BRIDGE-PROP: REASONING-GUIDED BRIDGE PROPAGATION FOR MULTI-HOP RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-hop question answering often requires combining evidence linked through entities that are absent from the question itself. In reasoning-based iterative GraphRAG, a language model may infer such bridge entities during multi-round retrieval. However, these entities remain in unstructured reasoning text and may be omitted or diluted in later retrieval steps, rather than being retained as explicit, reusable representations in the graph. As a result, they cannot directly guide subsequent graph propagation. We introduce BRIDGE-Prop (Reasoning-Guided Bridge Propagation), an iterative GraphRAG framework that grounds predicted bridge entities to graph nodes and retains them as a cumulative bridge memory across rounds. These bridges act as explicit seeds for the existing Personalized PageRank graph-ranking algorithm, allowing retrieval to pursue multiple intermediate connections. We evaluate BRIDGE-Prop on MuSiQue and 2WikiMultiHopQA on the HippoRAG-2 backbone. It raises 2WikiMultiHopQA recall@5 from .889 to .962 over the strongest iterative baseline, ITER-RETGEN, and lifts answer EM from .608 to .709. By question type and reasoning depth, the gain is largest where a single query cannot gather all the evidence at once: 2Wiki bridge-comparison questions and deeper MuSiQue hops. The retrieval advantage holds for both a proprietary and an open-weight model. BRIDGE-Prop shows how free-text LLM reasoning can be grounded into discrete, symbolic structure, tightening the loop between neural reasoning and structured graph-based retrieval.

1 INTRODUCTION

Retrieval-augmented generation (RAG) augments a language model with retrieved evidence, which yields more accurate answers on knowledge-intensive question answering than a model using only parametric knowledge (Lewis et al., 2020; Guu et al., 2020). A basic RAG system indexes a corpus offline, then at inference retrieves passages for the query and generates an answer from them. This direct pipeline works well for simple questions, but fails for more complex multi-hop question answering (QA) which may require revision of the query or gather evidence over several steps (Trivedi et al., 2022). Recent work lets the model's own reasoning steer retrieval instead of following a fixed pipeline (Singh et al., 2026): planning what to look for, deciding when and what to retrieve, assessing whether the evidence suffices, and re-retrieving until it does. For example, ReAct interleaves thought and action, so each thought decides the next action and each observation shapes the next thought (Yao et al., 2023), while Self-RAG learns when to retrieve and judges whether each passage supports its answer (Asai et al., 2024). FLARE looks ahead to the sentence it is about to generate and retrieves new information in case the generated conclusion is uncertain (Jiang et al., 2023). In each, inference-time reasoning steers retrieval to answer multi-hop questions a single pass cannot.

These methods retrieve by matching offline passages to the query by means of semantic similarity. However, in multi-hop QA, evidence often shares little overlap with the query: a passage can only be reached through an intermediate entity, which matches neither the query's words nor its meaning. A simple similarity search can miss it. A knowledge graph can form a stronger representation, linking passages through shared entities and facts so that retrieval can follow those links to evidence a query cannot match through semantic similarity (Peng et al., 2026; Edge et al., 2025). HippoRAG 2 (Gutiérrez et al., 2025) builds it as an associative memory modeled on human recall, retrieving with Personalized PageRank (PPR) (Jeh & Widom, 2003) that can reach passages several hops from the query. But also single-pass graph retrieval falls short on deep-hop questions and complex rea-

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.

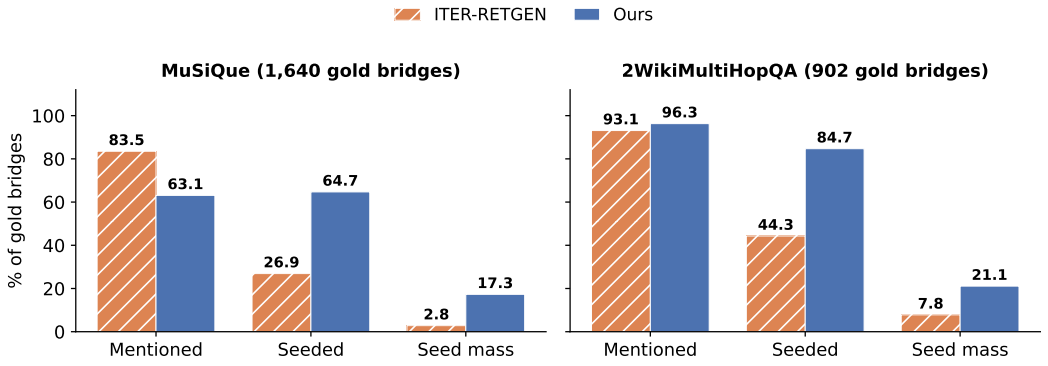

Figure 1: **The grounding gap.** Gold bridge entities are the intermediate chain entities of each question (MuSiQue 1,640, 2Wiki 902). *Mentioned*: the bridge appears in the method's reasoning. *Seeded*: it is grounded onto the graph as a retrieval seed. *Seed mass*: its share of the initial seed weight that PPR restarts from, which shapes the final relevance distribution over passages.

soning, where the answer may lie far from the query on the graph. To answer *which country the director of Get Carter (2000) is from*, for instance, the model must first find the director, *Stephen Kay*, a bridge entity that the question never names. We take a knowledge graph and its PPR propagation as our backbone, and study reasoning-guided retrieval on two graph types: a semantically sparse named-entity (NER) graph (Nadeau & Sekine, 2007) and HippoRAG-2's richer OpenIE fact graph (Gutiérrez et al., 2025; Banko et al., 2007).

In our evaluation, simply using chain-of-thought reasoning to guide retrieval on the graph is effective: it beats single-pass graph retrieval and the same loop over a dense retriever, on MuSiQue and 2WikiMultiHopQA. But this reasoning is rarely transferred onto the graph in a way that steers retrieval: a bridge it names is seldom turned into a seed, the starting node that graph retrieval propagates from. Our strongest iterative baseline, ITER-RETGEN, names the gold *bridge* nodes in its reasoning far more often than they are seeded onto the graph, and the bridges it does seed carry little mass (Figure 1). This points to a grounding bottleneck between bridge discovery and graph retrieval: the bridge is found in text but barely steers retrieval.

We close this gap with BRIDGE-Prop, a novel interface that turns intermediate reasoning into explicit graph seeds. Each round, the LLM reviews what the retrieved evidence still misses and predicts the bridge entities for the next hop. These bridges are resolved to graph nodes and kept across rounds as a cumulative, query-specific memory that augments the query seeds for the next round of PPR retrieval. To keep a high-degree bridge influential, we apply a degree-adaptive weight, since PPR would otherwise spread its seed mass thin across its many neighbors. Hence, BRIDGE-Prop grounds the gold bridges as graph seeds far more often, and with far more mass, than existing methods (Figure 1).

BRIDGE-Prop improves substantially over single-pass graph retrieval. Relative to ITER-RETGEN, the strongest iterator on the graph, it consistently improves retrieval on the cases a single query cannot resolve on its own, such as the deeper MuSiQue hops and 2WikiMultiHopQA bridge-comparison questions. The grounded bridges can also serve as reusable memory: replaying a query's accumulated bridges in a single pass clearly improves its retrieval, and they partly transfer to related queries.

Our contributions are: (i) an empirical characterization of the grounding gap in iterative GraphRAG; (ii) BRIDGE-Prop, a training-free interface that grounds free-text bridge reasoning into a discrete, cumulative graph memory, and improves retrieval on complex questions over strong iterative baselines; and (iii) evidence of inference-time bridge memory reuse, within-query and partly cross-query.

2 Related Work

For knowledge-intensive multi-hop QA, how the retriever represents and searches the corpus largely determines what evidence it can reach. Classical text retrieval primarily matches on surface

terms (Robertson & Zaragoza, 2009), missing evidence phrased differently from the query. Dense retrieval instead embeds queries and passages so that relevant pairs lie close in vector space: from supervised dual-encoder training that outperformed lexical retrieval on open-domain QA (Karpukhin et al., 2020), to unsupervised contrastive pretraining that generalizes without labels (Izacard et al., 2022), to LLM-based encoders that top general embedding benchmarks (Lee et al., 2025). Yet similarity alone does not guarantee the right evidence: dense retrieval scores each passage against the query independently, so its top-ranked passages are each similar to the query but not linked to one another, and they cannot form a connected chain of evidence. This limitation motivates graph-structured retrieval.

A knowledge graph makes these relations explicit, connecting passages through shared entities and facts (Hogan et al., 2021). Systems differ in what they put on the graph. Microsoft GraphRAG builds an entity graph with LLM-generated community summaries, and RAPTOR builds a hierarchy of recursive document summaries (Edge et al., 2025; Sarthi et al., 2024). These summaries expose corpus-level themes that flat retrieval misses, but the added abstraction layer can introduce noise and drift from the source text. HippoRAG-2 (Gutiérrez et al., 2025) instead grounds the graph in the corpus itself: it extracts schema-less (subject, relation, object) triples and keeps every passage as a node linked to the phrases it contains, so retrieval propagates over facts and passages taken directly from the source. This makes it a faithful backbone: a bridge that an LLM names while reading the retrieved passages can be grounded back in the graph as a seed, rather than left in free text. This is the property our method builds on.

Iterative LLM reasoning can infer missing information and approach a distant answer step by step. The ReAct paradigm (Yao et al., 2023) lets an LLM reason and act on an external environment in turn, using each action's observations to guide the next reasoning step. IRCoT (Trivedi et al., 2023) interleaves chain-of-thought reasoning with retrieval, ITER-RETGEN (Shao et al., 2023) feeds each round's generation back as context for the next retrieval query, and Self-Ask (Press et al., 2023) decomposes the question into sub-questions with interleaved retrieval. RAG-Fusion (Rackauckas, 2024) instead issues K independent query rewrites at once and merges their rankings with Reciprocal Rank Fusion (Cormack et al., 2009). All are text-based iterators over a plain, non-graph retriever: they do not exploit the corpus's graph structure, and the reasoning reaches retrieval only as a rewritten query. Because each query rewrite is conditioned on the earlier rounds' generation, an early error carries forward into every later round (Shao et al., 2023).

To couple LLM reasoning with graph retrieval, recent reasoning-on-graph methods let the LLM navigate the graph itself: GraphReader (Li et al., 2024) and ToG (Sun et al., 2024) have the LLM explore and prune nodes or relations at each step, so reaching the evidence a question needs can take many LLM calls. BRIDGE-Prop performs no step-by-step LLM navigation. Instead, each round it only shapes the seed distribution by predicting the bridge entities and grounding them as seeds, and Personalized PageRank propagates from there without further LLM calls.

3 BRIDGE-PROP

BRIDGE-Prop brings iterative LLM reasoning onto a graph retriever. Rather than coupling the two only through text, it adds an explicit interface between reasoning and graph propagation, letting each round's reasoning act on the graph. Given a question q and a fixed corpus graph $G = (V, E)$, BRIDGE-Prop retrieves passages with PPR from a seed distribution (Eq. 1). Each round, the LLM retrospectively explains what the retrieved evidence still misses and predicts the bridge entities needed for the next hop. These bridges are grounded to graph nodes and added to the seed distribution through a dual-channel entity seeding that combines them with the query seeds. A degree-adaptive weight offsets the PPR mass dilution that high-degree bridges suffer. Rankings from all rounds are then fused for the QA LLM. These steps are described in detail below.

3.1 GRAPH CONSTRUCTION AND ROUND-0 SEED MATCHING

We instantiate BRIDGE-Prop on two corpus graphs. The NER graph (Nadeau & Sekine, 2007) links entity and passage nodes through entity co-occurrence, entity–passage, and synonymy edges. The semantically richer OpenIE fact graph (Banko et al., 2007), following the HippoRAG-2 backbone,

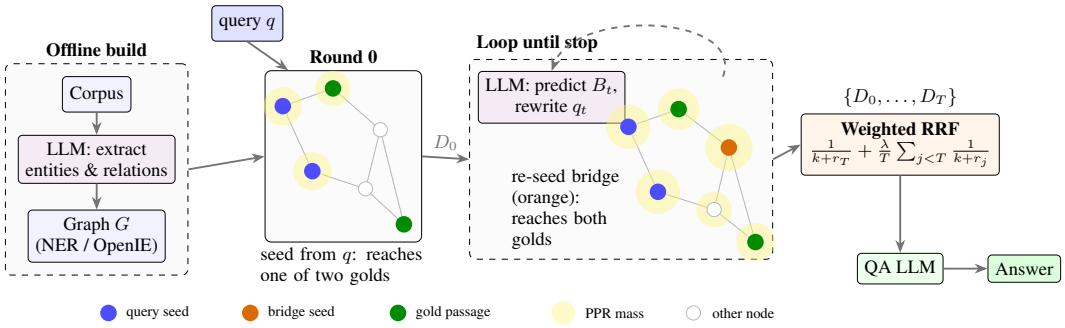

Figure 2: **Overview of BRIDGE-Prop.** Offline, an LLM builds the knowledge graph G from the corpus. Given a query q, round-0 PPR from the query seeds (blue) reaches only part of the gold evidence (green). Each loop round has the LLM predict a bridge entity B_t (orange) and rewrite the query q_t; injecting the bridge as a new seed, with mass set by a degree-adaptive weight (Eq. 3), reshapes the propagation to reach the remaining gold passages. The per-round rankings $\{D_0, \ldots, D_T\}$ are combined by weighted RRF, which gives the current round full weight and earlier rounds weight λ, and passed to the LLM for the answer. Halos mark PPR mass.

connects entity nodes through subject–relation–object facts (Gutiérrez et al., 2025). Appendix A.2 gives the construction and seed-matching details.

Round-0 seed matching. Round 0 matches the question to the graph, producing the initial query seed distribution $\mathbf{s}^q$; the NER and OpenIE matching procedures are given in Appendix A.2. We initialize $q_0 = q$, $\mathcal{B}_0 = \varnothing$, and $\mathbf{s}_0 = \mathbf{s}^q$.

3.2 Bridge Prediction Across Rounds

From the partial evidence retrieved so far, the LLM predicts the bridge targets for the next round. At round t, PPR (Jeh & Widom, 2003) ranks graph nodes from a personalization vector $\mathbf{s}_t$:

$$\mathbf{p}_t = (1-\alpha)M^\top \mathbf{p}_t + \alpha \mathbf{s}_t. \tag{1}$$

Here, M is the graph transition matrix and α the restart probability: with probability α the walk restarts from the *seed distribution* $\mathbf{s}_t$. Passage nodes are ranked by $\mathbf{p}_t$. Standard graph retrieval derives this vector from the question once.

BRIDGE-Prop starts from round-0 retrieval with the query seeds $\mathbf{s}^q$. Each later round calls the LLM on the question, the reasoning trace $r_{<t}$ so far, and the fused evidence retrieved to date, i.e., the top passages after weighted rank fusion of the per-round rankings (TopN over RRF, Eq. 4):

$$(r_t,\ B_t,\ q_t,\ \mathrm{stop}_t) = \mathrm{LLM}(q,\ r_{<t},\ \mathrm{TopN}(\mathrm{RRF}(\{D_0, \ldots, D_{t-1}\}))) \,.$$

The trace r_t first says what the retrieved evidence still misses, then names the entities that would bridge to it; these are the predicted bridges B_t. q_t is a rewritten query, and stop_t signals whether the evidence now suffices. Each predicted bridge is grounded to a node of G, and the unresolved ones are dropped (Appendix A.5). Writing $\widehat{B}_t$ for the grounded bridges of round t, the method accumulates them into a per-question memory $\mathcal{B}_t = \bigcup_{i=1}^{t} \widehat{B}_i$ until the LLM stops.

3.3 Dual-Channel Entity Seeding

To move toward the bridges without drifting from the question, BRIDGE-Prop seeds each round from two channels. Let $\mathbf{s}^{q_t}$ be the query seeds from the round-t rewritten query q_t, matched to the graph as in round 0 ($q_0 = q$). For each round $t \geq 1$, the seed distribution combines the round-t query seeds with the bridge memory accumulated through the current reasoning step:

$$\mathbf{s}_t = \mathrm{Normalize}\left(\mathbf{s}^{q_t} + \sum_{v \in \mathcal{B}_t} w(v)\mathbf{e}_v\right), \tag{2}$$

Table 1: Statistics of the retrieval corpora and the two graph views (NER and OpenIE). #Q is the number of questions; #P the number of candidate passages in the pooled corpus. Node, edge, and hub counts are reported as NER / OpenIE pairs, and hubs are entity nodes with degree at least 100 or 250.

Dataset	#Q	#P	Nodes	Edges	Hubs ≥ 100	Hubs ≥ 250
MuSiQue	1,000	11,656	74.5k / 107.7k	559k / 546k	1,931 / 1,839	232 / 130
2Wiki	1,000	6,119	45.6k / 55.2k	303k / 268k	421 / 315	89 / 10

where $\mathbf{e}_v$ is the indicator vector for node v, $w(v)$ is the degree-adaptive seed weight defined in Eq. 3 below, and $\mathrm{Normalize}(\cdot)$ rescales the result to a valid PPR restart distribution (Eq. 1). The query channel anchors retrieval to the question, and the bridge channel makes the inferred entities explicit propagation targets. Without the bridge channel, the method reduces to iterative query rewriting.

3.4 Degree-Adaptive Seeding and Rank Fusion

Under PPR, a high-degree bridge spreads its seed mass across many neighbors, weakening its pull on any single path. We counteract this with a degree-adaptive seed weight:

$$w(v) = w_0\big(1 + \log(\deg(v) + 1)\big), \tag{3}$$

where w_0 is the base bridge weight and $\deg(v)$ is the number of edges incident to node v in G. The logarithm compensates for the dilution without letting a few hubs dominate the seed distribution. Each round produces a passage ranking D_t, and BRIDGE-Prop fuses the rankings across rounds with a weighted Reciprocal Rank Fusion (RRF):

$$\mathrm{score}_t(d) = \frac{1}{k + \mathrm{rank}_{D_t}(d)} + \lambda \frac{1}{t} \sum_{j=0}^{t-1} \frac{1}{k + \mathrm{rank}_{D_j}(d)}, \qquad t \geq 1. \tag{4}$$

Here k is the rank-fusion constant, t is the current retrieval round, and $\mathrm{rank}_{D_j}(d)$ is the rank of passage d in the ranking D_j. The current ranking D_t receives full weight and earlier rankings share weight $\lambda < 1$. The fused top-5 passages are reused as evidence in later rounds, and the final output uses score_T, where T is the last completed retrieval round. When retrieval terminates at round 0, we use D_0 directly without rank fusion. Likewise, the first reasoning step reads D_0 directly. We allow at most three reasoning rounds and stop earlier when the LLM emits stop_t. Prompts, grounding details, and hyperparameters are given in Appendices A.9, A.5, and A.3, respectively.

4 Experimental Setup

Benchmarks and graphs. To evaluate different reasoning structures, we use **MuSiQue** (Trivedi et al., 2022) and **2WikiMultiHopQA** (Ho et al., 2020). The two differ in reasoning structure: MuSiQue questions follow compositional chains toward a single target, while 2Wiki spans several question types, including comparison and bridge questions that require evidence for two entities at once. Following HippoRAG-2 (Gutiérrez et al., 2025), we use its randomly sampled subsets of 1,000 questions per dataset and retrieve over each dataset's pooled candidate passages. We evaluate **NER** and semantically richer **OpenIE** graphs. Table 1 reports corpus and graph statistics, with construction and question-type details in Appendices A.2 and A.4.

The NER graphs have fewer nodes but more edges and high-degree hubs, reflecting repeated co-occurrence links among a smaller named-entity vocabulary. The OpenIE graphs include more descriptive entity phrases and have more nodes, but fewer edges and hubs.

Baselines and controlled comparison. We compare BRIDGE-Prop against a **closed-book LLM**, dense retrieval with **IRCoT** (Trivedi et al., 2023) and **ITER-RETGEN** (Shao et al., 2023), and single-pass graph retrieval using NER PPR or the **HippoRAG-2** backbone (Gutiérrez et al., 2025) on OpenIE. We also run IRCoT on NER, and ITER-RETGEN and BRIDGE-Prop on both graphs. To test the benefit of using intermediate bridge reasoning to seed graph propagation, matched BRIDGE-Prop–ITER-RETGEN comparisons share the same corpus, graph, encoder, LLM, retrieval budget, and maximum number of rounds.

Table 2: Main results on 1,000 MuSiQue and 2WikiMultiHopQA questions each. All methods use all-MiniLM-L6-v2 and a QA LLM over the top five passages. BRIDGE-Prop is evaluated on NER and OpenIE graphs; IRCoT on dense retrieval and the NER graph only. The OpenIE comparison uses HippoRAG, ITER-RETGEN, and BRIDGE-Prop. Best in bold. Significance tests for BRIDGE-Prop vs. ITER-RETGEN are in Table 8.

	MuSiQue			2WikiMultiHopQA		
Method	EM	F1	R@5	EM	F1	R@5
Closed-book LLM	.092	.141	–	.141	.171	–
Text-based iterative retrieval (dense retriever; no graph)						
+ IRCoT	.232	.317	.454	.333	.362	.597
+ ITER-RETGEN	.324	.421	.592	.505	.560	.746
NER graph (semantically sparse)						
NER (baseline)	.255	.340	.467	.379	.429	.659
+ IRCoT	.264	.353	.483	.499	.585	.677
+ ITER-RETGEN	.352	.454	.616	.555	.630	.837
+ BRIDGE-Prop (ours)	.398	.494	.638	.649	.721	.896
OpenIE graph / HippoRAG 2 (semantically rich)						
HippoRAG (baseline)	.280	.369	.521	.479	.541	.769
+ ITER-RETGEN	.374	.466	.656	.608	.682	.889
+ BRIDGE-Prop (ours)	**.425**	**.522**	**.679**	**.709**	**.781**	**.962**

Implementation. Our main configuration uses **qwen-plus** (Bai et al., 2023) for graph construction, intermediate bridge reasoning, and QA, and **all-MiniLM-L6-v2** (Wang et al., 2020; Reimers & Gurevych, 2019) for query-to-graph matching. The LLM receives the top five passages. All LLM calls use temperature 0 to minimize sampling variance. Within each matched configuration, the compared methods reuse the same cached graphs and embeddings, so any difference in results comes from the method and not from graph or encoder variation. We also evaluate **Llama-3.3-70B-Instruct** (Grattafiori et al., 2024) and repeat the 2Wiki NER comparison with **bge-base-en-v1.5** (Xiao et al., 2024). Additional implementation details are provided in Appendix A.1.

Metrics and statistical testing. Since BRIDGE-Prop is designed to improve evidence retrieval, we use **recall at five (R@5)** as our primary metric. R@5 is the fraction of gold supporting passages retrieved in the top five, macro-averaged across questions. We additionally report exact match (EM) and token-level F1 to assess downstream QA performance. The answer-extraction protocol and paired statistical testing procedures are detailed in Appendix A.1. Statistical results for the main qwen-plus and Llama experiments are reported in Tables 8 and 11, respectively.

5 Results

5.1 Overall Performance

Table 2 reports the main results under the qwen-plus configuration. Iterative retrieval helps on its own: interleaving reasoning with retrieval over a plain dense retriever improves substantially over the closed-book and single-pass baselines. Of the two text iterators, ITER-RETGEN is the stronger, reaching 2Wiki EM .505 and R@5 .746 against IRCoT's .333 and .597. We therefore take ITER-RETGEN as a strong text-based iterative baseline.

Moving iteration onto a graph improves both text iterators: on 2Wiki, a graph-based retriever raises IRCoT from R@5 .597 to .677 and ITER-RETGEN from .746 to .889. BRIDGE-Prop is the strongest method in every column, and its gain over the strong ITER-RETGEN baseline is graph-agnostic. On the semantically rich OpenIE graph, it lifts 2Wiki EM from .608 to .709 and R@5 from .889 to .962; on the sparse NER graph, it lifts 2Wiki EM from .555 to .649 and R@5 from .837 to .896. On MuSiQue the same pattern holds but the gains are smaller in both metrics (OpenIE EM .374 $\rightarrow$ .425, R@5 .656 $\rightarrow$ .679). All of BRIDGE-Prop's gains over ITER-RETGEN are sta-

Table 3: R@5 on MuSiQue by reasoning depth (2/3/4-hop; an n-hop question has n gold passages). Best per column in bold. 2WikiMultiHopQA is instead analyzed by question type (Table 4).

Method	2-hop (n=518)	3-hop (n=316)	4-hop (n=166)
NER graph (semantically sparse)			
NER (baseline)	.594	.378	.239
+ IRCoT	.626	.381	.229
+ ITER-RETGEN	.764	.509	.324
+ BRIDGE-Prop (ours)	.790	.520	**.389**
OpenIE graph / HippoRAG 2 (semantically rich)			
HippoRAG (baseline)	.661	.434	.238
+ ITER-RETGEN	.841	.519	.328
+ BRIDGE-Prop (ours)	**.845**	**.559**	.380

tistically significant under paired tests (see Table 8 in Appendix A.1). The difference between the two datasets motivates the reasoning-depth and question-type analyses below.

5.2 Performance by Reasoning Depth (MuSiQue)

Table 3 analyzes R@5 on MuSiQue by reasoning depth (2/3/4-hop questions). Retrieval gets harder with depth: every method's R@5 falls substantially from two to four hops. The single-pass baselines drop below .24 at four hops (NER .594 $\rightarrow$ .239, HippoRAG .661 $\rightarrow$ .238). IRCoT does not help on the sparse graph, at or below single-pass at four hops (.229), while ITER-RETGEN and BRIDGE-Prop degrade far less. BRIDGE-Prop's margin over ITER-RETGEN is largest on the deepest four-hop questions (NER $+.065$, OpenIE $+.052$), where a single rewritten query least reaches the final hop, and smallest on two-hop (NER $+.026$, OpenIE $+.004$). The gains concentrate on deeper hops, so BRIDGE-Prop's overall improvement on MuSiQue is modest.

5.3 Performance by Question Type (2Wiki): Parallel Evidence

Table 4 breaks down 2WikiMultiHopQA by question type. On the HippoRAG-2 backbone, BRIDGE-Prop exceeds .90 R@5 on every 2Wiki question type even with the lightweight MiniLM encoder, approaching the retrieval ceiling. Gains over ITER-RETGEN are largest on bridge-comparison questions, which require two parallel bridge paths: R@5 rises from .638 to .794 on NER and from .720 to .924 on OpenIE. Comparison questions require evidence for two entities and show the next-largest gains ($+.062$ on NER, $+.076$ on OpenIE). Inference and compositional questions involve two-passage chains that ITER-RETGEN already retrieves well, leaving smaller gains, consistent with shallower MuSiQue chains. Across both graphs, BRIDGE-Prop's advantage is larger on questions requiring more parallel evidence.

Table 4: R@5 on 2WikiMultiHopQA by question type, on both graphs. Best per column in bold.

Method	comparison (n=244, 2 gold)	inference (n=108, 2 gold)	compositional (n=413, 2 gold)	bridge-comparison (n=235, 4 gold)
NER graph (semantically sparse)				
NER (baseline)	.809	.727	.662	.466
+ ITER-RETGEN	.877	.898	.910	.638
+ BRIDGE-Prop (ours)	.939	.907	.927	.794
OpenIE graph / HippoRAG 2 (semantically rich)				
HippoRAG (baseline)	.867	.778	.801	.609
+ ITER-RETGEN	.912	.926	.962	.720
+ BRIDGE-Prop (ours)	**.988**	**.963**	**.969**	**.924**

Table 5: Component ablations on 1000-sample MuSiQue. Each row below the full method removes one component; all use the NER graph and qwen-plus API.

Variant	EM	F1	R@5
NER + BRIDGE-Prop (all components)	.396	.492	.638
− bridge seeds (query rewrite + RRF only)	.358	.445	.516
− RRF fusion (last round ranking only)	.370	.473	.582
− degree-adaptive weights (flat bridge weight)	.373	.470	.584

Table 6: Generalization to an open-weight LLM (Llama-3.3-70B, Llama-built NER graph): overall R@5. EM, F1, per-type/per-hop breakdowns, and significance tests are in Appendix A.6.

Dataset	Single-pass	ITER-RETGEN	BRIDGE-Prop
MuSiQue	.464	.588	**.606**
2WikiMultiHopQA	.666	.789	**.892**

5.4 Ablations: Bridge Propagation Drives the Gain

Explicit bridge discovery and grounding contribute most to retrieval performance (Table 5). On MuSiQue, removing bridge seeds reduces R@5 from .638 to .516, compared with .582 without rank fusion and .584 without degree-adaptive weighting. All three components contribute, but explicit bridge seeds drive the gain.

The grounding analysis complements these ablations: both methods mention gold bridges, but BRIDGE-Prop grounds them as graph seeds more often and assigns them more seed mass (Figure 1). BRIDGE-Prop encourages the LLM to explore potential bridges beyond the original query. When round-0 retrieval finds no gold evidence, BRIDGE-Prop recovers gold evidence more often than ITER-RETGEN on the same NER graph: 69% vs. 55% on MuSiQue and 74% vs. 44% on 2Wiki (Appendix A.8). Its reasoning also repeats far less across rounds than ITER-RETGEN's, which tends to reuse its previous text and stay anchored to the same evidence (Table 19, Appendix A.8). Appendix A.10 illustrates one retrieval trajectory.

5.5 Generalization to other models

Open-weight LLM. On Llama-3.3-70B (Table 6), the R@5 pattern holds: BRIDGE-Prop improves over ITER-RETGEN on both datasets, clearly on 2WikiMultiHopQA (.789 $\rightarrow$ .892) and modestly on MuSiQue (.588 $\rightarrow$ .606). As in the main results, the largest gains are on difficult questions (Appendix A.6). EM, F1, and significance tests are in the same appendix.

Sentence encoder. Swapping the small all-MiniLM-L6-v2 for the larger bge-base-en-v1.5 on the 2Wiki NER comparison, BRIDGE-Prop keeps its edge over ITER-RETGEN (overall R@5 .950 vs. .895), still gaining $+.105$ on bridge-comparison (Appendix A.7).

5.6 Inference-Time Bridge Memory Reuse

Replaying the memory in one pass. To assess whether the bridges accumulated over a retrieval trajectory form a compact memory that can drive retrieval on its own, we run a new single-pass graph retrieval for the query, seeded with its original query seeds plus the bridges discovered earlier over the completed trajectory. Without query rewriting or an additional reasoning call, R@5 rises from .467 to .580 (Table 7), recovering a substantial portion of the full multi-round result of .638. Once obtained, the accumulated bridges therefore carry most of the multi-round reasoning gain.

Cross-query memory reuse. The memory can be reused across queries at inference time: a new query can borrow a related query's bridges instead of computing its own. For each MuSiQue query whose nearest neighbor under the MiniLM encoder has cosine similarity above 0.85 (n=182), we augment the target query's round-0 PPR with the nearest neighbor's accumulated bridge memory.

Table 7: Bridge reuse on MuSiQue (R@5, NER graph). Bridge seeds here are the grounded bridge entities a query accumulated over its retrieval rounds, reused as PPR seeds. Single-pass NER and BRIDGE-Prop are the references. Columns are the full set (n=1,000) and the close-neighbor subset (cosine >0.85, n=182); borrowing needs a close neighbor, so it has no full-set entry (dash).

Setting	Bridge seeds	Rounds	R@5 (all) n=1,000	R@5 (nearest neighbor) n=182
Single-pass NER	none	single	.467	.395
Own reuse	own query	single	.580	.482
Cross-query borrow	neighbor query	single	–	.439
BRIDGE-Prop	own query	multi	**.638**	**.516**

R@5 rises from .395 to .439 without running the target query's own bridge reasoning. The borrowed memory is adapted to the new query. A natural next step is to index bridges at the hop level, extending memory coverage to a broader range of related questions. An LLM can then assess similarity at the question and sub-question levels before reusing bridges, improving selection precision.

6 Discussion & Conclusion

We introduce BRIDGE-Prop, a novel interface between LLM reasoning and graph propagation. Existing iterative methods keep reasoning as free text and retrieve by dense or lexical matching, leaving a grounding gap: a key bridge entity must compete with the whole generation to shape retrieval. BRIDGE-Prop is instead driven by discovering discrete, important intermediate nodes: it predicts the bridge entities, grounds each to a graph node, and adjusts its Personalized PageRank seed mass with a degree-adaptive weight that offsets high-degree dilution.

Overall, BRIDGE-Prop is an effective iterator: across both LLMs and both datasets it improves retrieval substantially over single-pass retrieval and surpasses the strongest text iterator, ITER-RETGEN. The effect is not tied to a particular graph construction and we observe particularly large gains for the harder question types. What mainly drives the gain is that BRIDGE-Prop gives the intermediate reasoning a structured form: the bridge entities it uncovers become discrete units, and it manages them across rounds as an accumulating memory, so each bridge stands out from the large body of generated text and keeps steering retrieval. In addition, the gain could stem from a reduced early commitment to a single retrieval direction: rather than forcing each reasoning step to commit to one next retrieval action, it uses a set of distinct bridge entities as seeds each round, letting multiple entities guide propagation across iterations. This mechanism is consistent with our observations: explicit bridge propagation helps retrieval recover from failed states while reducing repetitive reasoning across iterations.

Existing work has mostly targeted reasoning quality or graph quality; our results suggest that how intermediate reasoning is propagated into retrieval deserves equal attention. BRIDGE-Prop's degree-adaptive weighting is a simple, purely local adjustment of this propagation: by scaling each bridge entity's influence to the local graph topology, it helps them stay effective in subsequent retrieval, and our ablation shows even this yields a meaningful gain. More generally, the broader interaction between the reasoning output and graph retrieval offers substantial room for further work.

Because bridges in BRIDGE-Prop are discrete graph nodes rather than free text, they form a reusable symbolic memory. Our reuse experiments suggest that a bridge's value outlives the reasoning that produced it: once discovered, it keeps steering retrieval on its own, and can even help a different but related question. This points to bridges as a persistent, shareable retrieval memory rather than a byproduct of one query's reasoning. How to index, share, and manage such memory across questions or tasks is a promising direction for future work.

To conclude, BRIDGE-Prop shows that intermediate reasoning does not need to remain confined to free-text context, but can be grounded as explicit structures that persists across retrieval rounds. We believe this provides a promising direction for more tightly coupling neural reasoning with structured graph-based retrieval.

AI USE STATEMENT

The research idea and the full system were designed by the authors themselves. Large language models were used as an assistant during the project. The authors wrote the initial and final drafts of the manuscript themselves; AI was used in between only to polish wording and reorganize existing material. AI helped implement this design in code, manage the experiments (including identifying and removing components that did not help), and write basic data-processing and analysis scripts. The authors take full responsibility for all content of this paper and the accompanying code.

REPRODUCIBILITY STATEMENT

We take several steps to make our results reproducible. The full method, including all equations and the round-by-round procedure, is given in Section 3. Model access, API versions, decoding settings, and the single-run protocol are described in Appendix A.1, together with the answer-extraction rule and the paired significance tests. Graph construction and query-to-graph seed matching for both the NER and OpenIE views are detailed in Appendix A.2, and all hyperparameters are listed in Appendix A.3 (Table 9). The evaluation subsets, their question-type composition, and the retrieval corpora follow the HippoRAG-2 protocol and are described in Section 4 and Appendix A.4. All prompts used for graph extraction, bridge prediction, and question answering are reproduced verbatim in Appendix A.9. The code will be released publicly upon acceptance.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 9112–9141, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/25f7be9694d7b32d5cc670927b8091e1-Paper-Conference.pdf.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In Manuela M. Veloso (ed.), *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pp. 2670–2676, 2007. URL http://ijcai.org/Proceedings/07/Papers/429.pdf.

Gordon V. Cormack, Charles L. A. Clarke, and Stefan Büttcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In James Allan, Javed A. Aslam, Mark Sanderson, ChengXiang Zhai, and Justin Zobel (eds.), *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*, pp. 758–759. ACM, 2009. doi: 10.1145/1571941.1572114. URL https://doi.org/10.1145/1571941.1572114.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization, 2025. URL https://arxiv.org/abs/2404.16130.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

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 Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 21497–21515. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/gutierrez25a.html.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3929–3938. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/guu20a.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 Donia Scott, Núria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pp. 6609–6625. International Committee on Computational Linguistics, 2020. doi: 10.18653/V1/2020.COLING-MAIN.580. URL https://doi.org/10.18653/v1/2020.coling-main.580.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia D'amato, {Gerard De} Melo, Claudio Gutierrez, Sabrina Kirrane, {José Emilio Labra} Gayo, Roberto Navigli, Sebastian Neumaier, {Axel-Cyrille Ngonga} Ngomo, Axel Polleres, {Sabbir M.} Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *ACM Computing Surveys*, 54(4), July 2021. ISSN 0360-0300. doi: 10.1145/3447772. Place: New York, NY, USA Publisher: Association for Computing Machinery.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Trans. Mach. Learn. Res.*, 2022, 2022. URL https://openreview.net/forum?id=jKN1pXi7b0.

Glen Jeh and Jennifer Widom. Scaling personalized web search. In Gusztáv Hencsey, Bebo White, Yih-Farn Robin Chen, László Kovács, and Steve Lawrence (eds.), *Proceedings of the Twelfth International World Wide Web Conference, WWW 2003, Budapest, Hungary, May 20-24, 2003*, pp. 271–279. ACM, 2003. doi: 10.1145/775152.775191. URL https://doi.org/10.1145/775152.775191.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7969–7992, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.495. URL https://aclanthology.org/2023.emnlp-main.495/.

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 Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Nv-embed: Improved techniques for training llms as generalist embedding models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 79310–79333, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/c4bf73386022473a652a18941e9ea6f8-Paper-Conference.pdf.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. Larochelle,

M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.

Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, Wenbo Su, and Bo Zheng. GraphReader: Building graph-based agent to enhance long-context abilities of large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 12758–12786, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.746. URL https://aclanthology.org/2024.findings-emnlp.746/.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticæ Investigationes*, 30(1):3–26, 2007. ISSN 0378-4169. doi: https://doi.org/10.1075/li.30.1.03nad. URL https://www.jbe-platform.com/content/journals/10.1075/li.30.1.03nad.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. Graph retrieval-augmented generation: A survey. *ACM Trans. Inf. Syst.*, 44(2): 35:1–35:52, 2026. doi: 10.1145/3777378. URL https://doi.org/10.1145/3777378.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.378. URL https://aclanthology.org/2023.findings-emnlp.378/.

Zackary Rackauckas. Rag-fusion: A new take on retrieval augmented generation. *International Journal on Natural Language Computing*, 13(1):37–47, February 2024. ISSN 2319-4111. doi: 10.5121/ijnlc.2024.13103. URL http://dx.doi.org/10.5121/ijnlc.2024.13103.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Stephen E. Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, 2009. doi: 10.1561/1500000019. URL https://doi.org/10.1561/1500000019.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher Manning. Raptor: Recursive abstractive processing for tree-organized retrieval. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 32628–32649, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/8a2acd174940dbca361a6398a4f9df91-Paper-Conference.pdf.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 9248–9274, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.620. URL https://aclanthology.org/2023.findings-emnlp.620/.

Aditi Singh, Abul Ehtesham, Saket Kumar, Tala Talaei Khoei, and Athanasios V. Vasilakos. Agentic retrieval-augmented generation: A survey on agentic rag, 2026. URL https://arxiv.org/abs/2501.09136.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 3868–3898, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/10a6bdcabbd5a3d36b760daa295f63c1-Paper-Conference.pdf.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475. 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 Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10014–10037, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.557. URL https://aclanthology.org/2023.acl-long.557/.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-pack: Packed resources for general chinese embeddings. In Grace Hui Yang, Hongning Wang, Sam Han, Claudia Hauff, Guido Zuccon, and Yi Zhang (eds.), *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pp. 641–649. ACM, 2024. doi: 10.1145/3626772.3657878. URL https://doi.org/10.1145/3626772.3657878.

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

A Additional Details

A.1 Reproducibility and Evaluation Details

Model access and inference. We accessed qwen-plus through the DashScope API using the floating `qwen-plus` alias. During our experiments from April to June 2026, this alias resolved to `qwen-plus-2025-12-01`. All qwen-plus calls used temperature 0 with thinking mode disabled. Llama-3.3-70B-Instruct was accessed through the DeepInfra API at temperature 0. Even at temperature 0 (greedy decoding), a hosted API can return different completions for the same request on different calls, because each call may land on different backend hardware or be batched with different requests, and the resulting floating-point nondeterminism can flip an occasional token. Still, each result aggregates over 1,000 questions and several reasoning rounds per question, so the aggregate metrics are stable across runs. Recall@5 is stable, since it is a macro-averaged retrieval metric that tolerates small ranking changes, whereas exact match is per-question and sensitive to the exact wording of the final answer. This residual variation comes from the API, not from our pipeline. Within each matched configuration, offline graph construction outputs and embeddings are cached and reused across methods. Exact end-to-end reproducibility additionally requires fixed model versions, prompts, inference settings, and a deterministic execution environment. These conditions cannot be fully controlled for hosted APIs. Hyperparameters and prompts are provided in Appendices A.3 and A.9.

Answer extraction. The QA prompt requests reasoning followed by an answer marked with `Answer:`. We extract the text after this marker as the predicted answer. For the experiments, we standardize answer extraction across the compared methods, including re-extracting ITER-RETGEN answers from its saved reasoning traces using the same rule. This post-processing affects QA scoring but does not change retrieval rankings or R@5. The reader prompt is reproduced in Appendix A.9.

Statistical testing. To assess differences between BRIDGE-Prop and ITER-RETGEN, we pair the two methods on the same questions and use McNemar's exact test for EM and a two-sided sign test for F1 and R@5. Because the tests are paired over the shared question set, they apply to the single-run main results as well as the Llama experiments. Table 8 reports the tests for the main qwen-plus results and table 11 for the Llama results.

Table 8: Significance of the main qwen-plus results (Table 2): BRIDGE-Prop vs. ITER-RETGEN, paired over the shared questions (McNemar for EM, sign test for R@5). Δ matches Table 2. $^{*}p < .05$, $^{**}p < .01$, $^{***}p < .001$.

Dataset	Graph	ΔR@5	p(R@5)	ΔEM	p(EM)
MuSiQue	NER	$+.022$	$.045^{*}$	$+.046$	$< .001^{***}$
MuSiQue	OpenIE	$+.023$	$.005^{**}$	$+.051$	$< .001^{***}$
2Wiki	NER	$+.059$	$< .001^{***}$	$+.094$	$< .001^{***}$
2Wiki	OpenIE	$+.073$	$< .001^{***}$	$+.101$	$< .001^{***}$

A.2 Graph Construction and Seed Matching

NER graph. The NER graph links passages through named entities that predicted bridges can resolve to. We split each passage into sentences and use an LLM to extract named entities from the passage. String matching maps each entity to the sentences and passages that mention it. Entity nodes that co-occur in a sentence are connected, with edge weights given by their corpus co-occurrence counts. Entity–passage edges have unit weight. Synonymy edges connect entity nodes whose embedding cosine similarity exceeds the threshold in Table 9, with similarity used as the edge weight.

OpenIE fact graph. The OpenIE graph follows the HippoRAG-2 backbone (Gutiérrez et al., 2025). An LLM extracts subject–relation–object triples at the passage level. Triple arguments form entity nodes, including descriptive phrases beyond named entities, and relations connect these nodes. Entity–passage and synonymy edges are constructed as in the NER graph. The extraction prompts for both graph views are provided in Appendix A.9.

Seed matching. For the NER graph, we retain sentence embeddings and the entity-to-sentence mapping. Each candidate entity pair is scored by averaging the query's cosine similarity to its pair embedding and to its source-sentence embedding. Maximal marginal relevance (MMR) then selects a diverse set of high-scoring pairs as seeds. For the OpenIE graph, seed matching follows the HippoRAG-2 query-to-fact procedure (Gutiérrez et al., 2025). Both procedures produce the query seed distribution used by PPR.

A.3 Hyperparameters

Table 9 lists the retrieval settings. BRIDGE-Prop allows at most three reasoning rounds after round-0 retrieval and stops early on the LLM's stop signal. The current ranking receives full weight in rank fusion, while earlier rankings share history weight $\lambda = 0.3$ (Eq. 4). Grounding uses exact matching followed by the embedding fallback described in Appendix A.5.

A.4 Question-Type Composition

Table 10 reports the composition of the evaluation subsets by MuSiQue hop count and 2WikiMultiHopQA question type.

Table 9: Retrieval hyperparameters. Seed-selection settings in the first five rows apply to the NER graph.

Component	Hyperparameter	Value
NER seed selection	MMR coefficient	0.7
	MMR candidate pool	30
	Selected seed pairs	5
	Entity-pair / sentence weight	0.5 / 0.5
	Passage seed weight	0.05
Graph	Synonymy threshold	0.8
Propagation	PPR restart probability α	0.5
Bridge seeding	Base weight w_0	0.5
Iteration	Maximum reasoning rounds	3
Rank fusion	RRF constant k	60
	History / current weight	0.3 / 1.0
Reader	Top-k passages	5

Table 10: Question-type composition of the 1,000-question evaluation subsets. A dash indicates that a question type is not present in that dataset.

Dataset	MuSiQue reasoning structures			2Wiki question types			
	2-hop	3-hop	4-hop	Comparison	Inference	Compositional	Bridge-comparison
MuSiQue	518	316	166	–	–	–	–
2Wiki	–	–	–	244	108	413	235

A.5 Bridge Grounding

Predicted bridges are grounded to graph nodes before seeding: an exact lowercased-string match first, else the nearest node by encoder cosine above 0.55, otherwise discarded. Grounding rarely fails: of the bridges the loop attempts to ground on MuSiQue, 99.4–99.9% resolve across the NER (qwen, Llama) and OpenIE (Llama) graphs. Most are exact matches (81% for qwen, 91–92% for Llama), the encoder fallback recovering the rest.

A.6 Generalization to another LLM (Llama-3.3-70B)

Table 11 gives the full EM/F1/R@5 comparison for the Llama-3.3-70B runs (summarized in Table 6 of the main paper), with paired significance tests. The following tables then break it down by reasoning depth (MuSiQue, Table 12) and question type (2WikiMultiHopQA, Table 13). R@5 is the primary. The structure matches the qwen main results (Tables 3–4).

Table 11: Generalization to Llama-3.3-70B (Llama-built NER graph), full metrics. Stars on Δ mark a significant difference from ITER-RETGEN (paired test: McNemar for EM, sign test for F1 and R@5; $^{*}p < .05$, $^{**}p < .01$, $^{***}p < .001$; $^{\text{n.s.}}$ not significant).

	MuSiQue			2WikiMultiHopQA		
Method	EM	F1	R@5	EM	F1	R@5
round-0 (single pass)	.252	.357	.464	.455	.509	.666
+ ITER-RETGEN	.383	.492	.588	.547	.622	.789
+ BRIDGE-Prop (ours)	.363	.477	.606	.643	.722	.892
Δ	$-.020^{\text{n.s.}}$	$-.015^{\text{n.s.}}$	$+.018^{*}$	$+.096^{***}$	$+.101^{***}$	$+.103^{***}$

A.7 Embedding Robustness

To test whether the retrieval gain depends on the sentence encoder, we rerun the 2WikiMultiHopQA NER comparison (Llama-3.3-70B) with a weak encoder (MiniLM) and a stronger one (bge-base); Table 14 reports R@5 by question type. BRIDGE-Prop's advantage over ITER-RETGEN holds

Table 12: MuSiQue R@5 by reasoning depth (Llama-3.3-70B, Llama-built NER graph, n=1000). Δ = BRIDGE-Prop − ITER-RETGEN.

Method	2-hop	3-hop	4-hop	All
round-0 (single pass)	.583	.387	.236	.464
+ ITER-RETGEN	.747	.481	.294	.588
+ BRIDGE-Prop (ours)	.758	.499	.337	.606
Δ	+.011	+.018	+.043	+.018

Table 13: 2WikiMultiHopQA R@5 by question type (Llama-3.3-70B, Llama-built NER graph, n=1000). Δ = BRIDGE-Prop − ITER-RETGEN.

Method	comparison	inference	compositional	bridge-comparison	All
round-0 (single pass)	.803	.727	.678	.477	.666
+ ITER-RETGEN	.828	.856	.885	.550	.789
+ BRIDGE-Prop (ours)	.924	.912	.923	.797	.892
Δ	+.096	+.056	+.038	+.247	+.103

under both and concentrates on bridge-comparison (+.217 under MiniLM, +.105 under bge-base). It is larger under the weaker MiniLM: bge-base already lifts ITER-RETGEN's recall near the ceiling on the single-bridge types (.92–.94), leaving less headroom, so the smaller gap reflects a ceiling effect.

Table 14: Embedding robustness on 2WikiMultiHopQA (Llama-3.3-70B, qwen-built NER graph). R@5 by question type and overall, under a weak (MiniLM, 22M) and a stronger (bge-base, 109M) sentence encoder, for ITER-RETGEN and BRIDGE-Prop. Δ = BRIDGE-Prop − ITER-RETGEN; stars mark a significant difference (sign test; $^{*}p < .05$, $^{***}p < .001$).

	ITER-RETGEN	BRIDGE-Prop	Δ
MiniLM (22M, 384-d), n=1000			
Overall	.808	.897	+.089***
comparison	.844	.939	+.094***
inference	.884	.907	+.023
compositional	.903	.932	+.029*
bridge-comparison	.570	.787	+.217***
bge-base (109M, 768-d), n=1000			
Overall	.895	.950	+.055***
comparison	.916	.992	+.076***
inference	.926	.944	+.019
compositional	.944	.967	+.023*
bridge-comparison	.772	.878	+.105***

A.8 Mechanistic Comparison: Win–Loss and Failure Attribution

We compare BRIDGE-Prop and ITER-RETGEN question by question on the NER backbone with qwen-plus, our main configuration. BRIDGE-Prop wins more often than it loses on both datasets (net +46 on MuSiQue, +94 on 2Wiki; Table 16). BRIDGE-Prop's edge is clearest when round-0 retrieves no gold passage: it recovers a gold passage in a later round far more often than ITER-RETGEN (Table 15, with fully-recovered rates in Table 18).

Where one method wins and the other loses, the loss is dominated by retrieval rather than answer generation: on 2Wiki, 88% of ITER-RETGEN's losses and 76% of BRIDGE-Prop's are cases where the gold passages were never fetched (Table 17); ITER-RETGEN's retrieval failures are mostly cases where it named the bridge in text but did not fetch the gold. Figure 3 shows the discovered bridges differ by dataset: on MuSiQue many are off-chain distractors, whereas on 2Wiki almost all lie on

Table 15: Recovery from questions with round-0 R@5 equal to zero.

Dataset	# Questions	ITER-RETGEN	BRIDGE-Prop
MuSiQue	185	55%	69%
2Wiki	27	44%	74%

the reasoning chain. This larger share of off-chain distractors makes the bridge signal noisier on MuSiQue, consistent with BRIDGE-Prop's smaller margin over ITER-RETGEN there.

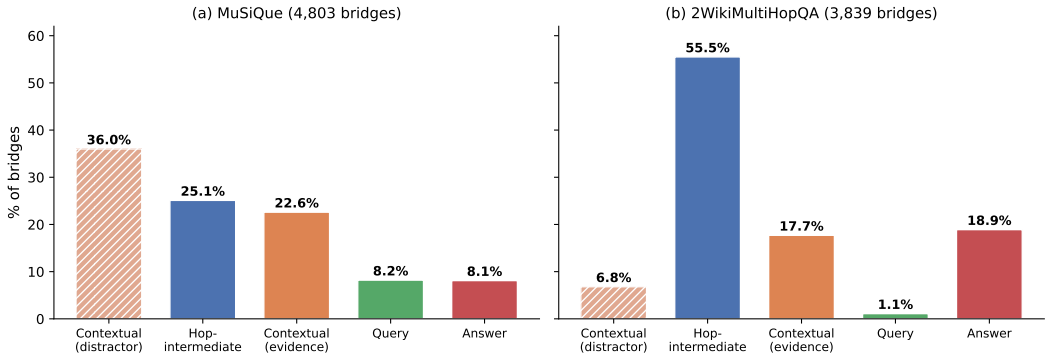

Figure 3: Composition of the bridge entities the LLM discovers. We label each bridge by matching it against the question, the intermediate decomposition answers, and the final answer; entities that match none of these are *contextual*, which we further split into evidence or distractor by whether the entity links to a gold passage in the graph. The two datasets behave differently: on **(a)** MuSiQue many bridges are off-chain distractors, whereas on **(b)** 2WikiMultiHopQA, whose reasoning chains are short and explicit, almost all bridges lie on the chain.

Table 16: Per-question outcomes of BRIDGE-Prop vs. ITER-RETGEN (NER backbone). For each question we record which method answers correctly; Table 17 analyzes why each method loses.

Dataset	Both correct	BRIDGE-Prop win	ITER win	Both wrong	Net
MuSiQue	285	113	67	535	+46
2WikiMultiHopQA	501	148	54	297	+94

Escaping echo loops. We also measure how much each method's reasoning repeats across rounds, by the overlap between consecutive reasoning traces (Table 19). ITER-RETGEN reuses much of its previous text each round, whereas BRIDGE-Prop's reasoning changes as it explores and names new bridges, giving substantially lower cross-round overlap on both datasets. High repetition tends to produce near-identical retrievals, one way ITER-RETGEN stays anchored to the same evidence.

Table 17: Why each method loses on 2WikiMultiHopQA. Among questions where one method is correct and the other is not, we classify the losing method's failure as retrieval (gold passages not fetched) or answer generation (gold fetched but answer wrong).

Failure cause	ITER-RETGEN fails (n=148, BRIDGE-Prop wins)	BRIDGE-Prop fails (n=54, ITER wins)
Retrieval failure (gold not fetched, R@5<1)		
Bridge not produced	2 (1.4%)	0 (0.0%)
Bridge produced, gold not fetched	128 (86.5%)	41 (75.9%)
Retrieval subtotal	**130 (87.8%)**	**41 (75.9%)**
Answer-generation (QA) failure (gold fetched, answer wrong)		
Reasoning error	6 (4.1%)	2 (3.7%)
Hallucination	2 (1.4%)	2 (3.7%)
Refusal	4 (2.7%)	5 (9.3%)
Format only (EM=0)	6 (4.1%)	4 (7.4%)
QA subtotal	**18 (12.2%)**	**13 (24.1%)**

Table 18: Detailed recovery after round-0 retrieves no gold passage. "Recovered" means that at least one gold passage is retrieved in a later round; "fully" means that all gold passages are retrieved.

Dataset	Method	Recovered	Fully	Mean R@5
MuSiQue (n=185)	ITER-RETGEN	55%	8%	.279
	BRIDGE-Prop	69%	12%	.381
2Wiki (n=27)	ITER-RETGEN	44%	15%	.306
	BRIDGE-Prop	74%	41%	.574

Table 19: Cross-round repetition of the reasoning generations (mean over consecutive rounds; NER graph, qwen-plus). $\mathrm{Jaccard}_1$ and $\mathrm{Jaccard}_2$ are word- and bigram-overlap. Lower means the reasoning changes more across rounds.

Method	$\mathrm{Jaccard}_1$	$\mathrm{Jaccard}_2$	ROUGE-L
MuSiQue			
ITER-RETGEN	.50	.33	.50
BRIDGE-Prop	**.36**	**.14**	**.30**
2WikiMultiHopQA			
ITER-RETGEN	.62	.48	.64
BRIDGE-Prop	**.38**	**.16**	**.36**

A.9 PROMPTS

The pipeline uses five fixed one-shot prompts, all run at temperature 0. We reproduce them verbatim below. Braces such as {passage} or {query} mark a runtime-filled slot, and [. . .] marks an in-context example passage abridged for space.

A.9.1 PASSAGE NAMED-ENTITY EXTRACTION (NER GRAPH)

```
[system]
Your task is to extract named entities from the given paragraph. Respond
     with a JSON list of entities.

[one-shot user]
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.

[one-shot assistant]
{"named_entities":
    ["Radio City", "India", "3 July 2001", "Hindi", "English", "May
    2008", "PlanetRadiocity.com"]
}

[user]
{passage}
```

A.9.2 QUERY KEY-ENTITY EXTRACTION (ROUND-0 SEED MATCHING)

```
[system]
You're a very effective entity extraction system.

[one-shot user]
Please extract all named entities that are important for solving the
    questions below. Place the named entities in json format.

Question: Which magazine was started first Arthur's Magazine or First
    for Women?

[one-shot assistant]
{"named_entities": ["First for Women", "Arthur's Magazine"]}

[user]
Question: {query}
```

A.9.3 OPENIE TRIPLE EXTRACTION (OPENIE GRAPH)

```
[system]
Your task is to construct an RDF (Resource Description Framework) graph
    from the given passages and named entity lists. Respond with a JSON
    list of triples, with each triple representing a relationship in the
     RDF graph.

Pay attention to the following requirements:
- Each triple should contain at least one, but preferably two, of the
    named entities in the list for each passage.
- Clearly resolve pronouns to their specific names to maintain clarity.

[one-shot user]
```

```
Convert the paragraph into a JSON dict, it has a named entity list and a
    triple list.
Paragraph: Radio City [... same passage as above ...]
{"named_entities": ["Radio City", "India", "3 July 2001", "Hindi", "
   English", "May 2008", "PlanetRadiocity.com"]}

[one-shot assistant]
{"triples": [
    ["Radio City", "located in", "India"],
    ["Radio City", "is", "private FM radio station"],
    ["Radio City", "started on", "3 July 2001"],
    ["Radio City", "plays songs in", "Hindi"],
    ["Radio City", "plays songs in", "English"],
    ["Radio City", "forayed into", "New Media"],
    ["Radio City", "launched", "PlanetRadiocity.com"],
    ["PlanetRadiocity.com", "launched in", "May 2008"],
    ["PlanetRadiocity.com", "is", "music portal"],
    ["PlanetRadiocity.com", "offers", "news"],
    ["PlanetRadiocity.com", "offers", "videos"],
    ["PlanetRadiocity.com", "offers", "songs"]
]}

[user]
Convert the paragraph into a JSON dict, it has a named entity list and a
    triple list.
Paragraph:
{passage}
{named_entity_json}
```

A.9.4 BRIDGE PREDICTION

```
[system]
You are a retrieval reasoning assistant inspired by how human memory
   uses both observed cues and hypothetical context. Given an original
   query, the documents retrieved so far, and optionally a reasoning
   trace, your job is to:

1. Analyze what information has been found and what is still missing.
2. Identify CONCRETE bridge entities you SEE in retrieved documents (
   these will guide structured retrieval).
3. Hypothesize a CONTEXT SCENARIO - natural language describing the
   likely time period, location, domain, or related concepts (this is
   your reasoning context, NOT the retrieval query).
4. Rewrite the query using ONLY bridge entities and the original
   question - keep it focused and short. Do NOT include the
   context_scenario in the rewritten_query.
5. Decide whether to continue retrieval or stop.

Respond in JSON format:
{
    "analysis": "Brief analysis of what's found vs missing",
    "discovered_entities": ["entity1", "entity2"],
    "context_scenario": "Hypothesized context: time period, location,
   domain, related concepts",
    "rewritten_query": "Focused natural language query with bridge
   entities only",
    "should_stop": false
}

Rules:
- "discovered_entities": ONLY entities you can directly SEE in retrieved
    documents. 1-5 entities max. Use lowercase. DO NOT guess.
```

```
- "context_scenario": Short natural language description of the
    hypothesized context. Stored as your reasoning trace and seen by
    future rounds, but NOT used directly for retrieval. Examples:
  * "Likely 16th century European religious reform, possibly Lutheran/
    Calvinist figures"
  * "North Carolina state administration history, 18-19th century
    capital designation"
  * "Modern NBA basketball event, post-2000 era"
- "rewritten_query": Focused query using ONLY bridge entities +
    reformulation of the original question. DO NOT add context_scenario,
     time periods, or hypothesized terms here. Keep it short and
    concrete.
- "should_stop": true ONLY if retrieved documents already contain a
    complete answer.

Example:
Q: "When did the city where Curry's college is located become NC's
    capital?"
After round 0 finding Stephen Curry and Davidson College:
{
    "analysis": "Found Curry attended Davidson College in NC. Need the
    city of Davidson and its history as state capital.",
    "discovered_entities": ["davidson college", "north carolina"],
    "context_scenario": "Likely 18-19th century North Carolina state
    administration history; possibly involving Raleigh or another NC
    city becoming the state capital.",
    "rewritten_query": "When did Davidson, North Carolina become the
    state capital?",
    "should_stop": false
}
```

A.9.5 READER (QUESTION ANSWERING)

```
[system]
As an advanced reading comprehension assistant, your task is to analyze
    text passages and corresponding questions meticulously. Your
    response start after "Thought: ", where you will methodically break
    down the reasoning process, illustrating how you arrive at
    conclusions. Conclude with "Answer: " to present a concise,
    definitive response, devoid of additional elaborations.

[one-shot user]
Wikipedia Title: Neville A. Stanton
Neville A. Stanton is a British Professor of Human Factors and
    Ergonomics at the University of Southampton. [...]

Wikipedia Title: Southampton
The University of Southampton, which was founded in 1862 and received
    its Royal Charter as a university in 1952, has over 22,000 students.
     [...]
[+ three more Wikipedia passages]

Question: When was Neville A. Stanton's employer founded?

Thought:

[one-shot assistant]
The employer of Neville A. Stanton is University of Southampton. The
    University of Southampton was founded in 1862.
Answer: 1862.

[user]
Wikipedia Title: {passage}
```

```
...
Question: {query}
Thought:
```

A.10 CASE STUDY

Figure 4 traces one 2WikiMultiHopQA question through both methods under the same setup: qwen-plus, all-MiniLM-L6-v2 embeddings, the NER graph, and five passages retrieved per round. The question asks which country the director of *Get Carter* (2000) is from, a two-hop chain: the film leads to its director Stephen Kay, whose biography states he was born in New Zealand. Both methods first retrieve the film passage, which calls the film "an American action thriller film" and names Stephen Kay but not his origin. ITER-RETGEN answers from this passage alone, infers an American director from the "American film" label, and stays anchored to it across all three rounds (R@5 $= 0.50$), repeating "United States". BRIDGE-Prop instead names *Stephen Kay* as a bridge entity and seeds the next retrieval with him; his biography is retrieved (R@5 $0.50 \rightarrow 1.00$) and the answer, New Zealand, is read from it.

The two runs differ only in how reasoning feeds retrieval: BRIDGE-Prop turns the inferred bridge into an explicit retrieval target, while ITER-RETGEN relies on generated text and cannot escape the passage it already holds. This matches the aggregate results: BRIDGE-Prop recovers from a failed round-0 retrieval markedly more often than ITER-RETGEN (Appendix A.8). The gap is propagation, not discovery.

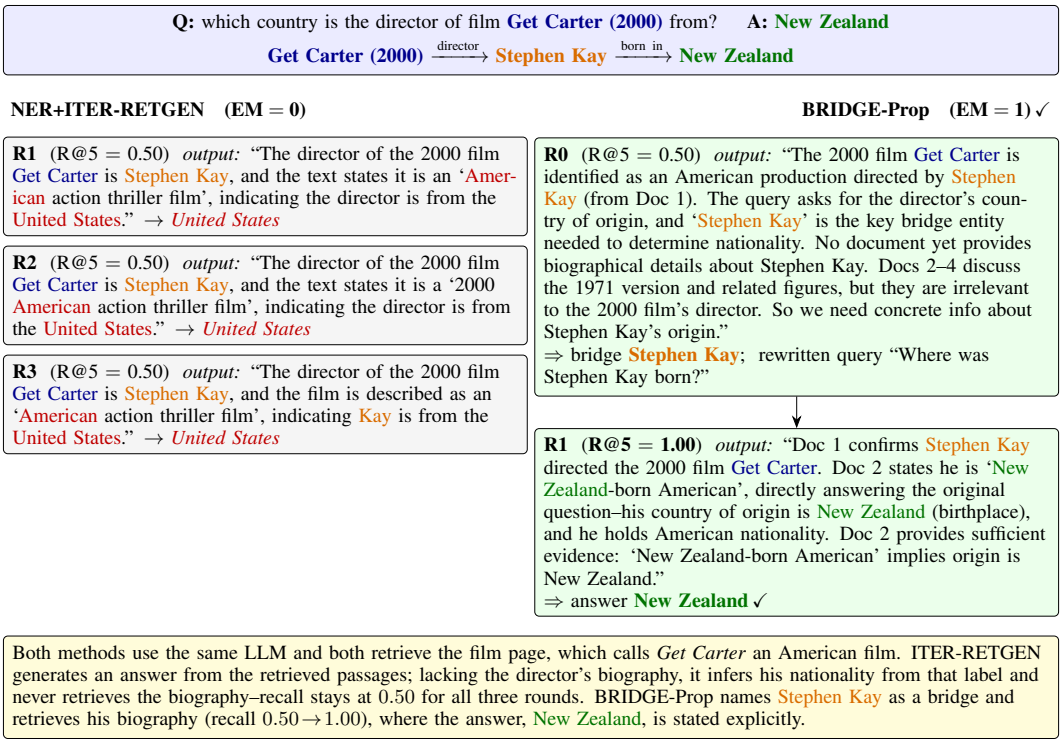

Figure 4: Information flow for one 2WikiMultiHopQA question (NER graph, qwen-plus, five passages per round). Both methods retrieve the film's page, but only BRIDGE-Prop reaches the director's biography: ITER-RETGEN infers the director's nationality from the film's "American" label, while BRIDGE-Prop seeds the director as a bridge entity and reads the answer from his biography. Each box shows that round's R@5; column headers show the final EM.