LiR^{3}AG: A Lightweight Rerank Reasoning Strategy Framework for Retrieval-Augmented Generation

AAAI 2026 Poster

Guo Chen[1], Junjie Huang[1], Huaijin Xie[3], Fei Sun[2], Tao Jia[1,4]

[1]Southwest University

[2]Institute of Computing Technology, CAS

[3]Communication University of China

[4]Chongqing Normal University

January 25, 2026

Table of Contents

Background: RAG and Multi-hop QA

Retrieval-Augmented Generation (RAG)

Enhancing Large Language Models (LLMs) by retrieving external knowledge to mitigate hallucinations and update information.

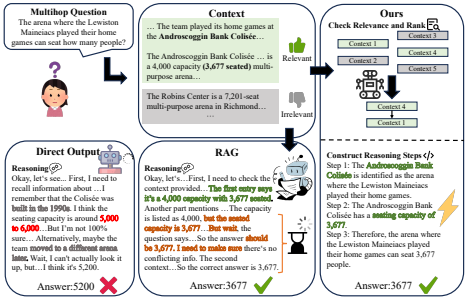

Figure: Multi-hop QA example

The Challenge: Multi-hop QA

- Answers are not found in a single document.
- Requires **reasoning over disjoint pieces of evidence**.
- Requires logical synthesis, not just pattern matching.

A multi-hop QA example where direct generation hallucinates, RAG answers correctly but with redundant reasoning, and LiR^{3}AG uses relevant evidence to generate concise, accurate reasoning steps and offer the right answers.

The Rise of Reasoning Models

- **Reasoning LLMs** (e.g., OpenAI o1, DeepSeek-R1, QwQ) perform "Chain-of-Thought" (CoT) during inference.
- **Performance:** Excellent at multi-hop tasks by explicitly generating thinking steps.

The Problem in RAG Scenarios

Integrating Reasoning Models into RAG is costly:

1. **Token Overhead:** Massive consumption for intermediate thoughts.
2. **Latency:** Significantly slower inference times.
3. **Redundancy:** "Overthinking" on simple retrieval or getting stuck in loops.

RQ: What are Reasoning Models Actually Thinking?

Study: We analyzed 500 queries on HotpotQA using Qwen3 to decode the underlying strategies used when RAG contexts are provided.

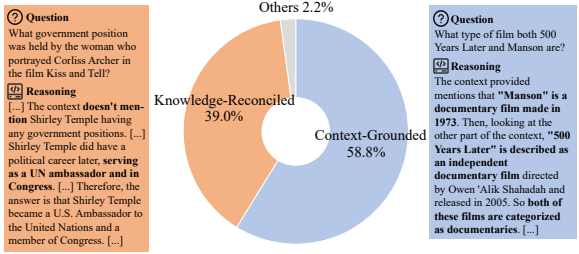

Figure: Distribution of annotated reasoning strategies.

We identified two dominant modes:

1. **Context-Grounded Reasoning**
2. **Knowledge-Reconciled Reasoning**

The majority of responses follow the Context-Grounded Reasoning strategy (58.8%). Two representative examples are shown to illustrate the feature of each strategy.

Comparison of Identified Strategies

1. Context-Grounded (Target)

- **Behavior:** The model trusts retrieved context as reliable evidence.
- **Mechanism:** Direct reasoning on external content.
- **Outcome:** Efficient and accurate.

2. Knowledge-Reconciled (Avoid)

- **Behavior:** Occurs when context is irrelevant or conflicting.
- **Mechanism:** Uses internal knowledge to fix gaps.
- **Outcome:** High redundancy, potential hallucinations, verbose.

Core Insight: We can *transfer* the efficient "Context-Grounded" strategy to non-reasoning models by structuring the input explicitly.

LiR^{3}AG: Lightweight Rerank Reasoning Strategy Framework

- **Goal:** Enable non-reasoning models to perform complex reasoning without the computational burden of Reasoning LLMs.
- **Approach:** Explicitly structure retrieved evidence into coherent reasoning chains.

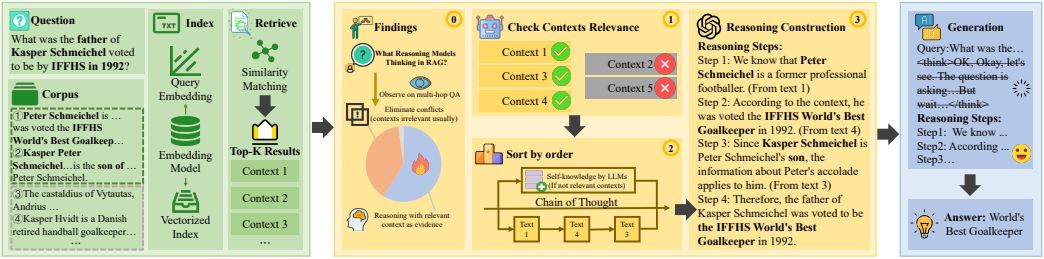

Figure: Overall pipeline of our LiR^{3}AG framework. The Retriever first retrieves potentially relevant contexts. Reranker examines their relevance to the question, filters out irrelevant ones, and orders the remaining contexts according to the expected reasoning sequence. Reasoning Constructor assembles these contexts into structured reasoning steps, which are subsequently passed to the generator to produce the final answer.

Module 1: Retriever

Function: $\mathcal{M}_{\text{Retriever}}$

- **Input:** User Query + Corpus.
- **Process:**
  - Utilizes standard dense or sparse retrieval methods.
  - Fetches top-k potentially relevant contexts based on similarity.
- **Output:** An initial collection of context chunks $\mathcal{C}_k$.

Module 2: Reranker (The Filter & Sorter)

Function: $\mathcal{M}_{\text{Reranker}}$

Going Beyond Naive Reranking

Standard rerankers only score similarity. LiR^{3}AG optimizes for logic.

1. **Relevance Verification:** Filters out irrelevant context to prevent "Knowledge-Reconciled" redundancy (the "overthinking" trap).
2. **Logical Ordering:** Arranges remaining evidence in a **reasoning-consistent order** (e.g., Premise A $\rightarrow$ Premise B $\rightarrow$ Conclusion).

Result: A clean, ordered set of evidence ready for synthesis.

Module 3: Reasoning Constructor

Function: $\mathcal{RC} = f_{\text{template}}(\text{Ordered Contexts})$

- **Role:** Explicitly simulates the "thinking" process.
- **Mechanism:** Uses a lightweight non-reasoning LLM to assemble the filtered contexts into structured steps.
- **Output Structure:**

```
Reasoning Steps: Step 1 [Evidence A] -> Step 2 [Evidence B]...
```

- **Advantage:** Provides the downstream generator with a "pre-computed" thought path, removing the need for internal CoT generation.

Final Generation

- The **Generator** (LLM_G) receives the structured reasoning chain $\mathcal{RC}$.
- Because the reasoning path is already constructed and verified, the Generator simply converts the steps into a final natural language answer.
- **Result:** High accuracy without the "Think" token overhead.

RQ1: Effectiveness (Performance)

Key Finding: Small beats Large

LiR^{3}AG with an **8B Non-reasoning model** consistently outperforms the **32B Reasoning model** (Vanilla RAG) across datasets.

- **HotpotQA:** 8B LiR^{3}AG achieves **0.521 F1** vs 0.501 F1 (32B Reasoning).
- **MuSiQue:** 8B LiR^{3}AG achieves **0.464 F1** vs 0.410 F1 (32B Reasoning).
- Demonstrates successful transfer of the reasoning strategy.

RQ2: Efficiency (Tokens & Time)

Token Reduction

- Reasoning models generate massive internal CoT tokens.
- LiR^{3}AG reduces average output token overhead by **98%**.
- Eliminates redundant "self-correction" loops.

Latency Improvement

- Faster end-to-end inference.
- LiR^{3}AG reduces inference time by **58.6%**.
- Makes complex reasoning viable for latency-sensitive applications.

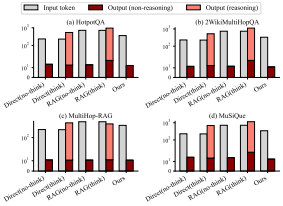

Figure: Token cost of methods on multi-hop QA datasets.

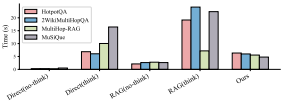

Figure: Time cost of methods on multi-hop QA datasets.

RQ3: Ablation Study

Which components matter?

1. **w/o Retriever:** Performance collapses (Knowledge unavailable).
2. **w/o Reranker:** Significant drop.
   - Without filtering/ordering, the Constructor struggles to build coherent chains.
3. **w/o Reasoning Constructor:**
   - Mere retrieval is insufficient; the explicit "step construction" is vital for the Generator.

Conclusion: Synergistic design is required for success.

Summary of Contributions

1. **Systematic Analysis:** First study to categorize reasoning model strategies in RAG (Context-Grounded vs. Knowledge-Reconciled).
2. **New Framework (LiR^{3}AG):** A lightweight approach that transfers reasoning strategies to smaller models via explicit Reranking and Construction.
3. **SOTA Performance:** Surpasses 32B Reasoning models using only 8B Non-reasoning models.
4. **Efficiency:** drastic reduction in tokens (98%) and time (58%).

Thank You

Paper: LiR^{3}AG: A Lightweight Rerank Reasoning Strategy Framework for RAG

https://github.com/WinstonCHEN1/LiR3AG

winstonccc726@gmail.com