UNIREL: RELATION-CENTRIC KNOWLEDGE GRAPH QUESTION ANSWERING WITH RL-TUNED LLM REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge Graph Question Answering (KGQA) has largely focused on *entity-centric* queries that return a single answer entity. However, many real-world questions are inherently relational, aiming to understand how entities are associated rather than which entity satisfies a query. In this work, we introduce *relation-centric* KGQA, a complementary setting in which the answer is a subgraph that represents the semantic relations among entities. The main challenge lies in the abundance of candidate subgraphs, where trivial or overly common connections often obscure the identification of unique and informative answers. To tackle this, we propose *UniRel*, a unified modular framework that combines a subgraph retriever with an LLM fine-tuned using reinforcement learning. The framework uses a reward function to prefer compact and specific subgraphs with informative relations and low-degree intermediate entities. Experiments show that UniRel not only produces more coherent and informative relational answers than existing baselines, but also generalizes effectively to unseen entities and relations. UniRel can also be applied to conventional entity-centric KGQA, achieving competitive or improved performance in several settings. A human evaluation further shows that participants prefer UniRel's answers over the baselines.

1 INTRODUCTION

Traditional knowledge graph question answering (KGQA) is largely *entity-centric*, aiming to return a single target entity. For example, a query such as "Who is Meghan Markle's husband's grandmother?" yields "Queen Elizabeth II" as the answer (see Figure 1(a)). Alternatively, users also often pose *relation-centric* queries that seek to uncover the relationships between entities rather than retrieve an individual fact. For example, "How are Meghan Markle and Queen Elizabeth II associated?" requires constructing a subgraph that captures their semantic connections. Such queries are common in real-world settings but remain beyond the scope of traditional KGQA systems.

Existing work in KGQA has primarily adhered to the entity-centric paradigm, often augmenting LLMs with structured knowledge to improve accuracy. Retrieval-augmented generation (RAG) incorporates KG facts as contextual input (Linders & Tomczak, 2025), while other approaches combine LLMs with graph neural networks (GNNs) for joint reasoning over text and structure (Xu et al., 2025; Yasunaga et al., 2021; He et al., 2024). To enhance interpretability, some methods generate explicit reasoning paths for multi-hop queries (Zhou et al., 2018; Chakraborty, 2024; Zhang et al., 2018), and others employ reinforcement learning (RL) or search strategies such as Monte Carlo Tree Search (MCTS) to explore candidate paths (Shen et al., 2025). Despite these advances, current systems ultimately return a single entity, leaving relation-centric queries unexplored.

In this work, we introduce *relation-centric* KGQA, a complementary setting to the standard *entity-centric* KGQA. Instead of producing an entity, the answer is a subgraph that captures the underlying relational structure among the seed entities. As demonstrated in Figure 1(a), multiple subgraphs may serve as answers. While Answer A1 is among the shortest valid answers, it reflects an overly common relation and conveys little information, whereas Answers A2 and A3 provide more unique and informative connections. This illustrates the central challenge of relation-centric KGQA: Among many candidate subgraphs, trivial or generic ones often obscure unique and informative answers.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

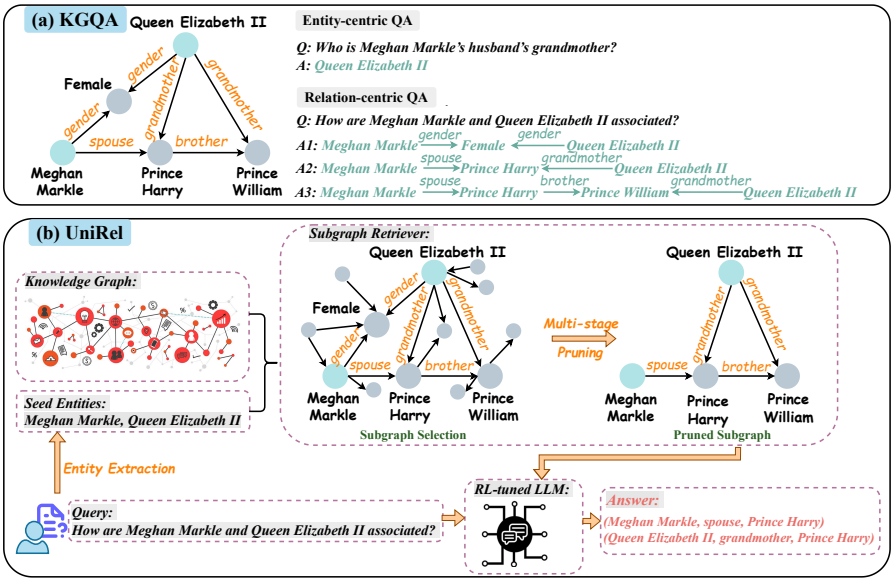

Figure 1: An overview of KGQA and our proposed UniRel framework. (a) An illustrative example contrasting entity-centric KGQA with relation-centric KGQA. (b) The UniRel pipeline, integrating subgraph retriever and RL-tuned LLM reasoning.

To address this challenge, we propose *UniRel*, a unified modular framework for KGQA, with a particular focus on relation-centric queries. UniRel integrates *subgraph retriever* and *RL-tuned LLM* reasoning into a coherent pipeline. Figure 1(b) illustrates the overall workflow. Starting from a natural language query, UniRel first identifies seed entities and then applies a subgraph retriever module that combines candidate selection with multi-stage pruning, producing a compact and task-relevant subgraph. The retrieved subgraph is textualized and provided alongside the query to an RL-tuned LLM, where the reward design encourages concise and informative outputs by favoring smaller structures with rarer relations and lower-degree intermediates. Overall, UniRel produces answers that are not only valid but also capture distinctive and meaningful relations.

We conduct extensive experiments across seven benchmark knowledge graphs and a diverse set of LLMs from the Qwen and Llama families, two widely adopted open-source model families. We build relation-centric query sets covering both two-entity and multi-entity settings, evaluated under in-domain and cross-domain scenarios. Results show that UniRel consistently outperforms existing baselines by retrieving more connected and informative relational subgraphs, while generalizing effectively to unseen entities and relations. Interestingly, Qwen models are substantially more sensitive to the removal of semantic information than Llama models, highlighting a fundamental difference in how these families leverage semantic cues. UniRel also extends to conventional entity-centric KGQA through an entity-centric retriever, achieving competitive performance compared to existing baselines. Beyond these reward-based metrics, a human evaluation shows that participants prefer UniRel's answers over the baselines without being told what our objective favors.

2 Problem Statement

In relation-centric KGQA, a knowledge graph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{T})$, where $\mathcal{V}$ is the set of entities, $\mathcal{R}$ the set of relation types, and $\mathcal{T} \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ the set of triples. Given a query q, let $\mathcal{V}_q \subseteq \mathcal{V}$ denote the set of seed entities extracted from q. The task is to construct a subgraph: $\mathcal{G}^* = (\mathcal{V}^*, \mathcal{R}^*, \mathcal{T}^*) \subseteq \mathcal{G}$. A subgraph $\mathcal{G}^*$ is considered *valid* if all entities in $\mathcal{V}_q$ are connected in the undirected projection of $\mathcal{G}^*$, with $\mathcal{V}_q \subseteq \mathcal{V}^*$, ensuring that the resulting answer is grounded in the semantics of $\mathcal{G}$. When multiple valid subgraphs exist, user preferences may vary depending on their background knowledge or task requirements. In this work, we prioritize subgraphs that are more *informative*, rather than those dominated by trivial or overly common structures. *Accordingly, our optimization objective balances compactness, relation rarity, and lower-degree intermediate entities*, ensuring that the resulting subgraphs emphasize non-trivial and distinctive semantic relations. Alternative optimization objectives can be defined, depending on the specific application scenario.

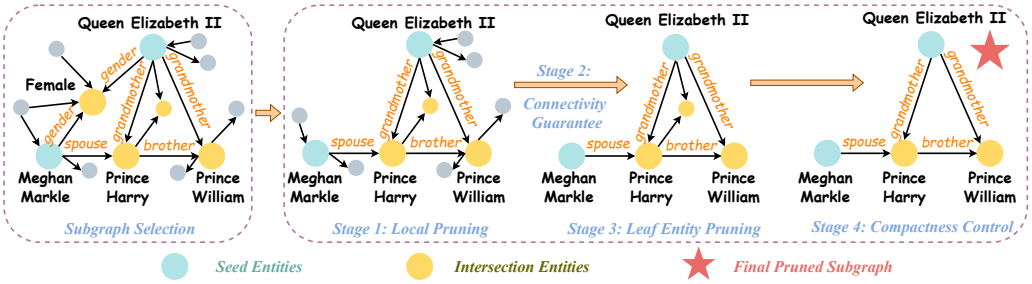

Figure 2: An example of multi-step graph pruning. From the initial subgraph (left), Stage 1 removes overly common entities (e.g., *Female*), Stage 2 enforces connectivity through intersections, Stage 3 prunes leaf entities, and Stage 4 enhances compactness by removing high-penalty intersection nodes (yellow). The final subgraph (right) retains the most informative relations among the seed entities.

3 Framework: UniRel

UniRel first builds a task-specific subgraph with a retriever that applies multi-stage pruning (Sec. 3.1.1 and 3.1.2), and then uses an RL-tuned LLM to generate answers that emphasize informative relations (Sec. 3.2). Changing the pruning and reward design adapts the framework to other objectives.

3.1 Subgraph Retriever

Subgraph retriever constructs a compact, task-relevant subgraph for downstream reasoning, consisting of subgraph selection and multi-step pruning. Details are provided in Appendix A.

3.1.1 Subgraph Selection

Given a query q, the seed entity set is extracted from q as $\mathcal{E}_q = \{e_1, e_2, \ldots, e_m\}$. For each seed entity $e_i \in \mathcal{E}_q$, we define its k-hop expansion as $\mathcal{V}^k(e_i) = \{v \in \mathcal{E} \mid \mathrm{dist}(e_i, v) \leq k\}, \mathcal{T}^k(e_i) = \{(u, r, v) \in \mathcal{T} \mid u, v \in \mathcal{V}^k(e_i)\}$, where $\mathrm{dist}(\cdot, \cdot)$ denotes the distance in the undirected projection of $\mathcal{G}$.

The candidate subgraph $\mathcal{G}' = (\mathcal{V}', \mathcal{R}', \mathcal{T}')$ is then obtained by aggregating the k-hop neighborhoods of all seed entities, where $\mathcal{V}' = \bigcup_{e_i \in \mathcal{E}_q} \mathcal{V}^k(e_i), \mathcal{T}' = \bigcup_{e_i \in \mathcal{E}_q} \mathcal{T}^k(e_i)$, and $\mathcal{R}' = \{r \in \mathcal{R} \mid (u, r, v) \in \mathcal{T}'\}$. The subgraph $\mathcal{G}'$ serves as the initial candidate, refined through pruning in the next stage.

3.1.2 Multi-step Graph Pruning

The candidate subgraph $\mathcal{G}' = (\mathcal{V}', \mathcal{R}', \mathcal{T}')$ often contains redundant structures and overly generic entities that obscure meaningful relational patterns. To mitigate this issue, we refine $\mathcal{G}'$ through a multi-stage pruning process. As a measure of entity generality, we introduce the *hub penalty*, defined for each $e \in \mathcal{E}$ as:

$$\mathrm{HubPenalty}(e) = \log\big(1 + \deg(e)\big), \tag{1}$$

where $\deg(e)$ denotes the degree of e in the undirected projection of $\mathcal{G}$. The penalty increases with entity degree, such that highly connected entities (e.g., *male*) receive larger values, while rarer and more specific entities (e.g., *president*) are penalized less. This formulation provides a criterion for filtering out uninformative entities and guiding the pruning of $\mathcal{G}'$.

Figure 2 shows the multi-stage pruning on the example in Figure 1, with each stage explained below.

Stage 1: Local Pruning. For each expansion set $\mathcal{V}^k(e_i)$, we remove every **non-seed** entity v with $\mathrm{HubPenalty}(v) \geq \rho$, together with its incident relations. We then keep only the connected component containing e_i, giving the pruned neighborhoods $\mathcal{V}^k_\rho(e_i)$ and $\mathcal{T}^k_\rho(e_i)$.

Stage 2: Connectivity Guarantee. Following local pruning, the filtered neighborhoods $\{\mathcal{V}^k_\rho(e_i)\}$ may fail to ensure connectivity among all seed entities. To evaluate connectivity, we compute pairwise intersections $\mathcal{I}_{ij} = \mathcal{V}^k_\rho(e_i) \cap \mathcal{V}^k_\rho(e_j)$ and construct an auxiliary graph $\mathcal{H}_\rho = (\{\mathcal{V}^k_\rho(e_i)\}, \{(i, j) \mid \mathcal{I}_{ij} \neq \emptyset\})$, where each node represents a pruned neighborhood $\mathcal{V}^k_\rho(e_i)$, and an edge (i, j) is introduced whenever their intersection is non-empty. If $\mathcal{H}_\rho$ is connected, then all seed entities are jointly connected in the candidate subgraph. Otherwise, ρ is incrementally relaxed by a fixed step $\Delta\rho$ (i.e., $\rho \leftarrow \rho + \Delta\rho$) and Stages 1–2 are repeated until graph $\mathcal{G}_\rho = (\mathcal{V}_\rho, \mathcal{R}_\rho, \mathcal{T}_\rho)$ becomes connected.

Stage 3: Leaf Entity Pruning. Once connectivity is ensured, we iteratively eliminate non-seed entities in $\mathcal{G}_\rho$ whose neighbor set $\mathcal{N}_\rho(v) = \{u \in \mathcal{V}_\rho \mid \exists r \in \mathcal{R}_\rho : (v, r, u) \in \mathcal{T}_\rho \text{ or } (u, r, v) \in \mathcal{T}_\rho\}$ contains only a single distinct element ($|\mathcal{N}_\rho(v)| = 1$). Such leaf-like entities and their incident relations are iteratively pruned until no removal is possible.

Stage 4: Compactness Control. If the refined subgraph $\mathcal{G}'_\rho$ remains large, we further enforce compactness by leveraging the intersections $\{\mathcal{I}_{ij}\}$ defined in Stage 2. Let $\mathcal{I}^+ = \{\mathcal{I}_{ij} \mid \mathcal{I}_{ij} \neq \emptyset\}$ denote the collection of all non-empty intersections. We then select a subset $\mathcal{I}^c \subseteq \mathcal{I}^+$ whose index pairs form a connected spanning subgraph over the seed indices, i.e., the auxiliary graph $\mathcal{H}^c = \big(\{0, 1, \ldots, m-1\},\ \{(i,j) \mid \mathcal{I}_{ij} \in \mathcal{I}^c\}\big)$ is connected. For each valid subset $\mathcal{I}^c$, a candidate subgraph is constructed in two steps. First, within each intersection $\mathcal{I}_{ij}$, we retain s entities with lowest hub penalties: $\widehat{\mathcal{I}}_{ij} = \arg\min_{\mathcal{U} \subseteq \mathcal{I}_{ij},\, |\mathcal{U}| = \min(s,\, |\mathcal{I}_{ij}|)} \sum_{v \in \mathcal{U}} \mathrm{HubPenalty}(v)$. Second, each selected set $\widehat{\mathcal{I}}_{ij}$ is expanded via a k-hop search restricted to the corresponding pruned neighborhoods: $\mathcal{N}^k(\widehat{\mathcal{I}}_{ij}) = \widehat{\mathcal{I}}_{ij} \cup \{u \in \mathcal{V}^k_\rho(e_i) \cup \mathcal{V}^k_\rho(e_j) \mid \mathrm{dist}(u, v) \leq k,\ v \in \widehat{\mathcal{I}}_{ij}\}$. The candidate node set associated with $\mathcal{I}^c$ is then $\mathcal{V}(\mathcal{I}^c) = \bigcup_{\mathcal{I}_{ij} \in \mathcal{I}^c} \mathcal{N}^k(\widehat{\mathcal{I}}_{ij})$, and its induced subgraph is then simplified using Stage 3. Finally, the most compact subgraph under parameters (s, ρ) is obtained as $\mathcal{G}^s_\rho = (\mathcal{V}^s_\rho, \mathcal{R}^s_\rho, \mathcal{T}^s_\rho)$, where $\mathcal{V}^s_\rho = \mathcal{V}\big(\arg\min_{\mathcal{I}^c \text{ valid}} |\mathcal{V}(\mathcal{I}^c)|\big)$, and $\mathcal{T}^s_\rho = \{(u, r, v) \in \mathcal{T}'_\rho \mid u, v \in \mathcal{V}^s_\rho\}$, $\mathcal{R}^s_\rho = \{r \in \mathcal{R}'_\rho \mid (u, r, v) \in \mathcal{T}^s_\rho\}$.

Stage 5: Iterative Reduction. If the subgraph still exceeds the target size, we decrease s and reapply Stage 4 until a sufficiently compact subgraph is obtained.

3.2 Relational Answer Generation via LLM

Given the refined subgraph $\mathcal{G}^s_\rho$, an RL-tuned LLM generates a relational answer conditioned on the query and the textualized subgraph, trained with the task-specific rewards below.

3.2.1 Reinforcement Learning Algorithms

RL with Verifiable Reward (RLVR). Our approach builds on the RLVR paradigm (Lambert et al., 2024), which uses a rule-based verifier that assigns a reward of 1 if a response passes a task-specific correctness check and 0 otherwise. This works well for tasks with clear success criteria such as math and code. We design a corresponding rule-based reward to verify generated relations (Sec. 3.2.2).

Group Relative Policy Optimization (GRPO). For policy optimization, we adopt GRPO (Shao et al., 2024), which evaluates responses based on relative performance within a sampled group from the policy. Given a query q, we sample from the previous policy $\pi_{\theta_{\text{old}}}(\cdot \mid q)$ to get G responses $\{o_i\}_{i=1}^G$ with rewards $\{r_i\}$. The advantage A_i is the group-normalized reward, defined as: $A_i = (r_i - \overline{r})/(\sigma_r + \varepsilon_{\text{stab}})$, where $\overline{r}$ and σ_r are the mean and standard deviation of the group rewards, and $\varepsilon_{\text{stab}}$ ensures numerical stability. With $\rho_i = \pi_\theta(o_i \mid q)/\pi_{\theta_{\text{old}}}(o_i \mid q)$, GRPO loss is defined as a clipped surrogate similar to PPO: $\mathcal{L}_{\text{GRPO}}(\theta) = -[\frac{1}{G}\sum_{i=1}^G \min\big(\rho_i A_i,\ \mathrm{clip}(\rho_i, 1-\alpha, 1+\alpha)\, A_i\big) - \beta\, D_{\text{KL}}\big(\pi_\theta(\cdot|q) \,\|\, \pi_{\text{ref}}(\cdot|q)\big)]$, which favors above-average responses while the KL term keeps π_θ close to the reference policy π_{ref}.

3.2.2 Reward Design

In relation-centric KGQA, a good answer must follow the format, connect the seed entities, and emphasize informative entities and relations. We therefore define a composite reward for each answer a with four components: *Format*, *Connectivity*, *Entity Informativeness*, and *Relation Informativeness*.

Format Reward. The format reward is defined as $R_{\text{fmt}}(a) \in \{-1, 1\}$, where $R_{\text{fmt}}(a) = 1$ when a conforms to the required format and $R_{\text{fmt}}(a) = -1$ otherwise.

Connectivity Reward. The connectivity reward evaluates whether the output subgraph connects the extracted entities $\mathcal{E}_q$ and is defined as $R_{\text{con}}(a) \in \{-\lfloor |\mathcal{E}_q|/2 \rfloor, \ldots, \lceil |\mathcal{E}_q|/2 \rceil - 1\}$, where the minimum indicates full disconnection, intermediate values reflect partial connectivity (e.g., $-\lfloor |\mathcal{E}_q|/2 \rfloor + l$ for $(l+1)$-connected entities), and the maximum denotes full connectivity.

Entity Informativeness Reward. This component favors informative over generic entities. For an answer a with entity set $\mathcal{E}(a)$, the reward is the sum of normalized hub penalties (Equation 1):

$$R_{\text{ent}}(a) = \sum\nolimits_{e \in \mathcal{E}(a)} R_{\text{ent}}(e), \quad R_{\text{ent}}(e) = -\frac{\mathrm{HubPenalty}(e)}{\max_{v \in \mathcal{E}} \mathrm{HubPenalty}(v)}. \tag{2}$$

Relation Informativeness Reward. This component favors infrequent relations, which are more informative. The informativeness of a relation r is measured by its inverse document frequency (IDF) (Robertson, 2004): $\mathrm{IDF}(r) = \log(|\mathcal{T}|/|\{(u, r, v) \in \mathcal{T}\}|)$, where $|\mathcal{T}|$ is the number of triples in $\mathcal{G}$ and the denominator is the frequency of r. Frequent relations (e.g., *gender*) receive lower IDF scores, while rarer ones (e.g., *invention*) score higher. Given an answer a with relation set $\mathcal{R}(a)$, the reward is the sum of normalized IDF scores:

$$R_{\mathrm{rel}}(a) = \sum\nolimits_{r \in \mathcal{R}(a)} R_{\mathrm{rel}}(r), \quad R_{\mathrm{rel}}(r) = \frac{\mathrm{IDF}(r)}{\max_{s \in \mathcal{R}} \mathrm{IDF}(s)} - 1. \tag{3}$$

Overall Reward. The final reward for an answer a aggregates all four components:

$$R(a) = R_{\mathrm{fmt}}(a) + R_{\mathrm{con}}(a) + \tfrac{1}{2}\left(\tfrac{R_{\mathrm{ent}}(a)}{x} + \tfrac{R_{\mathrm{rel}}(a)}{y}\right), \tag{4}$$

where x and y are normalization constants ensuring $R_{\mathrm{ent}}(a)/x, R_{\mathrm{rel}}(a)/y \in [-1, 0]$, with values determined by the maximum permitted subgraph size. By construction, the reward is bounded as a function of the number of seed entities $|\mathcal{E}_q|$: $R(a) \in [-\lfloor |\mathcal{E}_q|/2 \rfloor - 2, \lceil |\mathcal{E}_q|/2 \rceil)$. If $R_{\mathrm{fmt}}(a) = -1$, then $R(a) = -\lfloor |\mathcal{E}_q|/2 \rfloor - 2$. If $R_{\mathrm{con}}(a) = -\lfloor |\mathcal{E}_q|/2 \rfloor$, then $R(a) = -\lfloor |\mathcal{E}_q|/2 \rfloor$.

4 Experiment

4.1 Experiment Setup

Datasets and Query Construction. We evaluate on seven benchmark KG datasets, including Freebase13 (Socher et al., 2013), FB15k-237 (Toutanova & Chen, 2015), MetaQA (Zhang et al., 2018), DBpedia50/500 (Auer et al., 2007), YAGO3-10 (Mahdisoltani et al., 2015), and UMLS (Bodenreider, 2004). Since existing KGQA benchmarks are entity-centric, we construct 2,500 relation-centric queries per dataset (2,000 train and 500 test) using two-entity queries with $\mathrm{dist}(e_i, e_j) \leq 4$. We further construct three- and four-entity queries on DBpedia50 to evaluate scalability. Additional details are provided in Appendix B.2.

Models. To assess generalizability, we evaluated UniRel on LLMs from the Qwen (Yang et al., 2024) and Llama (Grattafiori et al., 2024) families, covering diverse architectures and scales. Specifically, we used Qwen-2.5-3B/7B/14B-Instruct, Llama-3.2-3B-Instruct, and Llama-3.1-8B-Instruct.

Prompts. We designed structured prompts that encode the textualized knowledge graph as node and edge tables and incorporate the query, requiring the model to output the corresponding subgraph in a standardized triple format. Prompt templates are provided in Appendix B.2.

Baselines. We compare UniRel with representative end-to-end KGQA pipelines, including SubgraphRAG (Li et al., 2025a) and GNN-RAG (Mavromatis & Karypis, 2024), which are compatible with large input graphs in our setting. Other methods do not transfer directly. Agentic search methods such as ToG (Sun et al., 2024) and PoG (Chen et al., 2024) list all neighbors of the current entity in the prompt at each hop, which is impractical when a hub such as *Male* has tens of thousands of neighbors. RoG (Luo et al., 2024) trains its planner on paths to a gold answer entity, which our setting does not have. We instead include applicable methods as retriever and reasoning ablations (Sec. 4.3).

Evaluation Metrics. We primarily evaluate performance using *Connectivity Ratio* (C) and *Average Reward* ($\bar{R}$). C measures whether the generated subgraph connects all seed entities, i.e., whether it forms a valid answer. $\bar{R}$ reflects how well the generated subgraph satisfies the preference criteria defined in our framework. **Notably, our setting is preference-based, where multiple valid subgraphs may exist and a single ground-truth answer is not well-defined. We therefore evaluate with a reward-defined objective rather than a single ground-truth answer, and validate this objective with a human evaluation in Section 5**. Nonetheless, for completeness, we report *Subgraph F1* as a supplementary metric, comparing the predicted subgraph with a reference subgraph obtained via exhaustive search under the same objective. Details are provided in Appendix B.2.

4.2 Main Results

Parameter Choice. We calibrate x and y once on DBpedia500 and fix $x = 7$ and $y = 6$ across all datasets and models, without dataset-specific tuning (Appendix B.3).

Connectivity Comparisons. Table 1 presents the performance of UniRel and the baselines across datasets. Since a valid relation-centric answer must connect the queried entities, we first examine

Table 1: Performance comparison between baselines and UniRel (connectivity ratio, average reward), where all outputs are format-valid.

Models \ Datasets		Freebase13	FB15k-237	MetaQA	DBpedia50	DBpedia500	YAGO3-10	UMLS
Qwen3B	SubgraphRAG	(23.8%, -0.65)	(1.0%, -0.99)	(1.0%, -0.99)	(6.4%, -0.88)	(0.2%, -0.99)	(1.8%, -0.98)	(5.0%, -0.92)
	GNN-RAG	(32.0%, -0.39)	(41.6%, -0.27)	(34.2%, -0.43)	(17.8%, -0.67)	(33.2%, -0.45)	(50.2%, -0.15)	(12.6%, -0.77)
	UniRel	(86.0%, 0.64)	(62.0%, -0.02)	(72.8%, 0.07)	(91.0%, 0.55)	(64.8%, -0.06)	(67.4%, -0.03)	(69.0%, 0.08)
Qwen7B	SubgraphRAG	(42.0%, -0.38)	(2.6%, -0.98)	(1.4%, -0.98)	(14.0%, -0.76)	(0.2%, -0.99)	(2.0%, -0.98)	(17.2%, -0.86)
	GNN-RAG	(32.6%, -0.38)	(42.1%, -0.27)	(42.6%, -0.21)	(24.6%, -0.56)	(35.6%, -0.42)	(52.4%, -0.12)	(13.4%, -0.75)
	UniRel	(93.0%, 0.75)	(70.6%, 0.15)	(79.0%, 0.15)	(93.6%, 0.59)	(72.6%, 0.05)	(76.4%, 0.09)	(86.2%, 0.24)
Qwen14B	SubgraphRAG	(41.6%, -0.39)	(5.8%, -0.95)	(1.6%, -0.98)	(25.0%, -0.58)	(0.4%, -0.99)	(2.4%, -0.98)	(29.6%, -0.83)
	GNN-RAG	(41.0%, -0.24)	(82.6%, 0.43)	(83.2%, 0.18)	(40.0%, -0.40)	(84.8%, 0.13)	(90.0%, 0.39)	(13.4%, -0.76)
	UniRel	(97.2%, 0.83)	(**83.6%**, **0.48**)	(**87.4%**, **0.22**)	(98.4%, 0.67)	(84.8%, 0.14)	(92.1%, 0.40)	(97.6%, 0.36)
Llama3B	SubgraphRAG	(31.4%, -0.54)	(0.6%, -0.99)	(1.0%, -0.99)	(7.8%, -0.85)	(0.2%, -0.99)	(3.6%, -0.95)	(19.8%, -0.80)
	GNN-RAG	(10.8%, -0.61)	(28.2%, -0.50)	(38.0%, -0.37)	(13.0%, -0.76)	(17.2%, -0.62)	(36.0%, -0.38)	(1.1%, -0.98)
	UniRel	(95.8%, 0.80)	(70.6%, 0.14)	(70.6%, 0.12)	(60.6%, 0.04)	(38.8%, -0.47)	(75.4%, 0.09)	(34.6%, -0.34)
Llama8B	SubgraphRAG	(40.4%, -0.40)	(4.6%, -0.97)	(2.2%, -0.97)	(13.6%, -0.66)	(0.6%, -0.99)	(4.4%, -0.95)	(27.0%, -0.78)
	GNN-RAG	(29.8%, -0.45)	(77.2%, 0.32)	(55.4%, -0.08)	(33.8%, -0.41)	(42.6%, -0.38)	(92.6%, 0.41)	(12.6%, -0.78)
	UniRel	(**98.6%**, **0.85**)	(82.6%, 0.47)	(82.6%, 0.20)	(**99.0%**, **0.67**)	(**85.0%**, **0.14**)	(**93.6%**, **0.45**)	(**98.8%**, **0.38**)
Optimal Reward		0.92	0.77	0.65	0.72	0.65	0.67	0.79

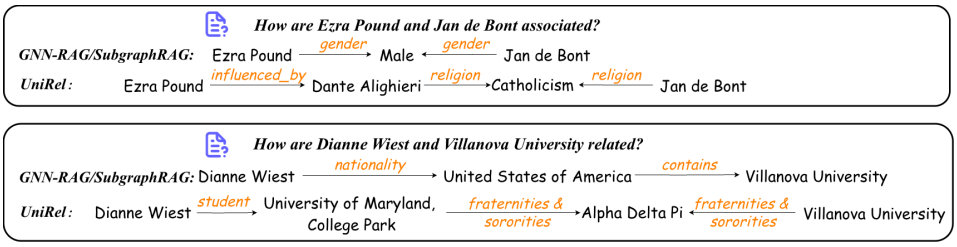

Figure 3: Case studies.

connectivity. UniRel matches or exceeds all baselines in every setting. In contrast, SubgraphRAG drops to as low as 0.2% because it selects triples individually by query relevance without a mechanism to connect the queried entities, often resulting in disconnected fragments.

Reward Comparisons. UniRel achieves substantially higher average reward than both baselines, reflecting a stronger preference for informative relations over trivial connections. Even when GNN-RAG connects the seeds, its path scoring favors short paths and ignores informativeness, so its output often passes through uninformative hubs such as *Male*. The case studies in Figure 3 show that UniRel instead finds meaningful intermediates (e.g., *influenced by Dante Alighieri*), since it prunes generic nodes and optimizes for informativeness (see Appendix B.3 for more examples).

Model Comparisons. Within both the Qwen and Llama families, larger models generally perform better, and UniRel further improves performance through task-specific optimization. Qwen-14B achieves the best results on FB15k-237 and MetaQA, while Llama-8B performs best on the remaining datasets. Performance on Freebase13 and DBpedia50 approaches the optimal reward, likely due to their relatively high connectivity. Additional details are provided in Appendix B.3.

4.3 Ablation Study

We conduct ablation studies on DBpedia50 with Llama-3.1-8B-Instruct to analyze the contribution of each component.

Retriever Ablation. All alternative retrievers, including GNN-based Retriever (Mavromatis & Karypis, 2024), PCST (G-Retriever) (He et al., 2024), and SubgraphRAG-Retriever (Li et al., 2025a), lead to substantial performance drops compared to UniRel (Table 2a), highlighting the importance of our retriever for constructing connected and informative relational subgraphs. Our retriever is also efficient, taking only milliseconds per query on a single CPU thread (Appendix A.5).

Reasoning Ablation. RL-based optimization is essential for preference-aware reasoning (Table 2b). Prompting conveys the output format but not the preference, and SFT-based RoE (Han et al., 2026) fails almost entirely. SFT on the reward-optimal subgraph $\mathcal{G}_{gt}$ performs much better, yet RL still improves both metrics. The advantage of RL over SFT is larger on denser subgraphs (Appendix B.4). A dense subgraph admits many valid answers, and SFT imitates only one of them, whereas RL rewards any answer that scores well.

Table 2: Ablation study on DBpedia50 with Llama-3.1-8B-Instruct. Each setting varies a single component while keeping all others fixed as in UniRel.

(a) Retriever Ablation

Variant	C	$\bar{R}$
GNN-based	36.2%	-0.43
PCST	1.6%	-0.98
SubgraphRAG	35.0%	-0.43
UniRel (Ours)	**99.0%**	**0.67**

(b) Reasoning Ablation

Variant	C	$\bar{R}$
Prompting	38.8%	-0.31
SFT-based RoE	1.4%	-0.99
SFT on $\mathcal{G}_{gt}$	93.0%	0.58
RL-tuned (Ours)	**99.0%**	**0.67**

(c) Reward Ablation

Variant	C	R_{ent}	R_{rel}
w/o R_{con}	92.8%	-0.32	-0.39
w/o R_{ent}	92.9%	-0.35	-0.41
w/o R_{rel}	98.0%	-0.29	-0.85
Full (Ours)	**99.0%**	**-0.29**	**-0.34**

Table 3: Cross-dataset generalization of UniRel trained on DBpedia500 (connectivity, average reward).

Datasets \ Models		Qwen3B	Qwen7B	Qwen14B	Llama3B	Llama8B
Freebase13	Original	(80.0%, 0.52)	(83.4%, 0.59)	(94.0%, 0.77)	(81.8%, 0.55)	(91.0%, 0.72)
	Modified	(69.6%, 0.33)	(74.8%, 0.43)	(84.0%, 0.60)	(81.6%, 0.55)	(87.0%, 0.65)
MetaQA	Original	(60.6%, -0.12)	(69.8%, 0.04)	(78.0%, 0.11)	(64.2%, -0.06)	(80.0%, 0.18)
	Modified	(29.2%, -0.56)	(49.0%, -0.21)	(60.8%, -0.02)	(67.2%, 0.06)	(72.6%, 0.15)
YAGO3-10	Original	(56.6%, -0.18)	(67.4%, 0.01)	(80.6%, 0.15)	(44.8%, -0.36)	(83.0%, 0.18)
	Modified	(23.6%, -0.66)	(41.4%, -0.34)	(48.2%, -0.24)	(44.6%, -0.31)	(62.2%, -0.03)

Reward Ablation. Each reward component contributes meaningfully to answer quality (Table 2c): removing R_{con} leads to disconnected subgraphs, while removing R_{ent} or R_{rel} reduces informativeness.

4.4 Generalization to Cross-Domain Setting

To examine transferability, we train on DBpedia500 and evaluate directly on the remaining datasets. Table 3 reports partial results, with full results in Appendix B.5.

Settings. *Original* keeps entity and relation names, while *Modified* replaces them with random identifiers ($\text{ENT}_1, \ldots, \text{ENT}_n$, $\text{REL}_1, \ldots, \text{REL}_m$), removing semantic cues but preserving the structure.

Cross-Domain Transfer. Under *Original*, models produce valid subgraphs for unseen entities and relations, substantially outperforming the baselines in Table 1 and often approaching in-domain UniRel performance across all metrics. This indicates effective cross-domain transfer.

Semantic vs. Structural Reliance. The *Modified* setting reveals a sharp split between model families. Qwen degrades drastically while Llama remains largely stable. On YAGO3-10, Qwen-3B drops by **58.3% (−33.0 points)** in C and **266.67% (−0.48 points)** in $\bar{R}$, whereas Llama-3B changes by only **0.45% (−0.2 points)** and **13.89% (+0.05 points)**, the latter a slight improvement. Llama-8B degrades more than Llama-3B but remains far more stable than Qwen, so Qwen is more sensitive to the removal of semantic information.

To probe this further, we replace a controlled fraction of entities and relations on MetaQA with random identifiers, where 25%, 50%, and 75% interpolate between *Original* (0%) and *Modified* (100%). As the ratio grows, Qwen degrades on both metrics, while Llama remains far more stable (Figure 4). Llama-3B fluctuates only mildly, and Llama-8B drops by only **9.25% (−7.4 points)** in C and **16.67% (−0.03 points)** in $\bar{R}$.

Together, these results suggest that Qwen relies on semantic regularities for generalization, whereas Llama draws more on structural connectivity to sustain stable performance.

4.5 Scalability to Multi-Entity Queries

To assess scalability, we extend the evaluation to three- and four-entity queries on DBpedia50, reporting pairwise connectivity (only two are connected) and full connectivity (all three are connected) for three entities, and pairwise, triple, and full connectivity for four (Table 4). UniRel achieves substantial full connectivity and average reward in both settings, and when full connectivity is not reached, the pairwise and triple ratios show that it still recovers partial structures. Across backbones, Llama is notably stronger than Qwen at small scale, and the gap closes at larger scale, where both approach the optimal reward on three-entity queries. These findings demonstrate the scalability of the framework to more complex multi-entity KGQA tasks.

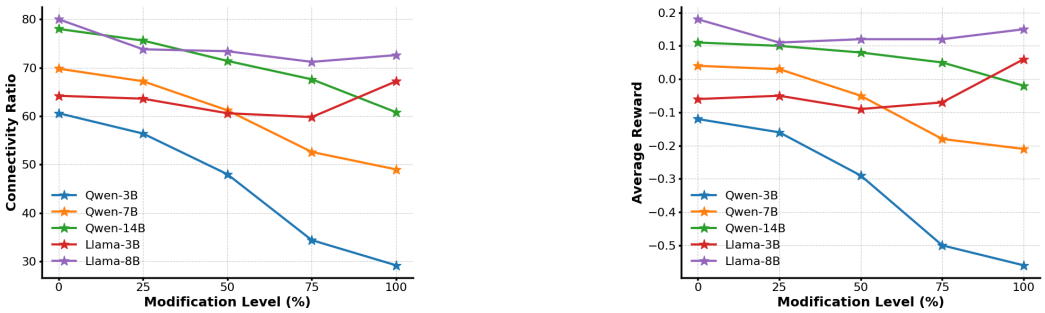

Figure 4: Comparison of connectivity ratio and average reward under different modification levels in MetaQA.

Table 4: Performance of UniRel on DBpedia50 across three-entity and four-entity queries.

Models	**Three Entities**			**Four Entities**			
	Full Conn.	Pairwise Conn.	Reward	Full Conn.	Triple Conn.	Pairwise Conn.	Reward
Qwen3B	69.2%	17.2%	1.17	8.2%	18.6%	50.8%	-0.29
Qwen7B	73.4%	18.0%	1.27	55.4%	16.4%	18.0%	0.78
Qwen14B	95.4%	2.0%	1.59	83.2%	6.1%	4.3%	1.26
Llama3B	90.8%	7.6%	1.57	48.4%	18.4%	24.2%	0.68
Llama8B	94.6%	4.8%	1.61	80.0%	5.6%	10.2%	1.26
Optimal Reward	1.74			1.63			

4.6 UniRel for Entity-Centric KGQA

UniRel adapts to entity-centric KGQA by replacing our subgraph retriever with an entity-centric one, while keeping the RL-tuned LLM unchanged. We use LLaMA2-7B-chat (Touvron et al., 2023) as the base LLM and evaluate on WebQSP (Yih et al., 2016) and CWQ (Talmor & Berant, 2018) with Hit@1 and F1 over the entities in the generated subgraph (Appendix B.7).

Table 5: Entity-centric KGQA results.

Retriever	Pipeline	WebQSP		CWQ	
		Hit@1	F1	Hit@1	F1
PCST (G-retriever) He et al. (2024)	Original	70.49	53.41	–	–
	UniRel	**70.50**	**58.90**	49.53	44.70
KAPING Retriever Baek et al. (2023)	Original	52.64	–	–	–
	UniRel	**72.00**	61.40	50.40	46.20
ROG Retriever Luo et al. (2024)	Original	**85.70**	70.80	**62.60**	56.20
	UniRel	80.00	**72.30**	59.20	**57.20**
GNN Retriever Mavromatis & Karypis (2024)	Original	**80.60**	**71.30**	61.70	59.40
	UniRel	80.20	71.10	**65.20**	**61.70**
GNN-RA Retriever Mavromatis & Karypis (2024)	Original	**82.80**	73.50	62.80	62.40
	UniRel	82.70	**77.10**	**66.50**	**63.70**

Table 5 compares each retriever's original pipeline with UniRel under the same retriever and base LLM, using the original numbers reported in prior work. UniRel improves F1 in most settings while keeping Hit@1 competitive. The gains are largest on WebQSP with the weaker PCST and KAPING retrievers, and on CWQ with the stronger GNN and GNN-RA retrievers, suggesting that relational reasoning helps most in complex multi-hop settings.

5 Human Evaluation of Relational Answers

Our automatic metrics all rely on the reward we optimize. To evaluate UniRel *independently* of this reward, we asked human participants, under approval from our university's IRB, to rank outputs from UniRel against baseline and ablation variants. Study materials, collected data, and analysis code are provided in the supplementary material.

Study Design. We recruited 50 participants via Prolific and retained 49 after an attention check. Each participant evaluated four randomly sampled relation-centric queries with four anonymized answers: UniRel, GNN-RAG/SubgraphRAG (identical outputs here), and the reward ablations w/o R_{rel} and w/o R_{ent} from Table 2c.

Task. Participants ranked the answers separately on *informativeness* (how much it reveals about the relationship), *coherence* (whether it forms a sensible chain), and *usefulness* (whether they would share it with someone who asked). The instructions explained what makes an answer valid, but not what makes one better.

Results. Table 6 reports how many participants ranked each system *first*. UniRel leads in all twelve query–criterion pairs and attains the best mean rank on every criterion, with 1.67 on informativeness, 1.96 on coherence, and 1.79 on usefulness, where 2.5 corresponds to random ordering.

Table 6: Number of participants (out of 49) ranking each system first, as informativeness/coherence/usefulness. Baseline: GNN-RAG/SubgraphRAG.

	UniRel	w/o R_{rel}	w/o R_{ent}	Baseline
Q1	**22/27/28**	18/8/12	6/8/8	3/6/1
Q2	**30/24/26**	7/4/4	3/4/6	9/17/13
Q3	**33/22/29**	7/17/11	8/4/7	1/6/2
Q4	**23/17/21**	13/12/13	6/5/5	7/15/10

UniRel is preferred over GNN-RAG/SubgraphRAG in 80%–92% of informativeness judgments and 67%–96% of usefulness judgments, significant on every query ($p \leq 0.011$). The gap on coherence is smaller (61%–80%, significant only on Q1), as short chains through generic entities such as *Male* read easily even when they say little. UniRel is also significantly preferred over w/o R_{ent} in eleven of twelve pairs and over w/o R_{rel} in seven, so both reward components matter to readers.

Asked what guided their choices, many participants said they preferred a specific connection over a shared generic attribute, although the instructions never mentioned degree, relation frequency, or compactness. Since the study covers four queries, this agreement holds on these queries rather than across all domains. More analysis is provided in Appendix C.

6 Related Work

Knowledge Graph Question Answering. Recent KGQA methods incorporate LLMs, either by retrieving relevant triples as context (Linders & Tomczak, 2025) or by combining LLMs with GNNs for joint reasoning (Xu et al., 2025; Yasunaga et al., 2021; He et al., 2024). For interpretability, other work generates explicit reasoning paths for multi-hop questions (Zhou et al., 2018; Chakraborty, 2024; Zhang et al., 2018), refines them with RL or search such as MCTS (Shen et al., 2025), or lets an LLM agent explore the graph hop by hop (Sun et al., 2024; Chen et al., 2024). Embedding-based methods instead capture multi-hop inference implicitly (Saxena et al., 2020; Shi et al., 2021). These methods take a question that names the relations to follow and return the entity that satisfies it, whereas relation-centric KGQA names no relation and asks how the given entities connect.

Improving LLM Reasoning via RLVR. RLVR has been applied to mathematical reasoning (Guo et al., 2025), code generation (Wang et al., 2025a), multi-modal reasoning (Huang et al., 2025b; Wang et al., 2025b; Li et al., 2025b), relation extraction (Dai et al., 2025b), GUI navigation (Shi et al., 2025), and parallel thinking (Zheng et al., 2025). Other work improves its training paradigm through self-play (Liu et al., 2025; Huang et al., 2025a) and test-time RL (Zuo et al., 2025), or its optimization algorithms (Yu et al., 2025; Yue et al., 2025; Dai et al., 2025a; Wang et al., 2025c; Zhou et al., 2025). We extend RLVR to relational subgraph generation, with a verifier that scores both validity and informativeness.

7 Conclusion & Discussion

In this paper, we introduced relation-centric KGQA, where the answer is a subgraph that captures how the given entities are associated. We proposed UniRel, which combines an algorithmic subgraph retriever with an RL-tuned LLM to produce compact, connected, and informative answers. Experiments across seven knowledge graphs and five backbones, together with a human evaluation, show that UniRel improves answer quality over existing baselines, generalizes to unseen entities, relations, and multi-entity queries, and transfers to entity-centric KGQA.

By returning explicit relational subgraphs, UniRel may support applications that need transparent reasoning, such as education, scientific exploration, and decision support. However, its answers may inherit gaps or biases from the underlying knowledge graph and reward design, which could lead users to over-trust them in sensitive settings.

Several limitations remain. UniRel depends on the coverage and quality of the knowledge graph, which may degrade performance in sparse or noisy domains. The reward encodes one shared notion of informativeness, while users may prefer different answers based on their background. Finally, the formulation targets "how" questions answerable by relational paths, not "why" questions that require causal context. Future directions include personalized rewards, uncertainty-aware reasoning, and more scalable retrieval for large and dynamic knowledge graphs.

AI USE STATEMENT

UniRel uses LLMs as a component of the method itself: Qwen and Llama models are fine-tuned as the answer generator, and an LLM extracts entity mentions in the entity-linking experiment. These are part of the research object rather than assistance in conducting the research.

Among tasks with required disclosure, we used ChatGPT only to simplify long Freebase relation names into short aliases, and all aliases were manually verified (Appendix B.2). We did not use generative AI tools for the other tasks with required disclosure.

Additionally, we used generative AI tools for tasks with recommended disclosure: formulating questions for the human evaluation, debugging code, editing the paper to improve readability, identifying relevant literature, formatting references, and suggesting a structure for the paper. All AI-assisted outputs were reviewed by the authors. The survey questions were checked by the authors before the study, debugged code was tested for correctness, and all cited literature was verified manually. We take responsibility for the final content of this work.

ETHICS STATEMENT

Human subjects. Our human evaluation was approved by our university's IRB. Participants were recruited through Prolific, gave informed consent, and were paid $2.00 for a task of about 8 minutes. We collected no personally identifying information, and the released responses are fully anonymized.

Data. All knowledge graphs used in this work are publicly available research datasets, and we use them under their original licenses. The relation-centric queries we construct are derived from these graphs and contain no private information beyond what the graphs already include.

REPRODUCIBILITY STATEMENT

The supplementary material contains our code, the constructed relation-centric query sets, and the human evaluation materials, collected responses, and analysis code. Section 3.1.1 and Section 3.1.2 specify the subgraph retriever, and Appendix A gives its complexity analysis and measured runtime. Section 3.2.2 defines every reward component. Appendix B.1 lists all training hyperparameters, hardware, and variance across random seeds. Appendix B.2 describes dataset processing, query construction, the prompt template, and how reference subgraphs for Subgraph F1 are computed. Appendix C describes the protocol of the human evaluation.

REFERENCES

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: a nucleus for a web of open data. In *Proceedings of the International Semantic Web Conference and the Asian Semantic Web Conference (ISWC/ASWC)*, pp. 722–735, 2007.

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136*, 2023.

Olivier Bodenreider. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(1):267–270, 2004.

Abir Chakraborty. Multi-hop question answering over knowledge graphs using large language models. *arXiv preprint arXiv:2404.19234*, 2024.

Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pp. 37665–37691, 2024.

Runpeng Dai, Linfeng Song, Haolin Liu, Zhenwen Liang, Dian Yu, Haitao Mi, Zhaopeng Tu, Rui Liu, Tong Zheng, Hongtu Zhu, et al. Cde: Curiosity-driven exploration for efficient reinforcement learning in large language models. *arXiv preprint arXiv:2509.09675*, 2025a.

Runpeng Dai, Tong Zheng, Run Yang, Kaixian Yu, and Hongtu Zhu. R1-re: Cross-domain relation extraction with rlvr. *arXiv preprint arXiv:2507.04642*, 2025b.

Stuart E Dreyfus and Robert A Wagner. The steiner problem in graphs. *Networks*, 1(3):195–207, 1971.

Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. Talk like a graph: Encoding graphs for large language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, pp. 43909–43934, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Haoyu Han, Kai Guo, Harry Shomer, Yu Wang, Yucheng Chu, Hang Li, Li Ma, and Jiliang Tang. Reasoning by exploration: A unified approach to retrieval and generation over graphs. In *Proceedings of the ACM Web Conference (WWW)*, pp. 2408–2418, 2026.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pp. 132876–132907, 2024.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-zero: Self-evolving reasoning llm from zero data. *arXiv preprint arXiv:2508.05004*, 2025a.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025b.

Nathan Lambert, Jacob Daniel Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, et al. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Mufei Li, Siqi Miao, and Pan Li. Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation. In *Proceedings of the Conference on Learning Representations (ICLR)*, 2025a.

Zongxia Li, Wenhao Yu, Chengsong Huang, Rui Liu, Zhenwen Liang, Fuxiao Liu, Jingxi Che, Dian Yu, Jordan Boyd-Graber, Haitao Mi, et al. Self-rewarding vision-language model via reasoning decomposition. *arXiv preprint arXiv:2508.19652*, 2025b.

Jasper Linders and Jakub M Tomczak. Knowledge graph-extended retrieval augmented generation for question answering. *arXiv preprint arXiv:2504.08893*, 2025.

Bo Liu, Leon Guertler, Simon Yu, Zichen Liu, Penghui Qi, Daniel Balcells, Mickel Liu, Cheston Tan, Weiyan Shi, Min Lin, et al. Spiral: Self-play on zero-sum games incentivizes reasoning via multi-agent multi-turn reinforcement learning. *arXiv preprint arXiv:2506.24119*, 2025.

L Luo, YF Li, G Haffari, and S Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *Proceedings of the Conference on Learning Representations (ICLR)*, 2024.

Farzaneh Mahdisoltani, Joanna Asia Biega, and Fabian M. Suchanek. Yago3: A knowledge base from multilingual wikipedias. In *Proceedings of the Conference on Innovative Data Systems Research (CIDR)*, 2015.

Costas Mavromatis and George Karypis. Gnn-rag: Graph neural retrieval for large language model reasoning. *arXiv preprint arXiv:2405.20139*, 2024.

Stephen Robertson. Understanding inverse document frequency: on theoretical arguments for idf. *Journal of documentation*, 60(5):503–520, 2004.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the annual meeting of the association for computational linguistics (ACL)*, pp. 4498–4507, 2020.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Tiesunlong Shen, Jin Wang, Xuejie Zhang, and Erik Cambria. Reasoning with trees: faithful question answering over knowledge graph. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pp. 3138–3157, 2025.

Jiaxin Shi, Shulin Cao, Lei Hou, Juanzi Li, and Hanwang Zhang. Transfernet: An effective and transparent framework for multi-hop question answering over relation graph. *arXiv preprint arXiv:2104.07302*, 2021.

Yucheng Shi, Wenhao Yu, Zaitang Li, Yonglin Wang, Hongming Zhang, Ninghao Liu, Haitao Mi, and Dong Yu. Mobilegui-rl: Advancing mobile gui agent through reinforcement learning in online environment. *arXiv preprint arXiv:2507.05720*, 2025.

Richard Socher, Danqi Chen, Christopher D. Manning, and A. Ng. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2013.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of EMNLP-IJCNLP*, pp. 2380–2390, 2019.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *Proceedings of International Conference on Learning Representations (ICLR)*, pp. 3868–3898, 2024.

Alon Talmor and Jonathan Berant. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*, 2018.

Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of Workshop on Continuous Vector Space Models and their Compositionality*, pp. 57–66, 2015.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Haozhe Wang, Long Li, Chao Qu, Fengming Zhu, Weidi Xu, Wei Chu, and Fangzhen Lin. To code or not to code? adaptive tool integration for math language models via expectation-maximization. *arXiv preprint arXiv:2502.00691*, 2025a.

Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhu Chen. Vl-rethinker: Incentivizing self-reflection of vision-language models with reinforcement learning. *arXiv preprint arXiv:2504.08837*, 2025b.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025c.

Derong Xu, Xinhang Li, Ziheng Zhang, Zhenxi Lin, Zhihong Zhu, Zhi Zheng, Xian Wu, Xiangyu Zhao, Tong Xu, and Enhong Chen. Harnessing large language models for knowledge graph question answering via adaptive multi-aspect retrieval-augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 25570–25578, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*, 2021.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 201–206, 2016.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 5773–5784, 2022.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander Smola, and Le Song. Variational reasoning for question answering with knowledge graph. In *Proceedings of the AAAI conference on artificial intelligence (AAAI)*, 2018.

Tong Zheng, Hongming Zhang, Wenhao Yu, Xiaoyang Wang, Xinyu Yang, Runpeng Dai, Rui Liu, Huiwen Bao, Chengsong Huang, Heng Huang, et al. Parallel-r1: Towards parallel thinking via reinforcement learning. *arXiv preprint arXiv:2509.07980*, 2025.

Mantong Zhou, Minlie Huang, and Xiaoyan Zhu. An interpretable reasoning network for multi-relation question answering. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pp. 2010–2022, 2018.

Yujun Zhou, Zhenwen Liang, Haolin Liu, Wenhao Yu, Kishan Panaganti, Linfeng Song, Dian Yu, Xiangliang Zhang, Haitao Mi, and Dong Yu. Evolving language models without labels: Majority drives selection, novelty promotes variation. *arXiv preprint arXiv:2509.15194*, 2025.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, et al. Ttrl: Test-time reinforcement learning. *arXiv preprint arXiv:2504.16084*, 2025.

A Complexity Analysis of the Subgraph Retriever

A.1 Subgraph Selection

Computing the k-hop neighborhood for a single seed e_i can be implemented via a truncated BFS on the undirected projection of the knowledge graph, which takes $O(|\mathcal{V}^k(e_i)| + |\mathcal{T}^k(e_i)|)$ time. Aggregating over all m seed entities, the total time complexity of subgraph selection is

$$O\left(\sum_{i=1}^{m} \left(|\mathcal{V}^k(e_i)| + |\mathcal{T}^k(e_i)|\right)\right).$$

Maintaining the union of nodes and edges using hash-based data structures incurs only linear overhead in the size of the resulting candidate graph and does not change the asymptotic complexity.

A.2 Multi-step Graph Pruning

Computing $\deg(v)$ for all $v \in \mathcal{V}'$ requires looking up each entity's global degree from a precomputed table over $\mathcal{G}$. Assuming $O(1)$ lookups (e.g., via a hash table built once during preprocessing), this takes $O(|\mathcal{V}'|)$ time.

Stage 1: Local Pruning. Thresholding nodes by ρ requires comparing each entity's precomputed global degree against ρ, taking $O(|\mathcal{V}'|)$ time. Removing incident edges can be implemented by filtering $\mathcal{T}'$ in one pass (keeping only edges whose endpoints survive), which costs $O(|\mathcal{T}'|)$. Finally, restricting the remaining graph to the connected component containing the seed e_i can be done by a single BFS/DFS from e_i in $O(|\mathcal{V}'| + |\mathcal{T}'|)$ time, which simultaneously discards isolated nodes and disconnected fragments. Overall, the time complexity of Stage 1 is

$$O(|\mathcal{V}'| + |\mathcal{T}'|).$$

Note that ρ only affects the size of the retained node/edge sets after pruning; the worst-case running time remains linear in $|\mathcal{V}'| + |\mathcal{T}'|$.

Stage 2: Connectivity Guarantee. Assume each pruned neighborhood is stored as a hash set. Then computing one intersection $\mathcal{I}_{ij}$ takes $O(\min\{|\mathcal{V}_\rho^k(e_i)|, |\mathcal{V}_\rho^k(e_j)|\})$ time by iterating over the smaller set and checking membership in the larger set. Thus, computing all pairwise intersections costs

$$O\left(\sum_{1 \le i < j \le m} \min\left\{|\mathcal{V}_\rho^k(e_i)|, |\mathcal{V}_\rho^k(e_j)|\right\}\right).$$

In the worst case, this is upper bounded by $O(m^2 \max_i |\mathcal{V}_\rho^k(e_i)|)$. Building $\mathcal{H}_\rho$ while checking whether $\mathcal{I}_{ij}$ is empty is included in the above cost, and checking connectivity of $\mathcal{H}_\rho$ via BFS/DFS takes $O(m + |E(\mathcal{H}_\rho)|) \le O(m^2)$. Let $N_k(\rho) = \max_i |\mathcal{V}_\rho^k(e_i)|$. Hence, the overall per-round time complexity of Stage 2 is dominated by

$$O(m^2 N_k(\rho)).$$

Let $R(\rho)$ be the number of relaxation rounds until $\mathcal{H}_\rho$ becomes connected. Since each round reruns Stages 1–2, we account for the Stage 1 rerun cost separately in the aggregation. The total time spent on Stage 2 across all rounds is

$$O\left(\sum_{t=1}^{R(\rho)} m^2 \, N_k(\rho_t)\right) \;\le\; O\big(m^2 R(\rho)\, N_k(\rho^\star)\big),$$

where $\rho^\star$ denotes the final threshold at which connectivity is restored.

Stage 3: Leaf Entity Pruning. This process can be implemented using a queue-based peeling procedure. We first compute the degree of each node in $O(|\mathcal{T}_\rho|)$ time, enqueue all non-seed nodes with degree one, i.e., with only one neighbor, and then iteratively remove them while updating the

degrees of their neighbors. Each node is enqueued and removed at most once, and each relation in $\mathcal{T}_\rho$ is deleted at most once. Therefore, the total time complexity of Stage 3 is

$$O(|\mathcal{V}_\rho| + |\mathcal{T}_\rho|)\,.$$

Stage 4: Compactness Control. Let $\mathcal{I}^+ = \{\mathcal{I}_{ij} \mid \mathcal{I}_{ij} \neq \emptyset\}$ denote the collection of non-empty intersections obtained in Stage 2, where $\mathcal{I}_{ij} = \mathcal{V}_\rho^k(e_i) \cap \mathcal{V}_\rho^k(e_j)$.

We select a single $\mathcal{I}^c \subseteq \mathcal{I}^+$ that forms a connected spanning subgraph over the seed indices via a greedy procedure: the elements of $\mathcal{I}^+$ are interpreted as edges of $\mathcal{H}_\rho$, sorted by $|\mathcal{I}_{ij}|$ in ascending order, and added one at a time using union-find until all m seed indices belong to a single connected component. This yields a spanning tree of $\mathcal{H}_\rho$ and runs in $O(m^2 \log m)$ time, since $|\mathcal{I}^+| \leq \binom{m}{2}$. We denote $\lambda = |\mathcal{I}^c| = m - 1$.

For each $\mathcal{I}_{ij} \in \mathcal{I}^c$, we retain a subset $\hat{\mathcal{I}}_{ij} \subseteq \mathcal{I}_{ij}$ of size $\min(s, |\mathcal{I}_{ij}|)$ corresponding to the entities with the lowest hub penalties. This operation can be implemented by partial selection (e.g., using a heap of size s), taking $O(|\mathcal{I}_{ij}| \log s)$ time. Since $N_k(\rho) = \max_i |\mathcal{V}_\rho^k(e_i)|$ and $|\mathcal{I}_{ij}| \leq N_k(\rho)$, the total cost of this step is

$$O\left(\sum_{\mathcal{I}_{ij} \in \mathcal{I}^c} |\mathcal{I}_{ij}| \log s\right) \leq O(\lambda N_k(\rho) \log s).$$

Next, for each retained set $\hat{\mathcal{I}}_{ij}$, we perform a truncated k-hop expansion restricted to the corresponding pruned neighborhoods $\mathcal{V}_\rho^k(e_i) \cup \mathcal{V}_\rho^k(e_j)$. This can be implemented as a multi-source BFS and visits each node and relation at most once within the restricted region. Let $T_k(\rho) = \max_i |\mathcal{T}_\rho^k(e_i)|$ denote the maximum number of relations in a pruned k-hop neighborhood. The cost is $O(N_k(\rho) + T_k(\rho))$ per intersection, and $O(\lambda(N_k(\rho) + T_k(\rho)))$ in total.

The induced subgraph over the expanded node set is then constructed and further simplified using Stage 3, both linear in the resulting subgraph size and not affecting the asymptotic complexity. Putting everything together, the overall time complexity of Stage 4 is

$$O\big(m^2 \log m + \lambda N_k(\rho) \log s + \lambda(N_k(\rho) + T_k(\rho))\big)\,,$$

where the first term accounts for the greedy selection of $\mathcal{I}^c$.

Stage 5: Iterative Reduction. Let $R(s)$ denote the number of reduction rounds, i.e., the number of times Stage 4 is reapplied with decreasing values of s. Since the greedy selection of $\mathcal{I}^c$ depends only on intersection sizes and is independent of s, it is computed once and reused across all reduction rounds; each round therefore re-executes only the partial selection of $\hat{\mathcal{I}}_{ij}$, the k-hop expansion, and the Stage 3 simplification. Using the per-round cost from Stage 4, the overall time complexity of Stage 5 is

$$O\left(m^2 \log m + \sum_{t=1}^{R(s)} \big(\lambda N_k(\rho) \log s_t + \lambda(N_k(\rho) + T_k(\rho))\big)\right),$$

where s_t denotes the value of s at the t-th reduction round, and the first term accounts for the one-time greedy selection.

In practice, s is initialized to a small constant and monotonically decreased, and $R(s)$ is typically very small. Therefore, the complexity of Stage 5 is dominated by a constant number of invocations of Stage 4 and reduces to

$$O\big(m^2 \log m + \lambda(N_k(\rho) + T_k(\rho))\big)$$

up to a small multiplicative factor.

A.3 Overall Complexity

We summarize the time complexity of the entire subgraph retrieval and pruning pipeline. Recall that $m = |\mathcal{V}_q|$ denotes the number of seed entities. We denote by $N_k = \max_i |\mathcal{V}^k(e_i)|$ and $T_k = \max_i |\mathcal{T}^k(e_i)|$ the maximum numbers of nodes and relations in an unpruned k-hop neighborhood,

and by $N_k(\rho) = \max_i |\mathcal{V}_\rho^k(e_i)|$ and $T_k(\rho) = \max_i |\mathcal{T}_\rho^k(e_i)|$ the corresponding maxima after pruning at threshold ρ. By construction, $N_k(\rho) \le N_k$ and $T_k(\rho) \le T_k$. Here, $R(\rho)$ denotes the number of threshold relaxation rounds in Stage 2, $\lambda = |\mathcal{I}^c|$ the number of selected intersections in Stage 4, and $R(s)$ the number of reduction rounds in Stage 5.

Aggregating the costs of all stages, the total time complexity is

$$T_{\text{total}} \;=\; O\Big(\sum_{i=1}^{m} \big(|\mathcal{V}^k(e_i)| + |\mathcal{T}^k(e_i)|\big) \;+\; R(\rho)\,(|\mathcal{V}'| + |\mathcal{T}'|) \;+\; m^2 R(\rho)\, N_k(\rho^*) \\ +\; m^2 \log m \;+\; R(s)(\lambda N_k(\rho^*) \log s + \lambda(N_k(\rho^*) + T_k(\rho^*))) \;+\; |\mathcal{V}'_{\rho^*}| + |\mathcal{T}'_{\rho^*}| \Big),$$

where ρ^* denotes the final threshold at which connectivity is restored. The terms correspond to subgraph selection, $R(\rho)$ reruns of Stage 1 within the relaxation loop, Stage 2 connectivity checks, the one-time greedy selection of $\mathcal{I}^c$ in Stage 4, the per-round cost of Stages 4–5, and Stage 3 simplifications.

We simplify this expression by identifying dominated terms. Using $\mathcal{V}' = \bigcup_{i=1}^{m} \mathcal{V}^k(e_i)$ (and analogously for $\mathcal{T}'$), the unpruned candidate sizes satisfy $|\mathcal{V}'| = O(mN_k)$ and $|\mathcal{T}'| = O(mT_k)$, so the subgraph-selection and Stage 1 rerun costs combine into

$$\sum_{i=1}^{m} \big(|\mathcal{V}^k(e_i)| + |\mathcal{T}^k(e_i)|\big) + R(\rho)\,(|\mathcal{V}'| + |\mathcal{T}'|) = O\big(R(\rho)\, m\, (N_k + T_k)\big).$$

The pruned linear terms $|\mathcal{V}'_{\rho^*}| + |\mathcal{T}'_{\rho^*}| \le |\mathcal{V}'| + |\mathcal{T}'|$ are dominated by the above.

Next, in Stages 4–5 the partial-selection term $\lambda N_k(\rho^*) \log s$ reduces to $O(\lambda N_k(\rho^*))$ since s is a small constant. The Stage 4 greedy-selection cost $m^2 \log m$ is dominated by the Stage 2 cost $m^2 R(\rho)\, N_k(\rho^*)$ since $N_k(\rho^*) \ge \log m$ in any non-trivial knowledge graph. Finally, $R(s)$ is typically a small constant, so Stage 5 contributes only a constant multiplicative factor over Stage 4.

Combining these observations and dropping dominated terms, the overall complexity is dominated by

$$T_{\text{total}} = O\big(R(\rho)\, m\, (N_k + T_k) \;+\; m^2 R(\rho)\, N_k(\rho^*) \;+\; R(s)\, \lambda\, (N_k(\rho^*) + T_k(\rho^*))\big)\,.$$

A.4 Discussion

A natural formulation for connecting multiple seed entities is Steiner tree problem. While Steiner-tree approaches are theoretically appealing, their computational characteristics and operational assumptions differ substantially from the requirements of query-driven reasoning on large knowledge graphs. Below we discuss key considerations motivating our design.

Computational complexity. Exact Steiner tree computation on general graphs is NP-hard and exhibits exponential dependence on the number of terminals m. Typical exact algorithms, such as the Dreyfus–Wagner dynamic program over subsets of terminals, have worst-case running time

$$O(3^m|\mathcal{V}| + 2^m(|\mathcal{T}| + |\mathcal{V}| \log |\mathcal{V}|)),$$

which quickly becomes prohibitive even for moderately sized multi-entity queries.

Approximate or heuristic Steiner methods avoid the exponential dependence on m, but their complexity is typically dominated by repeated shortest-path computations among terminals. For example, constructing a metric closure by running single-source shortest-path algorithms from each terminal followed by a minimum spanning tree incurs

$$O\big(m\,(|\mathcal{T}| + |\mathcal{V}| \log |\mathcal{V}|) + m^2 \log m\big)\,,$$

where the dominant term arises from m shortest-path searches on the (sub)graph.

In contrast, our approach avoids exponential dependence on m. As shown in our analysis, the dominant cost scales as

$$O\big(R(\rho)\, m\, (N_k + T_k) + R(\rho)\, m^2\, N_k(\rho^*) + R(s)\, \lambda\, (N_k(\rho^*) + T_k(\rho^*))\big)\,,$$

Table 7: Wall-clock time per query for the retriever, in milliseconds. Total is the sum of the stage timings, up to rounding.

Dataset	k	Total	Selection	Stage 1–2	Stage 3	Stage 4–5
Freebase13	2	29.3	16.7	12.1	<0.1	0.5
DBpedia50	2	6.3	3.7	1.4	<0.1	1.2
DBpedia50	3	19.1	9.2	3.3	0.3	6.3
DBpedia50	4	49.4	23.4	7.8	1.5	16.6

which is polynomial in m and bounded by the sizes of k-hop neighborhoods (both unpruned and pruned). Crucially, the dominant runtime depends on *local* quantities N_k and $N_k(\rho^*)$ rather than global graph statistics $|\mathcal{V}|, |\mathcal{T}|$, in contrast to Steiner-tree formulations whose shortest-path subroutines scale with the entire graph.

Locality of computation. A key distinction lies in the scope of computation. Steiner-based formulations rely on shortest-path searches whose cost depends on *global* graph structure ($|\mathcal{V}|, |\mathcal{T}|$). In contrast, all major operations in our framework—neighborhood expansion, pairwise intersections, connectivity checks, and compactness control—are confined to *k-hop local neighborhoods* of the seeds and their pruned counterparts. Consequently, the complexity is governed by N_k, T_k (unpruned) and $N_k(\rho^*)$, $T_k(\rho^*)$ (pruned) rather than global graph statistics. On real-world KGs, $N_k(\rho^*)$ is typically several orders of magnitude smaller than $|\mathcal{V}|$, yielding substantial efficiency gains in practice.

Controlled trade-off between connectivity and compactness. Our method exposes explicit parameters (ρ, k, λ, s) that directly regulate the trade-off between connectivity and compactness. The threshold ρ and the number of relaxation rounds $R(\rho)$ control how aggressively neighborhoods are pruned, while λ and s bound the number and size of retained intersections. This results in a predictable and tunable computational budget at query time. By contrast, Steiner-tree formulations do not provide similarly explicit query-level controls over the explored subgraph size.

Pairwise interactions versus global optimization. Although our approach involves a quadratic dependence on m due to pairwise interactions among seed entities, these interactions are limited to set intersections and bounded neighborhood expansions. This is fundamentally different from global optimization objectives in Steiner-tree formulations, where pairwise relationships are mediated through shortest-path computations over large graphs. As a result, the quadratic term in our method remains tractable in practice and does not introduce super-polynomial behavior.

A.5 Empirical Runtime

Table 7 shows the wall-clock time per query for the full retriever on a MacBook Pro (Apple M2 Pro, 16 GB), single-threaded, averaged over the test queries of each dataset, on Freebase13 with $k = 2$ and DBpedia50 with $k \in \{2, 3, 4\}$. Degree and IDF tables are precomputed once per KG and excluded from the timing. Stages 1–2 and Stages 4–5 are reported jointly, since each pair forms a loop.

- **Scaling with graph size.** At $k = 2$, moving from DBpedia50 to Freebase13 increases the KG from 30,449 entities and 43,756 triples to 75,043 and 375,514, roughly 8.6× more triples, while retrieval time grows only 4.6×, from 6.3 ms to 29.3 ms. Runtime thus grows more slowly than the graph, consistent with Appendix A.4, where the dominant cost is governed by local neighborhood sizes rather than by global graph statistics.
- **Scaling with k.** On DBpedia50, time increases from 6.3 ms to 19.1 ms to 49.4 ms as k goes from 2 to 4, with selection growing by a near-constant factor of 2.5 per hop rather than accelerating.

A.6 Retrieval Fidelity

Our retriever prunes the candidate subgraph heuristically, so it may remove part of the best answer before the LLM sees it. We measure this loss stage by stage. Retrieval-then-reasoning pipelines in entity-centric KGQA report retrieval quality separately in the same way, as answer coverage or retrieval recall (Sun et al., 2019; Zhang et al., 2022; Luo et al., 2024).

Table 8: Retrieval recall of the reward-optimal structure after each stage, and the reward gap after pruning.

Dataset	Selection	Stage 1–2	Stage 3	Stage 4–5	Reward gap
DBpedia50	100%	97.7%	97.3%	98.5%	0.041
Freebase13	100%	91.7%	91.5%	91.1%	0.045
MetaQA	100%	87.7%	83.4%	80.8%	0.220
YAGO3-10	100%	21.2%	18.7%	65.5%	0.185

Metrics. Entity-centric KGQA measures whether the retrieved subgraph contains the gold answer entity. Our setting has no single answer entity, so we apply the same idea at the level of triples. Let $\mathcal{T}_{gt}$ be the triples of the reward-optimal structure $\mathcal{G}_{gt}$, and $\mathcal{T}_s$ the triples of the subgraph surviving stage s. Retrieval recall at stage s, averaged over queries, is

$$\mathrm{Recall}_s = \frac{|\mathcal{T}_{gt} \cap \mathcal{T}_s|}{|\mathcal{T}_{gt}|}.$$

This is the recall term of Subgraph F1, computed at each stage rather than on the final answer. To measure what the loss costs, we define the reward gap as the reward of the optimal structure minus the best reward still attainable within the pruned subgraph $\mathcal{G}_{\text{pruned}}$, averaged over queries:

$$\mathrm{Gap} = R(\mathcal{G}_{gt}) - \max_{\mathcal{G} \subseteq \mathcal{G}_{\text{pruned}},\ \mathcal{G} \text{ valid}} R(\mathcal{G}).$$

Table 8 reports both on four datasets. Stages 1–2 and Stages 4–5 are reported jointly, since each pair forms a loop.

Where the loss occurs. Selection retains the optimal structure in full on all four datasets, so any loss comes from pruning rather than from the k-hop expansion.

Recall is not monotone across stages. Stage 4 does not simply continue from Stage 3. It rebuilds a subgraph from the Stage 2 intersections and re-expands within the pruned neighborhoods, so structure removed by leaf pruning can reappear. On YAGO3-10, this raises recall from 18.7% to 65.5%.

Fidelity differs across datasets. DBpedia50 and Freebase13 retain over 91% of the optimal structure, with a reward gap below 0.05. MetaQA and YAGO3-10 lose more, with gaps of 0.220 and 0.185. On YAGO3-10, most of the loss happens in Stage 1, where hub-penalty thresholding removes intermediate entities of the optimal structure.

Optimality. The exact objective is a Steiner-tree problem, so we make no optimality guarantee. The reward gap instead measures how far retrieval moves the system from the optimum. It bounds what any downstream generator can achieve, since no LLM can recover structure that pruning has already removed.

B Details of Experiments

B.1 Hyperparameters

Table 10 summarizes the key hyperparameters used in our experiments. We use 4× NVIDIA A100 80GB GPUs to train the 14B models, 2× NVIDIA A100 80GB GPUs to train the 7B/8B models, and 1× NVIDIA A100 80GB GPU to train the 3B models.

Training Stability and Variance. During training, the KL divergence from the reference policy increases in the early phase and then stabilizes within $[0.2, 0.35]$, rather than diverging as training proceeds. To measure variance across runs, we repeat training with five random seeds on FB15k-237 and MetaQA using Llama-3.1-8B-Instruct (Table 9). Both datasets are mid-range in connectivity rather than near-saturated, so ceiling effects do not understate the variance. Average reward varies by at most 0.007 and connectivity by at most 1.13 points across seeds.

B.2 Implementation Details

Datasets Table 11 summarizes the datasets used in our experiments, including their sources, number of entities, and number of triples. For FB15k-237, we replace machine identifiers

Table 9: Variance over five random seeds with Llama-3.1-8B-Instruct.

Dataset	C	$\bar{R}$
FB15k-237	82.90 ± 0.42	0.470 ± 0.007
MetaQA	82.80 ± 1.13	0.204 ± 0.005

Table 10: Hyperparameter settings used across all experiments.

Category	**Setting**
Data / Input Configuration	
Max Prompt Length	8192
Max Response Length	2048
Validation Batch Size	1024
Random Seed	1
Optimization	
Global Batch Size (Actor/Critic)	64
Learning Rate	1×10^{-6}
Weight Decay	1×10^{-2}
Optimizer	AdamW (bf16)
Max Gradient Norm	1.0
Rollout Configuration	
Number of Rollouts	5 (train), 1 (validation)
Rollout Temperature	1.0 (train), 0.6 (validation)
Rollout Top-p	1.0 (train), 0.95 (validation)
Max Batched Tokens	10240
KL Control	
KL Coefficient (λ_{KL})	1×10^{-2}
Training Setup	
Total Epochs	20 (2-entities), 40 (3-/4-entities)

with their corresponding human-readable entity names. To address the issue of long and complex relation names in Freebase, we transform each relation into a simplified alias with the assistance of ChatGPT, followed by manual verification. For example, the original relation `/film/actor/film./film/performance/film` is shortened to `acted_in`.

Query Construction For *two-entity* queries, we sampled pairs of seed entities such that the shortest path distance $\mathrm{dist}(e_i, e_j) \leq 4$ in the knowledge graph. This 4-hop cap is consistent with the maximum reasoning depth in widely-used entity-centric KGQA benchmarks (typically 1–3 hops). Beyond this distance, paths between entities tend to traverse loosely related intermediates and rarely correspond to coherent semantic associations, which would weaken the informativeness signal targeted by our reward design. A portion of our two-entity queries are converted from such entity-centric KGQA benchmarks, where the head entity and the gold answer naturally form a seed pair under relation-centric reformulation. The remaining queries are randomly sampled following Algorithm 1. The sampling procedure performs repeated random selection of a source entity and then selects a target entity reachable within k hops, ensuring that queries follow the graph's local connectivity.

To evaluate scalability, we further constructed queries involving *three* and *four* entities on the DBpedia50 dataset. For triplets (u, v, w), we required that the third entity w is within k hops of at least one of the first two entities (u *or* v). For quadruplets (u, v, w, z), we extended this requirement so that the fourth entity z is within k hops of at least one of $\{u, v, w\}$. This OR-connectivity guarantees that multi-entity queries yield semantically well-connected subgraphs.

After sampling seed entities, we converted them into natural language questions using a set of manually designed templates. For example, given the pair (*Meghan Markle*, *Queen Elizabeth II*), we generate the query: *"How are Meghan Markle and Queen Elizabeth II associated?"* We also used alternative phrasings such as *"For what reason are [entity1] and [entity2] related?"* to improve linguistic diversity.

Prompts For each experiment, we constructed prompts that contextualize the knowledge graph and the query into a unified text input. Specifically, the graph is represented as **node and edge tables**,

Table 11: Statistics of knowledge graph datasets.

Datasets	Source	#Entities	#Relations	#Triples
Freebase13	Freebase	75,043	13	375,514
FB15k-237	Freebase	14,265	237	310,116
MetaQA	Movie-domain	40,151	9	134,741
DBpedia50	Subset of DBpedia (Wikipedia)	30,449	365	43,756
DBpedia500	Larger subset of DBpedia	490,598	573	4,268,614
YAGO3-10	Subset of YAGO3 (Wikipedia, WordNet, GeoNames)	123,182	37	1,089,040
UMLS	Unified Medical Language System	135	46	6,529

Algorithm 1 Entity Pair Sampling

1: Randomly sample an entity u from the graph.
2: Run BFS up to k hops to obtain reachable set $\mathcal{N}(u, k)$.
3: Randomly sample $v \in \mathcal{N}(u, k)$ with $d(u, v) \leq k$.
4: Return (u, v) as the entity pair.

followed by the natural language query, together with task-specific instructions. This design ensures that the model has access to both structural and semantic information.

An example prompt is shown below.

```
You are given a directed graph as two CSV-like sections in this order:

1) Node table (header included):
node_id, node_attr

2) Edge table (header included):
src, edge_attr, dst

Task
- Use ONLY edges from the Edge table to answer the question by outputting
  a path.
- When printing each edge, replace IDs with the exact node_attr from the
  Node table.
- Output MUST be text triples, not numeric IDs.

Output format (STRICT | no extra text):
GRAPH:
("subject"|predicate|"object")
...
END

Rules
- Use only listed edges; do NOT invent edges.
- Map IDs → node_attr; preserve node_attr exactly.
- Output NOTHING outside the SUBGRAPH block.
- If no subgraph exists, output exactly:
GRAPH:
END

Graph:
node_id, node_attr
1, Meghan Markle
2, Prince Harry
3, Prince William
4, Queen Elizabeth II

src, edge_attr, dst
1, spouse, 2
4, grandmother, 2
4, grandmother, 3
2, brother, 3
```

Table 12: Statistics of subgraphs after subgraph selection.

	Freebase13	FB15k-237	MetaQA	DBpedia50	DBpedia500	YAGO3-10	UMLS
Average Number of Entities	54319.6	8576.3	2581.1	465.7	31110.4	51322.5	135.0
Average Number of Relations	266679.8	324601.9	7399.9	1239.8	256889.1	159730.5	13058.0

Table 13: Statistics of pruned subgraphs after multi-stage pruning.

	Freebase13	FB15k-237	MetaQA	DBpedia50	DBpedia500	YAGO3-10	UMLS
Average Number of Entities	6.4	17.9	16.6	15.4	24.2	30.2	10.2
Average Number of Relations	10.8	43.4	30.3	28.2	51.4	83.6	36.5

```
Question: How are Meghan Markle and Queen Elizabeth II associated?

Your output must be ONLY the SUBGRAPH block.
```

Statistics of Subgraphs We show the statistics on the size of the initial (Table 12)and pruned subgraphs (Table 13).

Additional Evaluation Metrics Besides Connectivity Ratio and Average Reward, we report *Subgraph F1* as a supplementary metric, computed as the triple-level overlap between the predicted subgraph and a ground-truth reference:

$$\text{Precision} = \frac{|\mathcal{T}_p \cap \mathcal{T}_{gt}|}{|\mathcal{T}_p|}, \quad \text{Recall} = \frac{|\mathcal{T}_p \cap \mathcal{T}_{gt}|}{|\mathcal{T}_{gt}|}, \quad \text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{5}$$

where $\mathcal{T}_p$ and $\mathcal{T}_{gt}$ denote the predicted and ground-truth triple sets.

We construct the ground-truth subgraph $\mathcal{G}_{gt}$ for each query as the connected subgraph that maximizes the reward objective (Eq. 4) while connecting all seed entities. The construction depends on the number of seeds:

- *Two-entity queries*: We enumerate all simple paths of length $\leq k$ between the two seeds via a bounded BFS, score each path with the reward function, and select the one achieving the highest reward.
- *Multi-entity queries* ($m \geq 3$): We compute the exact Steiner tree connecting all seed entities using the Dreyfus–Wagner dynamic programming algorithm (Dreyfus & Wagner, 1971), which is tractable for the small numbers of seeds in our setting ($m \leq 4$). The resulting tree is then refined by reward-guided local search (greedy addition/removal of incident nodes) until no further reward improvement is possible.

Example 1 *Consider the query "How are Meghan Markle and Queen Elizabeth II associated?" Three candidate paths exist between the two seeds:*

- $P_1 = \{$*(Meghan Markle, gender, Female), (Queen Elizabeth II, gender, Female)*$\}$ *(2 triples)*
- $P_2 = \{$*(Meghan Markle, spouse, Prince Harry), (Queen Elizabeth II, grandmother, Prince Harry)*$\}$ *(2 triples)*
- $P_3 = \{$*(Meghan Markle, spouse, Prince Harry), (Prince Harry, brother, Prince William), (Queen Elizabeth II, grandmother, Prince William)*$\}$ *(3 triples)*

P_2 achieves the highest reward, because (i) Prince Harry has a lower hub penalty than the highly-connected Female; (ii) spouse and grandmother have higher IDF than the frequent gender; (iii) P_2 is more compact than P_3. Thus $\mathcal{T}_{gt} = P_2$.

We then compute Subgraph F1 against $\mathcal{T}_{gt}$ for each candidate prediction:

- *If the model predicts $\mathcal{T}_p = P_1$, then $|\mathcal{T}_p \cap \mathcal{T}_{gt}| = 0$, giving* $\text{Precision} = \text{Recall} = \text{F1} = 0$.
- *If the model predicts $\mathcal{T}_p = P_2$, the prediction matches $\mathcal{T}_{gt}$ exactly, giving* $\text{F1} = 1$.
- *If the model predicts $\mathcal{T}_p = P_3$, the intersection is the single triple (Meghan Markle, spouse, Prince Harry), so $|\mathcal{T}_p \cap \mathcal{T}_{gt}| = 1$, $|\mathcal{T}_p| = 3$, $|\mathcal{T}_{gt}| = 2$, yielding* $\text{Precision} = 1/3$, $\text{Recall} = 1/2$, *and* $\text{F1} = 0.40$.

Table 14: Prompt template variants on DBpedia50 with Llama-3.1-8B-Instruct.

Template	What changes	F	C	$\bar{R}$
Original (node/edge tables)	–	100%	99.0%	0.67
Instruction rewrite	Task description only	98.6%	90.0%	0.48
Edge list	Graph encoding	68.4%	60.8%	-0.68
Natural language	Graph encoding	12.2%	5.5%	-2.60

Table 15: Evaluation results of Qwen-2.5-3B-Instruct model on DBpedia500 under different parameter settings. Only reachable cases are reported for I_R and I_E.

x	y	F	C	I_R	I_E
5	4	100.0%	59.0%	-5.12	-5.84
6	5	100.0%	60.6%	-5.62	-6.43
7	6	100.0%	64.8%	-5.83	-7.24
8	7	100.0%	64.2%	-6.39	-7.95

Prompt Sensitivity Prompting is used only in answer generation (Sec. 3.2). The retriever is purely algorithmic and involves no language model. To test how the RL-tuned policy responds to prompt changes, we evaluate three alternative templates against the original on DBpedia50 with Llama-3.1-8B-Instruct in Table 14:

- *Instruction rewrite* paraphrases and reorders the task description, leaving the graph encoding unchanged.
- *Edge list* replaces the node and edge tables with one line per triple, removing the numeric identifiers and the ID-to-name mapping step.
- *Natural language* states each triple as a sentence.

All three keep the required output format unchanged. All variants are evaluated on the same RL-tuned checkpoint without retraining, so the results measure how the deployed model responds to a changed prompt. We additionally report format validity (F), the fraction of outputs that parse into the required triple format.

Under instruction rewriting, format validity stays high at 98.6% while connectivity drops by 9 points, so the policy tolerates changes in wording. Prompt wording alone is known to shift performance on graph tasks. With the graph encoding held fixed, edge-existence accuracy moves from 42.8% to 60.8% under a rephrased question (Fatemi et al., 2024).

The last two variants change how the graph itself is written, which departs from the format seen during training. Format validity falls to 68.4% and 12.2%, and connectivity decreases accordingly. Since the policy is fine-tuned on one encoding, these results reflect specialization rather than instability. Our claim is limited to improved generation under the encoding used for training, and retraining on a different encoding requires no change to the framework.

B.3 Details of Main Results

Parameter Choice. We calibrate the normalization terms x and y in Eq. 4 on DBpedia500 using Qwen-2.5-3B-Instruct. Since two-entity queries satisfy $\mathrm{dist}(e_i, e_j) \leq 4$, an optimal subgraph typically contains about 5 entities and 4 relations, motivating the initial choice $(x, y) = (5, 4)$. Table 15 reports I_E and I_R as the un-normalized entity and relation reward sums (Eq. 4, before division by x, y); higher values indicate more informative content.

Increasing (x, y) trades informativeness for connectivity. The step $(5, 4) \rightarrow (6, 5)$ gains 1.6% connectivity at an informativeness loss of 1.09 ($I_R : -0.50$, $I_E : -0.59$). The next step $(6, 5) \rightarrow (7, 6)$ gains 4.2% connectivity at a comparable loss of 1.02 ($I_R : -0.21$, $I_E : -0.81$), making $(7, 6)$ a strictly better trade-off. Further increasing to $(8, 7)$ no longer improves connectivity. We therefore fix $(x, y) = (7, 6)$ for all datasets and models with two-entity queries.

Table 16: Subgraph F1 (%) comparison between baselines and UniRel.

Models \ Datasets		Freebase13	FB15k-237	MetaQA	DBpedia50	DBpedia500	YAGO3-10	UMLS
Qwen3B	SubgraphRAG	18.4	1.0	0.7	4.5	0.6	2.5	2.6
	GNN-RAG	14.4	13.3	16.7	7.9	13.9	28.2	5.8
	UniRel	58.7	15.8	25.0	39.2	14.3	29.8	17.1
Qwen7B	SubgraphRAG	38.6	5.8	4.1	11.7	5.5	7.2	16.0
	GNN-RAG	14.6	8.0	10.2	21.1	20.6	27.2	5.3
	UniRel	60.2	18.3	26.8	41.0	25.9	30.7	17.6
Qwen14B	SubgraphRAG	34.3	8.8	8.8	18.1	9.4	5.7	21.8
	GNN-RAG	15.2	16.5	29.2	28.5	21.3	30.8	4.1
	UniRel	63.2	19.1	**30.1**	**43.8**	**22.7**	**37.9**	26.7
Llama3B	SubgraphRAG	30.2	0.7	1.1	6.0	0.4	2.1	16.3
	GNN-RAG	0.0	8.0	17.6	20.7	9.0	26.1	0.0
	UniRel	60.8	12.2	26.7	26.7	12.8	26.3	18.4
Llama8B	SubgraphRAG	37.9	9.8	5.5	22.1	9.3	19.3	24.7
	GNN-RAG	12.2	16.4	21.0	32.8	13.3	30.3	4.8
	UniRel	**65.2**	**20.1**	27.0	42.6	17.9	30.8	**27.8**

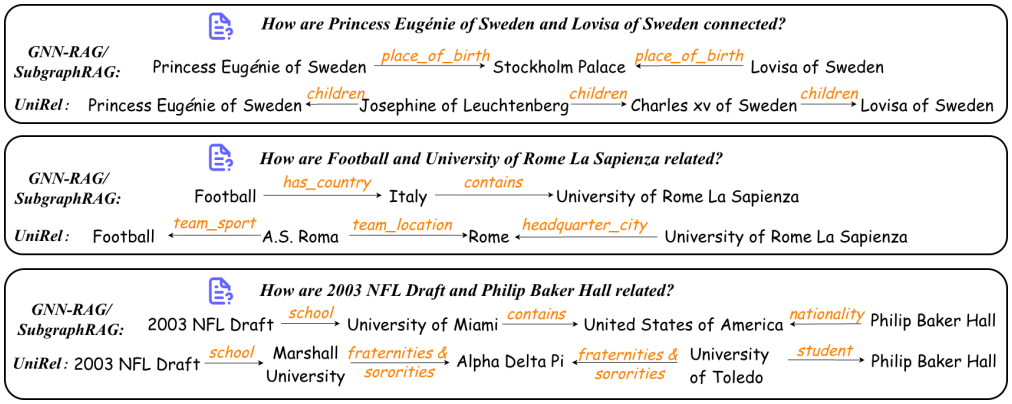

Figure 5: More case study.

Performance Comparison across Models and Datasets. Table 16 reports Subgraph F1 across the two baselines and UniRel. UniRel achieves the highest F1 in nearly all model-dataset combinations, consistently outperforming both SubgraphRAG and GNN-RAG by substantial margins. Among the baselines, GNN-RAG generally outperforms SubgraphRAG, consistent with our connectivity findings: SubgraphRAG's triple-level retrieval frequently yields disconnected fragments that directly lower its F1 against the connected ground-truth subgraph. UniRel's advantage further widens with model size, indicating that the benefits of hub-penalty pruning and reward-driven training compound with stronger LLM backbones.

Case Study. We present three additional case studies to further illustrate the differences between the baselines and UniRel.

B.4 Details of Ablation Study

Comparison with ICL and SFT. We compare RL with two alternatives that use the same retrieved subgraphs and the same backbone, Llama-3.1-8B-Instruct. *ICL* places three demonstrations of (subgraph, question, answer) in the prompt with no parameter updates. *SFT* performs behavior cloning on the reward-optimal subgraph $\mathcal{G}_{gt}$ of each training query, constructed by exact search. Table 17 reports the results on all seven datasets. The last row gives the density of the pruned subgraph, defined as the average number of relations divided by the average number of entities in Table 13. A higher density means more edges per node, so more distinct paths connect any given pair of seeds.

Table 17: Comparison of ICL, SFT, and RL on Llama-3.1-8B-Instruct, reported as (connectivity, average reward).

Method	Freebase13	FB15k-237	MetaQA	DBpedia50	DBpedia500	YAGO3-10	UMLS
ICL	(60.0%, 0.13)	(5.2%, -0.93)	(14.6%, -0.84)	(40.8%, -0.30)	(2.2%, -0.98)	(27.6%, -0.60)	(33.4%, -0.38)
SFT	(89.6%, 0.68)	(9.2%, -0.90)	(79.8%, 0.17)	(93.0%, 0.58)	(65.0%, -0.10)	(58.2%, -0.24)	(40.8%, -0.27)
RL (ours)	(**98.6%**, **0.85**)	(**82.6%**, **0.47**)	(**82.6%**, **0.20**)	(**99.0%**, **0.67**)	(**85.0%**, **0.14**)	(**93.6%**, **0.45**)	(**98.8%**, **0.38**)
Density	1.69	2.42	1.83	1.83	2.12	2.77	3.58

Table 18: Training dynamics on UMLS with Llama-3.1-8B-Instruct.

	F	C	R_{ent}	R_{rel}	$\bar{R}$
Phase 1	100%	33.6%	-0.73	-0.70	-0.38
Phase 2	100%	98.8%	-0.95	-0.43	0.30
Final	100%	98.8%	-0.88	-0.34	0.38

RL is best on all seven datasets and on both metrics, followed by SFT and then ICL. ICL is weak throughout. It attains negative average reward on six of seven datasets. Demonstrations in context convey the output format but not the preference the task requires.

SFT is a stronger baseline, but RL improves on it everywhere, and the margin depends on density. On the three sparsest datasets (Freebase13, MetaQA, and DBpedia50), SFT already recovers most of the performance, and RL adds at most 9 points of connectivity and 0.17 of reward. On the four denser ones, the gap widens to 20–73 points of connectivity and 0.24–1.37 of reward. A dense subgraph admits many valid answers. Cloning one fixed target gives no signal about which of them to prefer, whereas RL scores whatever the policy produces.

Training Dynamics. Table 18 tracks RL training on UMLS with Llama-3.1-8B-Instruct. Connectivity rises from 33.6% to 98.8% in the first part of training and then saturates. After that, both informativeness terms improve, with R_{ent} recovering from -0.95 to -0.88 and R_{rel} improving from -0.43 to -0.34. RL first learns to produce a valid answer at all, and then learns which valid answer is better. The second part is what a fixed target cannot supply.

B.5 Details of Generalization Results

Table 19 reports the complete cross-dataset generalization results of UniRel trained on DBpedia500 and evaluated on all benchmarks. Beyond the partial results discussed in the main paper, we provide additional analysis to better understand the observed performance variations across datasets.

We observe that generalization performance is relatively strong on DBpedia50, which may be attributed to its high similarity to the training dataset, DBpedia500, in terms of data source, schema, and graph structure. Both datasets are derived from DBpedia and share similar entity types, relation distributions, and structural patterns, which likely facilitates knowledge transfer under the UniRel framework.

In contrast, generalization performance on UMLS is comparatively weaker. A possible explanation is the substantial semantic gap between UMLS and DBpedia-based datasets. UMLS focuses on biomedical concepts and relations that differ significantly from the general-domain entities and relations seen during training. As a result, semantic similarity learned during pretraining and fine-tuning may be less effective in supporting cross-dataset transfer, limiting the benefits of semantic generalization.

Taken together, these observations suggest that cross-dataset generalization under UniRel is influenced by both semantic similarity and structural compatibility between the training and target knowledge graphs. Datasets that are closer to the training domain in terms of semantics and structure tend to exhibit stronger transfer performance, while datasets with larger semantic shifts pose greater challenges.

Table 19: Cross-dataset generalization of UniRel trained on DBpedia500, reported as (connectivity ratio, average reward, subgraph F1).

Datasets \ Models		Qwen3B	Qwen7B	Qwen14B	Llama3B	Llama8B
Freebase13	Original	(80.0%, 0.52, 44.7)	(83.4%, 0.59, 46.1)	(94.0%, 0.77, 52.7)	(81.8%, 0.55, 46.9)	(91.0%, 0.72, 54.1)
	Modified	(69.6%, 0.33, 41.3)	(74.8%, 0.43, 44.2)	(84.0%, 0.60, 50.4)	(81.6%, 0.55, 46.9)	(87.0%, 0.65, 51.2)
FB15k-237	Original	(47.0%, -0.23, 1.8)	(61.6%, 0.01, 1.7)	(76.6%, 0.16, 3.1)	(45.6%, -0.28, 1.6)	(69.8%, 0.08, 2.8)
	Modified	(20.8%, -0.71, 0.4)	(41.4%, -0.32, 1.2)	(52.4%, -0.12, 1.8)	(45.8%, -0.25, 1.7)	(59.4%, -0.03, 1.9)
MetaQA	Original	(60.6%, -0.12, 20.0)	(69.8%, 0.04, 21.7)	(78.0%, 0.11, 23.1)	(64.2%, -0.06, 21.7)	(80.0%, 0.18, 23.9)
	Modified	(29.2%, -0.56, 9.1)	(49.0%, -0.21, 16.5)	(60.8%, -0.02, 18.4)	(67.2%, 0.06, 22.5)	(72.6%, 0.15, 23.2)
DBpedia50	Original	(91.4%, 0.55, 40.1)	(96.6%, 0.64, 42.1)	(97.6%, 0.66, 42.3)	(56.8%, -0.02, 34.8)	(98.2%, 0.66, 42.5)
	Modified	(54.6%, -0.06, 34.0)	(73.0%, 0.27, 37.9)	(84.0%, 0.45, 40.7)	(84.6%, 0.45, 38.8)	(91.8%, 0.57, 41.7)
YAGO3-10	Original	(56.6%, -0.18, 11.5)	(67.4%, 0.01, 13.2)	(80.6%, 0.15, 15.0)	(44.8%, -0.36, 10.1)	(83.0%, 0.18, 15.8)
	Modified	(23.6%, -0.66, 7.23)	(41.4%, -0.34, 10.0)	(48.2%, -0.24, 11.4)	(44.6%, -0.31, 10.5)	(62.2%, -0.03, 13.8)
UMLS	Original	(49.0%, -0.19, 13.3)	(55.8%, -0.07, 13.6)	(74.0%, 0.10, 12.7)	(56.4%, -0.12, 13.4)	(66.6%, 0.05, 13.1)
	Modified	(41.0%, -0.30, 13.1)	(43.4%, -0.24, 13.4)	(50.0%, -0.14, 12.1)	(49.4%, -0.17, 13.6)	(55.0%, -0.07, 12.4)

Table 20: Performance of UniRel on DBpedia50 across three-entity and four-entity queries.

Models		Three Entities				Four Entities				
		Full Conn.	Pairwise Conn.	Reward	F1	Full Conn.	Triple Conn.	Pairwise Conn.	Reward	F1
Qwen3B	UniRel	69.2%	17.2%	1.17	44.6	8.2%	18.6%	50.8%	-0.29	35.2
Qwen7B	UniRel	73.4%	18.0%	1.27	45.5	55.4%	16.4%	18.0%	0.78	43.7
Qwen14B	UniRel	95.4%	2.0%	1.59	50.0	83.2%	6.1%	4.3%	1.26	46.9
Llama3B	UniRel	90.8%	7.6%	1.57	50.2	48.4%	18.4%	24.2%	0.68	43.9
Llama8B	UniRel	94.6%	4.8%	1.61	51.1	80.0%	5.6%	10.2%	1.26	46.8
Optimal Reward		1.74				1.63				

B.6 Scalability

We provide the detailed performance include SubgraphF1 in Table 20.

B.7 Details of UniRel for Entity-Centric KGQA

Datasets. We evaluate UniRel on two widely used multi-hop KGQA benchmarks, WebQSP and CWQ as shown in Table 21. Both datasets are built on Freebase and require reasoning over multiple entities and relations, making them well suited for evaluating relation-centric and subgraph-based reasoning methods.

WebQSP focuses on relatively constrained multi-hop questions, where answers can often be derived from compact subgraphs connecting a small number of entities. In contrast, CWQ contains more complex questions with longer reasoning chains and more diverse relational structures, posing greater challenges for subgraph extraction and relational reasoning.

Table 21: Dataset statistics.

Dataset	#Train	#Test	Max Hop
WebQSP	2,826	1,628	2
CWQ	27,639	3,531	4

Metrics. **F1**: measures the overlap between the predicted answer set and the ground-truth answer set by combining precision and recall. It reflects how well the model balances answer correctness and completeness, and is particularly suitable for questions with multiple correct answers.

Hit@1: measures whether the top-ranked predicted answer exactly matches the ground-truth answer. It captures the model's ability to place the correct answer at the highest rank and is commonly used for single-answer or ranking-based evaluation.

B.8 Robustness to Entity Linking

Our main experiments assume that seed entities are given. This follows standard KGQA practice. MetaQA annotates the head entity of every question, WebQSP and CWQ provide manually annotated topic entities, and methods such as SubgraphRAG, GNN-RAG, and RoG take these as input. We report two experiments on DBpedia50 with Llama-3.1-8B-Instruct. The first replaces the given seeds

Table 22: End-to-end entity linking on DBpedia50 with Llama-3.1-8B-Instruct.

Setting	Per-query EM	C	$\bar{R}$
Given seeds	100%	99.0%	0.67
LLM linking, original questions	95%	94.4%	0.61
LLM linking, rewritten entity names	84%	82.6%	0.42

Table 23: Sensitivity to one-hop seed substitution. C and $\bar{R}$ are computed against the original seeds.

Perturbation	C	$\bar{R}$	After selection	After pruning
None	99.0%	0.67	–	–
One seed replaced	29.8%	-0.52	100%	30.8%
Both seeds replaced	9.0%	-0.81	100%	9.8%

with an automatic entity linker. The second perturbs the seeds directly to measure sensitivity to linking errors.

End-to-end entity linking. We drop the given seeds. The backbone LLM extracts entity mentions from the question, and each mention is matched to a KG entity by exact label match, then alias lookup, then nearest-neighbor retrieval over name embeddings. Since linking one seed wrongly means answering a different question, we report per-query exact match (EM), the fraction of queries in which every seed is linked correctly.

On the original questions, EM is 95% and connectivity falls only from 99.0% to 94.4% (Table 22). Since our questions are instantiated from canonical KG labels, matching is close to trivial in this case. We therefore rewrite every entity name into a non-canonical form, such as an abbreviation, a partial name, or the name without an honorific. EM then drops to 84% and connectivity to 82.6%. In both settings, connectivity tracks EM closely, so the pipeline does not amplify linking errors.

Sensitivity to linking errors. We replace one or both seeds with a random 1-hop neighbor across all test queries. This models a near-miss link, where the intended entity remains adjacent in the graph. We omit substitution by an unrelated entity, which places the intended entity outside the k-hop expansion by construction and yields near-zero performance for any seed-anchored retriever. Table 23 also reports how often the replaced entity is still present in the subgraph after selection and after pruning.

Degradation is substantial. Connectivity falls to 29.8% with one seed displaced and 9.0% with both. The reward values match what fully disconnected outputs imply, so failures are disconnected answers rather than partially degraded ones. The cause lies in pruning rather than selection. The replaced entity is always present in the candidate subgraph, but it survives pruning in only 30.8% and 9.8% of cases, and connectivity matches these figures within one point.

A possible fix. Stages 1 and 3 currently protect only the linked entity from pruning. Protecting the top-k linking candidates instead would keep the intended entity whenever the linker ranks it highly, at the cost of a slightly larger subgraph. We leave this to future work.

C Details of Human Evaluation

C.1 Protocol

Participants. We recruited 50 fluent English speakers via Prolific. Each participant was paid $2.00. One participant failed the attention check, leaving 49 for analysis. The study was approved by our university's IRB, and all participants gave informed consent. We collected no personally identifying information.

Queries. Each participant evaluated the same four relation-centric queries.

Answers. For each query, participants saw four candidate answers: (1) UniRel, (2) GNN-RAG/SubgraphRAG, which produce identical outputs on these queries, (3) UniRel w/o R_{rel}, and

(4) UniRel w/o R_{ent}. All answers were shown as chains of triples in a uniform format and labelled A–D. The assignment of labels to systems varied across queries, and participants were not told which system produced which answer.

Task. For each query, participants produced three independent rankings of the four answers:

- *Informativeness*: how much each answer tells you about the relationship.
- *Coherence*: how well each answer forms a sensible chain.
- *Usefulness*: whether you would give the answer to someone who asked the question.

The instructions explained how to read an answer and what makes it valid, but did not say what makes one valid answer better than another. They never mentioned node degree, relation frequency, or compactness. The full instructions are included in the supplementary material.

C.2 Per-Query Results

Table 24 reports how many participants ranked each system first, broken down by query and criterion.

Table 24: Number of participants (out of 49) ranking each system first on informativeness (I), coherence (C), and usefulness (U). Bold marks the highest count for each query and criterion. Baseline: GNN-RAG/SubgraphRAG.

	UniRel			w/o R_{rel}			w/o R_{ent}			Baseline		
Query	I	C	U	I	C	U	I	C	U	I	C	U
Q1	**22**	**27**	**28**	18	8	12	6	8	8	3	6	1
Q2	**30**	**24**	**26**	7	4	4	3	4	6	9	17	13
Q3	**33**	**22**	**29**	7	17	11	8	4	7	1	6	2
Q4	**23**	**17**	**21**	13	12	13	6	5	5	7	15	10

Mean rank. UniRel attains the best mean rank on every criterion (1 = best, 2.5 under random ordering). On informativeness, its mean rank is 1.67, against 2.32, 2.66, and 3.35 for w/o R_{rel}, w/o R_{ent}, and the baseline. Its mean ranks on coherence and usefulness are 1.96 and 1.79.

C.3 Statistical Analysis

Agreement between participants. We measure agreement with Kendall's W, which is 0 when participants order the answers in unrelated ways and 1 when all give the identical ordering. We test whether it exceeds zero with a Friedman test at $p < 0.05$.

- Informativeness: $W \in [0.23, 0.43]$, significant on every query.
- Usefulness: $W \in [0.15, 0.48]$, significant on every query.
- Coherence: $W \in [0.05, 0.24]$, not significant on Q4.

Participants converge on which answer says more, but disagree more on which one reads better. The three criteria therefore capture distinct aspects rather than a single overall impression.

Preference over the baseline. We test whether participants rank UniRel above the baseline more often than not, using a one-sided exact binomial test against a 50% null ($n = 49$). UniRel is preferred in 80–92% of judgments on informativeness and 67–96% on usefulness, significant on all four queries for both ($p \leq 0.011$). On coherence, the margin narrows to 61–80% and reaches significance only on Q1. Baseline answers are shorter, typically two triples through a generic entity such as *Male*, and a short chain reads directly even when it conveys little.

Preference over the ablations. Applying the same test to the two ablation variants isolates the contribution of each reward component. UniRel is preferred over w/o R_{ent} in 61–86% of judgments, significant in eleven of twelve query–criterion pairs ($p \leq 0.011$). The exception is coherence on Q4 (61%, $p = 0.076$). Against w/o R_{rel}, the margin is smaller at 51–84%, significant in seven of twelve pairs ($p \leq 0.0047$).

C.4 PARTICIPANTS' OWN DESCRIPTIONS

At the end of the study, participants were asked what influenced their decisions. Many said they preferred answers that named a specific connection over ones that only pointed out something the entities had in common. Representative responses include:

- "a specific, direct family or professional link versus generic shared attributes like gender, nationality, or birthplace"
- "specificity over broad traits, direct family chains over trivial shared attributes"

Since the instructions never mention degree, relation frequency, or compactness, this preference was not suggested to participants.