DEPENDENCY-ALIGNED CHUNK-TO-EVENT GRAPH RETRIEVAL FOR MULTI-HOP QUESTION ANSWERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-hop question answering requires retrieving complementary evidence distributed across documents, including intermediate facts that may be weakly related to the original question. Dense retrievers often miss such evidence, while graph-based methods may traverse corpus connections that are irrelevant to the required reasoning structure. We argue that subquery dependencies provide an explicit structural prior for determining which evidence regions should be connected. Based on this insight, we introduce DreamRAG, a chunk-to-event retrieval framework that decomposes a question into a dependency graph and represents each subquery-specific seed region with a virtual supernode. Its core mechanism, Dependency-Aligned Supernode Bridging, maps predicted dependencies to targeted minimum-cost path searches between the corresponding supernodes. DreamRAG further decouples chunk-level navigation from fine-grained event-level evidence selection. Experiments on HotpotQA, 2WikiMultihopQA, and MuSiQue demonstrate that DreamRAG significantly improves macro-average EM and F1 over strong dense and graph-based RAG baselines. Our code is available at https://anonymous.4open.science/r/Dream-9184/.

1 INTRODUCTION

Multi-hop question answering (MHQA) requires a system to retrieve and combine complementary facts distributed across multiple documents (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022; Welbl et al., 2018; Chen & Durrett, 2019). Unlike single-hop retrieval, where a passage can often be evaluated by its direct relevance to the question, multi-hop retrieval must recover intermediate evidence whose utility emerges only through its relation to other reasoning steps. Consider the question "What nationality is the author of the novel that inspired the film directed by X?" Answering it requires identifying the film, its source novel, the author, and finally the author's nationality. Evidence about the author may have little direct similarity to the original question, yet it is indispensable to the answer. A successful retrieval system must therefore recover not only individually relevant passages, but also the connections among evidence required by the question's latent reasoning structure (Lewis et al., 2020; Gao et al., 2023).

Existing retrieval paradigms capture only part of this requirement. Dense retrievers rank passages primarily by semantic similarity to the input question (Lewis et al., 2020; Karpukhin et al., 2020). Because candidate passages are largely scored independently, evidence for an intermediate step may be overlooked when it is weakly related to the original question (Trivedi et al., 2023). Graph-based RAG methods mitigate this limitation by organizing corpus elements through entities, relations, communities, or associative links (Edge et al., 2024; Guo et al., 2025; Gutiérrez et al., 2025). More recent methods further exploit higher-order relations or retrieve and prune relational paths (Lien et al., 2026; Chen et al., 2026). These approaches improve cross-document reachability, but corpus connectivity alone does not determine which evidence regions should be connected for a particular question. Unconstrained or weakly conditioned traversal may connect semantically plausible regions that do not participate in the same reasoning dependency, introducing irrelevant intermediate nodes and redundant context.

Question decomposition provides a complementary source of structure. Iterative approaches expose intermediate information needs through follow-up questions or interleaved reasoning and retrieval (Press et al., 2023; Trivedi et al., 2023). More structured methods organize complex questions using decomposition trees, progressive dependencies, or query graphs (Li et al., 2024; Yixing

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.

et al., 2024; Zhu et al., 2025). These studies demonstrate that decomposed subqueries are useful retrieval targets. However, an underexplored question remains: *how should the dependencies among subqueries constrain graph traversal over the corpus?* We argue that decomposition should determine not only *what* to retrieve, but also *which retrieved evidence regions should be bridged*. Evidence regions associated with dependent subqueries should be considered for structural connection, whereas independent branches—such as the two sides of a comparison question—need not be connected merely because they contribute to the same final answer.

Based on this insight, we introduce **DreamRAG** (**D**ependency-aligned **R**etrieval with **E**vent-**A**ware **M**ulti-hop generation), a chunk-to-event graph retrieval framework that uses subquery dependencies as an explicit structural prior for evidence acquisition. DreamRAG first decomposes a question into subqueries and a directed dependency graph, then retrieves a set of seed chunks for each subquery. Rather than treating all seeds as interchangeable endpoints, DreamRAG preserves their subquery assignments by introducing one query-level virtual supernode for each seed set.

Our central mechanism, **Dependency-Aligned Supernode Bridging (DASB)**, maps every predicted subquery dependency to a corresponding supernode-to-supernode routing task. For each dependency edge, DASB searches the chunk graph for a minimum-cost path between the associated supernodes, using an objective that balances path compactness with query-aware semantic relevance. Because each supernode is connected to all seeds retrieved for its subquery, a single graph search jointly considers alternative seed endpoints without explicitly issuing a separate search for every seed pair. Moreover, paths are searched only for subquery pairs connected by an explicit dependency. DASB thus converts the reasoning topology of the question into a set of targeted evidence-bridging operations, instead of using query structure merely to order independent retrieval calls.

DreamRAG further separates the evidence granularity needed for graph navigation from that needed for answer generation. Chunks retain sufficient context and entity information to serve as stable routing units, but they often contain sentences unrelated to the target reasoning chain. DreamRAG therefore augments the chunk graph with an event index containing fine-grained, self-contained factual units extracted from each chunk. After DASB identifies the relevant chunk region, the retrieved chunks are expanded into events and ranked against both the original question and its subqueries. This chunk-to-event design assigns distinct roles to the two layers: chunks preserve cross-document reachability, while events provide compact evidence to the generator. In this way, dependency-aligned bridging determines *where* to search, and event-level ranking determines *what* to present. Our main contributions are as follows:

- **Dependency structure as a graph-retrieval prior.** We formulate the dependencies among decomposed subqueries as an explicit prior for deciding which subquery-specific evidence regions should be connected during corpus-graph retrieval.
- **Dependency-Aligned Supernode Bridging.** We introduce DASB, which represents each subquery's seed set with a virtual supernode and maps each predicted dependency to a targeted minimum-cost bridging task. This preserves the correspondence between reasoning steps and retrieved evidence while avoiding unnecessary searches between unrelated subqueries.
- **Chunk-to-event dual-layer retrieval.** We decouple structural navigation from evidence presentation: chunks act as routing units for cross-document traversal, whereas fine-grained events serve as compact, query-aware evidence for answer generation.
- **Multi-benchmark evaluation and structural analysis.** Experiments on three datasets demonstrate consistent improvements over dense and graph-based RAG baselines. Ablations and query-graph analyses further characterize the contributions and robustness of the proposed components.

2 Related Work

Multi-Hop Retrieval and Question Decomposition. Retrieval-Augmented Generation (RAG) typically retrieves passages according to their semantic relevance to an input question and conditions a language model on the resulting context (Lewis et al., 2020; Karpukhin et al., 2020; Gao et al., 2023). Retrieve–rerank–generate pipelines such as Re2G improve candidate selection by reranking an initial retrieval set (Glass et al., 2022). Nevertheless, these methods largely evaluate passages independently. In multi-hop question answering, intermediate evidence may have weak similarity to the original question and become useful only through its relation to other evidence. Increasing the retrieval budget can improve coverage, but does not explicitly recover these relations or organize

passages according to a question-specific reasoning structure. Self-RAG adaptively determines when to retrieve and uses self-reflection to assess retrieved evidence (Asai et al., 2024), while long-context studies show that providing more text does not guarantee effective evidence use (Liu et al., 2024).

Question decomposition makes intermediate information needs explicit. Min et al. decompose multi-hop questions into simpler subquestions and rescore their answers (Min et al., 2019), whereas QDMR represents reasoning steps as an ordered natural-language sequence (Wolfson et al., 2020). Self-Ask generates follow-up questions (Press et al., 2023); IRCoT interleaves retrieval with chain-of-thought reasoning (Trivedi et al., 2023); and ReAct interleaves reasoning traces with external actions (Yao et al., 2023). More structured methods explicitly model relationships among subquestions. KELDaR uses a decomposition tree to guide knowledge-graph retrieval (Li et al., 2024), Chain-of-Question performs answer-conditioned progressive decomposition (Yixing et al., 2024), and RoleRAG organizes retrieval and reasoning through a query graph (Zhu et al., 2025). These methods show the value of decomposing complex information needs. DreamRAG goes beyond using subqueries as retrieval prompts or an execution plan: it treats their dependency graph as a routing prior that determines which subquery-specific evidence regions should be connected in the corpus graph.

Graph-Based Retrieval and Structured Reasoning. Graph-based RAG represents corpus associations explicitly to improve cross-document reachability. GraphRAG constructs entity-centric graphs and hierarchical communities (Edge et al., 2024), LightRAG combines graph and vector retrieval (Guo et al., 2025), and HippoRAG2 supports associative retrieval over graph-based memory (Gutiérrez et al., 2025). Related evidence-organization methods include RAPTOR, which recursively clusters and summarizes chunks (Sarthi et al., 2024), and HippoRAG, which combines a knowledge graph with Personalized PageRank (Gutiérrez et al., 2024). HyperRAG models higher-order relations with hypergraphs and constructs query-conditioned relational chains (Lien et al., 2026), while PathRAG retrieves relational paths and prunes redundant graph information (Chen et al., 2026). Although these methods exploit corpus structure, their traversal is primarily governed by corpus relations and aggregate query relevance, without explicitly preserving which subquery produced each seed region or which pairs of regions should be connected. A complementary line of work represents reasoning itself as a graph. ReasonGraphQA introduces fine-grained evidence graphs for questions involving multiple chains and hops (Zhu et al., 2024), while GRS-QA models reasoning pathways using textual nodes and logical-flow edges (Yone et al., 2026).

DreamRAG differs from prior graph-retrieval methods in two coupled design choices. First, it preserves the assignment from each subquery to its retrieved seed chunks through a query-level virtual supernode. This supernode represents an evidence region rather than a single fixed endpoint. Second, Dependency-Aligned Supernode Bridging (DASB) invokes a minimum-cost path search only for supernode pairs associated with explicit dependency edges. A search can therefore select among multiple candidate seed endpoints while avoiding bridging requests between unrelated subqueries. The resulting graph search is conditioned not only on what is relevant to the overall question, but also on which reasoning steps depend on one another.

3 METHOD

Given a multi-hop question q, our goal is to retrieve a compact set of complementary facts that supports the intermediate reasoning steps required to answer q. This setting presents two challenges. First, evidence for an intermediate step may be only weakly related to the original question. Second, evidence retrieved for different steps should not be connected indiscriminately: the useful connections depend on the logical dependencies among the corresponding information needs. To address these challenges, we propose **DreamRAG** (**D**ependency-aligned **R**etrieval with **E**vent-**A**ware **M**ulti-hop generation), which uses the dependency structure of a decomposed question as an explicit routing prior. DreamRAG consists of an offline indexing stage and an online retrieval stage. During indexing, we construct (i) a chunk graph for cross-document navigation and (ii) an event index containing fine-grained factual units. At retrieval time, DreamRAG: (i) decomposes the question into single-hop subqueries and predicts their dependency graph; (ii) retrieves a set of seed chunks for each subquery and associates each seed set with a virtual supernode; (iii) performs a graph search only between supernodes whose subqueries are connected by a predicted dependency; and (iv) expands the resulting chunks into events and selects a compactevent-level context for answer generation. This design separates two roles that are typically conflated in graph-based retrieval. Chunks retain the

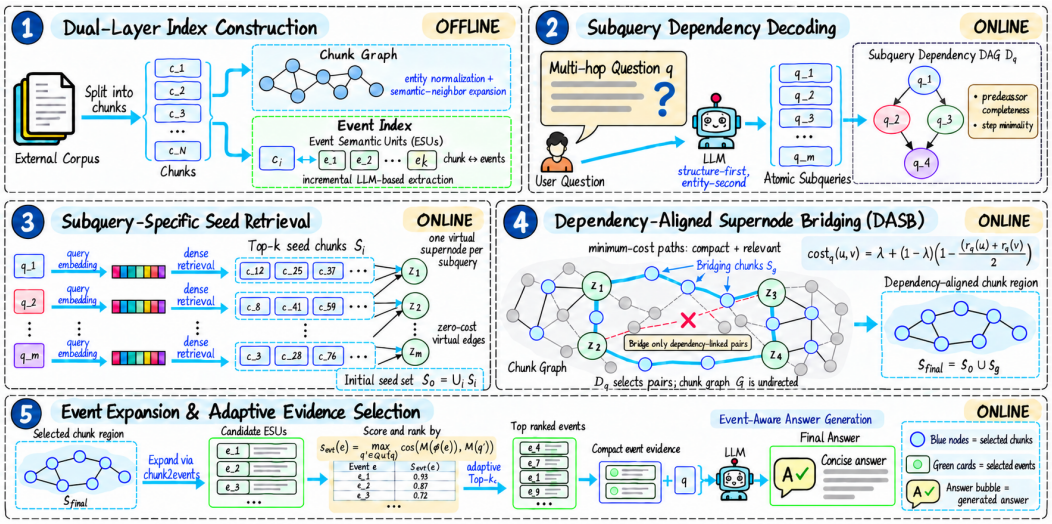

Figure 1: Overview of DreamRAG

context and entity information required for reliable graph navigation, whereas events provide concise factual evidence to the generator. Figure 1 illustrates the overall framework.

3.1 Dual-Layer Index Construction

The foundation of DreamRAG is a dual-layer index that jointly supports structural traversal and fine-grained semantic extraction. The first layer is a chunk graph that preserves structural connectivity across documents. The second layer is an event index that exposes fine-grained factual statements inside each chunk.

Chunk Graph Construction. Let $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$ denote the complete set of text chunks obtained by segmenting the external corpus. We construct an undirected chunk graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{C}$. An edge indicates two chunks share at least one entity after normalization and semantic expansion. The resulting graph defines the corpus topology used for cross-document navigation.

For each chunk c_i, let K_i denote its original entity set extracted by named entity recognition. We first apply an entity normalization function $\mathrm{Norm}(\cdot)$, which lowercases entity mentions, removes punctuation, and merges duplicated normalized strings:

$$K_i^{\mathrm{norm}} = \{\mathrm{Norm}(k) \mid k \in K_i\}. \tag{1}$$

To alleviate missing connections caused by lexical variation, we further expand each normalized entity with its semantic neighbors. For every entity k, ColBERT retrieves the top-h candidates from the global entity vocabulary. Given the ColBERT matching score $s_{\mathrm{col}}(k, k')$, we normalize it by the maximum score returned for k:

$$\bar{s}_{\mathrm{col}}(k, k') = \frac{s_{\mathrm{col}}(k, k')}{\max_z s_{\mathrm{col}}(k, z)}. \tag{2}$$

We retain only candidates whose normalized scores exceed the entity-expansion threshold τ_e:

$$\mathrm{NN}_h(k; \tau_e) = \{k' \in \mathrm{Top}_h(k) \mid \bar{s}_{\mathrm{col}}(k, k') > \tau_e\}. \tag{3}$$

Since the original nearest-neighbor relation is directional, we symmetrize the retained relations. The resulting semantic neighborhood of entity k is

$$\mathrm{SNN}_h(k; \tau_e) = \mathrm{NN}_h(k; \tau_e) \cup \{k' \mid k \in \mathrm{NN}_h(k'; \tau_e)\}. \tag{4}$$

The augmented entity set associated with chunk c_i is then defined as

$$\widetilde{K}_i = K_i^{\text{norm}} \cup \bigcup_{k \in K_i^{\text{norm}}} \text{SNN}_h(k; \tau_e). \tag{5}$$

For two chunks c_i and c_j, if $\widetilde{K}_i \cap \widetilde{K}_j \neq \varnothing$,then an edge exists between c_i and c_j, namely,

$$(c_i, c_j) \in \mathcal{E} \quad \Longleftrightarrow \quad \widetilde{K}_i \cap \widetilde{K}_j \neq \varnothing. \tag{6}$$

Event Index Construction. Chunks retain the context and entities needed to establish reliable graph connections, but they may be too coarse for answer generation. A retrieved chunk can contain one relevant statement together with several unrelated facts. We therefore decompose each chunk into Event Semantic Units (ESUs). Let the events extracted from chunk c_i be $\mathcal{H}_i = \{e_{i1}, e_{i2}, \ldots, e_{in_i}\}$. Each event is represented as $e = (x_e, p_e, k_e)$, where x_e is a self-contained factual statement, p_e contains concise descriptions of the information needs that the fact may answer, and k_e contains salient entities and concepts. We require factual statements to contain explicit subjects and to avoid unresolved pronouns so that events remain interpretable independently of their source chunks.

We use an incremental LLM-based procedure to improve factual coverage. The first pass extracts salient atomic facts from a source chunk. Each subsequent pass receives the source chunk and the events extracted so far and is asked to identify salient facts that remain uncovered. Extraction terminates when no new event is produced or when a predefined extraction budget is reached. We retain the bidirectional mappings

$$\text{chunk2events}(c_i) = \mathcal{H}_i, \qquad \text{event2chunk}(e_{ij}) = c_i. \tag{7}$$

These mappings allow DreamRAG to navigate over contextualized chunks and subsequently recover fine-grained facts from the selected chunk region. We do not construct a separate event-level traversal graph: chunks support structural navigation, whereas events support evidence presentation.

3.2 Dependency-Aligned Graph Retrieval

Subquery Dependency Decoding. Given an input question q, DreamRAG predicts a set of atomic subqueries $\mathcal{Q} = \{q_1, q_2, \ldots, q_m\}$ and a directed acyclic dependency graph $D_q = (\mathcal{V}_q, \mathcal{A}_q)$. Each node $v_i \in \mathcal{V}_q$ corresponds to subquery q_i. A directed edge $(v_i, v_j) \in \mathcal{A}_q$ indicates that the information requested by q_i is a prerequisite for resolving q_j. The edge direction represents logical dependency in the question rather than a required traversal direction in the undirected corpus graph.

We employ a *structure-first, entity-second* decoding strategy. The model first outlines the abstract dependency skeleton with masked entities, and subsequently instantiates the concrete entities. We enforce two constraints during decomposition. First, *predecessor completeness*: every information requirement of a subquery must be available either from the original question or from one of its predecessor subqueries. Second, *step minimality*: the decomposition should avoid redundant subqueries whose information is already available from previous steps. These constraints yield a compact dependency graph that is sufficiently expressive for retrieval but does not introduce unnecessary traversal edges.

Subquery-Specific Seed Retrieval. For each decoded subquestion q_i, we independently perform dense retrieval to identify its top-k seed chunks, denoted as $S_i = \text{TopK}(q_i)$. The initial seed set is:

$$S_0 = \bigcup_{i=1}^{m} S_i.$$

To evaluate the semantic utility of any chunk c within the broader reasoning context, we define a query-aware relevance score:

$$r_q(c) = \max_{q' \in \mathcal{Q} \cup \{q\}} \cos\left(M(c), M(q')\right),$$

where $M(\cdot)$ is the dense embedding encoder.

Dependency-Aligned Supernode Bridging. The central retrieval mechanism of DreamRAG is Dependency-Aligned Supernode Bridging (DASB). DASB maps every predicted dependency edge in D_q to a targeted connection request between the corresponding seed regions in the chunk graph. This

design preserves an explicit correspondence among subqueries, their retrieved seed regions, and the pairs of regions that should be bridged. Consequently, the dependency graph controls not only the retrieval targets but also the topology of the subsequent corpus-graph search.

For each subquery q_i, we introduce a virtual supernode z_i and connect it to every seed chunk in S_i with a zero-cost virtual edge: $\mathcal{E}^{\text{virt}} = \{(z_i, c) \mid c \in S_i,\ 1 \le i \le m\}$. Let $\mathcal{Z} = \{z_1, z_2, \ldots, z_m\}$. The query-augmented graph is $\mathcal{G}_q^v = \big(\mathcal{V} \cup \mathcal{Z}, \mathcal{E} \cup \mathcal{E}^{\text{virt}}\big)$. A virtual supernode does not aggregate, summarize, or re-encode its seed chunks. Instead, it provides a multi-source and multi-target interface to the candidate evidence region associated with a subquery. At inference time, DreamRAG assigns each existing edge a cost conditioned on the current question. For each existing chunk-level edge $(u, v) \in \mathcal{E}$, its online traversal cost is

$$\text{cost}_q(u, v) = \lambda + (1 - \lambda)\left(1 - \frac{r_q(u) + r_q(v)}{2}\right), \tag{8}$$

where $\lambda \in [0, 1]$ balances an explicit per-edge penalty against query-aware semantic relevance. Because cosine similarity is bounded, the resulting edge costs are nonnegative and can be optimized with Dijkstra's algorithm. The two terms serve complementary purposes. The constant λ charges every traversed edge and explicitly favors shorter paths. The second term penalizes transitions whose endpoint chunks have low relevance to the original question and its subqueries. For a path $\pi = (x_0, x_1, \ldots, x_L)$, the total cost is $C_q(\pi) = \sum_{\ell=0}^{L-1} \text{cost}_q(x_\ell, x_{\ell+1})$, where virtual edges contribute zero cost. For every predicted dependency $(v_i, v_j) \in \mathcal{A}_q$, DASB computes a minimum-cost path between the corresponding supernodes:

$$\pi_{ij}^* = \operatorname*{argmin}_{\pi : z_i \rightsquigarrow z_j} C_q(\pi). \tag{9}$$

Although the query dependency graph is directed, the chunk graph is undirected. The direction $v_i \to v_j$ records that q_i supplies a prerequisite for q_j. DASB uses this dependency to decide that the associated evidence regions should be bridged, while the connecting corpus path may be traversed in either direction. Let $\text{Chunk}(\pi)$ denote the real chunk nodes on a path, excluding its virtual endpoints. The bridging chunk set is

$$S_g = \bigcup_{(v_i, v_j) \in \mathcal{A}_q} \text{Chunk}(\pi_{ij}^*). \tag{10}$$

If no corpus path exists between two dependency-associated seed regions, that dependency contributes no additional bridging chunks, while its original seed chunks remain available. The final chunk evidence set is $S_{\text{final}} = S_0 \cup S_g$.

Event Expansion and Adaptive Evidence Selection. DASB returns a dependency-aligned set of chunks, but directly supplying all of these chunks to the generator may introduce irrelevant or repetitive context. We therefore expand the selected chunks into their associated events: $\mathcal{H}_{\text{cand}} = \bigcup_{c_i \in S_{\text{final}}} \text{chunk2events}(c_i)$. Let $\phi(e)$ denote the textual representation used to embed an event, comprising its factual content, potential questions, and salient keywords. We score each candidate event against the original question and every subquery:

$$s_{\text{evt}}(e) = \max_{q' \in \mathcal{Q} \cup \{q\}} \cos\left(M(\phi(e)), M(q')\right). \tag{11}$$

The max-similarity objective retains an event when it strongly supports at least one intermediate step, even if it has low similarity to the original question. This is useful for bridge evidence whose relevance becomes explicit only after decomposition. Questions with more reasoning steps may require a larger evidence budget. We therefore determine the event budget from the number of subqueries:

$K_e = \min\left(K_{\max}, \max\left(K_{\min}, k_{\text{per}} \cdot m\right)\right)$. The final event evidence set is

$$\mathcal{H}_{\text{final}} = \text{TopK}_{e \in \mathcal{H}_{\text{cand}}}\left(s_{\text{evt}}(e), K_e\right). \tag{12}$$

The two layers consequently perform distinct operations: DASB identifies where complementary evidence may lie at the chunk level, while event-level ranking determines which fine-grained facts should be presented to the generator.

3.3 EVENT-AWARE ANSWER GENERATION

Finally, the selected events are ordered by their relevance scores, serialized as a compact evidence context, and supplied to the answer generator together with the original question. Let F_{gen} denote the generator. The predicted answer is $\widehat{a} = F_{\text{gen}}\left(q, \mathcal{H}_{\text{final}}\right)$. The generator is instructed to answer the question using the retrieved evidence and to return a concise response. No additional corpus-graph traversal is performed during generation.

4 EXPERIMENTS

We evaluate DreamRAG along four research questions:

- **RQ1:** Does dependency-aligned graph retrieval improve end-to-end multi-hop question answering over dense and graph-based RAG baselines?
- **RQ2:** Does the dual-layer index provide the intended structural and semantic benefits?
- **RQ3:** How much does each component of DreamRAG contribute to the final performance?
- **RQ4:** Is DreamRAG robust to retrieval design choices such as seed budget, event representation, embedding model, decomposer, and the DASB trade-off parameter?

Datasets. We evaluate DreamRAG on three standard multi-hop question answering benchmarks: MuSiQue (answerable) (Trivedi et al., 2022), 2WikiMultihopQA, abbreviated as 2WikiMQA (Ho et al., 2020), and HotpotQA (Yang et al., 2018). These datasets cover different types of compositional reasoning. HotpotQA emphasizes bridge and comparison reasoning over Wikipedia passages; 2WikiMQA contains multi-hop questions grounded in Wikidata-derived relations; and MuSiQue is constructed by composing multiple single-hop questions, making it particularly challenging for systems that fail to preserve intermediate reasoning dependencies.

Following the evaluation protocol adopted by HippoRAG2 (Gutiérrez et al., 2025), we sample 1,000 questions from the validation split of each dataset to control the total experimental cost. For each benchmark, we construct the retrieval corpus from the union of candidate passages associated with the selected questions, including both supporting and distractor passages. The resulting retrieval corpora contain 11,656 passages for MuSiQue, 6,119 passages for 2WikiMQA, and 9,221 passages for HotpotQA. The three benchmarks contain 3, 4, and 2 query types, respectively. This setting ensures that all retrieval-based methods operate over the same dataset-specific candidate pool under comparable conditions.

Baselines. We compare DreamRAG with three categories of baselines.

- **Direct Zero-shot LLM inference:** We evaluate GPT-3.5-turbo and GPT-4o-mini without external retrieval. These baselines measure how well the language model can answer multi-hop questions using only its parametric knowledge.
- **Vanilla retrieval-augmented generation:** We evaluate a standard dense retrieve-then-generate pipeline. Given an input question, the retriever selects the top-k passages according to semantic similarity and directly feeds them to the generator. We report VanillaRAG with $k \in \{1, 3, 5\}$. These baselines test whether increasing the number of independently retrieved passages is sufficient for recovering multi-hop evidence.
- **Graph-based retrieval-augmented generation:** We further compare with representative graph-enhanced RAG methods, including GraphRAG (Edge et al., 2024), LightRAG (Guo et al., 2025), HippoRAG2 (Gutiérrez et al., 2025), HyperRAG (Lien et al., 2026), and PathRAG (Chen et al., 2026). These methods exploit different forms of graph-structured information to improve evidence retrieval beyond independent dense retrieval. For HyperRAG, we evaluate its HyperRetriever variant. For methods that provide multiple configurations, we use the configuration designed for multi-hop retrieval whenever available.

Metrics. We report end-to-end question answering performance using Exact Match (EM) and token-level F1. EM measures whether the normalized predicted answer exactly matches the gold answer, while F1 measures token overlap between the prediction and the reference answer. We report EM and F1 on each dataset and compute the unweighted macro-average over the three datasets to evaluate overall multi-hop QA performance.

Table 1: Comparison of different methods on multi-hop question answering benchmarks. The best result in each column is shown in bold, and the second-best result is underlined.

Type	Model	HotpotQA		2WikiMQA		MuSiQue		Average	
		EM	F1	EM	F1	EM	F1	EM	F1
Direct Zero-shot LLM	GPT-3.5-turbo	28.1	39.6	29.1	36.0	5.3	15.8	20.8	30.5
	GPT-4o-mini	29.3	40.7	29.0	36.2	6.4	16.0	21.6	31.0
Vanilla RAG	VanillaRAG (Top-1)	27.8	36.8	5.7	6.8	11.4	18.0	15.0	20.5
	VanillaRAG (Top-3)	41.8	53.3	25.6	27.7	18.5	27.4	28.6	36.1
	VanillaRAG (Top-5)	44.3	56.2	31.8	34.2	21.6	31.6	32.6	40.7
Graph-based RAG	GraphRAG	44.4	56.8	32.9	39.4	20.4	30.7	32.6	42.3
	LightRAG	47.1	62.4	38.1	44.7	25.2	35.7	36.8	47.6
	HippoRAG2	<u>52.0</u>	<u>67.0</u>	<u>53.4</u>	<u>60.0</u>	<u>35.0</u>	<u>48.6</u>	<u>46.8</u>	<u>58.5</u>
	HyperRAG (HyperRetriever)	47.0	55.0	46.2	48.2	24.4	30.8	39.2	44.7
	PathRAG	38.9	57.3	20.3	37.7	17.4	31.3	25.5	42.1
Ours	DreamRAG	**58.4**	**75.4**	**58.5**	**71.6**	**41.5**	**57.0**	**52.8**	**68.0**

Table 2: Structural statistics of the chunk–event dual-layer index. GEC denotes gold-edge coverage, and ρ_{len} denotes the ratio between the average event length and the average chunk length.

Dataset	#Chunks	Density	GEC (%)	#Events	Events/Chunk	ρ_{len} (%)
HotpotQA	9,221	0.0425	–	62,011	6.72	13.85
2WikiMQA	6,119	0.0575	97.81	34,944	5.71	15.86
MuSiQue	11,656	0.0262	80.42	70,238	6.03	15.28

Implementation Details. To ensure controlled comparisons, we use `all-MiniLM-L6-v2` as the default embedding model for semantic retrieval across all applicable methods. For DreamRAG, the number of retrieved seed chunks per subquery is set to $k = 5$. All RAG methods use deepseek-v4-flash as the downstream generator with fixed configurations, ensuring that performance deltas primarily reflect retrieval quality rather than generator variance.

Exp-1: Overall Performance. Table 1 reports the end-to-end performance of all methods on HotpotQA, 2WikiMQA, and MuSiQue. The best result in each column is shown in bold, and the second-best result is underlined, and we find:

Comparison with dense retrieval. DreamRAG obtains a macro-average of 52.8 EM and 68.0 F1, compared with 32.6 EM and 40.7 F1 for VanillaRAG with five retrieved chunks. The gap is observed on all three datasets. These results are consistent with the motivation that independently ranked chunks may not recover or organize the intermediate evidence needed by multi-hop questions.

Comparison with graph-based retrieval. Among the reported baselines, HippoRAG2 is the strongest by macro-average performance, achieving 46.8 EM and 58.5 F1. DreamRAG improves over it by 6.0 EM and 9.5 F1 points on average. The improvement is consistent across the three datasets: 6.4 EM and 8.4 F1 on HotpotQA, 5.1 EM and 11.6 F1 on 2WikiMQA, and 6.5 EM and 8.4 F1 on MuSiQue.

Performance on compositional questions. MuSiQue is the most challenging benchmark in our evaluation, as reflected by the lower absolute scores of all methods. DreamRAG improves over HippoRAG2 from 35.0 to 41.5 EM and from 48.6 to 57.0 F1. This result is compatible with the hypothesis that explicit dependency modeling is particularly useful when a question contains heterogeneous multi-step compositions.

Exp-2: Structural Analysis of the Dual-Layer Index. We next examine whether the constructed chunk–event index exhibits the desired structural and semantic properties. At the chunk layer, graph density measures sparsity, while gold-edge coverage (GEC) measures how many annotated inter-passage reasoning transitions are preserved by the graph. At the event layer, we report the number of extracted events, the average number of events per chunk, and the relative event length ρ_{len}, defined as the ratio between the average event length and the average chunk length.

As shown in Table 2, the chunk graph is sparse but retains substantial reasoning connectivity. On 2WikiMQA, the graph covers 97.81% of annotated reasoning edges, indicating that most gold inter-

Table 3: Ablation study of evidence granularity, question decomposition, and graph-based evidence bridging. All configurations use $\lambda = 0.5$.

Configuration	HotpotQA		2WikiMQA		MuSiQue		Average	
	EM	F1	EM	F1	EM	F1	EM	F1
Chunk-level Evidence	56.1	73.8	56.9	70.3	38.9	53.1	50.6	65.7
w/o Question Decomposition	54.2	71.5	55.3	70.4	36.5	51.9	48.7	64.6
w/o Graph-based Bridging	56.6	73.8	57.6	71.4	41.2	56.6	51.8	67.3
Full DreamRAG: Event Evidence	**58.4**	**75.4**	**58.5**	**71.6**	**41.5**	**57.0**	**52.8**	**68.0**

passage transitions are explicitly represented by the constructed chunk graph. On MuSiQue, the GEC is 80.42%, which is lower but still substantial. This is consistent with the more compositional construction of MuSiQue, where some reasoning transitions may not be captured by direct entity-level associations. GEC is not reported for HotpotQA in the current evaluation.

The event layer provides a much finer granularity of evidence. Each chunk is decomposed into 5.71–6.72 events on average, while the average event length is only 13.85%–15.86% of the corresponding chunk length. These statistics show that events are substantially more compact than chunks. The dual-layer index therefore serves two complementary purposes: the chunk graph maintains cross-passage structural reachability, while the event index converts retrieved chunk regions into concise factual units for final ranking and generation.

Exp-3: Ablation Study. We conduct ablation studies to quantify the contribution of DreamRAG's main components. We evaluate three core design choices: the granularity of evidence supplied to the generator, the use of question decomposition, and dependency-aligned graph bridging. All variants use $\lambda = 0.5$. The chunk-level variant directly supplies retrieved chunks to the generator instead of expanding them into events. The variant without question decomposition retrieves using only the original question and returns the top-15 events. The variant without graph-based bridging bypasses DASB and ranks events expanded only from the initially retrieved seed chunks. The full model uses question decomposition, dependency-aligned bridging, and event-level evidence selection. Table 3 reports the component-removal results.

Question decomposition. Removing question decomposition produces the largest degradation, reducing macro-average EM by 4.1 points and F1 by 3.4 points. This result supports the role of decomposed information needs as a central retrieval signal. In the full system, decomposition affects both subquery-specific seed retrieval and the dependency structure used by DASB; this ablation therefore measures their combined contribution rather than only the quality of the dependency edges.

Event-level evidence. Replacing selected events with their source chunks reduces EM by 2.2 points and F1 by 2.3 points. The result indicates that the event layer provides useful evidence filtering beyond chunk-level navigation. Together with the observed length ratios, it supports the design choice of using different granularities for graph traversal and answer generation.

Graph-based bridging. Removing graph bridging while retaining the initially retrieved seeds reduces EM by 1.0 point and F1 by 0.7 points. This shows that path-derived chunks provide complementary information beyond independent subquery-specific seed retrieval. The smaller drop relative to removing decomposition is expected because top-ranked seeds already recover part of the required evidence. The result does not imply that every retrieved path corresponds to a complete gold reasoning chain; rather, bridging provides a measurable end-to-end benefit under the current retrieval setting.

5 Conclusion

We introduced DreamRAG, a chunk-to-event retrieval framework for multi-hop question answering. Its core mechanism, Dependency-Aligned Supernode Bridging, maps predicted subquery dependencies to targeted path searches between subquery-specific seed regions, while separating chunk-level navigation from event-level evidence selection. Experiments on HotpotQA, 2WikiMultihopQA, and MuSiQue show that DreamRAG achieves the best EM and F1 among the evaluated methods.

AI USE STATEMENT

In this work, we have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, interpret result. Formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs are not applicable to this work. Additionally, we used generative AI tools to create or modify scientific figures or images, and edit a research paper to improve readability. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Our code is available at https://anonymous.4open.science/r/Dream-9184/.

REFERENCES

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*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/25f7be9694d7b32d5cc670927b8091e1-Abstract-Conference.html.

Boyu Chen, Zirui Guo, Zidan Yang, Yuluo Chen, Junze Chen, Zhenghao Liu, Chuan Shi, and Cheng Yang. PathRAG: Pruning graph-based retrieval augmented generation with relational paths. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(36):30183–30191, 2026. doi: 10.1609/aaai.v40i36.40268. URL https://ojs.aaai.org/index.php/AAAI/article/view/40268.

Jifan Chen and Greg Durrett. Understanding dataset design choices for multi-hop reasoning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4026–4032, 2019. URL https://aclanthology.org/N19-1405/.

Darren Edge, Ha Trinh, Newman Cheng, et al. From local to global: A graph RAG approach to query-focused summarization, 2024. URL https://arxiv.org/abs/2404.16130.

Yunfan Gao, Yun Xiong, Xinyu Gao, et al. Retrieval-augmented generation for large language models: A survey, 2023. URL https://arxiv.org/abs/2312.10997.

Michael Glass, Gaetano Rossiello, Md Faisal Mahbub Chowdhury, Ankita Rajaram Naik, Pengshan Cai, and Alfio Gliozzo. Re2G: Retrieve, rerank, generate. In *Proceedings of NAACL-HLT*, pp. 2701–2715, 2022. doi: 10.18653/v1/2022.naacl-main.194. URL https://aclanthology.org/2022.naacl-main.194/.

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, 2025. doi: 10.18653/v1/2025.findings-emnlp.568. URL https://aclanthology.org/2025.findings-emnlp.568/.

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, 2024. URL https://papers.nips.cc/paper/2024/hash/6ddc001d07ca4f319af96a3024f6dbd1-Abstract-Conference.html.

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. URL https://proceedings.mlr.press/v267/gutierrez25a.html.

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. doi: 10.18653/v1/2020.coling-main.580. URL https://aclanthology.org/2020.coling-main.580/.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020. URL https://aclanthology.org/2020.emnlp-main.550/.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://papers.nips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Yading Li, Dandan Song, Changzhi Zhou, Yuhang Tian, Hao Wang, Ziyi Yang, and Shuhao Zhang. A framework of knowledge graph-enhanced large language model based on question decomposition and atomic retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 11472–11485, 2024. doi: 10.18653/v1/2024.findings-emnlp.670. URL https://aclanthology.org/2024.findings-emnlp.670/.

Wen-Sheng Lien, Yu-Kai Chan, Hao-Lung Hsiao, Bo-Kai Ruan, Meng-Fen Chiang, Chien-An Chen, Yi-Ren Yeh, and Hong-Han Shuai. HyperRAG: Reasoning n-ary facts over hypergraphs for retrieval augmented generation. In *Proceedings of the ACM Web Conference 2026*, pp. 2465–2476, 2026. doi: 10.1145/3774904.3792710. URL https://doi.org/10.1145/3774904.3792710.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. Multi-hop reading comprehension through question decomposition and rescoring. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6097–6109, 2019. doi: 10.18653/v1/P19-1613. URL https://aclanthology.org/P19-1613/.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah 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, 2023. doi: 10.18653/v1/2023.findings-emnlp.378. URL https://aclanthology.org/2023.findings-emnlp.378/.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/8a2acd174940dbca361a6398a4f9df91-Abstract-Conference.html.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. URL https://aclanthology.org/2022.tacl-1.31/.

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, 2023. doi: 10.18653/v1/2023.acl-long.557. URL https://aclanthology.org/2023.acl-long.557/.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302, 2018. doi: 10.1162/tacl_a_00021. URL https://aclanthology.org/Q18-1021/.

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics*, 8:183–198, 2020. doi: 10.1162/tacl_a_00309. URL https://aclanthology.org/2020.tacl-1.13/.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. 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*, 2018. URL https://aclanthology.org/D18-1259/.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Peng Yixing, Quan Wang, Licheng Zhang, Yi Liu, and Zhendong Mao. Chain-of-question: A progressive question decomposition approach for complex knowledge base question answering. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 4763–4776, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.283. URL https://aclanthology.org/2024.findings-acl.283/.

Khin Yone, Devasha Trivedi, Anish Pahilajani, Jincen Shuai, Samyak Rajesh Jain, Ryan Rossi, Nesreen K. Ahmed, Franck Dernoncourt, Yu Wang, and Namyong Park. Reasoning graph-structured question answering: Datasets and insights from LLM benchmarking. In *Proceedings of the Fifteenth Language Resources and Evaluation Conference*, pp. 5301–5316, 2026. doi: 10.63317/4zjpmtqxxtx4. URL https://aclanthology.org/2026.lrec-1.414/.

Minjun Zhu, Yixuan Weng, Shizhu He, Kang Liu, Haifeng Liu, Yang jun Jun, and Jun Zhao. Towards graph-hop retrieval and reasoning in complex question answering over textual database. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 16539–16549, Torino, Italia, May 2024. ELRA and ICCL. URL https://aclanthology.org/2024.lrec-main.1437/.

Yutao Zhu, Jiajie Jin, Hongjin Qian, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. Single LLM, multiple roles: A unified retrieval-augmented generation framework using role-specific token optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 4837–4856, 2025. doi: 10.18653/v1/2025.emnlp-main.243. URL https://aclanthology.org/2025.emnlp-main.243/.

A Appendix

A.1 Prompt Templates

This section reports the prompt templates used in the current implementation of DreamRAG. To ensure reproducibility, the instructions and few-shot demonstrations inside the prompt boxes are reproduced verbatim from the implementation. DreamRAG uses two groups of prompts: (1) event semantic unit extraction for constructing the event layer, and (2) query decomposition for constructing the subquery dependency graph. Event extraction is performed for at most two rounds: an initial extraction pass followed by an optional supplementary pass for recovering missed events. For query decomposition, HotpotQA, 2WikiMultihopQA, and MuSiQue share the same decomposition instruction and differ only in their dataset-specific few-shot demonstrations.

A.1.1 Event Semantic Unit Extraction

DreamRAG extracts fine-grained Event Semantic Units (ESUs) from each text chunk. Each ESU contains a self-contained factual statement, a set of potential questions that can be directly answered by the event, and a set of salient entity phrases. The implementation also permits an optional `context` field for short supporting information.

Initial Event Extraction. The first pass extracts the principal high-information events from the input chunk using the following prompt.

```
Role
----
You are an information extraction assistant for building an event semantic graph.

Task
----
```

Extract Event Semantic Units (ESUs) from the input text.
Each ESU should be a complete factual statement that is independently retrievable.
For each ESU, also provide:
1) potential user questions this ESU can directly answer,
2) key entity words from the fact.

Rules
−−−−−
1. Keep only high−information facts.
2. Every extracted sentence must have an explicit subject.
3. Do not use unresolved pronouns (he, she, they, it, this, that) as subject.
4. Split unrelated facts into separate events.
5. Keep wording close to source text; do not invent facts.
6. Prefer key relation facts useful for multi−hop QA (e.g., parent/spouse, performer/director/composer, birth/death place, release/publication date, country).
7. For each event, generate 1−3 concise potential questions answerable from this event.
8. For each event, provide 2−6 key entity words/phrases contained in this fact.
9. Do not output importance or attention weights.

Output Schema
−−−−−−−−−−−−−
Return JSON only:

```
{
  "events": [
    {
      "event_id": "E1",
      "content": "<self−contained factual sentence>",
      "context": "<optional short supporting context>",
      "potential": [
        "<question 1>",
        "<question 2>"
      ],
      "key_words": [
        "<entity phrase 1>",
        "<entity phrase 2>"
      ]
    }
  ]
}
```

If no valid events are found, return:
{ "events": [] }

Few−shot Examples
−−−−−−−−−−−−−−−−−
Example 1
Input:
Lothair II was the second son of Emperor Lothair I and Ermengarde of Tours. Ermengarde of Tours died on 20 March 851.

Output:

```
{
  "events": [
    {
      "event_id": "E1",
      "content": "Lothair II's mother was Ermengarde of Tours.",
      "context": "Lothair II was the second son of Emperor Lothair I.",
      "potential": [
```

```
      "Who was Lothair II's mother?"
     ],
     "key_words": [
      "Lothair II",
      "Ermengarde of Tours"
     ]
    },
    {
     "event_id": "E2",
     "content": "Ermengarde of Tours died on 20 March 851.",
     "context": "",
     "potential": [
      "When did Ermengarde of Tours die?"
     ],
     "key_words": [
      "Ermengarde of Tours",
      "20 March 851"
     ]
    }
  ]
}
```


Example 2
Input:
Aas Ka Panchhi is a 1961 Hindi movie. Phoolwari is a 1946 Indian Bollywood film. Phoolwari was directed by Chaturbhuj Doshi.

Output:


```
{
  "events": [
    {
     "event_id": "E1",
     "content": "Aas Ka Panchhi was released in 1961.",
     "context": "",
     "potential": [
      "When was Aas Ka Panchhi released?"
     ],
     "key_words": [
      "Aas Ka Panchhi",
      "1961"
     ]
    },
    {
     "event_id": "E2",
     "content": "Phoolwari was released in 1946.",
     "context": "",
     "potential": [
      "When was Phoolwari released?"
     ],
     "key_words": [
      "Phoolwari",
      "1946"
     ]
    },
    {
     "event_id": "E3",
     "content": "Phoolwari was directed by Chaturbhuj Doshi.",
     "context": "",
```

```
    "potential": [
      "Who directed Phoolwari?"
    ],
    "key_words": [
      "Phoolwari",
      "Chaturbhuj Doshi"
    ]
   }
 ]
}

Text: {input_text}
Output:
```

Supplementary Event Extraction. A single extraction pass may omit secondary but still useful facts. DreamRAG therefore optionally performs a second pass conditioned on both the original text and the event contents obtained in the first pass. The supplementary prompt requests only previously missed events and explicitly prohibits repetition or paraphrasing of already extracted events.

```
Role
----
You are an information extraction assistant for building an event semantic graph.

Task
----
You already extracted some event semantic units from the text.
Extract ONLY additional high-information events that were missed previously.
For each new event, include potential and key_words.

Rules
-----
1. Do not repeat or paraphrase already extracted events.
2. Keep only new, self-contained factual statements.
3. Every sentence must have an explicit subject.
4. If no additional events remain, return { "events": [] }.
5. For each event, include 1-3 concise potential questions as potential.
6. For each event, include 2-6 key entity words/phrases as key_words.

Output Schema
-------------
Return JSON only:
{
 "events": [
   {
    "event_id": "E1",
    "content": "<self-contained factual sentence>",
    "context": "<optional short supporting context>",
    "potential": ["<question 1>"],
    "key_words": ["<entity phrase 1>", "<entity phrase 2>"]
   }
 ]
}

Already Extracted:
{extracted_events}
```

Text: {input_text}
Output:

If the supplementary pass returns no new event, the extraction procedure terminates immediately. The current implementation performs at most two event-extraction passes and does not iterate indefinitely.

A.1.2 QUERY-GRAPH CONSTRUCTION

DreamRAG decomposes each multi-hop question into single-hop subqueries together with explicit predecessor dependencies. HotpotQA, 2WikiMultihopQA, and MuSiQue use one shared decomposition instruction. The shared instruction and the corresponding dataset-specific few-shot demonstrations are concatenated as the system message, while the original multi-hop question is provided separately as the user message.

Shared Decomposition Instruction. The shared decomposition instruction is:

Decompose multi−hop questions into a sequence of single−hop subqueries with explicit dependencies.
Each of the single−hop query must be answerable using a single word. Merge overlapping information about the same word into one subquery.
Each follow−up questions should have an indicator of the previous question they are building upon at the beginning (like <Q1>). Do not add
any noise and use the keywords that are present in the input query alone.

The generated decomposition follows the general form:

Q1) <independent single−hop subquery>
Q2) <Q1> <subquery depending on Q1>
Q3) <Q2> <subquery depending on Q2>

A predecessor marker such as `<Q1>` indicates that the current subquery depends on the answer to the referenced subquery. For example, the output above induces the directed chain $Q_1 \rightarrow Q_2 \rightarrow Q_3$. Subqueries without predecessor markers are treated as independent roots.

HotpotQA Few-Shot Demonstrations. HotpotQA uses the shared instruction together with the following two demonstrations.

INPUT: 'What was the other name for the war between the Cherokee people and white settlers in 1793?'

OUTPUT: Q1) What was the war between the Cherokee people and white settlers in 1793 called?
Q2) <Q1> What was the other name for the war between the Cherokee people and white settlers in 1793?

INPUT: 'Which university has more campuses, Dalhousie University or California State Polytechnic University, Pomona?'
OUTPUT: Q1) How many campuses does Dalhousie University have?
Q2) How many campuses does California State Polytechnic University, Pomona have?

The first demonstration illustrates a two-node dependency chain $Q_1 \rightarrow Q_2$, whereas the second contains two independent comparison branches without an inter-subquery dependency.

2WikiMQA Few-Shot Demonstrations. 2WikiMQA uses the shared instruction together with the following five demonstrations.

INPUT: 'Which film was released first, Aas Ka Panchhi or Phoolwari?'
OUTPUT: Q1) When was the film Aas Ka Panchhi released?
Q2) When was the film Phoolwari released?

INPUT: 'Which film has the director who died first, The Goose Woman or You Can No Longer Remain Silent?'
OUTPUT: Q1) Who is the director of The film Goose Woman?
Q2) Who is the director of the film You Can No Longer Remain Silent?
Q3) <Q1> When did the director of The film Goose Woman die?
Q4) <Q2> When did the director of the film You Can No Longer Remain Silent die?

INPUT: 'Who lived longer, Ludwig Elsbett or Pamela Ann Rymer?'
OUTPUT: Q1) How long did Ludwig Elsbett live?
Q2) How long did Pamela Ann Rymer live?

INPUT: 'What is the place of birth of the director of film Gaby: A True Story?'
OUTPUT: Q1) Who is the director of film Gaby: A True Story?
Q2) <Q1> What is the place of birth of the director of film Gaby: A True Story?

INPUT: 'Who is the father–in–law of Sisowath Kossamak?'
OUTPUT: Q1) Who husband/wife of Sisowath Kossamak?
Q2) <Q1> Who is the father–in–law of Sisowath Kossamak?

These demonstrations cover the major dependency patterns used in 2WikiMQA: two independent nodes for comparison questions, two parallel dependency chains for bridge-comparison questions, and a two-node chain for inference or composition questions.

MuSiQue Few-Shot Demonstrations. MuSiQue uses the shared instruction together with the following three demonstrations.

INPUT: 'Who married the publisher of abolitionist newspaper The North Star?'
OUTPUT: Q1) Who is the publisher of abolitionist newspaper The North Star?
Q2) <Q1> Who married the publisher of abolitionist newspaper The North Star?

INPUT: 'In what state is the district where the man who wanted to reform the religion practiced by Innocenzo Ferrieri preached a sermon on Marian devotion?'
OUPUT: Q1) What is the religion of Innocenzo Ferrieri?
Q2) <Q1> Who wanted to reform the religion practiced by Innocenzo Ferrieri?
Q3) <Q2> What is the district where the man who wanted to reform the religion practiced by Innocenzo Ferrieri preached a sermon on Marian devotion?
Q4) <Q3> In what state is the district where the man who wanted to reform the religion practiced by Innocenzo Ferrieri preached a sermon on Marian devotion?

INPUT: 'The Beach was filmed in what location of the country that contains the birth city of Siddhi Savetsila?'
OUTPUT: Q1) What is the birth city of Siddhi Savetsila?
Q2) <Q1> In which country is the birth city of Siddhi Savetsila located?
Q3) <Q2> The Beach was filmed in what location of the country that contains the birth city of Siddhi Savetsila?

These demonstrations emphasize sequential dependency chains with different reasoning depths, including two-, three-, and four-step decompositions.

A.2 Addtional Experiments

Exp-4: Robustness Across Retrieval Choices. We further study whether DreamRAG depends heavily on particular implementation choices in the retrieval pipeline. Starting from the full configuration, we vary one factor at a time: the event retrieval representation, the number of seed chunks

Table 4: Single-factor retrieval ablations on three multi-hop QA datasets. The full setting uses Event Content with keywords, five seed chunks per subquery, GPT-4o, and all-MiniLM. MPNet denotes all-mpnet-base-v2, and Granite denotes granite-embedding-125m-english. All runs use $\lambda = 0.5$. Average is the unweighted mean over the three datasets.

Alternatives		HotpotQA		2WikiMQA		MuSiQue		Average	
		EM	F1	EM	F1	EM	F1	EM	F1
Full DreamRAG		58.4	75.4	**58.5**	71.6	41.5	57.0	52.8	68.0
Event retrieval representation	Content only	57.0	74.1	58.2	**72.4**	41.2	57.1	52.1	67.9
Seed chunks per subquery	3	57.1	73.9	57.9	71.7	41.3	56.5	52.1	67.4
	4	57.7	75.2	58.0	72.3	41.1	56.3	52.3	67.9
Subquery decomposer	GPT-4o-mini	57.4	74.7	57.6	71.7	40.0	56.7	51.7	67.7
	DeepSeek-V4-Flash	49.0	64.7	58.4	72.2	32.1	46.0	46.5	61.0
Encoder model	MPNet	57.6	74.4	56.4	70.9	**42.2**	57.4	52.1	67.6
	Granite	**59.6**	**76.3**	**58.5**	**72.4**	41.5	**58.6**	**53.2**	**69.1**

per subquery, the subquery decomposer, and the semantic encoder. All variants retain event-level answer generation and use $\lambda = 0.5$. Table 4 reports EM and F1 on all three datasets together with their macro-average.

Event representation and seed budget. Using only event content, without keywords, slightly reduces the macro-average from 52.8/68.0 to 52.1/67.9 EM/F1. The effect is most visible on HotpotQA, while changes on 2WikiMQA and MuSiQue are small. This suggests that event content is the dominant retrieval signal, while keywords provide a complementary signal that is useful when entity matching or salient concept matching is important.

The seed-budget results show a mild coverage–efficiency trade-off. Reducing the seed budget from five to four causes only a small macro-average change, from 52.8/68.0 to 52.3/67.9. Reducing it further to three causes a larger decline, especially in F1. These results indicate that DreamRAG does not require a large number of seeds per subquery, but overly aggressive pruning can reduce evidence coverage before graph bridging is applied.

Encoder and decomposer robustness. DreamRAG remains effective under different semantic encoders. MPNet achieves 52.1/67.6 macro-average EM/F1, close to the default all-MiniLM setting, while Granite improves the macro-average to 53.2/69.1 and obtains the highest F1 on all three datasets. This indicates that DreamRAG is not tightly coupled to a specific embedding model, and stronger encoders can further improve evidence ranking.

The choice of decomposer has a larger effect. GPT-4o-mini remains close to GPT-4o, with only a modest macro-average decrease. In contrast, DeepSeek-V4-Flash performs competitively on 2WikiMQA but substantially worse on HotpotQA and MuSiQue. This pattern suggests that DreamRAG can tolerate moderate variation in decomposition quality, but poor or inconsistent dependency graphs can still propagate errors into seed retrieval and DASB. Overall, the decomposer is the most sensitive component among the tested implementation choices, which is expected because it defines the reasoning topology used by the retrieval process.

Exp-5: Effect of the Trade-off Parameter λ. Finally, we examine the effect of the DASB trade-off parameter λ. We vary $\lambda \in \{0, 0.2, 0.5, 0.8, 1.0\}$ while keeping all other settings fixed. Specifically, we set $k_{\text{per}} = 15$, $K_{\min} = 0$, and $K_{\max} = 45$. Smaller values of λ emphasize the semantic relevance of intermediate chunks, while larger values increasingly favor structurally compact paths. The endpoint $\lambda = 0$ removes the explicit path-length preference, whereas $\lambda = 1$ reduces routing to shortest-hop search with uniform edge cost.

Table 5 shows that DreamRAG is stable across a broad range of λ values. The best setting varies by dataset, reflecting different balances between semantic relevance and structural compactness. On 2WikiMQA, the semantic endpoint $\lambda = 0$ gives the best EM and F1, but the full variation across all settings is small, indicating limited sensitivity. On HotpotQA, the best performance occurs at $\lambda = 0.5$, suggesting that semantic relevance and path compactness provide complementary signals for bridge-style reasoning. On MuSiQue, EM peaks at $\lambda = 0.8$, while F1 peaks at $\lambda = 0.5$, indicating

Table 5: Sensitivity to the DASB trade-off parameter λ. Values are absolute EM and F1 scores; bold indicates the best result for each dataset and metric.

λ	HotpotQA		2WikiMQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
0.0	56.9	74.2	**58.8**	**72.0**	40.8	56.7
0.2	57.3	74.3	58.3	71.6	41.2	56.4
0.5	**58.4**	**75.4**	58.5	71.6	41.5	**57.0**
0.8	58.0	74.9	58.7	71.9	**42.0**	56.7
1.0	57.5	74.6	58.3	71.3	41.8	56.6

that more compact structural paths can help exact-answer recovery, whereas token-level overlap remains less sensitive.

Importantly, $\lambda = 1$ does not achieve the best result on any dataset. This confirms that pure shortest-hop routing is insufficient and that semantic relevance should remain part of the DASB path objective. At the same time, the maximum variation over the sweep is bounded by 1.5 EM and 1.2 F1 points, showing that DreamRAG is not brittle to fine-grained tuning of λ. Overall, the sensitivity analysis characterizes λ as a meaningful but robust inductive bias: it controls the balance between semantic fidelity and structural compactness without being the main source of DreamRAG's performance gains.

Exp-6: Query-Decomposition Structure Evaluation DASB directly uses the predicted subquery dependency graph to determine which subquery-specific evidence regions should be structurally bridged. We therefore evaluate the decomposition stage not only by whether it generates plausible subquestions, but also by whether it recovers the intended reasoning topology.

The structural evaluation is conducted on MuSiQue and 2WikiMQA, for which a reference dependency topology can be derived from the available dataset annotations. HotpotQA is not included in this structural analysis because the current evaluation protocol does not derive an equivalent gold dependency graph from its annotations.

Reference Query-Graph Construction. For all datasets, each atomic subquestion is represented as a graph node. A directed edge $v_i \to v_j$ indicates that answering the j-th subquestion requires the answer to the i-th subquestion. The generated decomposition is converted into a predicted graph using the same dependency semantics: a predecessor marker such as `<Q1>` induces a directed edge from the referenced node to the current node.

- MuSiQue provides human-authored decompositions together with explicit references to intermediate answers. We construct one node for each annotated subquestion and add an edge from v_i to v_j whenever the j-th subquestion refers to the answer of the i-th subquestion. This construction preserves chain, branching, and longer dependency patterns without relying on lexical similarity between subquestions. The predicted graph is constructed analogously from the predecessor references in the generated decomposition.
- For 2WikiMQA, the reference topology is determined by the annotated question type. Comparison questions are represented by two independent nodes. Inference and composition questions are represented by two nodes connected by one directed edge. Bridge-comparison questions contain four nodes and two dependency edges, $v_1 \to v_3$ and $v_2 \to v_4$. These predefined structures are used as the reference graphs for structural evaluation.

We evaluate structural correctness using directed graph isomorphism. Two directed graphs are considered isomorphic if there exists a one-to-one mapping between their nodes that preserves all directed edges. This criterion therefore evaluates the topology of the reasoning structure rather than the surface wording of individual subquestions. A prediction is counted as incorrect if it adds or removes a node or changes the dependency structure.

For an evaluation set $\mathcal{Q}$, the isomorphic accuracy is defined as

$$\mathrm{IsoAcc} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbb{I}\big[G_q^{\mathrm{pred}} \cong G_q^{\mathrm{gold}}\big]\,. \tag{13}$$

Table 6: Isomorphic accuracy of generated query decompositions. Higher is better.

Model	2WikiMQA	MuSiQue
GPT-4o	**0.981**	**0.677**
GPT-4o-mini	0.979	0.657
DeepSeek-V4-Flash	0.954	0.441

Table 6 shows a clear difference in structural difficulty between the two datasets. On 2WikiMQA, all three decomposers achieve high isomorphic accuracy: GPT-4o reaches 0.981, GPT-4o-mini reaches 0.979, and DeepSeek-V4-Flash reaches 0.954. This result is consistent with the fact that 2WikiMQA uses a small family of predefined reasoning topologies.

MuSiQue is substantially more challenging because its annotations contain more heterogeneous two-, three-, and four-hop dependency structures. GPT-4o achieves the highest accuracy of 0.677, followed by GPT-4o-mini at 0.657 and DeepSeek-V4-Flash at 0.441. The larger separation on MuSiQue indicates that recovering diverse reasoning structures places a stronger demand on the decomposer than matching a small set of fixed graph templates.

These structural results also help interpret the decomposer ablation in the main text. The relatively small difference between GPT-4o and GPT-4o-mini in isomorphic accuracy is consistent with their similar downstream performance, whereas the substantially lower MuSiQue structural accuracy of DeepSeek-V4-Flash coincides with a larger degradation in end-to-end question answering. Overall, the analysis supports the role of query decomposition as a structural prior for DASB: more reliable dependency recovery provides a more accurate constraint on which evidence regions should be connected during graph retrieval.