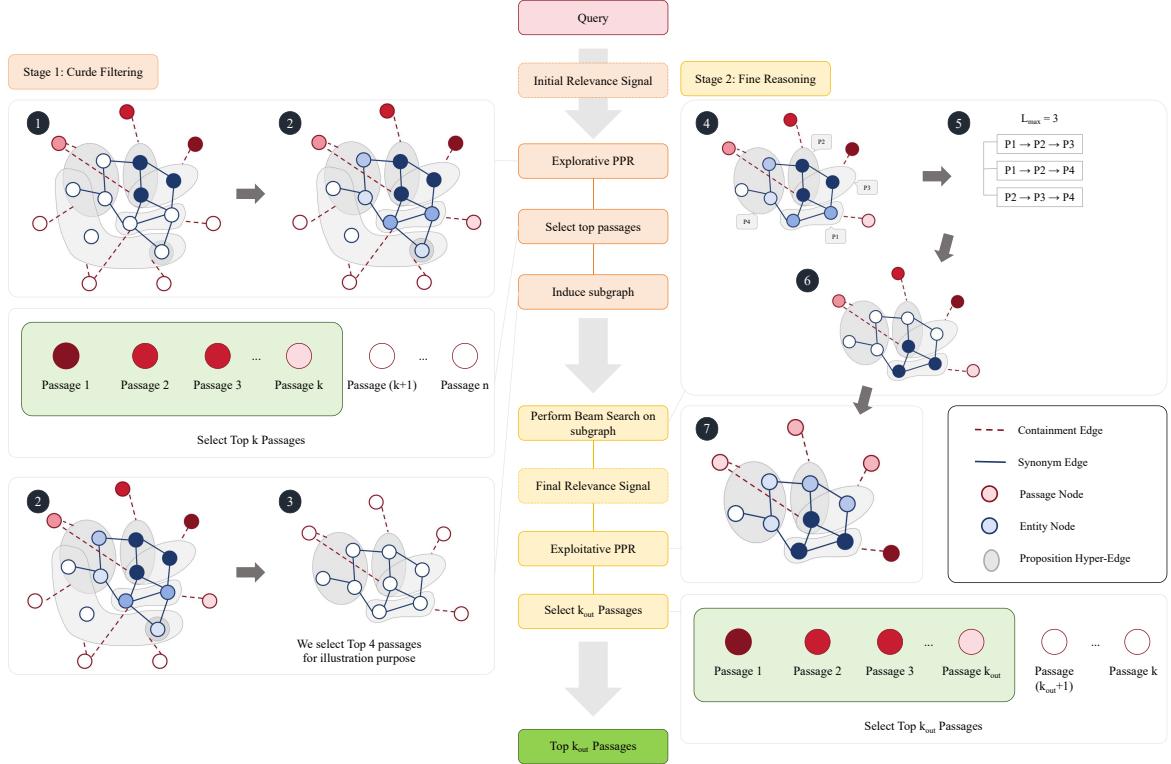


Figure 2: The two-stage online retrieval architecture of PropRAG. **Stage 1 (Coarse Filtering):** Employs exploratory PPR (high damping factor) on the full proposition graph G to induce a focused, relevant subgraph (G_{sub}). **Stage 2 (Fine Reasoning):** Executes a graph-guided beam search on G_{sub} to discover explicit reasoning paths (illustrated in Figure 3), generates refined relevance signals based on these paths, applies exploitative PPR (low damping factor) on G_{sub} using the refined signals, and selects the final top- k_{out} evidence passages.

constitute the focused subgraph G_{sub} . This localized graph serves as the operational search space for Stage 2.

5.2.2 Stage 2: Fine-grained Path Discovery and Ranking

The objective of Stage 2 is to explicitly discover multi-proposition reasoning paths within G_{sub} and leverage these paths to determine the final ranking of evidence passages.

1. Beam Search Path Exploration: The beam search algorithm (detailed in Section 5.3) systematically explores G_{sub} to identify connected proposition paths up to a maximum length L_{max} . It maintains a beam of the top- B paths ranked by relevance to the query q . (*Running Example, Fig. 3*): Initiating from P1, the beam search traverses the graph via the "Saint Peter" entity link, discovering P2 ("St. Peter's Basilica is located in Vatican City"). Subsequently, it connects P2 to P3 via "Vatican City", constructing the complete path P1 → P2 → P3.

2. Refined Seed Set Construction (S_{final}): A refined seed set is generated by integrating relevance signals from two sources: *Exploration Seeds* (derived from initial query similarity) and *Exploitation Seeds* (entities central to the top-scoring paths identified by the beam search).

3. Exploitative Path-informed Ranking (PPR): A second iteration of PPR is executed, constrained to G_{sub} . The reset probability vector incorporates normalized scores from both exploration and exploitation seeds (Appendix A.6). A lower damping factor (e.g., 0.45) is employed to intensify the focus on these refined relevance signals.

4. Final Evidence Ranking: Passages are ranked according to their final PPR scores, prioritizing those strongly connected to the salient reasoning paths discovered during the beam search.

Query (Q): What year did the Governor of the city where the basilica named after the same saint as the one that Mantua Cathedral is dedicated to die?

Gold Answer: 1952

Initial Propositions (Stage 1 Retrieval):

- (P1) 0.4275 - Mantua Cathedral is a Roman Catholic cathedral dedicated to Saint Peter.
- 0.3851 - Mantua Cathedral is the seat of the Bishop of Mantua.
- ...
- (P3) 0.3135 - No successor was appointed to the post of Governor of Vatican City after Marchese Camillo Serafini's death in 1952.

Beam Search (Stage 2, Depth 2/3): *Path expansion from P1.*

- 0.4792 - (P1) Mantua Cathedral... dedicated to Saint Peter. → (P2) St. Peter's Basilica is located in Vatican City. (via entity link: "Saint Peter" → "St. Peter")
- 0.4756 - (P1) → Alfredo Ormando died on 23 January 1998 in Rome. (via entity link: "Roman" → "Rome")
- ...

Beam Search (Stage 2, Depth 3/3): *Path expansion from P2.*

- **0.5989** - (P1) Mantua Cathedral... → (P2) St. Peter's Basilica is located in Vatican City. → (P3) No successor was appointed... after Marchese Camillo Serafini's death in 1952. (via entity link: "Vatican City")
- 0.5674 - (P1) → (P2) → Marchese Camillo Serafini held the post of Governor... until his death in 1952.
- ...

Observation: The crucial intermediate proposition P2 exhibited low initial relevance to Q but was discovered via the graph-guided beam search, enabling the construction of the complete reasoning path $P1 \rightarrow P2 \rightarrow P3$.

Figure 3: Running Example: Beam search execution ($L_{max} = 3$) for a MuSiQue query, illustrating the discovery of a multi-hop reasoning path in PropRAG. Proposition text is abridged for clarity.

5.3 Beam Search Algorithm for Path Discovery

The core of Stage 2 is a specialized beam search algorithm designed to efficiently explore the proposition graph and identify high-relevance reasoning chains without requiring online LLM inference. It heuristically seeks paths $P = (p_1, \dots, p_L)$ that maximize the relevance score $Score(P, q)$.

Algorithmic Procedure:

- **Initialization:** The beam is initialized with paths of length 1, comprising the propositions most semantically similar to the query q .
- **Path Expansion (Graph-Guided):** At each step, paths are extended by identifying candidate next propositions p_{next} . Candidates are

primarily selected from propositions connected to the current path's terminal proposition p_k via shared or synonymous entity links within G_{sub} . (*Running Example, Fig. 3*): P1 is expanded to P2 due to the shared entity "Saint Peter". To mitigate the risk of local optima, the top-3 initial query-relevant propositions are also included as candidates ("jump points"), regardless of direct connectivity to p_k .

- **Path Scoring (Online Embedding Computation):** The relevance of an expanded path $P_{new} = (P, p_{next})$ is estimated as $Score(P_{new}, q) \approx sim(emb(P_{new}), emb(q))$. A computationally efficient average embedding is used for preliminary scoring. The top- M candidates are subsequently re-ranked using a more precise score derived from an embedding of the concatenated text of all propositions in P_{new} . This step involves efficient online computation using the embedding model, distinct from LLM inference.
- **Selection and Pruning:** From all expanded candidates, the top- B (beam width) paths with the highest scores are retained for the next iteration; others are pruned.
- **Termination:** The search concludes when paths reach the maximum length L_{max} or no further valid expansions are possible.

This online beam search operates exclusively on the pre-computed graph structure and efficient online embedding computations. Crucially, it ensures low-latency inference by **avoiding any costly online LLM calls** for path generation or scoring.

6 Experiments

6.1 Setup

Datasets: We evaluate on five diverse QA datasets: NaturalQuestions (NQ) (Wang et al., 2024) and PopQA (Mallen et al., 2023) for single-hop reasoning, and 2WikiMultihopQA (2Wiki) (Ho et al., 2020), HotpotQA (Yang et al., 2018), and MuSiQue-Ans (Trivedi et al., 2022b) for multi-hop reasoning. We use the 1000-query samples and associated corpora provided by Gutiérrez et al. (2025) for direct comparability and to manage experimental costs.

Baselines: We compare against a comprehensive set of baselines, including classic retrievers (BM25,

Table 1: Passage Retrieval Performance (Recall@5). Baselines from Gutiérrez et al. (2025). Best overall result in bold.

Method	NQ	PopQA	MuSiQue	2Wiki	HotpotQA
<i>Simple Baselines</i>					
BM25	56.1%	35.7%	43.5%	65.3%	74.8%
Contriever	54.6%	43.2%	46.6%	57.5%	75.3%
GTR (T5-base)	63.4%	49.4%	49.1%	67.9%	73.9%
<i>Large Embedding Models (Base Retriever)</i>					
GTE-Qwen2-7B	74.3%	50.6%	63.6%	74.8%	89.1%
GritLM-7B	76.6%	50.1%	65.9%	76.0%	92.4%
NV-Embed-v2 (7B)	75.4%	51.0%	69.7%	76.5%	94.5%
<i>Structure-Augmented RAG</i>					
RAPTOR	68.3%	48.7%	57.8%	66.2%	86.9%
HippoRAG	44.4%	53.8%	53.2%	90.4%	77.3%
HippoRAG 2	78.0%	51.7%	74.7%	90.4%	96.3%
PropRAG (Ours)					
$L_{max} = 1$ (No beam search)	78.4%	56.3%	75.6%	92.0%	95.7%
$L_{max} = 2$	78.1%	56.1%	77.6%	93.4%	97.2%
$L_{max} = 3$ (Default)	77.9%	56.2%	78.3%	94.1%	97.4%
$L_{max} = 4$	77.8%	56.0%	77.6%	93.7%	97.0%

Table 2: End-to-End QA Performance (F1 Score) with Llama-3.3-70B-Instruct Reader. Baselines from Gutiérrez et al. (2025). Best overall result in bold.

Method	NQ	PopQA	MuSiQue	2Wiki	HotpotQA	Avg
<i>No Retrieval (Parametric)</i>						
Llama-3.3-70B-Instruct	54.9%	32.5%	26.1%	42.8%	47.3%	40.7%
<i>Simple Baselines</i>						
Contriever	58.9%	53.1%	31.3%	41.9%	62.3%	49.5%
GTR (T5-base)	59.9%	56.2%	34.6%	52.8%	62.8%	53.3%
<i>Large Embedding Models (Base Retriever)</i>						
GTE-Qwen2-7B	62.0%	56.3%	40.9%	60.0%	71.0%	58.0%
GritLM-7B	61.3%	55.8%	44.8%	60.6%	73.3%	59.2%
NV-Embed-v2 (7B)	61.9%	55.7%	45.7%	61.5%	75.3%	58.0%
<i>Structure-Augmented RAG</i>						
RAPTOR	50.7%	56.2%	28.9%	52.1%	69.5%	51.5%
GraphRAG	46.9%	48.1%	38.5%	58.6%	68.6%	52.1%
LightRAG	16.6%	2.4%	1.6%	11.6%	2.4%	6.9%
HippoRAG	55.3%	55.9%	35.1%	71.8%	63.5%	56.3%
HippoRAG 2	63.3%	56.2%	48.6%	71.0%	75.5%	62.9%
PropRAG (Ours)						
$L_{max} = 1$ (No beam search)	62.2%	56.1%	52.6%	73.5%	75.7%	64.0%
$L_{max} = 2$	61.9%	56.1%	53.4%	74.9%	76.0%	64.4%
$L_{max} = 3$ (Default)	62.5%	56.4%	53.9%	75.3%	76.1%	64.9%
$L_{max} = 4$	62.8%	56.0%	53.0%	75.3%	76.1%	64.7%

Contriever, GTR), large embedding models serving as base retrievers (GTE-Qwen2, GritLM, NV-Embed-v2), and leading structure-augmented RAG methods (RAPTOR, GraphRAG, LightRAG, HippoRAG, HippoRAG 2). Baseline results are primarily adopted from Gutiérrez et al. (2025).

Implementation and Reproducibility: PropRAG uses Llama-3.3-70B-Instruct (AI@Meta, 2024) for offline proposition extraction and as the QA reader, and NV-Embed-v2 (7B) (Lee et al., 2025) for all embeddings. Key parameters include

beam width $B = 4$ and max path length $L_{max} = 3$. A full list of hyperparameters is provided in Appendix A.2.

Metrics: We report standard metrics: Passage Recall@5 for retrieval quality, and QA F1 score for end-to-end performance, following MuSiQue evaluation scripts (Trivedi et al., 2022b).

6.2 Results and Discussion

We present the main retrieval (Recall@5) and end-to-end QA (F1 Score) results in Table 1 and Table

Table 3: Ablation Study on Beam Width (B) using $L_{max} = 3$. Default $B = 4$. Performance shown as Recall@5 / F1 Score.

Beam Width (B)	MuSiQue (R@5 / F1)	2Wiki (R@5 / F1)	HotpotQA (R@5 / F1)	Average (R@5 / F1)
1 (Greedy Search)	76.6% / 52.9%	92.1% / 74.5%	97.0% / 76.2%	88.5% / 67.9%
2	77.4% / 52.9%	94.4% / 75.8%	97.4% / 75.7%	89.7% / 68.1%
3	78.0% / 53.1%	94.2% / 75.7%	97.5% / 76.2%	89.9% / 68.3%
4 (Default)	78.3% / 53.9%	94.1% / 75.3%	97.4% / 76.1%	89.9% / 68.5%
5	77.8% / 54.4%	93.7% / 75.4%	97.4% / 76.1%	89.6% / 68.6%
6	77.8% / 53.9%	93.2% / 74.6%	97.2% / 75.7%	89.4% / 68.1%

Table 4: Full Ablation Study Results (Recall@5). PropRAG uses $L_{max} = 3$, $B = 4$ unless noted. HippoRAG 2 (no filter) results from [Gutiérrez et al. \(2025\)](#) Table 4.

Configuration	MuSiQue	2Wiki	HotpotQA	Avg
Full PropRAG	78.3%	94.1%	97.4%	89.9%
<i>Comparison Baselines (Impact of Propositions)</i>				
PropRAG (Propositions only, no two-stage PPR, no beam search)	75.4%	90.4%	95.9%	87.2%
HippoRAG 2 (Triples, no LLM rerank, comparable setting)	73.0%	90.7%	95.4%	86.4%
<i>Retrieval Strategy Ablations (Impact of Special Beam Search Settings)</i>				
Exploration Seeds Only	75.6%	92.0%	95.6%	87.7%
Exploitation Seeds Only	77.9%	91.4%	97.6%	89.0%
No Graph Guidance (Similarity only)	77.4%	92.9%	96.6%	89.0%

2, respectively.

Overall Performance: PropRAG (Default: $L_{max} = 3, B = 4$) achieves a state-of-the-art average F1 score of 64.9%. This outperforms the previous best structured RAG, HippoRAG 2, by 2.0 points, and the base retriever, NV-Embed-v2, by a substantial 6.9 points. This highlights the synergy of context-rich propositions and explicit, LLM-free online path discovery.

The Benefit of Propositions ($L_{max} = 1$): Even without multi-step path expansion ($L_{max} = 1$, effectively disabling the beam search but retaining the two-stage PPR on the proposition graph), PropRAG significantly outperforms baselines. For instance, on MuSiQue, $L_{max} = 1$ achieves 52.6% F1, a +4.0% improvement over HippoRAG 2 (48.6%). This demonstrates the inherent benefit of using the higher-fidelity proposition graph structure, even before extensive beam search exploration.

The Benefit of Beam Search ($L_{max} > 1$): Explicit path discovery via beam search further boosts performance, particularly on multi-hop datasets. Comparing the default configuration ($L_{max} = 3$) to the $L_{max} = 1$ ablation, we observe gains in average F1 by +0.9% (to 64.9%). The impact on retrieval is even more pronounced, with MuSiQue Recall@5 increasing by +2.7% (from 75.6% to 78.3%). This confirms that exploring 2-3 hop paths uncovers crucial evidence often missed by meth-

ods focused only on initial similarity. We find $L_{max} = 3$ provides the optimal balance.

Performance on Simpler Tasks: On single-hop datasets NQ and PopQA, PropRAG remains robust (e.g., 62.5% NQ F1, 56.4% PopQA F1 with $L_{max} = 3$), demonstrating that the added complexity does not degrade performance on tasks with less multi-hop dependency.

6.3 Ablation Studies

We conduct ablation studies on the multi-hop datasets (MuSiQue, 2Wiki, HotpotQA) to validate key design choices. Tables 3 and 4 show our findings.

Propositions vs. Triples: To isolate the impact of the knowledge representation, we compare PropRAG using only its first stage PPR (PPR on the proposition graph) against HippoRAG 2 in a comparable setting (no filter, using triples and PPR; results from [Gutiérrez et al. \(2025\)](#) Table 4). PropRAG (Stage 1 PPR only) achieves an average Recall@5 of 87.2%, compared to 86.4% for HippoRAG 2 (+0.8%). The difference is most notable on MuSiQue (75.4% vs. 73.0%, +2.4%). This confirms that the richer context captured in propositions provides a stronger foundation for graph-based retrieval, even without multi-step beam search.

Impact of Beam Width (B): We vary the beam width B while keeping $L_{max} = 3$. Increasing

B from 1 (greedy search) to 4 (default) improves average R@5 by +1.4% (from 88.5% to 89.9%) and average F1 by +0.6%. $B = 4$ offers a strong balance between exploration and computational cost.

Importance of Graph Guidance: We test the importance of the graph structure guiding the beam search by allowing path expansion based purely on embedding similarity, without requiring graph connectivity (beyond the initial "jump points"). This configuration ("No Graph Guidance") achieves an average R@5 of 89.0%, which is 0.9% lower than the full PropRAG (89.9%). The drop is significant on 2Wiki (92.9% vs. 94.1%). This demonstrates that leveraging the explicit connections in the proposition graph effectively guides the search towards more coherent reasoning paths, rather than relying solely on semantic similarity which can be noisy.

These ablations collectively confirm the contributions of the higher-fidelity proposition representation, the effectiveness of the graph-guided beam search, and the robustness of the two-stage retrieval architecture.

6.4 Efficiency Analysis

A key design goal of PropRAG is to enable sophisticated multi-hop reasoning efficiently by strategically investing in offline knowledge structuring to avoid costly online LLM calls.

Offline Cost-Benefit Trade-off: The offline indexing phase requires LLM calls for proposition extraction. As detailed in Appendix A.5 (Table 6), PropRAG required 21.1M total LLM tokens (Input+Output) to index the MuSiQue dataset. While this is higher than HippoRAG 2 (12.2M tokens), it is significantly lower than other structured RAG methods like GraphRAG (151.6M tokens) and LightRAG (86.8M tokens). Crucially, PropRAG achieves vastly superior performance (53.9% F1 on MuSiQue) compared to GraphRAG (38.5%) and LightRAG (1.6%). PropRAG thus achieves a superior cost-benefit balance, creating higher-quality knowledge units with less computational overhead than comparable methods.

Online Latency: While PropRAG avoids online LLM calls, the two-stage PPR and beam search (which includes online embedding computations for path scoring) add latency compared to a standard dense retriever. We quantify this trade-off through a runtime comparison on the MuSiQue dataset using our hardware (NVIDIA RTX 4090).

A standard dense retriever (NV-Embed-v2) takes approximately 50ms per query. PropRAG takes approximately 500-1000ms per query. In contrast, an LLM-dependent retrieval step (e.g., using Llama-3-70B to generate a next-hop query) typically takes 2-5 seconds or more per hop. Therefore, PropRAG is significantly faster (2-10x) online than LLM-in-the-loop retrieval alternatives while providing superior retrieval quality.

7 Conclusion

PropRAG represents a significant advancement in RAG by addressing the limitations of both standard retrieval and traditional structured RAG. By shifting from context-poor triples to richer propositions, PropRAG creates a high-fidelity knowledge representation. Coupled with a novel, LLM-free online beam search mechanism, it enables the explicit discovery of multi-step reasoning paths. This dual approach demonstrably improves the quality of retrieved evidence, particularly for complex multi-hop queries. Our experiments show that PropRAG sets new state-of-the-art results for zero-shot RAG systems on several challenging benchmarks. PropRAG underscores the value of explicit, algorithmic modeling of reasoning processes over high-fidelity, pre-structured knowledge, offering a promising direction for developing LLMs with more robust and associative non-parametric memory.

Acknowledgments

Research was supported in part by the National Science Foundation IIS-19-56151, NSF IIS 25-37827, the Molecule Maker Lab Institute: An AI Research Institutes program supported by NSF under Award No. 2019897, and the Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE) by NSF under BRIES Program No. HR0011-24-3-0325. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies. We also wish to thank Manting Liao for creating the figures.

Limitations

PropRAG's primary limitations include the computational overhead of beam search, which, while LLM-free online, is more intensive than simpler retrieval methods. The system's performance is sensitive to the quality of the offline proposition extrac-

tion phase; errors or omissions here can propagate. Although online LLM calls are avoided during retrieval, the initial proposition generation relies on an LLM, and its quality can influence downstream results. Furthermore, the graph construction process, particularly the accuracy of entity linking and synonymy detection, plays a crucial role and can be a source of error. The current path scoring relies on embedding similarity, which might not capture all semantic nuances required for perfect path evaluation.

References

AI@Meta. 2024. [Llama 3 model card](#).

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *International Conference on Learning Representations (ICLR)*.

Tong Chen, Hongwei Wang, Sihao Chen, Wenhao Yu, Kaixin Ma, Xinran Zhao, Hongming Zhang, and Dong Yu. 2024. Dense x retrieval: What retrieval granularity should we use? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15159–15177.

Roni Cohen, Eran Biran, Ori Yoran, Amir Globerson, and Mor Geva. 2024. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics (TACL)*, 12:283–298.

Derek Edge, Hoang Trinh, Nicholas Cheng, Joshua Bradley, Allen Chao, Ajay Mody, Shayne Truitt, and Jason Larson. 2024. From local to global: A graph rag approach to query-focused summarization. [http://arxiv.org/abs/2404.16130](#). ArXiv:2404.16130.

Jia-Chen Gu, Hao-Xiang Xu, Jia-Yu Ma, Pu Lu, Zheng-Hua Ling, Kai-Wei Chang, and Nanyun Peng. 2024. Model editing harms general abilities of large language models: Regularization to the rescue. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 16801–16819.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From rag to memory: Non-parametric continual learning for large language models. [http://arxiv.org/abs/2502.14802](#). ArXiv:2502.14802.

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

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sébastien Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions of the Association for Computational Linguistics (TACL)*, 10:641–655.

Zhouyu Jiang, Mengshu Sun, Lei Liang, and Zhiqiang Zhang. 2024. Retrieve, summarize, plan: Advancing multi-hop question answering with an iterative approach. <http://arxiv.org/abs/2407.13101>. ArXiv:2407.13101.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Gary Klein, Brian Moon, and Robert R. Hoffman. 2006. Making sense of sensemaking 1: Alternative perspectives. *IEEE Intelligent Systems*, 21(4):70–73.

Chang-Bin Lee, Rishav Roy, Mengjiao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. Nv-embed-v2: Improved techniques for training llms as generalist embedding models. [http://arxiv.org/abs/2405.17428](#). ArXiv:2405.17428.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9459–9474.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 9802–9822.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. 2022. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9844–9855.

Priyanka Sarthi, Sameer Abdullah, Abhilash Tuli, Sakshi Khanna, Abhijit Goldie, and Christopher D. Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations (ICLR)*.

Wendy A. Suzuki. 2007. Making new memories: the role of the hippocampus in new associative learning. *Annals of the New York Academy of Sciences*, 1097:1–11.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022a. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint arXiv:2212.10509*.

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

Yuhao Wang, Ruiyang Ren, Junyi Li, Wayne Xin Zhao, Jing Liu, and Ji-Rong Wen. 2024. Rear: A relevance-aware retrieval-augmented framework for open-domain question answering. *arXiv preprint arXiv:2402.17497*.

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

Jiahao Zhang, Haiyang Zhang, Dongmei Zhang, Yong Liu, and Shen Huang. 2023. End-to-end beam retrieval for multi-hop question answering. *arXiv preprint arXiv:2308.08973*.

A Appendix

A.1 Proposition Graph Construction Details

The PropRAG Proposition graph $G = (V, E)$ is constructed to facilitate reasoning over interconnected propositions. The vertex set V comprises two main types of nodes:

- V_{entity} : Nodes representing entities extracted from the text corpus.
- $V_{passage}$: Nodes representing the original text passages from which propositions and entities were derived.

The edge set E includes the following key types, designed to capture relationships within and between propositions, and to link entities back to their source contexts:

- **Entity Clique Edges (Implicit Proposition Hyper-edge):** For each proposition p extracted from the corpus, which contains a set of entities $\mathcal{E}(p)$, we add undirected edges connecting all pairs of distinct entities $\{e_i, e_j\}$ such that $e_i, e_j \in \mathcal{E}(p)$ and $e_i \neq e_j$. This forms a clique (a fully connected subgraph) among all entities co-occurring within that single proposition. This clique structure implicitly represents the proposition p as a hyper-edge, contextually linking all its constituent entities together, rather than relying on potentially ambiguous predicate-labeled edges between only two entities as in traditional triple stores.
- **Passage Containment Edges:** An undirected edge connects each entity node $e \in V_{entity}$ to the passage node $d \in V_{passage}$ corresponding to the text passage from which entity e (and its associated propositions) were originally extracted. These edges ground entities and propositions in their source documents.
- **Synonymy Edges:** An undirected edge connects two distinct entity nodes $e_i, e_j \in V_{entity}$ if their pre-computed embeddings are highly similar, i.e., $\text{sim}(\text{emb}(e_i), \text{emb}(e_j)) \geq \tau_{syn}$, where τ_{syn} is a predefined similarity threshold. These edges help bridge different textual mentions of the same underlying concept.

This graph structure allows for traversal algorithms (like PPR and beam search) to navigate through the rich context embedded in propositions (via the entity cliques/hyper-edges) and to connect entities

back to their original passages, facilitating comprehensive evidence aggregation.

A.2 Implementation Details

PropRAG leverages Llama-3.3-70B-Instruct for offline proposition extraction (and as the final QA reader for experiments) and NV-Embed-v2 (7B) as the base embedding model for passages, entities, and propositions, ensuring consistency with the HippoRAG 2 baseline setup. Default parameters used in PropRAG experiments are as follows:

- Beam width for path discovery (B): 4
- Maximum path length for beam search (L_{max}): 3
- Initial PPR damping factor (Stage 1, exploration): 0.75
- Final PPR damping factor (Stage 2, exploitation): 0.45
- Number of passages in subgraph (K): 50
- Number of top paths for exact scoring (beam search internal re-ranking) (M): 40
- Number of top initial seeds for final PPR ($B_{initial}$): 5
- Number of top propositions to select seeds from for final PPR ($P_{initial}$): B (Beam width)
- Number of top beam-derived seeds for final PPR (B_{beam}): 5
- Number of top beam-derived paths to select seeds from for final PPR (P_{beam}): 5
- Synonymy embedding similarity threshold (τ_{syn}): 0.8
- Number of initial propositions for seeding Stage 1 PPR (N_{prop}): 20
- Number of initial entities from the top- N_{prop} propositions for seeding Stage 1 PPR (N_{entity}): 40
- Weight for passage direct retrieval score in final PPR ($\lambda_{passage}$): 0.05

These parameters were determined based on empirical performance on development sets or adopted from common practices in related research where applicable. The choice of $L_{max} = 3$ was based on achieving the best average F1 score across development datasets.

A.3 LLM Prompts

This section details the prompts used for entity and proposition extraction with Llama-3.3-70B-Instruct, crucial for the offline indexing phase of PropRAG.

A.3.1 Entity Extraction Prompt

This prompt is designed for inclusive entity identification. Unlike strict Named Entity Recognition (NER) often used for triple extraction, this step aims to capture a broader set of concepts relevant for constructing rich propositions. It explicitly asks the LLM to identify named entities, dates, important generic entities, and entities involved in predicate relations. This provides a comprehensive list for the subsequent proposition generation phase, which only uses entities from this pre-identified set. (The prompt is shown in Figure 4)

A.3.2 Proposition Extraction Prompt

This prompt guides the LLM to decompose a passage into atomic, yet contextually complete, propositions. It strictly uses the entities identified in the previous step (Figure 4). The core focus is on maintaining high fidelity by preserving complex relationships, conditions, and the full context, which are often lost or oversimplified in traditional triple extraction processes. (The prompt is shown in Figure 5)

A.4 Proposition Graph Statistics

The proposition graphs constructed for each dataset vary in size and complexity, reflecting the nature of the underlying corpora. Table 5 provides key statistics for the graphs used in our experiments. These include the number of extracted propositions, the number of passage nodes (corresponding to unique passages in the corpus subset), the number of unique entity nodes identified, and the total number of edges in the constructed graph (encompassing entity clique edges, passage containment edges, and synonymy edges).

A.5 Cost and Efficiency

The offline indexing phase of PropRAG involves LLM-based proposition and entity extraction, as well as embedding computation. For embedding, we run a float16 version of NV-Embed-v2 on an NVIDIA RTX 4090 GPU. For proposition and entity extraction, we utilize the Llama-3.3-70B-Instruct model via Nebius AI Studio’s API endpoint. Processing each passage for proposition and

entity extraction takes approximately 2 seconds with this setup. As a concrete example, indexing the 11,656 passages from the MuSiQue dataset completed within approximately 40 minutes, at a monetary cost of around \$4 USD using the API.

The token cost for the offline LLM-based proposition extraction is an important consideration. Table 6 compares the input and output token counts for PropRAG on the MuSiQue dataset against those reported for other structure-augmented RAG methods by Gutiérrez et al. (2025) for their respective offline knowledge structuring phases.

PropRAG’s token cost for proposition extraction is higher than methods like HippoRAG 2 (which uses OpenIE for triple extraction, often less LLM-intensive) or RAPTOR (which focuses on summarization). This is attributable to the detailed instructions and the generation of full-sentence propositions, which are richer but require more tokens. However, PropRAG’s costs are considerably lower than methods like LightRAG and GraphRAG, which may involve more extensive LLM-based processing for their graph construction or summarization steps. The trade-off is between the upfront offline cost of generating high-fidelity propositions and the downstream benefits in retrieval accuracy and the avoidance of online LLM calls during retrieval. The online retrieval phase of PropRAG, involving PPR and beam search, is entirely LLM-free and computationally efficient, relying on pre-computed embeddings and graph operations.

A.6 Entity Score Calculation from Paths

After the beam search identifies a set of high-scoring proposition paths (as detailed in Section 5.3), PropRAG determines the importance of individual entities based on their participation in these paths. This entity scoring is crucial for generating the final set of seed nodes (S_{final}) used in the Stage 2 PPR (Section 5). The scoring process adheres to the following principles:

1. **Path Score Inheritance:** Each proposition within an identified path is considered to have the same relevance score as the overall path it belongs to.
2. **Entity Score Aggregation:** An entity’s total score is determined by summing the scores of all propositions (and thus, all paths) in which it appears. If an entity is part of multiple high-scoring paths or multiple propositions within a

Entity Extraction Prompt

Instruction: Your task is to extract entities from the given paragraph. Respond with a JSON dictionary only, with a "entities" key that maps to an non-empty list of entities. All named entities and dates must be included in the list. All generic entities important to the theme of the passage must be included in the list. All entities that is involved in a predicate relation to the above entities must be included in the list. All dates must be included in the list.

Demonstration:

Example Paragraph: Radio City Radio City is India's first private FM radio station and was started on 3 July 2001. It plays Hindi, English and regional songs. Radio City recently forayed into New Media in May 2008 with the launch of a music portal - PlanetRadiocity.com that offers music related news, videos, songs, and other music-related features.

Example Output:

```
{"entities":  
  ["Radio City", "India", "private FM radio station", "3 July 2001", "Hindi",  
   "English", "New Media", "May 2008", "PlanetRadiocity.com", "music portal",  
   "news", "videos", "songs"]  
}
```

Input Format:

Passage: \${passage}

Figure 4: LLM prompt for Entity Extraction. This prompt aims for comprehensive entity identification beyond standard NER.

Proposition Extraction Prompt

Instruction:

Your task is to analyze text passages and break them down into precise, atomic propositions using a specified list of named entities. A proposition is a fully contextualized statement that expresses a single unit of meaning with complete specificity about the relationships described.

For each proposition:

1. Extract a complete, standalone statement that preserves the full context
2. Use ONLY the entities provided in the named_entities list - do not introduce new entities
3. Ensure each proposition contains only ONE claim or relationship
4. Be extremely specific about which entities are involved in each relationship
5. Maintain clear causal connections between related statements

Respond with a JSON object containing a list of propositions, where each proposition is an object with:

- "text": The full proposition text as a complete, contextualized statement
- "entities": An array of entities from the named_entities list that appear in that proposition

Critical Guidelines:

- ONLY use entities from the provided named_entities list
- Make relationships explicit and specific - clarify exactly which entities relate to which other entities
- Clarify precisely which entity a modifier applies to (e.g., specify which product had "80% improvement")
- Establish clear connections between related facts (e.g., "Adobe optimized their applications FOR THE M1 CHIP")
- Connect comparative statements to their specific reference points (e.g., "Adobe's applications on the M1 chip improved by 80% compared to Intel-based Macs")
- Preserve temporal information and causal relationships between events
- Make each proposition stand alone with all necessary context
- Include ALL relevant entities from the named_entities list in both the proposition text and entities array
- Ensure the collection of propositions captures ALL meaningful information in the passage

Demonstration:

Passage: In 2020, after Apple launched the M1 chip, major software companies like Adobe optimized their applications, improving performance by up to 80% compared to Intel-based Macs.

Named entities:

```
["Apple", "M1 chip", "2020", "Adobe", "Adobe's applications", "Intel-based
    ↪ Macs", "80% performance improvement"]

{
  "propositions": [
    {
      "text": "Apple launched the M1 chip in 2020.",
      "entities": ["Apple", "M1 chip", "2020"]
    },
    {
      "text": "Adobe optimized their applications specifically for the M1 chip
        ↪ after its launch.",
      "entities": ["Adobe", "Adobe's applications", "M1 chip"]
    },
    {
      "text": "Adobe's applications running on the M1 chip improved performance
        ↪ by up to 80% compared to the same applications running on Intel-
        ↪ based Macs.",
      "entities": ["Adobe", "Adobe's applications", "M1 chip", "80% performance
        ↪ improvement", "Intel-based Macs"]
    }
  ]
}
```

Input Format:

```
Passage: ${passage}
Named entities: ${entities_json_list}
```

Figure 5: LLM prompt for Proposition Extraction. This prompt emphasizes contextual completeness and adherence to pre-identified entities.

Table 5: Statistics of Constructed Proposition Graphs per Dataset.

Statistic	NQ	PopQA	MuSiQue	2Wiki	HotpotQA
# Propositions	55536	57624	59028	30099	53566
# Passage Nodes	9633	8676	11656	6119	9811
# Entity Nodes	62368	73577	76928	43444	75608
# Total Edges	1.27M	1.17M	1.34M	0.86M	1.31M

Table 6: Offline LLM Token Costs (Input/Output) for Knowledge Structuring on MuSiQue Dataset (Millions of Tokens). Baseline data from [Gutiérrez et al. \(2025\)](#).

Method	Input Tokens (M)	Output Tokens (M)
RAPTOR	1.7	0.2
HippoRAG 2	9.2	3.0
PropRAG (Ours)	16.5	4.6
LightRAG	68.5	18.3
GraphRAG	115.5	36.1

single path, its score accumulates, reflecting its centrality and repeated relevance.

3. Emphasis on Connecting Entities: The scoring mechanism gives additional weight to entities that form crucial links within a reasoning path, particularly for synonymous connections.

- **Synonymous Connections Boost:** When a proposition P_A (containing entity E_A) connects to proposition P_B (containing entity E_B) via a synonymous link where $E_A \approx E_B$, the *connected entity* (E_B in P_B) receives an additional score increment equivalent to the path's score. This effectively elevates the importance of E_B , treating it as a strong continuation of a central concept from P_A . The rationale is that E_B is vital for identifying the passage associated with P_B . The original connecting entity (E_A in P_A) contributes its score through its presence in P_A but does not receive this specific connection-based score enhancement itself. If P_A was connected from a preceding proposition, its own central entities would have been accounted for similarly.

- **Exact Connections:** Entities that are shared exactly between two consecutive propositions in a path (forming an exact connection) naturally contribute to the score aggregation through their appearance in both propositions. Their role as direct bridges is thus inherently emphasized by the summation of scores from both propo-

sitions they are part of.

4. Initial Proposition Entities: For entities appearing in the very first proposition of a path (which do not have a preceding "connection" within that path), their initial relevance is captured through the "exploration seeds" ($S_{initial}$). Many entities from these initial top query-relevant propositions are directly considered as exploration seeds. This ensures their potential importance is factored into the final seed set, even if they don't benefit from the connection-based score enhancements that apply to entities deeper within a path.

Following the aggregation of scores for all entities involved in the discovered paths, the entities are ranked by their total accumulated scores. This ranked list is then used to select the top- B_{beam} "exploitation seeds." These exploitation seeds, rich in path-derived relevance, are combined with the "exploration seeds" ($S_{initial}$) to form the final seed set S_{final} for the concluding PPR stage, ensuring a comprehensive and robust final ranking of evidence passages.