RILO: QUERY PLANNING WITH LATENT RELATIONAL OPERATORS FOR MULTI-HOP RAG

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-hop retrieval requires connecting evidence across documents and deciding where to search next. The scale and sparsity of the surface-relation space complicate direct planning of multi-hop retrieval. We introduce RILO, a retrieval planner that learns a compact vocabulary of composable latent relational operators. These operators capture transformations shared across surface relations and are trained on corpus-derived paths to predict evidence endpoints. Given a question and retrieved evidence, RILO composes the operators into candidate multi-step plans. Each plan and its predicted endpoint guide a single corpus query without online graph traversal. A budget-aware controller selects the next query or stops based on expected evidence gain. RILO then updates its evidence state and replans as new passages arrive. In controlled evaluations on 2WikiMultiHopQA, HotpotQA, and MuSiQue, RILO achieves 73.10% macro complete-support coverage within the top ten passages (SC@10). With at most three corpus calls, it exceeds the strongest evaluated baseline by 3.86 percentage points.

1 INTRODUCTION

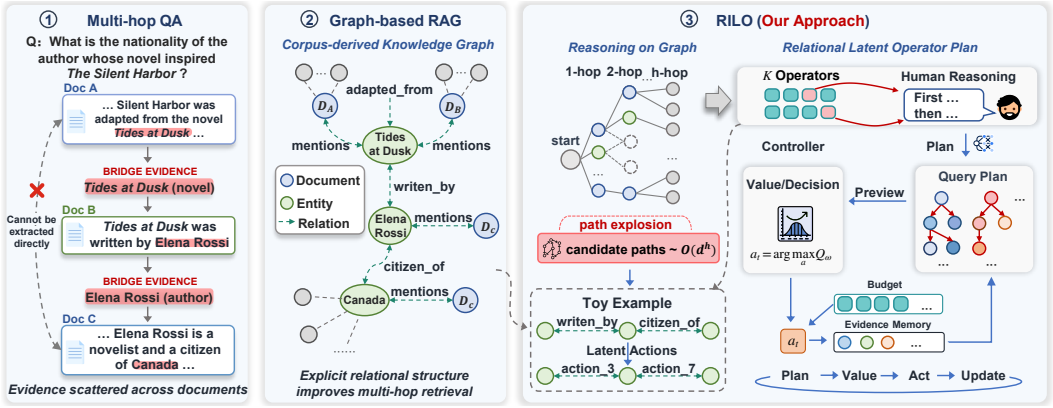

Figure 1: **From scattered evidence to executable retrieval plans.** (i) Evidence is distributed across documents; (ii) a corpus-derived graph exposes its relational links; and (iii) RILO replaces online path expansion with a fixed-size vocabulary of composable latent operators, predicts candidate evidence endpoints before corpus access, and uses a budget-aware controller to select an action and replan from returned evidence.

Answering multihop questions requires assembling evidence distributed across documents (Yang et al., 2018; Ho et al., 2020). The connections between these documents often depend on intermediate entities or facts that link one piece of evidence to the next. RAG provides access to this evidence at inference time (Lewis et al., 2020). A retrieved passage may provide part of the evidence and reveal entities that guide the search for the remaining evidence. Graph-based retrievers make these connections explicit through Wikipedia hyperlinks or extracted entity relations (Asai et al., 2020; Gutiérrez et al., 2024). These relations expose many plausible continuations from the current evidence. Some continuations may lead to useful evidence only after several steps, so their value extends beyond the next passage. Existing graph and iterative retrievers typically choose among these continuations

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.

during corpus interaction through graph expansion, path search, or query rewriting (Gutiérrez et al., 2024; Xiong et al., 2021; Trivedi et al., 2023). The central problem is therefore to decide which continuation is worth pursuing based on the evidence it may eventually reveal. To address this problem, **RILO anticipates where candidate continuations may lead before issuing the next corpus query.**

Relational operators provide compact primitives for retrieval planning. Planning these continuations directly over surface relations is difficult because the corpus contains many distinct relation labels. Many labels occur infrequently, and the number of possible relation sequences grows rapidly with planning depth. If each step admits up to d relation choices, there can be up to d^h candidate sequences of length h. However, different surface relations can induce similar evidence transitions. For example, *written_by*, *directed_by*, and *composed_by* all move from a creative work to its creator. An intuitive analogy is human reasoning through abstract relational transitions, as illustrated in Figure 1. These shared transitions suggest that retrieval planning can use a smaller set of reusable primitives. Accordingly, RILO learns a compact vocabulary of latent relational operators from relation paths extracted from the corpus. Each operator captures a transition shared across multiple relations. These operators are trained to predict the evidence reached by the extracted paths. Composing these operators in sequence represents longer evidence continuations within the same compact vocabulary.

***Key Insight:* A compact vocabulary preserves retrieval-relevant structure.** The key question is whether compressing the surface-relation space into a small operator vocabulary still preserves where those relations lead in retrieval space. Figure 2 varies the operator vocabulary size K. Endpoint R@10 measures top-ten target-hit rate on held-out paths, excluding the source passage. $K = 8$ retains 90.4% of raw-relation R@10 with a 779× smaller vocabulary. Consequently, this compression reduces the space of candidate transitions, and preserves most of the endpoint-retrieval signal needed for planning.

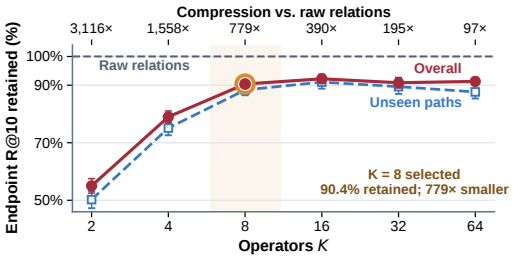

Figure 2: A compact vocabulary retains most retrieval performance, including on unseen paths.

Our Approach. RILO uses this vocabulary to plan where to search next. Starting from the question and retrieved evidence, it composes operators into candidate continuations and predicts their endpoints. Each plan and its predicted endpoint guide the construction of a single corpus query, even when the plan spans several relational steps. Before execution, RILO compares candidate queries by their expected downstream evidence gain within the remaining budget and chooses the next query or stops. Returned passages then update the evidence state and provide the context for the next round of planning, grounding subsequent decisions in the newly retrieved evidence.

Contributions. We make three contributions to multi-hop retrieval: ***(i)*** a compact, path-supervised vocabulary of composable relational operators learned from corpus-derived relation paths; ***(ii)*** an endpoint-conditioned query mechanism that realizes a multi-step relational plan as one corpus search; and ***(iii)*** a budget-conditioned retrieval policy for action selection, stopping, and evidence-grounded replanning as new evidence is retrieved. Across three multi-hop benchmarks, RILO achieves 73.10% macro SC@10, exceeding the second-best method by 3.86 points. Ablations show that relational abstraction, composition, and replanning each contribute to these gains.

2 RELATED WORK

Multi-hop retrieval. Dense retrieval encodes questions (Karpukhin et al., 2020; Lewis et al., 2020) or hypothetical documents (Gao et al., 2023) to search a passage index. Multi-hop methods condition subsequent retrieval on evidence (Xiong et al., 2021), search passage chains (Zhang et al., 2024), generate queries (Zhuang et al., 2024), or learn decomposition-free retrieval (Erker et al., 2026). IRCoT and CoRAG interleave retrieval and reasoning (Trivedi et al., 2023; Wang et al., 2025). Graph retrievers propagate relevance or search neighborhoods and paths (Gutiérrez et al., 2024; Shen et al., 2025; Liu et al., 2025; Mavromatis & Karypis, 2025; Ni et al., 2025). RILO uses graph structure for offline supervision rather than online traversal: grounded graph paths supervise a reusable operator vocabulary. At retrieval time, its plans query a passage index without accessing the supervision graph.

Query planning. LevelRAG separates atomic-query planning from retriever-specific rewriting (Zhang et al., 2025b); PlanRAG organizes atomic queries into cost-guided logical query trees (Xu et al., 2026). RILO instead represents a continuation as a sequence of learned discrete operators and realizes the entire sequence as one query, without executing individual operators as separate subqueries. Among latent retrieval methods, L-RAG uses intermediate LLM representations to retrieve next-hop evidence (Lin et al., 2025). LAnR derives retrieval vectors from a designated prediction token and learns a stopping head within a unified model (Ha Lan N.T et al., 2026). In contrast, RILO constructs text queries from composed plans and their predicted endpoints for a frozen corpus encoder.

Adaptive retrieval. Retrieval can adapt to question complexity (Jeong et al., 2024; Tang et al., 2025), uncertainty, or reflection (Jiang et al., 2023; Asai et al., 2024; Su et al., 2024). Stop-RAG learns value-based stopping (Park et al., 2025); DynaKRAG jointly learns continuation and selection among feasible evidence operations (Zhou et al., 2026a). RILO applies budget-conditioned control to compositional actions built from path-supervised operators, alongside direct dense retrieval and stopping. This separates two decisions: operator depth sets the relational lookahead within one query, whereas the controller allocates corpus calls across queries.

3 PROBLEM FORMULATION

Retrieval setting. Given a question q and a passage corpus $\mathcal{C}$, the task is to recover supporting evidence using at most B corpus calls. The retriever must decide what to search for next and when to stop as evidence accumulates. A corpus call denotes one search over the passage index. An initial dense retrieval forms the ranked memory M_0 and consumes one call, leaving $b_0 = B - 1$. At round t, the state s_t records the question q, the current ranked memory M_t, and the action history. The policy makes its next decision using s_t and the remaining budget b_t.

Retrieval actions. At each round, the policy can stop, issue a direct dense query, or execute a query guided by a compositional action. In RILO, a compositional action $a = (p, o_{1:\ell})$ specifies a relational continuation from an entity p observed in the question or M_t. Here, $o_{1:\ell}$ is an ordered sequence of ℓ operators drawn from a learned vocabulary $\mathcal{O}$ of size K. The sequence describes a predicted evidence continuation and is realized as a single query. Executing this query consumes one corpus call, regardless of ℓ. The supervision graph provides paths for offline operator learning; online planning uses the fixed learned model. Before the next corpus search, RILO constructs the candidate set $\mathcal{A}^{\text{comp}}(s_t)$. Given (s_t, b_t), policy μ selects an action a_t from $\mathcal{A}(s_t) = \{\text{STOP}, \text{DENSE}\} \cup \mathcal{A}^{\text{comp}}(s_t)$. If a query is executed, the returned passages O_t update the evidence memory. The transition $s_{t+1} = \mathcal{T}(s_t, a_t, O_t)$ records this update and the chosen action, while $b_{t+1} = b_t - 1$.

Objective. Retrieval ends when the policy selects STOP or exhausts the budget. Let $U(s)$ denote evidence utility, rewarding both support coverage and completeness within the top passages of memory. Let τ be the number of queries executed after the initial retrieval. We seek a policy that maximizes final evidence utility while accounting for the cost of additional searches.

$$\max_{\mu} \mathbb{E}_{\mu}[U(s_\tau) - \lambda\tau], \qquad 0 \le \tau \le B - 1, \tag{1}$$

where $\lambda \ge 0$ assigns a cost to each additional query. With initialization fixed, an equivalent incremental reward is $U(s_{t+1}) - U(s_t) - \lambda$ for each executed query; STOP yields zero additional reward. Support annotations define U exclusively for offline learning. Appendix A.5 specifies the utility definition.

4 METHOD

RILO uses relational patterns learned from corpus paths to reason about where to search next (Figure 3). The central idea is to guide query construction with predictions of where the current evidence may lead. We capture these patterns in a compact **relational language**, whose operators are learned offline from paths extracted from the corpus graph. Using these learned transitions, RILO looks several relational steps beyond the current evidence to guide a single corpus query. The search proceeds only when its expected evidence gain justifies another call. Learned relational structure serves as a lightweight reasoning layer and gives each search a clearer direction.

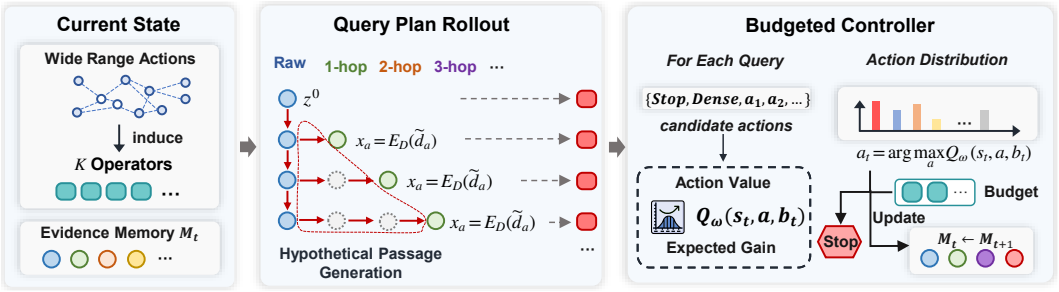

Figure 3: Overview of RILO's compositional retrieval loop. Grounded graph paths induce a compact relational language whose operator compositions guide query construction before corpus access. A budget-conditioned controller selects the next action or stops, while returned evidence grounds subsequent planning.

4.1 LEARNING A COMPACT RELATIONAL LANGUAGE

For retrieval planning, a compact relation vocabulary must preserve information about where evidence may lead. RILO learns shared transition primitives from corpus-derived paths, using endpoint prediction to supervise which surface relations can share an operator.

Shared transition primitives. We define a vocabulary $\mathcal{O} = \{o_0, \ldots, o_{K-1}\}$, where each o_k is a learned code that conditions a shared transition model G_θ. A frozen relation encoder E_r and a learned router F_ϕ assign each surface relation r to a single operator:

$$q_\phi(\cdot \mid r) = \mathrm{softmax}\left(\frac{F_\phi(E_r(r))}{\tau_{\mathrm{route}}}\right), \qquad k(r) = \arg\max_k q_\phi(k \mid r),$$

where τ_{route} is the routing temperature.

Table 1: Operator assignments at $K = 8$ on 2WikiMultiHopQA. Usage denotes the percentage assigned to each latent operator.

Operator	Use (%)	Representative surface relations
o_0	8.3	directed; written_by; created; founded
o_1	13.0	stars; released; produced; appeared_in
o_2	15.8	released_in; year; published_in
o_3	15.6	located_in; born_in; died_in; in
o_4	13.1	includes; married; part_of; from
o_5	4.1	produced_by; won; nominated_for
o_6	10.3	son_of; died_on; daughter_of; is_from
o_7	19.8	is; directed_by; was; nationality

Relations routed to the same code share a conditional transformation, but its output still depends on the source evidence. Table 1 reports representative assignments and operator usage at $K = 8$. For example, o_0 groups *directed*, *written_by*, *created*, and *founded*. These assignments are learned through endpoint-prediction supervision, without predefined semantic classes.

Learning through composition. A corpus-derived directed path supplies a source passage d_{src}, an ordered relation sequence $r_{1:L}$, and a target passage d^+ containing the terminal entity. Starting from the source passage representation, a continuous teacher T_ψ processes the relation embeddings $E_r(r_i)$, while the discrete model processes the corresponding operator codes $o_{k(r_i)}$. Both models update their recurrent states in relation order and decode the final states into endpoint predictions $\widetilde{z}_L$ and $\widehat{z}_L$, respectively. Each predicted intermediate state serves as the input to the next transition. For example, the schematic *written_by–citizen_of* rollout is trained to retrieve Doc C from Doc B in Figure 1.

Training the operator vocabulary. We first train the teacher with endpoint retrieval supervision and freeze it. For both one-step and multi-step paths, the discrete model minimizes

$$\mathcal{L}_{\mathrm{op}} = \mathcal{L}_{\mathrm{ret}} + \lambda_{\mathrm{dist}}\mathcal{L}_{\mathrm{dist}} + \lambda_{\mathrm{bal}}\mathcal{L}_{\mathrm{bal}} + \lambda_{\mathrm{ent}}\mathcal{L}_{\mathrm{ent}}, \tag{2}$$

where $\mathcal{L}_{\mathrm{ret}}$ trains $\widehat{z}_L$ to retrieve d^+ against the source passage and sampled competitors, using frozen passage embeddings. $\mathcal{L}_{\mathrm{dist}}$ aligns $\widehat{z}_L$ with the frozen teacher prediction $\widetilde{z}_L$ for the same complete path. Both losses supervise the final endpoint. The routing penalties encourage broad aggregate operator usage ($\mathcal{L}_{\mathrm{bal}}$) and confident individual assignments ($\mathcal{L}_{\mathrm{ent}}$). A straight-through estimator propagates gradients through hard operator assignments, allowing the router, codes, and transition model to be trained jointly. Appendix A.3 specifies the full operator, query-realization, and value-learning losses. At retrieval time, the planner selects operators directly from $\mathcal{O}$ and composes their transitions with G_θ (Section 4.2).

4.2 COMPOSITIONAL QUERY PLANNING

RILO performs relational lookahead before accessing the corpus. It composes the operators learned in Section 4.1 into evidence-anchored plans. Each plan and its predicted endpoint then guide query construction for the frozen passage index. A composition thus specifies a single candidate search.

Planning from observed evidence. At round t, each plan $a = (p, o_{1:\ell})$ starts from an entity p mentioned in the question q or retrieved memory M_t (Figure 4). We retain at most five distinct pivots, taking question entities first and then memory entities ordered by the highest-ranked passage containing them. These pivots anchor candidate continuations in the evidence available at the current round. For each pivot, z^0 represents its observed context. The proposal p_η scores the next operator using the retrieval state s_t and the endpoint predicted for the current prefix. The state s_t remains fixed during the rollout, while each predicted endpoint guides the next operator choice. Let $\mathrm{Rollout}_\theta$ denote the recurrent application of G_θ and its endpoint readout. For a plan of length $\ell \in \{1, \dots, H\}$, the prefix endpoints and cumulative proposal score are

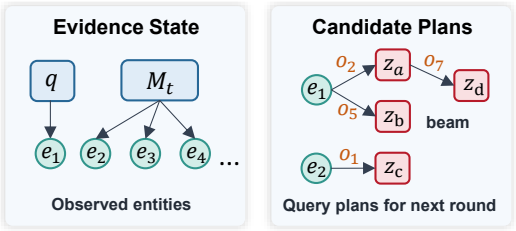

Figure 4: Evidence-grounded planning before corpus access. Entities observed in the question and memory anchor operator compositions. A bounded beam retains endpoint-distinct candidate plans for query realization.

$$z^i = \mathrm{Rollout}_\theta(z^0, o_{1:i}), \qquad S_\eta(a \mid s_t) = \sum_{i=1}^{\ell} \log p_\eta(o_i \mid s_t, z^{i-1}), \quad i = 1, \dots, \ell. \tag{3}$$

Using these scores, a bounded beam expands prefixes up to depth H, ranks candidates by S_η, and merges plans with similar predicted endpoints. The highest-scoring surviving plans form $\mathcal{A}^{\mathrm{comp}}(s_t)$, which contains at most J plans, each paired with its endpoint z^ℓ. The resulting set provides endpoint-distinct continuations for query construction. The proposal is trained offline on routed relation sequences compatible with training questions, as described in Section 4.4.

Turning plans into queries. For each retained plan, a tiny-T5 query realizer C_γ (18.2M parameters) combines the retrieval state s_t, the full plan a, and its endpoint z^ℓ. These inputs connect the current information need with the predicted evidence continuation. It generates an evidence-focused query $\widetilde{d}_a$ with a 96-token cap (32 in the short-query comparison), encoded by the frozen corpus encoder:

$$\widetilde{d}_a = C_\gamma(s_t, a, z^\ell), \qquad x_a = E_D(\widetilde{d}_a). \tag{4}$$

For example, the continuation from *Beloved* to its author and nationality could yield the short query "Beloved author nationality". The DENSE action complements the compositional candidates with a direct retrieval option. A direct query generator C_{direct} uses only the question and current memory to produce a retrieval query, giving $x_{\mathrm{DENSE},t} = E_D(C_{\mathrm{direct}}(q, M_t))$.

Since the query is conditioned on M_t, it can be revised as new evidence is retrieved. This revised query uses the same frozen passage index and the same memory update rule as the compositional queries. Both dense and compositional queries are constructed before the next corpus search. The controller in Section 4.3 then evaluates their corresponding actions under the current state and remaining budget, and selects either a retrieval action or STOP. Once a retrieval action is selected, its query searches the frozen passage index, and the returned passages update M_t. Therefore, each selected retrieval action consumes one corpus call, regardless of the number of operators in a compositional plan.

4.3 BUDGETED RETRIEVAL CONTROL

The controller evaluates the candidates from Section 4.2 using the current evidence state and the remaining budget of corpus calls. Their predicted values determine which query to execute next and when to stop retrieval.

Budgeted action values. The value function $Q_\omega(s_t, a, b_t)$ estimates the expected additional evidence utility of executing a and continuing within budget b_t, after subtracting retrieval costs. The estimate combines immediate evidence gain with gains from subsequent searches, accounting for the costs of both current and future calls. For example, Doc B in Figure 1 identifies the novelist, enabling a

later search for nationality evidence in Doc C. Its value therefore depends on whether the remaining budget allows that continuation. The inputs to Q_ω comprise the retrieval state, budget, action representation, and query vector, with the predicted endpoint additionally supplied for compositional actions. Both compositional actions and DENSE are evaluated relative to stopping, whose value is fixed at $Q_\omega(s_t, \text{STOP}, b_t) = 0$.

Closed-loop execution. At round t with $b_t > 0$, the controller selects the action with the highest value from $\mathcal{A}(s_t)$, defined in Section 3,

$$a_t = \arg\max_{a \in \mathcal{A}(s_t)} Q_\omega(s_t, a, b_t). \tag{5}$$

The controller selects STOP when all retrieval candidates have non-positive predicted values, resolving ties in favor of stopping. A selected retrieval action has positive predicted net gain and searches the frozen passage index with its query vector x_{a_t}. This search returns the k_{ret} highest-scoring passages as O_t and consumes one corpus call, updating the state and budget according to

$$O_t = \operatorname*{Top}_{k_{\text{ret}},\, v \in \mathcal{C}} x_{a_t}^\top e_v, \qquad (s_{t+1}, b_{t+1}) = \big(\mathcal{T}(s_t, a_t, O_t),\, b_t - 1\big), \tag{6}$$

where e_v is the frozen embedding of passage v, k_{ret} counts passages returned per call, and K counts operators. The transition $\mathcal{T}$ incorporates the returned passages into memory and records the action and outcome. The updated evidence supplies pivots and context for constructing and evaluating candidates in the next round. Algorithm 1 summarizes this loop: ConstructActions builds compositional actions and their queries x_a (Section 4.2), while C_{direct} supplies the DENSE query. $\text{Retrieve}(x_a, k_{\text{ret}})$ performs the passage search in Equation 6. The loop returns the final retrieval state when the controller stops or the budget is exhausted.

Algorithm 1 Budgeted retrieval with compositional actions

Require: question q, initial memory M_0, budget B
$s \leftarrow (q, M_0, \emptyset)$; $b \leftarrow B - 1$
while $b > 0$ **do**
  $\mathcal{A}^{\text{comp}}(s) \leftarrow \text{ConstructActions}(s)$
  $\mathcal{A}(s) \leftarrow \{\text{STOP}, \text{DENSE}\} \cup \mathcal{A}^{\text{comp}}(s)$
  $a \leftarrow \arg\max_{a' \in \mathcal{A}(s)} Q_\omega(s, a', b)$
  if $a = \text{STOP}$ **then**
    break
  $O \leftarrow \text{Retrieve}(x_a, k_{\text{ret}})$
  $s \leftarrow \mathcal{T}(s, a, O)$; $b \leftarrow b - 1$
return s

4.4 OFFLINE TRAINING

Offline learning separates supervision for constructing a query from feedback on its retrieval value. We first learn action construction from corpus paths and training questions, then fit the controller to the outcomes of the resulting queries.

Query construction. We learn the relational language as described in Section 4.1. For proposal training, connecting entities identify training pivots, while training questions and support annotations identify compatible continuations. Their routed relation sequences supervise p_η, and all compatible sequences are retained as valid targets.

For query training, a question and an observed path prefix define s. A compatible continuation defines a and a later target passage d_a^+, and rolling out a gives z^ℓ. Given (s, a, z^ℓ), C_γ is trained with

$$\mathcal{L}_{\text{query}} = \mathcal{L}_{\text{text}} + \lambda_e \left[1 - \cos\big(\widehat{e}_a, E_D(d_a^+)\big)\right]. \tag{7}$$

Here, λ_e weights evidence alignment and is set to 0.1. The token loss $\mathcal{L}_{\text{text}}$ supervises target passage tokens with teacher forcing. The alignment term provides additional supervision for an auxiliary decoder prediction $\widehat{e}_a$ in the frozen passage embedding space. At inference, E_D encodes the generated text to form the retrieval query (Equation 4).

Retrieval value learning. We freeze action construction and query realization before training the controller. This keeps the mapping from each action to its query fixed while the controller learns from retrieval outcomes. To obtain these outcomes, we execute candidate retrieval actions on training questions and collect $\mathcal{D} = \{(s, a, b, r, s')\}$, where s' contains the returned passages. Training support annotations determine $r = U(s') - U(s) - \lambda$, the observed utility gain minus the call cost. This reward and the best predicted continuation under budget $b - 1$ define the target for fitting Q_ω,

$$\begin{aligned} y &= r + \max_{a' \in \mathcal{A}(s')} Q_{\bar{\omega}}(s', a', b - 1), \\ \mathcal{L}_Q &= \mathbb{E}_{\mathcal{D}}\left[\text{Huber}\left(Q_\omega(s, a, b), \text{stopgrad}(y)\right)\right]. \end{aligned} \tag{8}$$

The target function satisfies $Q_{\bar{\omega}}(s, a, 0) = Q_{\bar{\omega}}(s, \text{STOP}, b) = 0$. These boundaries give $y = r$ when $b = 1$, while larger budgets allow later retrieval value to contribute to the target. The target parameters $\bar{\omega}$ remain fixed within each fitting round and are refreshed from ω between rounds. Support annotations provide offline supervision, whereas online decisions use the question, retrieved evidence, action history, and remaining budget.

5 EXPERIMENTS

Table 2: **Benchmark comparison (%).** Three-run means; **bold**/underline indicate the best/second-best result for each metric. Lower calls are better.

	2WikiMultiHopQA			HotpotQA			MuSiQue		
Method	R@10	SC@10	F1	R@10	SC@10	F1	R@10	SC@10	F1
Dense	75.78	49.50	48.93	91.40	83.60	52.08	68.23	37.50	37.30
Hybrid	78.93	53.60	50.47	91.95	84.10	52.45	69.40	40.50	38.56
HyDE (2023)	81.92	58.97	50.88	92.47	85.03	52.76	70.58	42.77	39.14
MDR (2021)	85.57	66.60	51.63	92.87	85.90	53.67	72.58	44.63	40.18
EfficientRAG (2024)	86.33	68.03	52.35	93.22	86.60	53.26	73.76	46.90	**41.22**
QAFD (2026)	77.29	50.20	51.00	92.92	86.00	53.57	71.10	42.17	39.85
HippoRAG2 (2025)	87.07	68.53	49.51	93.42	87.00	51.50	72.64	45.07	36.85
IRCoT (2023)	85.71	65.43	52.97	94.43	89.03	54.62	71.79	43.57	37.83
LAnR (2026)	87.87	69.77	53.37	94.53	90.73	54.39	75.19	47.23	40.53
LevelRAG (2025)	87.43	68.73	53.09	94.27	90.07	54.07	74.71	46.63	40.27
RILO	**90.07**	**74.23**	**55.36**	**96.40**	**94.80**	**55.08**	**77.67**	**50.27**	41.01

Bamboogle frozen transfer	Hybrid	EfficientRAG (2024)	IRCoT (2023)	LAnR (2026)	LevelRAG (2025)	RILO
Answer F1 (%)	41.89	46.84	48.90	51.80	50.60	**53.24**
Corpus calls	**2.00**	2.43	2.35	2.21	2.56	2.19

5.1 EVALUATION SETUP

Benchmarks. We use 1,000 fixed questions each from 2WikiMultiHopQA (Ho et al., 2020), HotpotQA (Yang et al., 2018), and MuSiQue (Trivedi et al., 2022). The corresponding collections contain 6,119/9,811/11,656 passages. All learned modules use dataset-specific fitting and tuning splits. We evaluate the frozen 2Wiki-trained system on Bamboogle's 125 questions (Press et al., 2023), reporting F1 and corpus calls (Appendix A.4).

Baselines. Dense and Hybrid are implementation controls: dense retrieval (Karpukhin et al., 2020) and BM25–dense reciprocal rank fusion (Cormack et al., 2009); HyDE (Gao et al., 2023) tests query expansion. MDR (Xiong et al., 2021) performs iterative dense retrieval; EfficientRAG (Zhuang et al., 2024) updates queries through token selection, and IRCoT (Trivedi et al., 2023) retrieves with generated reasoning steps. HippoRAG2 (Gutiérrez et al., 2025) and QAFD (Zhou et al., 2026b) use graph structure. LAnR (Ha Lan N.T et al., 2026) directly produces latent queries with learned stopping; LevelRAG (Zhang et al., 2025b) plans atomic queries and rewrites them for its retrievers, here within a closed corpus.

Evaluation protocol. The controlled evaluation shares passages, Qwen3-Embedding-4B (Zhang et al., 2025a), and a non-thinking Qwen3-32B reader (Yang et al., 2025). LAnR retains its native retrieval encoder. Appendix E.3 details baseline adaptation and cost accounting. For graph construction, Qwen3-8B extracts entities and relation triples following OpenIE (Banko et al., 2007). Given this graph, RILO defaults to $K = 16$, $H = 3$, at most 12 proposals, and $B = 3$. R@10 measures support recall; SC@10 measures the fraction of questions with all supporting passages retrieved in the top ten. For answer F1, the top ten passages are concatenated as reader context, subject to the shared token limit. The main benchmark, matched controls, and independent-pipeline repeats are distinguished in Appendix E.1.

5.2 Evidence Recovery

Recovering complete evidence. Within three calls, RILO achieves 73.10% macro SC@10, 3.86 points above LAnR, the strongest baseline on this metric (Table 2). Macro answer F1 also improves by 1.05 points over LAnR. The completeness gain is consistent with directing searches toward missing relational links rather than further topical matches. On MuSiQue, F1 remains slightly below EfficientRAG despite higher coverage: retrieving the necessary evidence and extracting the answer remain distinct bottlenecks.

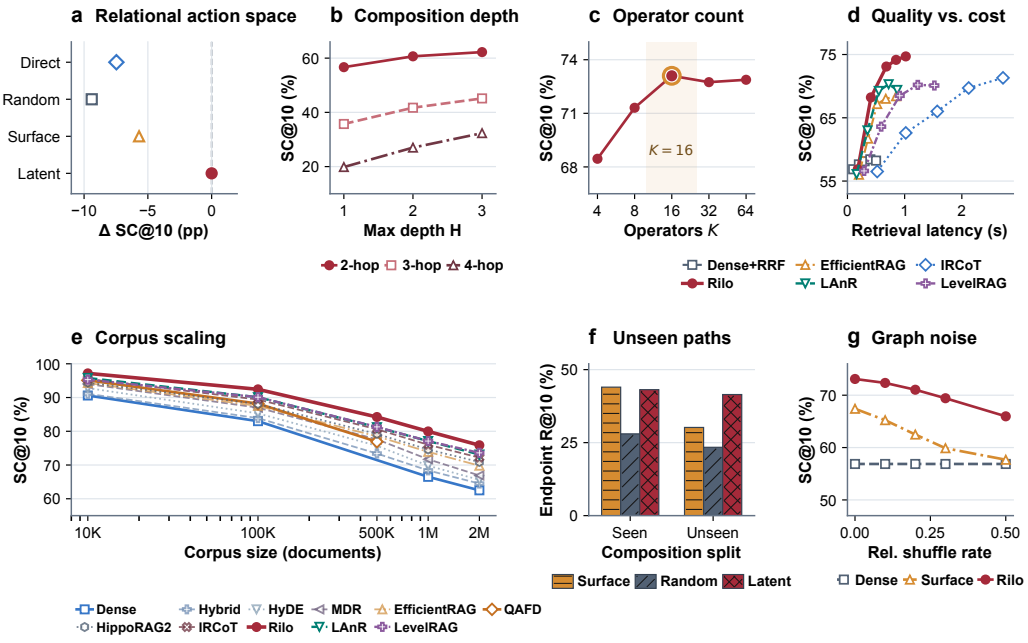

Figure 5: **Module contributions and operating limits.** Panels show (a) action representations, (b) composition depth, (c) operator count, (d) quality–latency trade-offs, (e) FullWiki scaling, (f) unseen paths, and (g) label corruption. Panel (b) uses 450/350/200 two-/three-/four-hop questions.

Role of operator composition. Figure 5a compares action representations; Direct excludes relational actions and is not Direct-continuous. Under matched short-query supervision, RILO reaches 73.71% macro SC@10, exceeding Direct-matched by 4.17 points and Direct-continuous by 1.84 points. The gain over Direct-continuous remains 1.50 points without learned value control. This pattern supports a contribution from structured query directions beyond learned stopping (Appendix D). Increasing H from one to three improves completeness at a fixed call budget (Figure 5b). Deeper compositions predict more distant evidence while still guiding a single search.

Ablating the retrieval loop. Shuffling endpoint–context correspondence reduces SC@10 by 9.26 points, the largest query ablation loss (Table 8). This loss indicates that endpoint–context alignment, rather than the endpoint feature alone, contributes to retrieval quality. Removing lookahead and replanning costs 5.73 and 2.57 points, respectively. Lookahead can retain searches with limited immediate gain but useful follow-up evidence; replanning updates this choice after observing actual retrieval results. Higher net utility under long-horizon control supports this interpretation (Appendix E.4.1).

5.3 Retrieval across Operating Conditions

Scaling retrieval. RILO averages 2.189 searches and 0.621 seconds of retrieval time, including candidate realization and STOP-time computation. At a three-call budget, it achieves higher SC@10 than LAnR and LevelRAG, but LAnR and EfficientRAG remain faster (Figure 5d). Expanding supplied-context pools to one million passages reduces macro SC@10 to 62.47 versus 60.91 for the continuous control (Appendix E.2). In the separate FullWiki evaluation, coverage declines with corpus size but remains above LAnR and LevelRAG at all tested scales (Figure 5e).

Generalization of learned operators. Shared operators improve endpoint R@10 on unseen paths by 11.27 points over surface relations but lose 0.87 points on seen paths (Figure 5f; 500 contexts per split, three repeats). This trade-off suggests that sharing pools evidence across relation phrases, improving reuse at some cost to surface-specific distinctions. The plateau beyond $K = 16$ favors a compact vocabulary (Figure 5c). Shuffling half the relation labels reduces SC@10 by 7.13 points versus 9.73 for surface relations (Figure 5g), consistent with reduced dependence on individual labels. Ordered and source-only controls further probe this reuse (Appendix E.4.3).

Cross-dataset transfer. Whole-system transfer yields lower support coverage than in-domain training (Table 3), indicating that reusable transitions still benefit from dataset-specific fitting. On Bamboogle, the frozen 2Wiki-trained system reaches 53.24% F1, exceeding LAnR and LevelRAG by 1.44 and 2.64 points, respectively (Table 2). The F1 gain accompanies fewer corpus calls: 2.19 versus 2.21 for LAnR and 2.56 for LevelRAG. This result supports transfer beyond the fitting dataset under the evaluated conditions. Coverage gains exceed F1 gains on the core benchmarks; paired reader analysis identifies context truncation and answer extraction as downstream bottlenecks (Appendix E.4.4).

Table 3: **Cross-dataset transfer.** R@10/SC@10 (%). Rows: whole-system training source; columns: evaluation dataset. Off-diagonal: frozen transfer.

Source	2Wiki	HotpotQA	MuSiQue
2Wiki	90.07/74.23	94.95/90.97	77.15/49.60
HotpotQA	88.35/69.83	96.40/94.80	75.88/48.20
MuSiQue	88.40/70.53	94.90/90.63	77.67/50.27

5.4 Case Analysis

Case evidence. Figure 6 asks which film's director was born earlier. The comparison requires two complete evidence chains: *Toto and Cleopatra* leads to Fernando Cerchio's birth date in 1914, whereas *Land of Peace* leads to Kamal El Sheikh's in 1919. The earlier date identifies *Toto and Cleopatra*. Both chains reuse the film–director–birth-date structure; o_0 and o_3 depict schematic steps. The archived trace retrieves all four supports in three calls, with the biographies ranked seventh and tenth. Completing both branches, rather than retrieving more film descriptions, makes the comparison answerable.

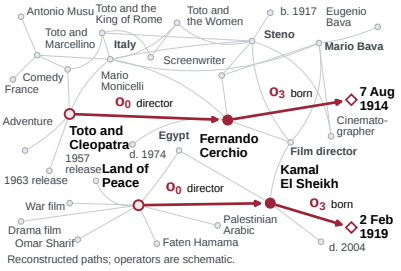

Figure 6: **Movie case.** Reconstructed support paths.

6 Discussion

Impact. RILO makes predicted evidence transitions explicit components of retrieval planning. Shared operators pool supervision across relation phrases and compose into query directions toward missing evidence. A single search realizes each multi-step plan within the available retrieval budget. The analyses connect this benefit to operator reuse and endpoint alignment with observed evidence. Budgeted replanning then adjusts evidence acquisition as retrieval results arrive.

Limitations. The operator vocabulary and proposal bounds are fixed after training; adaptive vocabularies and state-dependent search widths could accommodate changing corpora. Extracted-path noise also motivates confidence-weighted supervision, evaluated against both endpoint accuracy and planning cost. Dataset-specific corpora and adapted retrieval interfaces delimit the present evaluation; open-domain applicability and deployment latency require assessment under the intended corpus and hardware conditions.

7 Conclusion

We introduced RILO, a retrieval framework that learns a compact relational language and uses its compositions to guide evidence acquisition. Endpoint-conditioned queries connect predicted continuations to corpus search, while budgeted control coordinates retrieval and replanning as evidence accumulates. The evaluated benchmarks support the value of this connection for recovering multi-hop evidence. This framework provides a concrete basis for integrating learned relational structure with sequential retrieval decisions in resource-constrained RAG.

AI USE DISCLOSURE

We used AI-assisted tools to support manuscript writing and language polishing. The authors take full responsibility for the final content.

REPRODUCIBILITY STATEMENT

The RILO implementation, training configurations, and evaluation scripts are available at `https://anonymous.4open.science/status/RILO-6E20`. The appendices detail graph construction, path-derived supervision, model training, and evaluation protocols.

REFERENCES

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

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2024.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pp. 2670–2676, 2007. URL `https://www.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 *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 758–759, 2009.

Justus-Jonas Erker, Nils Reimers, and Iryna Gurevych. GRITHopper: Decomposition-free multi-hop dense retrieval. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 77–94, 2026.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pp. 1762–1777, 2023.

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.

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. PMLR, 2025. URL `https://proceedings.mlr.press/v267/gutierrez25a.html`.

Ha Lan N.T, Minh-Anh Nguyen, and Dung D. Le. Latent abstraction for retrieval-augmented generation. *arXiv preprint arXiv:2604.17866*, 2026.

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.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 7036–7050, 2024.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7969–7992, 2023.

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*, pp. 6769–6781, 2020.

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

Jiaen Lin, Jingyu Liu, and Yingbo Liu. Optimizing multi-hop document retrieval through intermediate representations. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 15798–15809, 2025. doi: 10.18653/v1/2025.findings-acl.816. URL https://aclanthology.org/2025.findings-acl.816/.

Hao Liu, Zhengren Wang, Xi Chen, Zhiyu Li, Feiyu Xiong, Qinhan Yu, and Wentao Zhang. HopRAG: Multi-hop reasoning for logic-aware retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 1897–1913, 2025.

Costas Mavromatis and George Karypis. GNN-RAG: Graph neural retrieval for efficient large language model reasoning on knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 16682–16699, 2025.

Tengjun Ni, Xin Yuan, Shenghong Li, Kai Wu, Ren Ping Liu, Wei Ni, and Wenjie Zhang. StepChain GraphRAG: Reasoning over knowledge graphs for multi-hop question answering. *arXiv preprint arXiv:2510.02827*, 2025.

Jaewan Park, Solbee Cho, and Jay-Yoon Lee. Stop-RAG: Value-based retrieval control for iterative RAG. *arXiv preprint arXiv:2510.14337*, 2025.

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. URL https://aclanthology.org/2023.findings-emnlp.378/.

Zhili Shen, Chenxin Diao, Pavlos Vougiouklis, Pascual Merita, Shriram Piramanayagam, Enting Chen, Damien Graux, Andre Melo, Ruofei Lai, Zeren Jiang, Zhongyang Li, Ye Qi, Yang Ren, Dandan Tu, and Jeff Z. Pan. GeAR: Graph-enhanced agent for retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 12049–12072, 2025.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. DRAGIN: Dynamic retrieval augmented generation based on the real-time information needs of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 12991–13013, 2024.

Xiaqiang Tang, Qiang Gao, Jian Li, Nan Du, Qi Li, and Sihong Xie. MBA-RAG: A bandit approach for adaptive retrieval-augmented generation through question complexity. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 3248–3254, 2025.

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.

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*, pp. 10014–10037, 2023.

Liang Wang, Haonan Chen, Nan Yang, Xiaolong Huang, Zhicheng Dou, and Furu Wei. Chain-of-retrieval augmented generation. In *Advances in Neural Information Processing Systems*, 2025.

Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oğuz. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations*, 2021.

Ganlin Xu, Linghao Zhang, Zhitao Yin, Hongda Xi, Chen Yang, Jiaqing Liang, Weijia Lu, Sihang Jiang, Yanghua Xiao, and Deqing Yang. When rag meets query planning: Logical query trees for resolving exploratory reasoning problems. *arXiv preprint arXiv:2607.00508*, 2026.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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*, pp. 2369–2380, 2018.

Jiahao Zhang, Haiyang Zhang, Dongmei Zhang, Yong Liu, and Shen Huang. End-to-end beam retrieval for multi-hop question answering. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1718–1731, 2024.

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

Zhuocheng Zhang, Yang Feng, and Min Zhang. LevelRAG: Enhancing retrieval-augmented generation with multi-hop logic planning over rewriting augmented searchers. *arXiv preprint arXiv:2502.18139*, 2025b. URL https://arxiv.org/abs/2502.18139.

Chenyu Zhou, Yaqi Wu, Xiaolei Guo, Jiaqi Huang, Xianfa Zhang, Junxu Zhang, Zhuo Yu, Zhubo Shi, Jianghao Lin, and Dongdong Ge. DynaKRAG: A unified framework for learnable evidence control in multi-hop retrieval-augmented generation. *arXiv preprint arXiv:2607.06507*, 2026a.

Zhuoping Zhou, Davoud Ataee Tarzanagh, Sima Didari, Wenjun Hu, Baruch Gutow, Oxana Verkholyak, Masoud Faraki, Heng Hao, Hankyu Moon, and Seungjai Min. Query-aware flow diffusion for graph-based RAG with retrieval guarantees. In *International Conference on Learning Representations*, 2026b.

Ziyuan Zhuang, Zhiyang Zhang, Sitao Cheng, Fangkai Yang, Jia Liu, Shujian Huang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. EfficientRAG: Efficient retriever for multi-hop question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3392–3411, 2024.

A Implementation Details

Dataset setup. The three core datasets provide support annotations and shared, frozen retrieval collections with fixed passage boundaries. Evaluation IDs were fixed before scoring and kept separate from tuning questions.

Evaluation metrics. For question q, let S_q be the set of annotated support passages and R_q^k the retrieved top-k set. Define

$$\mathrm{R@}k(q) = \frac{|S_q \cap R_q^k|}{|S_q|}, \qquad \mathrm{SC@}k(q) = \mathbf{1}[S_q \subseteq R_q^k]. \tag{9}$$

Scores are averaged over questions within each dataset; the macro average gives each core dataset equal weight. Passage IDs are deduplicated before scoring, and answer F1 uses dataset-appropriate normalization. The reader context concatenates the top ten retrieved passages under the shared 4,096-token limit.

Result aggregation. Core results average three runs on the same 1,000 questions per dataset; paired intervals appear in Table 7 and Appendix B.1. The path experiment uses 500 contexts per split. MuSiQue's two-, three-, and four-hop groups contain 450, 350, and 200 questions; their weighted scores recover the main $H = 3$ result. FullWiki Dense/QAFD results use one run on 1,000 questions.

A.1 Graph Construction

Passage representation. Each benchmark record remains one passage, serialized as title, newline, and passage text. Its ID is `chunk-` plus the MD5 digest of this UTF-8 serialization; original ID mappings are preserved. Collections deduplicate serialized passages and retain records without valid triples. Operator learning uses the in-domain corpus or, for transfer, only the source corpus.

Triple extraction. Locally hosted Qwen3-8B first extracts named entities, then receives these entities and the passage to produce subject–relation–object triples. Both stages use temperature zero, no thinking, and at most 2,048 output tokens. Triples retain source passage IDs and undergo structural validation and normalization, without a confidence cutoff. Unparseable responses yield no triples.

Entity matching. Entity and relation matching keys are lowercased, characters outside ASCII letters, digits, and spaces are replaced by spaces, and leading and trailing spaces are removed; internal whitespace is not collapsed, and Unicode NFKC normalization is not applied. Identical normalized entity strings receive the same node ID. Normalized relation keys support duplicate detection, while the original relation phrases, stripped of surrounding whitespace, supply the labels for operator training. The router F_ϕ learns the operator assignment directly from these phrases. Entity association is string-based and independent of external knowledge-base identifiers, so homonymous mentions can be conflated.

Qwen3-Embedding-4B encodes entity strings for auxiliary alias candidate generation. Candidate pairs are found among the top 2,047 cosine neighbors and require similarity at least 0.8; self-pairs are excluded, and source strings with at most two alphanumeric characters are skipped. Alias candidates are stored only in the auxiliary index. Directed supervision paths join successive triples through exact normalized identity.

Supervision graph. Two representations serve different purposes. The auxiliary passage–entity index combines symmetrized fact connections, passage incidence, and similarity links. Operator supervision instead uses the extracted triples, retaining their subject-to-object direction, relation phrase, and source passage. Formally, this supervision graph is $\mathcal{G} = (\mathcal{U}, \mathcal{E}, \mathcal{R})$, where $\mathcal{U}$ contains entities, $\mathcal{R}$ contains relation labels, and $\mathcal{E} \subseteq \mathcal{U} \times \mathcal{R} \times \mathcal{U}$ contains directed edges. Malformed triples and empty normalized fields are excluded from this relation view. No inverse relation is inferred from an undirected index connection. Passage–entity incidence defines $\rho(d)$ from the subject and object mentions in valid triples. Let $\mathcal{P}(u, r, v)$ denote the set of passages containing a given directed triple; this groups its witnesses without treating repeated mentions as new relation types. Generic nonempty predicates such as "is" are retained. Although self-loops are removed from the auxiliary index, the relation-path enumerator does not impose a simple-path constraint: entities or witness passages may recur.

A.2 PATH-DERIVED OPERATOR SUPERVISION

Operator learning uses frozen corpus-derived path examples with the following enumeration and training settings.

Directed path enumeration. Paths start with a source-passage triple and join normalized objects to successive subjects, preserving relation order, with $L \in \{1, 2, 3\}$. In stored triple order, enumeration inspects 32 source triples for one-step paths, retains at most 64 paths each at lengths two and three, and inspects at most 16 and eight outgoing triples at the first and second bridges, respectively. Within each source, deduplicate by length, normalized start/end entities, and ordered normalized relation keys, collapsing different witnesses or intermediate entities with the same key.

Source–target examples. For a path $\pi = u_0 \xrightarrow{r_1} \cdots \xrightarrow{r_L} u_L$ anchored at d_{src}, targets are drawn from the passage association

$$\mathcal{D}(u_L) = \{d \in \mathcal{C} : u_L \in \rho(d)\}, \qquad x = (d_{\mathrm{src}}, r_{1:L}, d^+), \quad d^+ \in \mathcal{D}(u_L).$$

Retain at most 16 targets per branch in passage-ID order as separate examples, excluding $d^+ = d_{\mathrm{src}}$. Targets need only contain the terminal entity in an extracted triple, not witness the final edge or provide annotated support. This weak supervision does not require $L + 1$ distinct passages.

Operator validation split. Examples are grouped by source passage, path length, and a structural descriptor containing the source title, length, and outgoing relation paths. Groups are shuffled with seed 42 and 10% are reserved for validation, keeping the examples of a group together. This group-level operator split permits a source passage to occur in different groups at different lengths, and entities and relation labels can occur on both sides. Corpus-derived vocabularies, embeddings, and frequency statistics are constructed before this split. Validation cases combine targets for the same group and relation sequence, exclude the source passage, and distinguish seen sequences, unseen combinations of seen atomic relations, and sequences containing unseen atomic relations. Question-conditioned training and final QA evaluation use the question-level split controls in Appendix A.4.

Training sample weights. Target frequency is the number of retained branch-to-target associations. Examples are assigned to logarithmic target-frequency buckets with base two. The weighting mixes the empirical bucket distribution with a uniform distribution at equal weight, then clips the resulting importance ratios to $[0.1, 10]$. After excluding source-equals-target examples and paths beyond the training horizon, at most 80,000 examples are selected. If this cap binds, selection is without replacement with probabilities proportional to these weights, using data seed 20260710. Otherwise all eligible examples are used. The selected examples are shuffled each epoch; their weights also multiply the retrieval loss after normalization to unit mean within each minibatch.

Recurrent operator implementation. Ordered surface-relation sequences are decomposed into atomic relation labels. An atomic relation uses its frozen single-relation embedding when available; otherwise it uses the normalized mean of available path embeddings containing that relation. If neither is available, a deterministic seeded vector is used. The continuous teacher projects relation embeddings into its transition space. The student routes each relation to one of $K = 16$ learned codes using a hard assignment with a straight-through gradient. Both models encode the source passage into a 256-dimensional hidden state, apply a GRU transition for each relation or code in sequence, and decode a gated residual from the final state. Adding this residual to the source embedding and normalizing gives the predicted target embedding. Intermediate hidden states are recurrent predictions, not teacher-forced embeddings of intervening passages.

Optimization schedule. The teacher is trained first with endpoint retrieval supervision and frozen before student training. Appendix A.3.1 defines the student objective and loss weights. Each stage uses five epochs, batches of 512, AdamW with learning rate 3×10^{-4} and weight decay 10^{-3}, gradient clipping at norm one, and router temperature annealed from 1.0 to 0.2.

Negative sampling. Each minibatch draws 126 passage IDs uniformly with replacement from the frozen corpus and shares this draw across its examples. For each example, the contrastive candidates comprise its positive target, its source passage, and these random passages. Cosine scores enter a cross-entropy loss with temperature 0.07 and the positive target as the correct class. Random entries equal to that example's target ID are masked. Repeated IDs retain their multiplicity; other valid targets can remain among the negatives.

A.3 Training Objectives

A.3.1 Operator Learning Objective

Endpoint notation. For a minibatch $\mathcal{B}$ of m examples from Appendix A.2, write $x_i = (d_i^{\mathrm{src}}, r_{i,1:L_i}, d_i^+)$ and $e(d) = E_D(d)/\|E_D(d)\|$. Let $\widehat{e}_i$ and $\widetilde{e}_i$ be the unit-normalized student and continuous-teacher predictions for the same source and complete relation sequence. Both predictions decode the final recurrent state as a gated residual added to the source embedding. The passage and relation encoders remain frozen throughout operator learning.

Contrastive retrieval. Let $c_{i0} = d_i^+$, $c_{i1} = d_i^{\mathrm{src}}$, and $c_{i,j+1} = n_j$ for the $M = 126$ batch-shared negative draws. Candidates form an ordered list, so repeated negative IDs retain their multiplicity. Define scores and per-example loss by

$$s_{ij} = \begin{cases} -10^4, & j \geq 2 \text{ and } c_{ij} = d_i^+, \\ \widehat{e}_i^\top e(c_{ij}), & \text{otherwise}, \end{cases}$$
$$\ell_{\mathrm{ret},i} = -\log \frac{\exp(s_{i0}/\tau_{\mathrm{ret}})}{\sum_{j=0}^{M+1} \exp(s_{ij}/\tau_{\mathrm{ret}})}, \qquad \tau_{\mathrm{ret}} = 0.07. \tag{10}$$

For target-frequency weights w_i defined in Appendix A.2, the batch objective is

$$\widetilde{w}_i = \frac{w_i}{m^{-1}\sum_{j=1}^m w_j}, \qquad \mathcal{L}_{\mathrm{ret}} = \frac{1}{m}\sum_{i=1}^m \widetilde{w}_i \ell_{\mathrm{ret},i}. \tag{11}$$

The continuous teacher minimizes this loss with its own endpoint prediction. It uses surface-relation embeddings and has no discrete-routing penalties.

Endpoint distillation. After teacher training, the student's additional matching loss is

$$\mathcal{L}_{\mathrm{dist}} = \frac{1}{m}\sum_{i=1}^m \left[1 - \widehat{e}_i^\top \operatorname{stopgrad}(\widetilde{e}_i)\right]. \tag{12}$$

Distillation uses an unweighted mean over final-endpoint predictions at all path lengths.

Routing regularization. Let p_v be the router distribution for valid occurrence $v = (i, t)$, obtained by applying softmax to $F_\phi(E_r(r_{i,t}))/\tau_{\mathrm{route}}$. Writing $\mathcal{V} = \{(i,t) : 1 \leq t \leq L_i\}$, define

$$\bar{p}_k = \frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}} p_{vk},$$
$$\mathcal{L}_{\mathrm{bal}} = \sum_{k=1}^K \bar{p}_k \log(K\bar{p}_k) = D_{\mathrm{KL}}(\bar{p} \,\|\, \mathrm{Unif}(K)), \tag{13}$$
$$\mathcal{L}_{\mathrm{ent}} = -\frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}\sum_{k=1}^K p_{vk}\log p_{vk}.$$

These averages count relation occurrences, not unique labels, and exclude padding; longer examples contribute more routing positions. Minimizing $\mathcal{L}_{\mathrm{bal}}$ promotes aggregate code usage, whereas minimizing $\mathcal{L}_{\mathrm{ent}}$ sharpens individual assignments. Probabilities are clamped below at 10^{-8} when evaluating these penalties. The forward assignment is one-hot, with surrogate gradient $p_v^{\mathrm{ST}} = \mathrm{onehot}(\arg\max_k p_{vk}) + p_v - \operatorname{stopgrad}(p_v)$. Retrieval and distillation therefore update both the transition model and router through the selected learned codes.

Equation (2) combines these four terms with weights $\lambda_{\mathrm{dist}} = 0.5$, $\lambda_{\mathrm{bal}} = 0.03$, and $\lambda_{\mathrm{ent}} = 0.002$. Only retrieval receives sample weights; neither distillation nor routing penalties do. The router temperature decreases linearly from 1.0 to 0.2 across epochs, independently of τ_{ret}. These values are fixed throughout operator training.

A.3.2 Query-Realization Loss

Text supervision. For the query objective in Equation (7), denote the conditioning input by $c_i = (s_i, a_i, z_i^\ell)$ and the target passage tokens by $y_{i,1:T_i}$. The length-normalized teacher-forced negative

log-likelihood is

$$\ell_{\text{text},i} = -\frac{1}{T_i^{\text{valid}}} \sum_{t=1}^{T_i} m_{it} \log P_\gamma(y_{it} \mid y_{i,<t}, c_i), \qquad T_i^{\text{valid}} = \sum_{t=1}^{T_i} m_{it}. \tag{14}$$

Here, m_{it} marks supervised target tokens and excludes padding and input prompt tokens. Teacher forcing supplies the gold target prefix to the decoder. Batch reduction uses the example mean $\mathcal{L}_{\text{text}} = m^{-1} \sum_i \ell_{\text{text},i}$.

Evidence alignment. Let $\widehat{e}_i$ denote the auxiliary vector predicted from the query decoder state, distinct from the operator endpoint used above. The second term in Equation (7) is the batch average

$$\mathcal{L}_{\text{align}} = \frac{1}{m} \sum_{i=1}^{m} \left[1 - \frac{\widehat{e}_i^\top E_D(d_i^+)}{\|\widehat{e}_i\| \, \|E_D(d_i^+)\|} \right], \qquad \mathcal{L}_{\text{query}} = \mathcal{L}_{\text{text}} + \lambda_e \mathcal{L}_{\text{align}}. \tag{15}$$

Alignment updates the prediction head and decoder, not the frozen passage encoder; corpus search remains outside the computation graph.

A.3.3 Budget-Conditioned Value Loss

Value targets. For a recorded retrieval transition i, set $r_i = U(s_i') - U(s_i) - \lambda$. The target in Equation (8) is equivalently

$$y_i = \begin{cases} r_i, & b_i = 1, \\ r_i + \max\big(0, \max_{a' \in \mathcal{A}(s_i') \setminus \{\text{STOP}\}} Q_{\bar{\omega}}(s_i', a', b_i - 1)\big), & b_i > 1. \end{cases} \tag{16}$$

STOP has target zero; budget-zero states admit no retrieval. The target is undiscounted; each search consumes one budget unit, regardless of composition length.

Value fitting. Let $\Delta_i = Q_\omega(s_i, a_i, b_i) - \text{stopgrad}(y_i)$. For threshold $\kappa = 1$, the Huber loss is

$$\mathcal{L}_Q = \frac{1}{m} \sum_{i=1}^{m} h_\kappa(\Delta_i), \qquad h_\kappa(\Delta) = \begin{cases} \frac{1}{2}\Delta^2, & |\Delta| \le \kappa, \\ \kappa(|\Delta| - \frac{1}{2}\kappa), & |\Delta| > \kappa. \end{cases} \tag{17}$$

Targets are detached; $\bar{\omega}$ is fixed within each fitting round and refreshed from ω between rounds. Only the value network is optimized.

A.4 Supervision Protocol

Question-level splits. Question data are separated into $\mathcal{Q}_{\text{fit}}$, $\mathcal{Q}_{\text{tune}}$, and $\mathcal{Q}_{\text{eval}}$. Evaluation IDs are fixed, while fitting and tuning questions come from the corresponding training split. Questions are grouped by NFKC-normalized, case-folded text with collapsed whitespace; a group cannot cross splits. Any training group matching a dataset-qualified evaluation ID or normalized question is excluded. Remaining groups are shuffled with seed 42 and split 90/10 for fitting/tuning. This rule is separate from the corpus-path grouping in Appendix A.2.

Module-specific supervision. Corpus-only extraction precedes operator learning, proposal/query training, freezing action construction, real transition collection, and controller fitting. Fitting questions and support labels select compatible pivots, sequences, and query targets. Label-derived rewards are stored separately from controller inputs (Table 4).

Evaluation regimes.

Dataset-specific in-domain training. Each main-table system uses its dataset's corpus and fitting questions; checkpoint and hyperparameter selection use that dataset's tuning split.

Source-only frozen transfer. The operator model, proposal, query generator, and controller are fitted and selected on the source only, then transferred together. All settings, including operator count, depth, beam/candidate limits, merging, prompts, decoding, utility weights, and stopping, are frozen. Target questions enter only at inference; target text is indexed with the frozen encoder. Target labels and metrics cannot inform fitting or selection. Bamboogle uses the 2Wiki-trained system and reports F1 and calls because support annotations are unavailable.

Reading the whole-system transfer matrix. Table 3 uses source rows and target columns: diagonal cells are in-domain, off-diagonal cells are frozen transfer.

Table 4: **Signal access by training stage.** "Targets" means offline supervision, not model input; "Tune" means checkpoint/hyperparameter selection without gradient updates.

Signal	Operator	Proposal / query	Controller	Online
Corpus text	Yes	Yes	Retrieved	Retrieved
Corpus triples / paths	Targets	Targets	No direct input	No
Fitting questions	No	Yes	Yes	Current question
Fitting support labels	No	Targets	Rewards only	No
Tuning support labels	No	Tune	Tune	No
Evaluation support labels	No	No	No	No
Answer labels	No	No	No	No
Executed retrieval results	No	Fitting traces	Transitions	After execution
Unexecuted future results	No	No	No	No

Controller features. Controller inputs comprise the question, observed passage history, candidate representation, remaining budget, and action history. Rewards, terminal utilities, support IDs, and oracle rankings are held in the target store used during offline fitting.

A.5 Controller Reward

Support recovery defines the utility used to label controller transitions.

Evidence utility. Annotated supports are mapped to canonical passage IDs using the retrieval evaluator, then both supports and ranked memory are deduplicated. $R_{10}(s)$ takes the first ten unique IDs in memory order, or all IDs if fewer than ten are available. The ranking is produced by the fixed retrieval and memory-update procedure. Questions with an empty or unmappable support set are excluded from support-based controller training. Datasets without support annotations, including the answering-only Bamboogle setting, supply no utility-training labels.

Let S_q be the nonempty set of gold support passage IDs. We define

$$U(s) = U_\beta(s) = \frac{C_{10}(s) + \beta I_{10}(s)}{1+\beta}, \qquad C_{10}(s) = \frac{|S_q \cap R_{10}(s)|}{|S_q|}, \qquad I_{10}(s) = \mathbf{1}[S_q \subseteq R_{10}(s)]. \tag{18}$$

Both metrics are fractional, so $0 \le U_\beta(s) \le 1$ for $\beta \ge 0$. Their question averages recover R@10 and SC@10 before reader-context truncation.

Transition reward. For an executed search, the reward is

$$r_t = U_\beta(s_{t+1}) - U_\beta(s_t) - \lambda, \qquad \sum_{t=0}^{\tau-1} r_t = U_\beta(s_\tau) - U_\beta(s_0) - \lambda\tau. \tag{19}$$

Initialization precedes s_0 and consumes one shared call, leaving $b_0 = B - 1$. STOP costs nothing and earns zero; the final search still earns its utility change minus cost (Appendix A.3.3). Differences are not clipped: unchanged support incurs $-\lambda$, and displacement can reduce reward further. Experiments and frozen transfer fix $k = 10$, $\beta = 1$, and $\lambda = 0.02$. This bounded utility satisfies Appendix C's reward condition.

A.6 Retrieval Cost Accounting

Search accounting. Charge every scored index search, including initialization and each search within a native operation. Hybrid's lexical/vector searches count separately, even in one API request; its main-table row uses two-search fusion. Operator rollout, graph propagation, candidate scoring, and generation incur latency/compute costs but no search count.

Timing protocol. Measure batch-one wall time on identical hardware with models/indexes loaded and accelerators synchronized around each request. Average over fixed questions, including encoding, planning, query realization, search, graph work, and reranking, but excluding reader generation.

A.7 Controlled Comparison Protocol

A.7.1 Baseline Evaluation Settings

Comparison identity. The controlled track shares passage IDs, the index, retrieval output limits, reader, and evaluation questions. The native track retains each method's trained query–document encoder pair and reports its own resource cost. Retrievers adapted to a replacement document encoder require a compatible query mapping. Appendix E.3 specifies the MDR-style adaptation and other baseline interfaces.

Table 5: **Baseline comparison settings.** Budgets count index searches including initialization. "Tune" permits only the declared fitting/tuning splits. Generation and graph computation are timed separately from search counts.

Method	Search cap	Permitted adaptation	Candidate handling / stopping
Dense	1	Tune index settings	One ranked list; fixed stop
Hybrid	2	Tune fusion	One lexical plus one dense list; RRF
HyDE	3	Tune prompt	Generated hypothesis; fixed search schedule
MDR	3	Fit compatible retriever	Native chain scoring; budget truncation
EfficientRAG	3	Fit native modules	Native update/stop; search cap
QAFD	3	Tune graph parameters	Native diffusion; count every index search
HippoRAG2	3	Tune graph parameters	Native graph ranking; search cap
IRCoT	3	Tune demonstrations	Reasoning-conditioned queries; native stop
RILO	3	Fit declared modules	At most 12 plans plus DENSE; learned STOP

For core in-domain comparisons, each dataset's training labels may be used only by the explicitly fitted baseline modules or tuning procedure. Source-only comparisons freeze all choices using the corresponding source fitting/tuning data (2Wiki for Bamboogle). Methods with no native supervised fitting stage retain their training-free design; the Direct controls provide supervision-matched attribution. The controlled retrieval interface returns at most 20 passages per index search. Any native reranker or graph stage with a larger candidate pool reports its actual pool size.

Cost ledger. Log per-question searches, generation calls/tokens, returned/reranked candidates, and latency (mean, median, p95), including unexecuted candidates and pre-STOP work. Appendices E.3 and E.6 give reader/offline costs and hardware. Collection permits 24,000 additional searches per method/dataset, or 24,900 including fitting-question initialization.

A.7.2 Executable Method Specification

Pivot-conditioned rollout. Select at most five pivots from extracted question spans followed by memory entities, resolving equal ranks by canonical passage ID and span offset. Use the highest-ranked observed passage containing the pivot as its context; when absent, use the question plus pivot text without another index search. Use the same frozen entity extractor and normalization as graph construction; empty normalized mentions are skipped. Preserve surface text and source occurrences to distinguish homonymous mentions. The context limit is 512 encoder tokens. With normalized context embedding e_0, the recurrent state and endpoint readout are distinct:

$$h_0 = \mathrm{Init}_\theta(e_0), \quad h_i = \mathrm{GRU}_\theta(h_{i-1}, o_i), \quad z_i = \mathrm{Normalize}(e_0 + \mathrm{Residual}_\theta(h_i, e_0)). \tag{20}$$

The hidden width is 256; endpoints use the frozen passage encoder's output dimension. Exact residual/gate tensors and initialization are exported with the model configuration. Online initialization uses observed context; the question-only fallback differs from passage-context training.

Candidate construction. Use $K = 16$, $H = 3$, beam width four per pivot, and at most 12 final plans. Expand each retained prefix by every code, retain four per pivot/depth, then rank all depths using the cumulative log-probability in the main method (length-normalization exponent zero). Break ties by pivot order then code sequence. Greedily merge within a pivot at endpoint cosine similarity 0.98, retaining the higher-scored plan; do not merge across pivots. If no plan survives, retain DENSE and STOP. Log selected depths and candidate counts. For multiple valid training continuations, the proposal objective averages teacher-forced token cross-entropy over valid routed sequences, then averages over training states.

Query interface. The 96-token configuration is specified below. The short-query configuration uses a 32-token inference cap and a 96-token training-target limit for all matched methods (Appendix E.3).

The query realizer uses `google/t5-efficient-tiny`, with hidden width 256, four encoder and four decoder layers, FFN width 1,024, and four attention heads. The realizer has 18.2M parameters, including endpoint conditioning, with tied embeddings counted once. It is fully fine-tuned, without LoRA, and frozen for online retrieval. Use 512 input tokens, 96 target/output tokens, and deterministic two-beam decoding with sampling disabled. Text decoding is distinct from beam search over relational plans. Layer normalization and a bias-free projection map the 2,560-dimensional endpoint to four soft-prefix tokens; a learned sigmoid gate also adds their mean to the text embeddings. Provide latent action IDs and pivot/context text to RILO, and the correspondingly budgeted continuous state to the continuous control. Direct-matched receives neither codes nor an endpoint input, but retains the same target-text and auxiliary alignment supervision. Query losses update the generator and projection heads, with frozen passage encoder and frozen operator construction at this stage. Token prediction and auxiliary alignment do not differentiate through discrete retrieval.

Memory interface. Deduplicate each returned list by canonical passage ID before assigning ranks. Fuse all executed lists using RRF with constant 60; break equal fused scores by passage ID. Retain all returned IDs and rank contributions (at most 60 distinct passages for three 20-passage searches), exposing the first ten to retrieval scoring and the reader. Concatenate in rank order under the 4,096-token reader limit and log actual visible spans. Recompute ranks after each search: top-ten support need not increase monotonically. Use exactly this fusion rule across the three matched variants, including the fixed-budget diagnostic; select the highest proposal-scored candidate there, with deterministic ties, and no learned value selection.

A.7.3 TRANSITION SAMPLING

Collection policy. After the common initial retrieval, collect real executions from fitting questions only. Allocate the 24,000-search allowance between remaining budgets two and one. At each selected state, prioritize half of the execution slots by proposal rank and sample the rest uniformly without replacement from remaining actions, rounding the priority share down. Subsample the next-state frontier to remain within the budget, and record selection probabilities and seeds. Unexecuted actions have no observed transition label. STOP has analytic zero continuation value. Fit the controller only after collection; record any subsequent data refresh as a separate round.

Table 6: **Transition collection counts.** One method/dataset; three seeds reuse the collection unless separately specified. Eligible actions exclude STOP and include DENSE.

Remaining budget	States	Eligible	Executed	Coverage (%)
2	900	11,700	8,100	69.23
1	2,100	27,300	15,900	58.24
Total	3,000	39,000	24,000	61.54

Coverage is executed state–action pairs divided by eligible pairs, using 13 eligible actions per selected state. The actual audit additionally groups counts by DENSE/compositional type, depth and proposal-rank band, and reports unseen action signatures on evaluation traces. A 100-question source-tuning diagnostic expands the retained two-step tree after initial retrieval, requiring at most $100(13 + 13^2) = 18{,}200$ additional searches. This source-tuning diagnostic measures calibration, premature STOP, and regret relative to the retained-tree oracle.

A.7.4 STATISTICAL PROTOCOL

Repeats. Controller-seed repeats use seeds 11, 23 and 37 with fixed action construction, collection, index, deterministic candidate-conditioning rules and two-beam query decoding. For each method, retain dataset-qualified question ID, run/seed, metric numerator and denominator, returned/visible passage IDs, search and token counts, and latency. Compute run SD with denominator $S - 1$ from complete-dataset run means.

Paired question uncertainty. Paired question-bootstrap resampling keeps all runs of each question together and uses identical draws for both methods, weighting datasets equally. Predeclare macro SC@10 and comparisons against EfficientRAG, IRCoT, Direct-matched and Direct-continuous; supplement intervals with paired randomization tests and Holm adjustment across these four comparisons. Shared-source/template cluster resampling is a sensitivity analysis when question dependence exists.

Table 7: **Paired comparisons.** Macro SC points. Point differences are computed from the main table; intervals quantify paired question-sampling uncertainty. Matched-control results appear in the attribution tables.

Comparison	Main-table difference	95% interval
RILO minus EfficientRAG	+5.92	[4.70, 7.13]
RILO minus IRCoT	+7.09	[5.82, 8.37]

Matched pretraining. Appendix B.1 specifies shared query supervision, candidate budgets, and independently fitted controllers for the Direct controls.

The pretraining-matched continuous variant uses the same selected corpus paths (at most 80,000), five epochs, batch size 512 and learning rate 3×10^{-4} as the operator stage, with identical endpoint retrieval targets and continuous transition width 256, without discrete routing penalties. Its question-conditioned stage uses the same 12,000 examples and 3,000 updates as the other matched variants.

B ATTRIBUTION DIAGNOSTICS

Table 8: **Component ablations.** Macro quality (%) over three datasets, $B = 3$.

Variant / factor changed	**SC@10**	**Ans. F1**	**Calls**
Surface relations (no latent codes)	67.41	48.27	2.18
Random relation-to-code routing	63.70	46.03	2.34
One-step operators ($H = 1$)	68.52	47.42	2.24
Endpoint query (no text realizer)	69.27	47.89	2.29
Text query (no endpoint input)	70.01	48.31	2.25
Endpoint–context shuffle	63.84	45.84	2.32
Immediate gain (no lookahead)	67.37	46.55	2.62
Open-loop plan (no replanning)	70.53	48.85	2.23
Fixed stopping (all B calls)	72.66	49.73	3.00
RILO (full model)	73.10	50.48	2.19

Diagnostic controls. The Direct comparison retains the state input, query-generation backbone, candidate cap, and controller architecture while replacing all compositional candidates with state-conditioned queries. Appendix B.1 specifies supervision and candidate diversity. The endpoint–context shuffle uses a fixed seeded permutation within each split. Seen and unseen path evaluations hold the encoder, training paths, negatives, and candidate width fixed. Controller comparisons share one candidate tree, with oracle labels confined to offline evaluation. The composition test uses a separate set of 1,000 held-out path contexts.

B.1 SUPERVISION-MATCHED ATTRIBUTION PROTOCOL

96-token matched-control configuration. This subsection reports the 96-token protocol and its uncertainty estimates. The short-query protocol and its supervision settings appear in Appendix D.

Table 9: 96-token configuration with adaptive control, $B \leq 3$. Macro quality (%), mean calls and latency (s). DM/DC: Direct-matched/Direct-continuous.

Method	R@10	SC@10	F1	Calls	Time (s)	Δ 95% CI
DM	84.57	69.54	49.60	2.16	0.53	+2.41 [1.94, 2.86]
DC	85.93	71.87	50.06	2.19	0.59	+0.09 [-0.34, 0.51]
RILO	86.07	71.96	50.11	2.17	0.62	–

Table 10: 96-token configuration with a fixed three-search budget and no learned value control. Macro quality (%); abbreviations as at left.

Method	R@10	SC@10	F1	Δ 95% CI
DM	85.23	70.60	49.83	+2.23 [1.78, 2.67]
DC	86.60	72.97	50.30	-0.13 [-0.56, 0.30]
RILO	86.55	72.83	50.28	–

Δ: RILO minus the row method in SC@10 percentage points, with paired question-bootstrap intervals. Action construction and controllers are trained separately on each core dataset.

Matched query supervision. Direct-matched, Direct-continuous, and RILO share manifest entries (question, observed prefix, target passage/tokens, weight, split), truncation, minibatch order, training

steps, and validation questions. All use Equation 7 with the same alignment weight; each decoder predicts its own auxiliary vector. Target embeddings provide supervision only. At inference, Direct-matched reads only the question and observed memory.

Continuous-state baseline. Direct-continuous recurrently updates a state from observed inputs, without online relation sequences or discrete codes. Table 11 gives its dimensions and capacity; Appendix D.1 specifies matched short-query and corpus-path endpoint/distillation supervision.

Candidate generation. All methods share the backbone, adaptation, input/output limits, and validation-search allowance, generating one query per condition with two-beam decoding. RILO's cap includes retained plans and the direct fallback. Direct variants use one full-state and up to 12 passage-focused conditions, ordered by memory rank, then canonical passage ID. Each retains the question and memory while marking one retrieved passage; DC computes its state from that condition, whereas DM receives no endpoint or code. Condition construction is fixed by source validation. Duplicate queries incur generation cost but merge before selection, retaining the first candidate. Query counts, tokens, and latency include unexecuted candidates and pre-STOP computation (Table 13); Table 11 identifies the shared query backbone.

Separately fitted controllers. Each method independently collects transitions and fits its controller on the same dataset-specific training questions, with matched collection rules, search budgets, backbone, utility, cost, and optimization schedule. Features come from its own generator; DM receives no operator-derived features. Table 11 gives the budgets.

Table 11: **Matched-control settings.**

Field	Direct-matched	Direct-continuous	RILO
Training examples / unique targets	12,000 / 4,800	12,000 / 4,800	12,000 / 4,800
Text / alignment loss (weight)	CE / cosine, 0.1	CE / cosine, 0.1	CE / cosine, 0.1
State dim. / depth / adapter	no latent state	256 / 3 / linear	256 / 3 / linear
Query backbone	tiny-T5	tiny-T5	tiny-T5
Query steps / batch size	3,000 / 16	3,000 / 16	3,000 / 16
Candidate cap / tokens per candidate	13 / 96	13 / 96	13 / 96
Query decoding (beams)	2	2	2
Collection searches after initialization	24,000	24,000	24,000
Controller fitting steps	2,000	2,000	2,000

Uncertainty estimation. Tables 12 and 9 report three paired runs on 1,000 questions per dataset and equal-weight macro means. SD uses complete-dataset run scores. Paired intervals use 2,000 within-dataset question-bootstrap draws, keeping each question's runs together and sharing draws across methods. They condition on the evaluated runs.

Fixed-budget diagnostic. Table 10 uses three searches, selecting the highest proposal-scored candidate with deterministic ties (Appendix A.7.2). It tests construction and fixed selection without learned value control or stopping.

C THEORETICAL ANALYSIS

We analyze control under a fixed relational language, realizer, and candidate rule, then relate operator approximation to candidate restriction. Policy values measure additional net utility relative to stopping; endpoint bounds concern evidence representations.

C.1 RETRIEVAL CONTROL FORMALISM

Action sets. Let $\mathcal{P}(s)$ contain all entities observed in the question and evidence memory. For a finite operator vocabulary $\mathcal{O}$ and depth limit H, define the full and retained action sets by

$$\begin{aligned} \mathcal{A}_H(s) &= \{\textsc{Stop}, \textsc{Dense}\} \cup \{(p, o_{1:\ell}) : p \in \mathcal{P}(s),\ 1 \le \ell \le H,\ o_{1:\ell} \in \mathcal{O}^\ell\}, \\ \mathcal{A}(s) &= \{\textsc{Stop}, \textsc{Dense}\} \cup \mathcal{A}^{\mathrm{comp}}(s) \subseteq \mathcal{A}_H(s). \end{aligned} \tag{21}$$

The full set precedes pivot/beam pruning and endpoint merging; the retained set has at most J compositions. Both sets are finite. Decoding and ties are fixed, with any proposal seed observed before selection, ensuring identical action-realization rules across systems.

Table 12: Per-dataset matched comparison under the 96-token configuration. Quality (%, mean ± run SD), mean calls and latency. Δ: RILO minus the row method in SC@10 points, with paired 95% intervals.

Dataset / method	R@10	SC@10	F1	Calls	Latency (s)	Δ [95% CI]
2WikiMultiHopQA (in-domain)						
Direct-matched	83.83±0.64	70.07±1.23	52.14±0.21	2.173	0.527	+3.57 [2.67, 4.47]
Direct-continuous	84.67±0.43	71.73±0.57	52.43±0.15	2.187	0.593	+1.90 [1.17, 2.70]
RILO	85.73±0.67	73.63±1.31	52.80±0.20	2.183	0.618	–
HotpotQA (in-domain)						
Direct-matched	95.53±0.58	91.60±0.87	54.33±0.21	2.157	0.534	+0.90 [0.17, 1.70]
Direct-continuous	96.32±0.53	92.87±0.99	54.60±0.20	2.194	0.586	-0.37 [-1.10, 0.40]
RILO	96.13±0.30	92.50±0.44	54.54±0.12	2.168	0.634	–
MuSiQue (in-domain)						
Direct-matched	74.33±0.14	46.97±0.81	42.32±0.04	2.162	0.529	+2.77 [2.00, 3.53]
Direct-continuous	76.81±1.01	51.00±1.48	43.16±0.29	2.188	0.591	-1.27 [-2.00, -0.57]
RILO	76.35±0.67	49.73±0.80	43.01±0.19	2.157	0.621	–

Table 13: Query-generation cost under the 96-token configuration. Mean candidates per decision round, including STOP rounds; mean tokens and p95 retrieval latency per question, excluding the reader.

Dataset / method	Generated	Distinct	Input tokens	Output tokens	p95 (s)
2WikiMultiHopQA					
Direct-matched	13.00	10.67	9,539	1,155	0.703
Direct-continuous	13.00	11.46	9,597	1,161	0.768
RILO	13.00	11.96	9,645	1,165	0.814
HotpotQA					
Direct-matched	13.00	10.69	9,565	1,149	0.696
Direct-continuous	13.00	11.41	9,552	1,157	0.774
RILO	13.00	11.96	9,512	1,163	0.817
MuSiQue					
Direct-matched	13.00	10.67	9,538	1,156	0.692
Direct-continuous	13.00	11.45	9,609	1,161	0.766
RILO	13.00	11.98	9,488	1,160	0.824

Retrieval dynamics. For a retrieval action, let $P(ds' \mid s, a)$ be the distribution of the next observed state after a real corpus query and the memory/history update. The state contains the history needed to make this transition Markov. All policies share P, the frozen query realizer, and the reward

$$r(s, a, s') = U(s') - U(s) - \lambda, \qquad |U(s)| \le U_{\max} < \infty, \quad 0 \le \lambda < \infty. \tag{22}$$

Policy value. Each retrieval consumes one budget unit, regardless of composition length. STOP terminates immediately without a query or additional reward. Budget zero also forces termination. For a policy π and its number $\tau \le b$ of additional queries, define

$$V^{\pi}(s, b) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\tau-1} r(S_t, A_t, S_{t+1}) \,\middle|\, S_0 = s, b\right]. \tag{23}$$

The empty sum is zero. Telescoping recovers $\mathbb{E}_{\pi}[U(S_\tau) - U(s) - \lambda\tau]$, consistent with Equation (1).

Reachable domain. Let $\mathcal{X}_H$ contain the starting state–budget pairs and be closed under successors of every full-set retrieval action: if $(s, b) \in \mathcal{X}_H$ and $b > 0$, then $(s', b-1) \in \mathcal{X}_H$ for P-almost every successor. All assumptions below hold uniformly on this full-action reachable domain. The maximum starting budget is finite.

Oracle values. For $\mathcal{D} \in \{\mathcal{A}_H, \mathcal{A}\}$, the optimal action and state values satisfy, for $b > 0$,

$$\begin{aligned} Q^*_{\mathcal{D}}(s,a,b) &= \mathbb{E}_{s'\sim P(\cdot|s,a)}[r(s,a,s') + V^*_{\mathcal{D}}(s',b-1)], \qquad a \neq \textsc{Stop}, \\ Q^*_{\mathcal{D}}(s,\textsc{Stop},b) &= 0, \\ V^*_{\mathcal{D}}(s,b) &= \max_{a\in\mathcal{D}(s)} Q^*_{\mathcal{D}}(s,a,b). \end{aligned} \tag{24}$$

All state and action values at budget zero are defined to be zero. We abbreviate $V^*_{\mathcal{A}_H} = V^*_H$ and $Q^*_{\mathcal{A}_H} = Q^*_H$. The option to stop implies $V^*_{\mathcal{D}} \geq 0$.

C.2 Approximation Assumptions

Candidate restriction. For an action-value function Q, define the restriction operator $(\mathsf{R}_{\mathcal{A}}Q)(s,b) = \max_{a\in\mathcal{A}(s)} Q(s,a,b)$. The candidate-restriction assumption is

$$0 \leq V^*_H(s,b) - (\mathsf{R}_{\mathcal{A}}Q^*_H)(s,b) \leq \delta \qquad ((s,b)\in\mathcal{X}_H,\ b>0). \tag{25}$$

Here Q^*_H allows full-set optimal continuation after the first action. Thus δ bounds the loss from restricting a single decision.

Bellman operator. The retained-set population Bellman operator is

$$(\mathcal{T}_{\mathcal{A}}Q)(s,a,b) = \begin{cases} 0, & b = 0 \text{ or } a = \textsc{Stop}, \\ \mathbb{E}_{s'\sim P(\cdot|s,a)}[r(s,a,s') + (\mathsf{R}_{\mathcal{A}}Q)(s',b-1)], & \text{otherwise}, \end{cases} \tag{26}$$

with $(\mathsf{R}_{\mathcal{A}}Q)(s,0) = 0$.

Residual assumption. The learned value function has exact boundaries $Q_\omega(s,\textsc{Stop},b) = Q_\omega(s,a,0) = 0$ and satisfies

$$\sup_{\substack{(s,b)\in\mathcal{X}_H,\ b>0 \\ a\in\mathcal{A}(s)}} |Q_\omega(s,a,b) - (\mathcal{T}_{\mathcal{A}}Q_\omega)(s,a,b)| \leq \varepsilon. \tag{27}$$

This population assumption uses the same Q_ω on both sides and requires verification beyond sampled training loss.

Deployed policy. Write $\widehat{V}_\omega = \mathsf{R}_{\mathcal{A}}Q_\omega$. The deployed policy π_ω maximizes Q_ω over the retained set, breaking ties in favor of STOP. Accordingly, $\widehat{V}_\omega(s,b) = 0$ whenever the policy stops, and it is also zero at budget exhaustion. Let $V^{\mathrm{RILO}} = V^{\pi_\omega}$ and $\bar{\tau}_\omega(s,b) = \mathbb{E}_{\pi_\omega}[\tau \mid s,b] \leq b$.

C.3 Adaptive-Horizon Control Bound

Proof. **Step 1. Candidate restriction.** Retained policies are full-set feasible, so $V^*_H \geq V^*_{\mathcal{A}}$. Both values vanish at $b = 0$. Inductively, for each retained retrieval action,

$$\begin{aligned} Q^*_H(s,a,b) - Q^*_{\mathcal{A}}(s,a,b) &= \mathbb{E}[V^*_H(s',b-1) - V^*_{\mathcal{A}}(s',b-1)] \\ &\leq (b-1)\delta. \end{aligned} \tag{28}$$

The difference for STOP is zero. Using $|\max_a f_a - \max_a g_a| \leq \max_a |f_a - g_a|$ and Equation (25) gives

$$\begin{aligned} V^*_H - V^*_{\mathcal{A}} &= \left[V^*_H - \mathsf{R}_{\mathcal{A}}Q^*_H\right] + \left[\mathsf{R}_{\mathcal{A}}Q^*_H - \mathsf{R}_{\mathcal{A}}Q^*_{\mathcal{A}}\right] \\ &\leq \delta + (b-1)\delta = b\delta, \end{aligned} \tag{29}$$

where functions in this display are evaluated at (s,b).

Step 2. Value approximation. The error is zero at $b = 0$. If $|\widehat{V}_\omega(s',b-1) - V^*_{\mathcal{A}}(s',b-1)| \leq (b-1)\varepsilon$ at all reachable successors, then for a retained retrieval action,

$$\begin{aligned} |Q_\omega(s,a,b) - Q^*_{\mathcal{A}}(s,a,b)| &\leq |Q_\omega(s,a,b) - (\mathcal{T}_{\mathcal{A}}Q_\omega)(s,a,b)| \\ &\quad + \mathbb{E}\big[|\widehat{V}_\omega(s',b-1) - V^*_{\mathcal{A}}(s',b-1)|\big] \\ &\leq \varepsilon + (b-1)\varepsilon = b\varepsilon. \end{aligned} \tag{30}$$

STOP has zero error. Taking maxima over the same retained set establishes

$$|\widehat{V}_\omega(s,b) - V^*_{\mathcal{A}}(s,b)| \leq b\varepsilon. \tag{31}$$

Step 3. Policy execution. Define the signed residual $e_\omega(s,a,b) = Q_\omega(s,a,b) - (\mathcal{T}_\mathcal{A} Q_\omega)(s,a,b)$. At any state where π_ω retrieves, write $a = \pi_\omega(s,b)$. Greediness implies

$$\widehat{V}_\omega(s,b) = e_\omega(s,a,b) + \mathbb{E}[r(s,a,s') + \widehat{V}_\omega(s', b-1)]. \tag{32}$$

The terminal predicted value is zero for STOP and budget exhaustion. Telescoping conditional expectations along the finite trajectory gives

$$\begin{aligned} \widehat{V}_\omega(s,b) - V^{\mathrm{RILO}}(s,b) &= \mathbb{E}_{\pi_\omega}\left[\sum_{t=0}^{\tau-1} e_\omega(S_t, A_t, b-t) \,\middle|\, s, b\right], \\ |\widehat{V}_\omega(s,b) - V^{\mathrm{RILO}}(s,b)| &\le \mathbb{E}_{\pi_\omega}[\tau \mid s, b]\varepsilon = \bar{\tau}_\omega(s,b)\varepsilon. \end{aligned} \tag{33}$$

Combining the bounds. The deployed policy is feasible for the full oracle, so its suboptimality is nonnegative. Equations (29), (31), and (33) yield

$$\begin{aligned} 0 \le V_H^* - V^{\mathrm{RILO}} &= (V_H^* - V_\mathcal{A}^*) + (V_\mathcal{A}^* - \widehat{V}_\omega) + (\widehat{V}_\omega - V^{\mathrm{RILO}}) \\ &\le b\delta + b\varepsilon + \bar{\tau}_\omega(s,b)\varepsilon \\ &\le b(\delta + 2\varepsilon). \end{aligned} \tag{34}$$

If RILO selects STOP at (s,b), then $\widehat{V}_\omega(s,b) = 0$, $V^{\mathrm{RILO}}(s,b) = 0$, and $\bar{\tau}_\omega(s,b) = 0$. Consequently, $V_H^*(s,b) \le b(\delta+\varepsilon)$, proving the stopping claim. At $b = 0$ the same statements hold with all terms equal to zero. □

Scope. The bound is conditional on the uniform assumptions within the fixed learned action system; $\delta = \varepsilon = 0$ gives optimality in that system. Composition depth changes the action sets and can change the error constants; the horizon counts corpus queries.

C.4 GRAPH-PATH APPROXIMATION

Ordered path encoding. Consider a grounded path $\pi = (u_0 \xrightarrow{r_1} \cdots \xrightarrow{r_\ell} u_\ell)$ in $\mathcal{G}$, with its path-associated evidence representations $z_0, \ldots, z_\ell$, which include evidence context. Write $G_o(z) = G_\theta(z,o)$. The router and operator model encode the path as

$$\sigma(\pi) = (o_{k(r_1)}, \ldots, o_{k(r_\ell)}), \qquad F_{\sigma(\pi)} = G_{o_{k(r_\ell)}} \circ \cdots \circ G_{o_{k(r_1)}}. \tag{35}$$

For concatenable graph paths, $\sigma(\pi_1 \cdot \pi_2)$ is the concatenation of their operator sequences. This preserves transition order, not graph adjacency or path identity: distinct paths can receive the same sequence, and a proposed sequence need not correspond to a graph path. The graph is used here as an offline reference, not accessed during retrieval.

Sufficient state for recurrent implementations. An endpoint-space map $G_o(z)$ requires the endpoint to be a sufficient transition state. For the GRU implementation, use $x_i = (h_i, e_0)$, $\mathcal{G}_o(x) = (\mathrm{GRU}_\theta(h,o), e_0)$ and endpoint readout $D(x) = \mathrm{Normalize}(e_0 + \mathrm{Residual}_\theta(h, e_0))$. The following recursion applies to x and $\mathcal{G}$ when grounded reference states and uniform transition bounds exist. If D is L_D-Lipschitz on the relevant region, the endpoint error is at most $L_D\|\widehat{x}_\ell - x_\ell\|$. For the normalized readout this additionally requires its pre-normalization norm to be bounded away from zero. The E_H used in the subsequent endpoint coverage argument must then be this readout-adjusted bound, not the unadjusted hidden-state bound. Grounded intermediate states and uniform constants are additional assumptions, not supplied by final-endpoint supervision.

Proposition 1 (Graph-path approximation)**.** For the path above, let $\widehat{z}_i = G_{o_{k(r_i)}}(\widehat{z}_{i-1})$, $e_i = \|\widehat{z}_i - z_i\|$, and $d_i = \|G_{o_{k(r_i)}}(z_{i-1}) - z_i\|$. Suppose each applied operator is L_G-Lipschitz, with $L_G \ge 0$, on a region containing its grounded and predicted inputs. Then

$$e_\ell \le L_G^\ell e_0 + \sum_{i=1}^{\ell} L_G^{\ell-i} d_i. \tag{36}$$

In particular, if $e_0 = 0$ and $d_i \le \epsilon_{\mathrm{op}}$, then $e_\ell \le E_\ell := \epsilon_{\mathrm{op}} \sum_{j=0}^{\ell-1} L_G^j$.

Proof. Adding and subtracting the operator prediction at the grounded input gives

$$\begin{aligned} e_i &\le \|G_{o_{k(r_i)}}(\widehat{z}_{i-1}) - G_{o_{k(r_i)}}(z_{i-1})\| + d_i \\ &\le L_G e_{i-1} + d_i. \end{aligned} \tag{37}$$

Induction yields Equation (36); substituting $d_i \le \epsilon_{\mathrm{op}}$ and $e_0 = 0$ gives the stated special case. □

Connection to operator training. With the teacher target $\widetilde{z}_i = T_\psi(z_{i-1}, E_r(r_i))$, the triangle inequality gives $d_i \le \|G_{o_{k(r_i)}}(z_{i-1}) - \widetilde{z}_i\| + \|\widetilde{z}_i - z_i\|$. The two terms separate student–teacher mismatch from teacher error. Distillation addresses the first, while retrieval supervision anchors the endpoint to target passages (Appendix A.3.1). Multi-step training applies both losses to the final endpoint. For $L_G < 1$, $E_\ell \le \epsilon_{\mathrm{op}}/(1 - L_G)$; for $L_G = 1$, $E_\ell = \ell\epsilon_{\mathrm{op}}$; for $L_G > 1$, the bound can grow geometrically with depth. Applying it requires uniform operator-error and Lipschitz bounds beyond average training losses.

Shared-code ambiguity. Suppose two grounded paths have identical initial representations and the same operator sequence, but terminal representations z_ℓ and z'_ℓ. Their deterministic operator rollout has a common prediction y, so

$$\max\{\|y - z_\ell\|, \|y - z'_\ell\|\} \ge \tfrac{1}{2}\|z_\ell - z'_\ell\|. \tag{38}$$

The triangle inequality gives the bound. It is attained at the midpoint when arbitrary Euclidean outputs are admissible; with unit-normalized outputs, that midpoint need not be feasible and attainment is not claimed. This endpoint ambiguity concerns identical inputs and codes; the query realizer additionally conditions on the question and state.

C.5 Coverage-Based Control Bound

A sufficient coverage condition. For $(s, b) \in \mathcal{X}_H$ with $b > 0$, let $a^\star$ maximize $Q^*_H(s, a, b)$ over $\mathcal{A}_H(s)$. A STOP or DENSE maximizer is already retained, giving zero restriction loss. For a compositional maximizer, assume a retained action $\bar{a}$ satisfies:

(i) Both actions encode grounded reference paths of length at most H, initialized at the representations used by their rollouts. Their predicted endpoints $\widehat{z}_a$ and reference endpoints z_a^{ref} satisfy $\|\widehat{z}_a - z_a^{\mathrm{ref}}\| \le E_H$ for $a \in \{a^\star, \bar{a}\}$, as in Proposition 1.

(ii) The retained reference endpoint covers the maximizing one: $\|z_{a^\star}^{\mathrm{ref}} - z_{\bar{a}}^{\mathrm{ref}}\| \le \xi$.

(iii) Full-oracle action values for this pair satisfy $|Q^*_H(s, a^\star, b) - Q^*_H(s, \bar{a}, b)| \le L_V\|\widehat{z}_{a^\star} - \widehat{z}_{\bar{a}}\|$ for a finite $L_V \ge 0$.

The third condition assumes value stability for realized queries and their full-oracle continuations, including the realizer's dependence on the pivot and operator sequence.

Restriction-loss bound. The triangle inequality gives $\|\widehat{z}_{a^\star} - \widehat{z}_{\bar{a}}\| \le 2E_H + \xi$. Since $\bar{a}$ remains available,

$$\begin{aligned} 0 \le V^*_H(s, b) - \max_{a \in \mathcal{A}(s)} Q^*_H(s, a, b) &\le Q^*_H(s, a^\star, b) - Q^*_H(s, \bar{a}, b) \\ &\le L_V(2E_H + \xi). \end{aligned} \tag{39}$$

If these conditions hold uniformly on $\mathcal{X}_H$, they supply the sufficient choice $\delta = L_V(2E_H + \xi)$ in the adaptive-horizon bound, yielding

$$V^*_H(s, b) - V^{\mathrm{RILO}}(s, b) \le bL_V(2E_H + \xi) + (b + \bar{\tau}_\omega(s, b))\varepsilon. \tag{40}$$

This separates operator approximation (E_H), retained path coverage (ξ), and value-estimation error (ε). This refinement requires grounded reference paths, endpoint coverage, and value stability. When these conditions are unavailable, δ remains a separate candidate-restriction assumption.

D Matched-Control Evaluation

The matched short-query comparison uses three runs of the same 1,000 questions per dataset. Table 14 reports the resulting RILO–Direct-continuous differences. The corresponding 2Wiki scores are 73.93% SC@10 and 53.23% F1, with retrieval latency of 0.616 seconds. Without learned value control, fixed-three-search SC@10 gains are +1.60/+1.30/+1.60 points for 2Wiki/HotpotQA/MuSiQue, averaging +1.50 points. This remaining gain is consistent with a benefit beyond learned stopping.

D.1 MATCHED RETRAINING PROTOCOL

Table 15 lists matched paths, endpoint/distillation targets, training allowances, dimensions, candidate limits, and query backbones. RILO and Direct-continuous share prefix/target examples, token/alignment losses, query backbone, and validation allowance; online access follows Table 4. Five paired seeds independently fit the operator/continuous model, proposal, query generator, and controller under matched allowances. These five-pipeline results remain separate from the three-run comparison. For three runs on N questions, SC@10 percentages lie on the $100/(3N)$ grid; the equal-weight macro over three 1,000-question datasets lies on the $100/9{,}000$ grid. Scores and differences are computed before rounding to two decimals, including within hop strata. These denominators describe score granularity, not independent sample sizes.

Table 14: **Matched-control differences.** SC@10 differences are RILO minus Direct-continuous, in percentage points. Differences use unrounded scores.

Comparison	RILO	Interval / SD
2Wiki, adaptive	+2.20	[0.24, 4.16] / 0.55
HotpotQA, adaptive	+1.77	[0.13, 3.41] / 0.62
MuSiQue, adaptive	+1.57	[-0.91, 4.05] / 0.71
Macro, adaptive	+1.84	[0.66, 3.02] / 0.33
Macro, fixed three searches	+1.50	[0.16, 2.84] / 0.51

D.2 SUPPLEMENTARY MATCHED COMPARISONS

Table 15: **Matched model configurations.**

Field	Direct-continuous	RILO
Corpus-path examples, maximum	80,000	80,000
Endpoint / same frozen-teacher targets	Yes / Yes	Yes / Yes
Pretraining epochs / batch size	5 / 512	5 / 512
Query examples / distinct target passages	12,000 / 4,800	12,000 / 4,800
Query updates / batch size	3,000 / 16	3,000 / 16
State width / maximum rollout depth	256 / 3	256 / 3
Query backbone	tiny-T5	tiny-T5
Candidate cap / output-token cap per candidate	13 / 32	13 / 32
Transition searches after initialization	24,000	24,000
Controller updates per fitted model	2,000	2,000
Independent full-pipeline seeds	5	5

Table 16: **Mechanism ablations.** SC@10 and F1 are macro percentages. Fixed-search results disable learned value selection.

Condition	SC@10	F1	RILO SC advantage
Adaptive control, at most three searches			
Direct-matched	69.54	49.60	+4.17
Direct-continuous	71.87	50.06	+1.84
Ordinary clustered relation codes	72.63	50.29	+1.08
RILO without multi-step operator training	72.47	50.21	+1.24
RILO	73.71	50.73	–
Fixed three searches, no learned value selection			
Direct-continuous	72.97	50.30	+1.50
RILO	74.47	50.90	–

Table 16 favors endpoint-trained routing over clustering and shows a benefit from multi-step supervision. Both controls retain question-conditioned training, candidate budgets, and separate controller fits. Their effects are not additive: removing multi-step supervision also changes training information. The SC@10 difference increases with QA hop count while absolute completeness declines. QA hops and operator rollout depth are distinct. The 450/350/200 split gives weighted means of 51.00% and 52.57%. Across three runs, the corresponding complete-support counts are 862/464/204 for Direct-continuous and 876/482/219 for RILO.

Table 17: **Source-disjoint composition evaluation.** Endpoint R@10 (%). Seen and unseen sequences contain atomic relations observed during training.

Dataset	Ordered composition	Continuous	RILO	Difference
HotpotQA	Seen sequence	77.40	78.90	+1.50
	Unseen sequence, seen atoms	65.20	67.70	+2.50
MuSiQue	Seen sequence	68.80	70.30	+1.50
	Unseen sequence, seen atoms	53.40	56.90	+3.50

Table 18: **MuSiQue results by annotated QA hop count.** SC@10 (%); three-run means on the 450/350/200-question strata. Weighted totals and differences are computed before rounding.

QA hops	Questions	Continuous	RILO	Difference
2	450	63.85	64.89	+1.04
3	350	44.19	45.90	+1.71
4	200	34.00	36.50	+2.50
Weighted total	1,000	51.00	52.57	+1.57

E SUPPLEMENTARY EVALUATION

E.1 EVALUATION PROTOCOL REGISTRY

Separate evaluation protocols. Table 19 separates protocols and repeat units; comparisons remain within protocol.

Table 19: **Evaluation registry.** SC@10 is a macro percentage. Main, matched-control, scaling, and independent-pipeline evaluations have distinct repeat units.

Protocol	SC@10	Evaluation	Interpretation / repeat unit
MAIN	73.10	Benchmark	Main benchmark; reported three runs
MATCH-96	71.96	96-token	Matched experiment; fixed generators, controller repeats
MATCH-32	73.71	Short-query	Adaptive matched-control evaluation
S-FIXED	74.47	Fixed budget	Three searches, no learned value selection
P-PIPELINE	73.71	Replication	Five independently trained pipelines
S-SCALE-1M	62.47	Scaling	One-million-passage collection
S-LANR	69.24	Baseline	Native latent retrieval; common-reader evaluation
S-LEVEL	68.48	Baseline	Closed-corpus LevelRAG adaptation, no web access

Additional main-table baselines. LAnR and LevelRAG use the shared 1,000 questions per dataset, collections, three-search cap, top-ten evaluation, and frozen Qwen3-32B reader. LAnR retains its native encoder, latent queries, and stopping head, replacing native answering with the common reader. LevelRAG retains atomic-query planning and rewriting but searches only the shared dense/sparse indexes, without web access. Each index invocation costs one call. Fitting/tuning uses source splits; evaluation support labels are unavailable online. Their macro SC@10/F1 scores are 69.24/49.43 and 68.48/49.14, respectively.

Statistical units. Main and 96-token matched comparisons use the paired question-bootstrap procedures in Appendices A.7.4 and B.1, respectively. Their intervals condition on the evaluated runs. Appendix E.6 separately reports five-pipeline training variation and defines the analytic sensitivity bands.

E.2 DISTRACTOR SCALING

Scale evaluation. Each benchmark base pool is expanded to 100,000, 1,000,000, and 5,000,000 passages with distinct Wikipedia-snapshot passages in an order independent of questions and labels. All methods share passage IDs/boundaries and the same 1,000 questions, including missing-support cases, without gold-support insertion. Models, prompts, thresholds, and tuning choices remain frozen; only encodings and indexes are rebuilt, with no additional operator training.

Table 20: **Wikipedia-augmented distractor scaling.** Macro quality percentages over three datasets, calls per question, and retrieval seconds measured from requests. This evaluation is separate from Figure 5e.

Passages	Method	R@10	SC@10	F1	Calls	Time (s)
100,000	EfficientRAG	80.63	64.83	47.61	2.437	0.482
	IRCoT	79.97	63.47	47.28	2.563	1.681
	Direct-continuous	82.91	68.11	48.83	2.247	0.668
	RILO	83.68	69.37	49.32	2.239	0.704
1,000,000	EfficientRAG	74.86	57.63	43.91	2.591	0.647
	IRCoT	74.12	56.37	43.52	2.704	1.923
	Direct-continuous	77.21	60.91	45.26	2.438	0.884
	RILO	78.34	62.47	45.93	2.417	0.936
5,000,000	EfficientRAG	68.27	49.83	39.87	2.713	0.936
	IRCoT	67.83	48.97	39.54	2.821	2.316
	Direct-continuous	71.16	53.71	41.18	2.593	1.237
	RILO	72.45	55.03	41.72	2.571	1.319

Scaling interpretation. All methods lose coverage and take longer at larger scales. At one million passages, RILO gains 1.56 SC@10 and 0.67 F1 points over DC at approximately 5.9% additional retrieval latency.

E.3 SYSTEM-LEVEL COST COMPARISON

Baseline configurations. Table 21 distinguishes native encoder pairs from shared-Qwen3-index adaptations and lists source-only permissions. MDR-style (adapted) freezes the document encoder and fits a compatible question/history query model on source retrieval targets.

Table 21: **Baseline adaptation settings.** Native encoders and controlled adaptations are distinguished explicitly.

Method family	Source-only allowance	Controlled-track identity
Dense / Hybrid	Index / fusion tuning	Shared encoder; Hybrid charges both searches
HyDE	Prompt and decoding tuning	Generated text encoded by shared encoder
MDR	Retrieval fitting and tuning	MDR-style adapted query model; native row separate
EfficientRAG	Native-module fitting and tuning	Document query/index compatibility and any adaptation
QAFD / HippoRAG2	Graph construction and tuning	Shared input corpus; graph access and work charged
IRCoT	Demonstration and stop-rule tuning	Shared search interface; reasoning generation timed
LAnR	Native latent-model fitting and stopping	Native retrieval encoder; common-reader adaptation
LevelRAG	Source prompt/search-rule tuning	Closed-corpus dense/sparse search; no web access
DM / DC / RILO	Declared matched fitting	Shared supervision permissions and separate controllers

Charging rules. Accounting follows Appendices A.6 and A.7, including unexecuted candidates and pre-STOP work. Serial end-to-end latency adds reader time to retrieval time.

Zero output-token entries indicate no autoregressive retrieval generation; encoder, tagger, and other neural computation remains included in timing. Hardware allocation and timing conditions are detailed in Appendix E.6. RILO uses approximately 1.2% more output tokens and 5.3% more retrieval time than DC, and remains slower than EfficientRAG.

Short-query configuration. DM, DC, and RILO generate concise retrieval queries with a 32-token output cap, deterministic two-beam decoding, and no reasoning trace or answer generation. The input cap is 512 tokens, with at most 13 candidates per decision. Two decision rounds permit at most

Table 22: **Per-dataset search counts.** All methods use a per-question cap of three; Dense and Hybrid operate at one and two searches, respectively.

Method	2Wiki	HotpotQA	MuSiQue	Macro calls
Dense	1.000	1.000	1.000	1.000
Hybrid	2.000	2.000	2.000	2.000
HyDE	1.000	1.000	1.000	1.000
MDR-style (adapted)	2.000	2.000	3.000	2.333
EfficientRAG	2.327	2.284	2.516	2.376
QAFD	2.000	2.000	2.000	2.000
HippoRAG2	2.000	2.000	2.000	2.000
IRCoT	2.481	2.367	2.684	2.511
LAnR	2.137	2.083	2.241	2.154
LevelRAG	2.473	2.397	2.681	2.517
Direct-matched	2.173	2.157	2.162	2.164
Direct-continuous	2.187	2.194	2.188	2.190
RILO	2.203	2.187	2.176	2.189

Table 23: **Online retrieval cost.** Tokens include unexecuted candidates and STOP rounds. Mean, median, and p95 are computed from actual per-request retrieval timings. Final answer generation is excluded.

Method	Input tokens	Output tokens	Mean (s)	Median (s)	p95 (s)
Dense	0	0	0.071	0.067	0.098
Hybrid	0	0	0.108	0.103	0.151
HyDE	412	87	0.438	0.419	0.612
MDR-style (adapted)	0	0	0.219	0.208	0.307
EfficientRAG	0	0	0.413	0.392	0.587
QAFD	527	29	0.536	0.511	0.782
HippoRAG2	684	41	0.748	0.706	1.094
IRCoT	3,862	246	1.571	1.492	2.263
LAnR	6,743	0	0.548	0.521	0.781
LevelRAG	4,286	139	0.914	0.873	1.327
Direct-matched	9,547	474	0.530	0.506	0.699
Direct-continuous	9,586	481	0.590	0.563	0.769
RILO	9,612	487	0.621	0.592	0.827

832 output tokens per question, including unexecuted candidates and queries generated before STOP. The teacher-forced training-target limit remains 96 tokens. Appendix B.1 specifies the 96-token configuration.

End-to-end latency. The serial RILO retrieval breakdown is 0.068 seconds for initial retrieval, 0.026 for proposal/rollout, 0.397 for query realization, 0.051 for encoding, 0.011 for value scoring, 0.049 for subsequent searches, and 0.019 for fusion/bookkeeping, totaling 0.621 seconds. The mean reader time is 1.846 seconds and end-to-end time is 2.467 seconds. For DC, reader time is 1.832 seconds and end-to-end time is 2.422 seconds. The end-to-end difference is approximately 1.9%. End-to-end quantiles use measured complete-request durations.

Offline costs. RILO preparation takes 28.4 accelerator-hours: graph extraction 8.6, corpus encoding/index preparation 3.4, teacher/operator training 4.7, proposal/query training 7.2, transition collection 3.1, and controller fitting 1.4. The corresponding DC total is 26.9 accelerator-hours. CPU time and storage are accounted for separately.

E.4 MECHANISM DIAGNOSTICS

Diagnostic settings. The diagnostics below use development-selected thresholds and Table 4's signal access.

E.4.1 BUDGETED CONTROL

S-CTRL evaluates 300 held-out questions, 100 per core dataset, with a fixed generator and a common retained candidate tree. With at most 13 retrieval actions and two searches after initialization,

exhaustive diagnostic expansion requires at most $300(13 + 13^2) = 54{,}600$ extra index searches, plus 300 initial searches; query generation is charged separately. The retained-tree oracle maximizes utility over the retrieved branches using offline support labels.

Full and immediate-gain policies use the same inputs and fitting allowance, but predict two-step and one-step returns, respectively. Simple stopping uses a development-selected novelty threshold and the immediate-gain action ranker. Fixed-budget control uses the full ranker but disables STOP. The oracle supplies a label-informed diagnostic ceiling. With the existing $\beta = 1$ and $\lambda = 0.02$, report

$$J = \mathbb{E}[U(s_\tau) - 0.02\tau] = \frac{\overline{R@10} + \overline{SC@10}}{200} - 0.02(\overline{\text{calls}} - 1).$$

Here the displayed recall and completeness are percentages; calls include initial retrieval. Rounded input means make displayed J approximate.

Table 24: **Controller comparison on 300 held-out questions.** All policies use the same retained candidate tree; calls include initialization.

Policy	R@10	SC@10	Calls	J
Immediate-gain	84.83	71.67	2.31	0.7563
Simple stopping	84.17	70.67	2.07	0.7528
Fixed budget	86.67	75.00	3.00	0.7684
RILO full controller	86.17	74.33	2.19	0.7787
Retained-tree oracle	87.50	77.00	2.23	0.7979

The full controller improves J by 0.0224 over immediate-gain selection and by 0.0103 over fixed-budget execution, despite the latter's higher completeness. This supports budget allocation that balances evidence coverage and query cost. The utility gap to the retained-tree oracle is 0.0192.

Controller errors. Continuation-value MAE is 0.031/0.044 at remaining budgets one/two; first-action regret is 0.009/0.016. Regret assumes oracle continuation after the selected action and differs from the full-policy utility gap. There are 7 premature stops among 92 STOP decisions (7.61%), versus 19/126 (15.08%) for simple stopping, using a 0.01-utility oracle-continuation threshold. Two-step MAE is 0.061 in the low-frequency stratum and 0.038 in the frequent stratum.

E.4.2 CANDIDATE PRUNING

S-PRUNE uses a separate balanced 300-question diagnostic, with identical pre-merge proposals and a 12-plan cap. Compare no merging, the current within-pivot endpoint threshold 0.98, and within-pivot query-vector merging at a development-selected threshold. Query merging must realize and encode all pre-merge candidates, including discarded ones. A separate 32-plan reference pool supplies the best realized immediate-utility action for retention auditing within that pool. Tied best actions count as retained if any survives. End-to-end SC@10 is measured separately after executing each variant's policy.

Table 25: **Candidate-pruning trade-offs.** Candidate counts and request times are means over the diagnostic evaluation.

Rule	Plans retained	Best-action retention (%)	SC@10	Seconds
No merge, cap 12	12.00	94.00	75.00	0.821
Endpoint merge, cap 12	9.37	92.00	74.33	0.643
Query-vector merge, cap 12	9.61	93.67	74.67	0.746

Endpoint merging reduces retrieval time by 21.68% relative to no merging, at a 0.67-point SC@10 cost. Before final truncation, 12/300 states (4.00%) lose more than 0.01 immediate utility through merging alone. The expanded-pool retention statistic additionally includes the final candidate cap. The realizer reads the full plan, so endpoint merging is a heuristic.

Depth bias. Under the same cap, SC@10 is 74.33 with the current unnormalized score, 74.67 with development-selected length normalization, and 74.33 with per-depth quotas.

E.4.3 Relational Information Beyond Endpoint Compression

S-REL extends Table 17 with source-document-disjoint splits and unseen ordered compositions of seen atomic relations. Relation statistics and routing are fitted on training data only. Controls match path length, source context, negatives, and relation-frequency bands. The split holds out source documents and ordered compositions, while allowing entity overlap and retaining seen atomic relations.

Table 26: **Unseen-composition representation controls.** Scores are endpoint R@10.

Representation	HotpotQA	MuSiQue
Source-only, fitted without relation inputs	59.83	47.61
Unordered code aggregation, retrained	63.47	51.83
Direction-collapsed relation inputs, retrained	64.13	52.47
Direct-continuous	65.20	53.40
Ordered RILO operators	67.70	56.90

The unordered control pools code embeddings without positional information and refits the endpoint predictor under the same supervision and parameter allowance. The direction-collapsed control removes available explicit inverse/direction annotations. Ordered operators exceed unordered pooling by 4.23/5.07 points and the continuous representation by 2.50/3.50 points. Together with the source-only control, this supports information from ordered relational transitions beyond contextual endpoint prediction. Removing direction changes both input information and inductive bias.

E.4.4 From Retrieved Support to Reader Success

S-READ uses a balanced set of 300 paired RILO/DC questions. Partition questions by complete support in the top ten before reader truncation, and use the same frozen reader, prompt, ranking contract, and 4,096-token input cap. The counts below give SC@10 of 74.33 for RILO and 71.67 for DC. No question is dropped from the paired analysis.

Table 27: **Paired evidence strata on 300 diagnostic questions.** Counts and conditional mean answer F1 use the same reader and context limit.

Complete support	Questions	RILO F1	DC F1
Both methods	201	64.31	63.87
Only RILO	22	48.73	39.26
Only DC	14	37.84	46.57
Neither	63	20.13	20.41

Within the retrieval-defined strata, newly completed support is associated with better answers on the 22 RILO-only questions, with the opposing pattern in the 14 DC-only questions. Weighting all strata gives F1 of 52.65 versus 52.13, a 0.52-point difference. After truncation, all annotated support spans remain visible for 193/223 (86.55%) complete-support RILO questions and 185/215 (86.05%) complete-support DC questions.

Error categories. For 100 sampled incorrect RILO answers under normalized exact match, the primary error categories are 18 extraction/pivot/routing, 24 composition/endpoint, 17 query realization, 11 candidate pruning, 14 premature stopping, and 16 context-delivery/reader failures.

E.5 Transfer Evaluation

Evaluation scope. S-TRANSFER uses frozen transfer; S-LANR/S-LEVEL costs concern in-domain main-table evaluations. Hardware follows Appendix E.6.

E.5.1 Source-Only Frozen Comparison

Apply Appendix A.4's frozen-transfer protocol from 2Wiki to 1,000 questions each on HotpotQA and MuSiQue, with matched RILO/DC source supervision and fitting allowances. LAnR retains its native encoder and stopping head; LevelRAG selects source prompts/search rules without weight fitting (Appendix E.1). All methods index the same supplied-context target collection using frozen encoders. Target demonstrations, tuning, and path training are prohibited; RILO/DC construct no

target supervision graph. Target support labels reach only the scorer. Frozen pivot extraction, fusion, and stopping rules apply, with a three-search cap (dense/sparse charged separately), common reader, and 4,096-token context limit.

Table 28: **Source-only frozen transfer.** Training and selection use 2Wiki; each target has 1,000 distinct questions. Scores are percentages and calls include initialization.

Target	Method	R@10	SC@10	F1	Calls
HotpotQA	Direct-continuous	94.03	88.67	53.37	2.217
	LAnR	92.97	87.43	52.83	2.163
	LevelRAG	93.17	87.97	52.61	2.487
	RILO	94.67	90.43	53.89	2.203
MuSiQue	Direct-continuous	76.19	47.93	40.61	2.253
	LAnR	73.81	44.97	39.17	2.287
	LevelRAG	74.13	45.63	39.43	2.713
	RILO	77.43	49.77	41.27	2.237

Transfer interpretation. Without target-label adaptation, RILO exceeds DC by 1.76/1.84 SC@10 and 0.52/0.66 F1 points on HotpotQA/MuSiQue. Target-only averages are 70.10 versus 68.30 SC@10, 47.58 versus 46.99 F1, and 2.220 versus 2.235 calls. LAnR/LevelRAG average 66.20/66.80 SC@10 under different pretraining and supervision; these are system-level comparisons.

E.5.2 COSTS OF THE ADDITIONAL BASELINES

The costs in Tables 22 and 23 include LAnR's prediction-token passes, stopping, native encoding, search, and evidence re-encoding; its input-token total includes repeated evidence. LevelRAG's tokens include planning, rewriting, and discarded text under development-selected stage limits. Answer generation is separate. Cross-model token totals are descriptive; request timings measure cost.

Table 29: **System resource costs.** Hours are accelerator-hours on four MetaX C550 accelerators. Retrieval, reader, and total times are measured request means.

Method	Preparation (h)	Retrieval (s)	Reader (s)	Total (s)
LAnR	26.3	0.548	1.819	2.367
LevelRAG	5.1	0.914	1.827	2.741
RILO	28.4	0.621	1.846	2.467

LAnR preparation comprises 3.8 accelerator-hours for corpus encoding/indexing, 18.7 for source-permitted model/head fitting, and 3.8 for tuning and validation. LevelRAG uses 3.4 hours for encoding/indexing and 1.7 for prompt/search-rule tuning, totaling 5.1 without weight fitting. These costs cover method-specific preparation across the three main-table datasets, exclude base-model pretraining, and count shared artifacts once.

Quality–cost interpretation. Relative to LAnR, RILO takes 13.32% longer in retrieval and 4.22% longer end to end. LevelRAG has lower preparation cost but higher online cost; RILO takes 32.06% less retrieval time and 10.00% less end-to-end time. These comparisons follow the operating-point scope in Appendix E.3.

E.6 REPRODUCIBILITY DETAILS

For absolute scores, the analytic sensitivity band is $100[p \pm 1.96\sqrt{p(1-p)/n}]$. Here p is the mean expressed as a proportion, with $n = 3{,}000$ across datasets and $n = 1{,}000$ for each frozen target. For paired differences d, the calculation uses variance $(q - d^2)/1{,}000$, where q takes the discordance proportions 0.10/0.07/0.16 for 2Wiki/HotpotQA/MuSiQue. Macro variance is the sum divided by nine. The fixed-search macro band uses discordance 0.14 over 3,000 questions.

Accelerator-hours count occupied device-hours. The 0.35 GiB index stores 2,560-coordinate FP32 embeddings for 27,586 passages, including index overhead. The 120 GiB final-artifact total excludes optimizer states, intermediate checkpoints, and full model copies per seed. All methods use the

Table 30: **Evaluation metadata.** Scores, SDs, band endpoints, and support ceilings use percentage units. Analytic sensitivity bands follow the calculation below; they are distinct from empirical bootstrap intervals and pipeline SDs.

Field	Value
Pipeline seed IDs	42, 43, 44, 45, 46; base seed 42 plus run index
Statistical decision threshold	0.05
Bootstrap analysis seed	42
LAnR macro SC band / SD	[67.59, 70.89] / 0.53
LevelRAG macro SC band / SD	[66.82, 70.14] / 0.50
100k scale: RILO SC band / SD	[67.72, 71.02] / 0.61
1M scale: RILO SC band / SD	[60.74, 64.20] / 0.74
5M scale: RILO SC band / SD	[53.25, 56.81] / 0.88
Indexed scale sizes	100,000 / 1,000,000 / 5,000,000 passages
Support ceiling, supplied-support pools	100.00 at all three scales; base supports retained
Separate label-blind full snapshot	21,000,000 passages
Full-snapshot SC / support ceiling	51.20 / 94.00; separate corpus protocol
Frozen HotpotQA SC band / SD	[88.61, 92.25] / 0.56
Frozen MuSiQue SC band / SD	[46.67, 52.87] / 0.79

Table 31: **Runtime configuration.** Installed device-memory capacity and measured peak usage are reported separately.

Field	Value
Per-run accelerator allocation	4 devices; reader tensor parallel degree 4
Device-memory capacity	64 GB per MetaX C550 accelerator
Host resources	64 CPU cores; 512 GiB RAM
Model precision / quantization	BF16; no weight quantization
Timing batch / warm-up / timed queries	1 / 50 / 1,000
Model residency / cache	resident weights and index; query-result cache off
Timing synchronization	all participating devices before and after timing
RILO reader mean / median / p95 (s)	1.846 / 1.789 / 2.699
RILO end-to-end mean / median / p95 (s)	2.467 / 2.411 / 3.335
DC reader mean / median / p95 (s)	1.832 / 1.776 / 2.679
DC end-to-end mean / median / p95 (s)	2.422 / 2.366 / 3.279
RILO preparation CPU-hours	42.0
Index / final artifact storage	0.35 / 120.0 GiB; three small collections combined
Peak device / host memory	48 GiB per device / 180 GiB
Independent timing repetitions	3

request-level accounting boundaries in Appendix E.3; end-to-end quantiles use measured complete-request durations.

E.6.1 DETAILED DIAGNOSTIC RESULTS

Table 32: **Per-seed complete-support counts.** Each run evaluates 1,000 questions per dataset, or 3,000 across all three; seed order is 42–46. Percent means and sample SDs are calculated from counts before rounding.

Method	Complete-support counts for five runs	Mean / SD (%)
Direct-matched	2069, 2081, 2094, 2080, 2107	69.54 / 0.49
Direct-continuous	2140, 2155, 2168, 2149, 2168	71.87 / 0.41
RILO	2197, 2208, 2221, 2210, 2220	73.71 / 0.33
LAnR	2056, 2070, 2089, 2075, 2096	69.24 / 0.53
LevelRAG	2036, 2048, 2056, 2055, 2077	68.48 / 0.50
DC, fixed searches	2174, 2180, 2195, 2191, 2205	72.97 / 0.41
RILO, fixed searches	2222, 2219, 2230, 2243, 2256	74.47 / 0.51

Model configuration. Excluding the query realizer, teacher, student operator/router, proposal, and controller sizes are 2.37M, 1.75M, 0.14M, and 0.20M parameters. The operator/router includes 16 codes of width 256; the teacher is frozen during student fitting. DM/DC/RILO generate 18.4/18.8/19.1

queries per question, with 16.2/17.1/17.8 distinct texts. Counts cover generated candidates across all decision states, including unexecuted candidates.

Composition split. The source-disjoint composition diagnostic uses 1,000 contexts per dataset/seen-status cell, split equally between lengths 2 and 3; Figure 5f uses 500. All atomic relations occur in training; held-out source documents and ordered compositions have no training overlap.

Table 33: **Two-step controller error partitions.** Each block partitions the same 300 states, with rounded weighted MAE 0.044.

Partition	Stratum	States	MAE
Action	DENSE / compositional	60 / 240	0.039 / 0.04525
Depth	0 (DENSE) / 1 / 2 / 3	60 / 120 / 72 / 48	.039 / .035 / .047 / .06825
Proposal rank	1 / 2–4 / 5–13	150 / 100 / 50	0.036 / 0.048 / 0.060
Frequency	Low / frequent	78 / 222	0.061 / 0.038
Value	$[0, .25), [.25, .5), [.5, .75), [.75, 1]$	60 / 120 / 90 / 30	.040 / .038 / .050 / .056

Depth-specific candidate retention. The query-vector cosine threshold is 0.97. Selected depths 1/2/3 have counts 150/90/60. A separate 300-state retention audit with the same stratum sizes retains 140/83/53 best-reference actions (93.33/92.22/88.33%; pooled 92.00%).