From Chains to Graphs: Self-Structured Reasoning for General-Domain LLMs

Yingjian Chen[1], Haoran Liu[2], Yinhong Liu[3], Sherry T. Tong[1], Aosong Feng[4], Jinghui Lu[5], Juntao Zhang[6], Yusuke Iwasawa[1], Yutaka Matsuo[1], Irene Li[1]*

[1]University of Tokyo, [2]Texas A&M University, [3]University of Cambridge, [4]Yale University, [5]Xiaomi EV, [6]Henan University.

irene.li@weblab.t.u-tokyo.ac.jp

Abstract

Large Language Models (LLMs) increasingly rely on explicit reasoning processes to solve complex question answering tasks. However, these reasoning processes are typically represented through linear textual traces, such as Chain-of-Thought (CoT). Although these traces are human-readable, their sequential structure can obscure logical dependencies and make inconsistencies difficult to identify. To improve reasoning structure, prior graph-based approaches introduce structured representations. However, they often rely on externally constructed graphs instead of enabling LLMs to perform graph-structured reasoning autonomously. To fill this gap, we propose Self-Graph Reasoning (SGR), a framework that enables LLMs to explicitly represent their reasoning process as a structured graph before producing the final answer. We also construct a graph-structured reasoning dataset by consolidating multiple candidate reasoning graphs into refined graph representations for training. Together, these components improve both the consistency and the interpretability of the reasoning. Experiments on five QA benchmarks across both general and specialized domains show that SGR consistently improves reasoning consistency and provides clearer, more interpretable graph-structured reasoning processes, achieving a 17.74% gain over the base model.[1]

1 Introduction

Large Language Models (LLMs) (Hurst et al., 2024; Dubey et al., 2024) have demonstrated strong performance across a wide range of open-domain natural language understanding and question-solving tasks (Zhao et al., 2023; Bang et al., 2023; Yang et al., 2024b; Teleki et al., 2025; Liu et al., 2025). Beyond direct answer prediction, eliciting explicit intermediate reasoning has become a widely adopted strategy for improving complex inference in LLMs. Chain-of-Thought (CoT) (Wei et al., 2022) is a representative paradigm, guiding models to generate intermediate reasoning steps before the final answer and thereby improving both task performance and interpretability (Ke et al., 2025; Patil and Jadon, 2025).

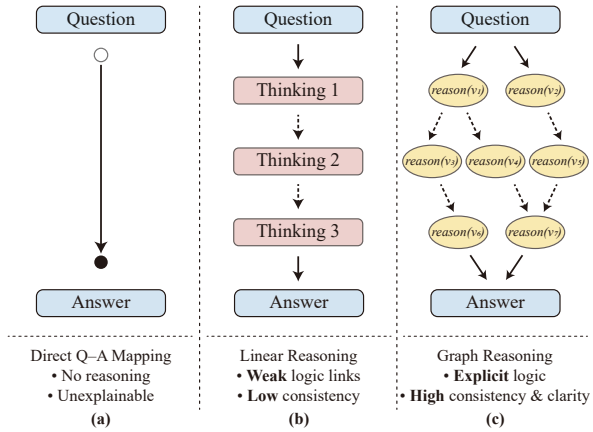


Figure 1: Comparison of reasoning and answering paradigms in general-domain question answering. (a) **Direct Answering** without explicit reasoning. (b) **Linear reasoning** with unclear logical dependencies among intermediate steps and the final answer. (c) Our **Graph-structured reasoning** with explicit logical connections, yielding higher reasoning consistency.

Despite their effectiveness, such reasoning approaches typically express reasoning as linear textual sequences. This sequential form makes the dependencies among premises, intermediate reasoning steps, and final conclusions implicit and difficult to inspect. As a result, it is often unclear whether the generated reasoning truly supports the final answer (Wang et al., 2025; Arcuschin et al., 2025; Turpin et al., 2023), as illustrated in Figure 1 (b). This limitation becomes more pronounced when reasoning involves multiple premises, parallel sub-problems, and their integration toward a final conclusion. Although such dependencies

*Corresponding author

[1]Our code is available at https://github.com/Yingjian-Chen/SGR-Self-Graph-Reasoning.

Findings of the Association for Computational Linguistics: IJCNLP-AACL 2026, pages 4141–4159
November 6-10, 2026 ©2026 Association for Computational Linguistics

can be described in a linear chain, they are often difficult to faithfully expose and verify in long free-form text. Consequently, factual errors or unsupported claims may be buried within otherwise plausible reasoning chains, making the reasoning process less reliable as evidence for the final conclusion. This issue is particularly challenging in general-domain question answering (QA), where intermediate reasoning units are often ambiguous and difficult to verify explicitly (Xu et al., 2024; Lee and Lee, 2025).

Existing graph-based reasoning methods either rely on externally provided or retrieved graphs (Jin et al., 2024; Luo et al., 2024; Han et al., 2025), or organize LLM-generated thoughts into a transformation graph through multiple rounds of generation and refinement, as in Graph-of-Thoughts (Yao et al., 2024). In general-domain QA, external graphs are often unavailable, and multi-step graph construction can propagate errors from incorrectly generated intermediate nodes or edges. In this setting, it remains largely unexplored whether LLMs can directly generate explicit reasoning graphs that make the support relations between intermediate steps and final answers checkable. To explore this question, we introduce *Self-Graph Reasoning* (SGR), a paradigm that guides LLMs to generate an explicit reasoning graph before producing the final answer. In SGR, nodes represent reasoning units and edges encode their dependencies. Unlike linear text-based reasoning, SGR provides a structured and clear representation of how intermediate steps support the final answer. To better support this paradigm, we construct a dataset for self-graph reasoning, in which multiple candidate reasoning graphs are generated and aggregated to provide diverse graph-based supervision signals. Experiments on five QA benchmarks, covering both general-domain and specialized-domain tasks, demonstrate that SGR improves the clarity and interpretability of reasoning while achieving competitive task performance. We further release the dataset to facilitate future research on structured and interpretable reasoning in LLMs.

In summary, our contributions are: (1) **Self-Graph Reasoning Method.** We introduce *Self-Graph Reasoning* (SGR), which, to the best of our knowledge, is among the first attempts to study whether LLMs can generate explicit reasoning graphs by themselves for question answering, making intermediate reasoning more transparent and checkable. (2) **Graph-Reasoning Dataset.** We construct a 10K-instance general-purpose graph-reasoning dataset that provides structured supervision for self-graph reasoning. (3) **Empirical Effectiveness.** Experiments on five QA benchmarks show that SGR achieves a 17.74% average improvement over its base model and remains effective in specialized domains, including mathematics and medicine.

2 Related Works

Linear Reasoning. Early language models mainly relied on direct answer prediction, where reasoning was largely implicit (Brown et al., 2020; Chowdhery et al., 2023; Li et al., 2025). Chain-of-Thought (CoT) was introduced to elicit intermediate reasoning steps before the final answer, improving model performance on complex reasoning tasks (Wei et al., 2022; Kojima et al., 2022; Zhou et al., 2022; Zhang et al., 2022). More recent reasoning-oriented LLMs further strengthen this paradigm through specialized training or inference-time reasoning mechanisms (Jaech et al., 2024; Guo et al., 2025; Yang et al., 2025). However, most existing reasoning paradigms still represent reasoning as linear textual traces, where the dependency relations between intermediate steps and final answers remain implicit. As a result, it is difficult to verify whether the generated reasoning actually supports the final prediction, and such traces may appear plausible while being unfaithful (Turpin et al., 2023; Lanham et al., 2023). In contrast, our work studies explicit graph-structured reasoning to make reasoning dependencies more transparent and checkable.

Graph-based Reasoning. Recent studies have explored graph-based reasoning with LLMs by incorporating structured graphs into the reasoning process (Luo et al., 2023; Sun et al., 2023; Jin et al., 2024; Tian et al., 2024; Cao, 2024; Chen et al., 2025). Existing methods generally follow two directions. One line relies on externally provided, retrieved, or task-specific graphs, such as knowledge or evidence graphs, to support answer generation (Luo et al., 2023; Wen et al., 2024). Although effective when reliable graphs are available, such structures are often absent in general-domain QA. Another line organizes LLM-generated thoughts into graph-like computation processes, as in Graph-of-Thoughts (Yao et al., 2024), which performs iterative generation, aggregation, and refinement over intermediate thoughts. However, in open-domain

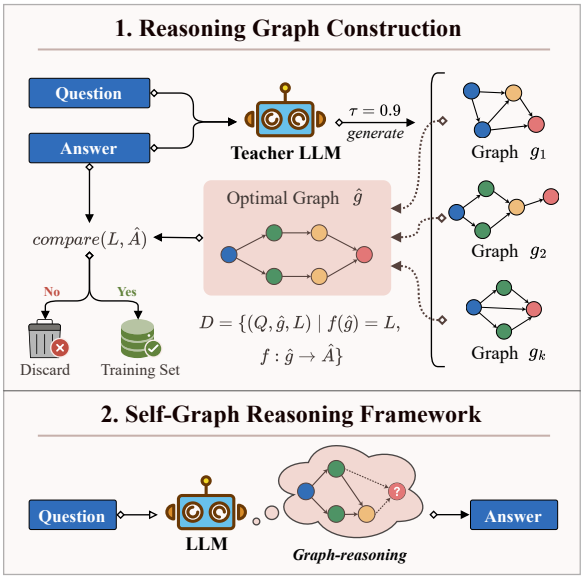


Figure 2: Overview of the proposed Self-Graph Reasoning (SGR).

QA, where reasoning paths are often uncertain, such frameworks require multi-step graph construction, making them vulnerable to error accumulation from incorrectly generated thoughts or edges. In contrast, SGR directly self-generates explicit reasoning graphs for question answering, providing a structured and checkable reasoning process.

3 Methods

In this section, we introduce *Self-Graph Reasoning* (SGR), a framework that guides LLMs to generate an explicit reasoning graph before producing the final answer. SGR organizes intermediate reasoning units and their dependencies into a graph structure, providing a clearer reasoning process and helps expose unsupported or inconsistent intermediate steps. The implementation of SGR consists of two main components, as illustrated in Figure 2: (1) Reasoning Graph Construction, which constructs a dataset of explicit reasoning graphs capturing the logical process between questions and answers, serving as supervision for model training; and (2) Self-Graph Reasoning Framework, where the model is fine-tuned on the constructed graph data to internalize the ability to perform self-graph reasoning before generating the final answer.

3.1 Reasoning Graph Construction

To provide the model with graph-structured reasoning supervision, we transform raw question-answer pairs into explicit reasoning graphs through candidate graph generation, graph refinement, and answer-consistency filtering.

Diverse Trajectory Exploration. To capture diverse reasoning trajectories for a general-domain question Q, we prompt a teacher LLM (GPT-4o) to generate multiple candidate reasoning graphs. General-domain questions may admit different valid reasoning routes, as they can be approached from different perspectives or intermediate assumptions. To encourage such diversity, we sample candidate graphs with a relatively high temperature ($\tau = 0.9$). Formally, for each question Q, we sample k independent candidate reasoning graphs, denoted as $\mathcal{S} = \{g_1, g_2, \ldots, g_k\}$. Each candidate graph $g_i = (V_i, E_i)$ consists of nodes V_i representing atomic reasoning steps and directed edges E_i representing dependencies among them.

Graph Refinement and Data Cleaning. Individual candidate graphs may contain fragmented reasoning steps, redundant claims, or inconsistent intermediate paths. To obtain a cleaner answer-consistent graph while preserving useful reasoning branches, we use the same teacher model to integrate multiple candidate graphs $\mathcal{S}$ into a single refined graph $\hat{g}$. During this process, the teacher model is instructed to merge semantically equivalent reasoning units, remove redundant or unsupported steps, and retain only reasoning branches that are consistent with the given question and answer. The detailed prompt is provided in Appendix H. The resulting graph $\hat{g}$ is then linearized into a structured template as follows:

```
<reasoning>
  <step> v_i → v_j </step>
  ...
</reasoning>
<answer> Final Answer </answer>
```

Answer-Consistency Filtering. To improve the reliability of the training data, we evaluate whether the refined graph $\hat{g}$ supports the known correct answer L for question Q. Specifically, we define a mapping $f(Q, \hat{g}) \to \hat{A}$, where f denotes an LLM-mediated decoding process that generates a predicted answer $\hat{A}$ conditioned on the question and the refined reasoning graph. We retain $\hat{g}$ in the final training set $\mathcal{D}$ only if the predicted answer matches the ground-truth label L:

$$\mathcal{D} = \{(Q, \hat{g}, L) \mid f(Q, \hat{g}) = L\}. \quad (1)$$

This filtering step removes graphs whose synthesized reasoning fails to support the correct answer,

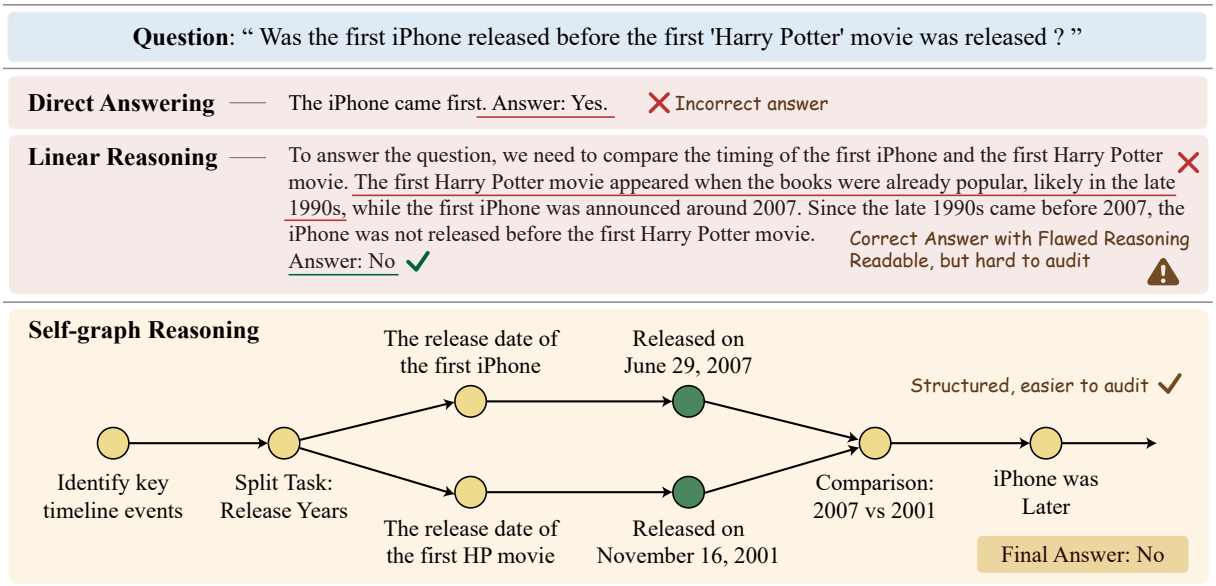


Figure 3: An illustrative example of the Self-Graph Reasoning framework. Our method constructs a structured reasoning graph where each node is explicitly grounded in its predecessors to ensure a logically consistent and clear path to the final answer. For comparison, we also illustrate other paradigms: Direct Answering lacks an explicit thinking process and is prone to errors, while Linear Reasoning can be difficult to audit because factual errors may be buried in long textual reasoning traces.

reducing the inclusion of answer-inconsistent graph supervision.

Supervised Graph Learning. To train the LLM to generate such structured reasoning traces, we perform Supervised Fine-Tuning (SFT) with LoRA (Hu et al., 2022) on the training data $\mathcal{D}$ constructed in Section 3.1. Each training instance is a triplet $(Q, \hat{g}, L)$, where $\hat{g}$ denotes the target reasoning graph and L denotes the ground-truth answer. The graph $\hat{g}$ serves as an intermediate structured representation that connects the question Q to the answer L. Formally, we optimize the model parameters θ by minimizing the standard cross-entropy loss over both the reasoning graph and the final answer:

$$\mathcal{L}(\theta) = -\sum_{(Q,\hat{g},L)\in\mathcal{D}} \log P_\theta(\hat{g}, L \mid Q). \quad (2)$$

By learning to generate $\hat{g}$ before L, the model is encouraged to organize its intermediate reasoning into explicit nodes and dependency edges. Compared with *Direct Answering*, which maps the question directly to an answer, and *Linear Reasoning* such as CoT, which presents reasoning as a sequential textual trace, SGR provides a structured representation in which the support relations among intermediate steps can be explicitly inspected. Specifically, each node v_j is associated with parent nodes $Pa(v_j)$ that indicate its supporting context, making it easier to identify unsupported steps, missing dependencies, or potential inconsistencies in the reasoning process. In this way, SGR improves the transparency and checkability of intermediate reasoning without relying on external graph structures.

Inference Stage. At inference time, given a question Q, the model generates the output in two consecutive parts: a structured reasoning graph $\hat{g}$ followed by the final answer L. This process follows the learned conditional distribution $P_\theta(\hat{g}, L \mid Q)$, where $\hat{g}$ serves as an explicit intermediate representation for answer generation. Specifically, the model produces reasoning units as graph nodes and dependency relations as edges, indicating how intermediate steps support one another and lead to the final answer. Compared with a sequential textual trace, the generated graph makes the dependency structure among intermediate reasoning units explicit and easier to inspect. An illustration of the inference output is shown in Figure 3.

4 Experimental Setup

4.1 Datasets

Training Dataset. To enable the LLM to perform self-graph reasoning before answering questions, we construct a dataset of about 10K samples based on the training subset of the LogiQA dataset (Liu

et al., 2020). LogiQA is a general-domain QA benchmark that involves complex logical reasoning, making it particularly suitable for supervising structured graph-based reasoning. The final 10K samples are obtained after data cleaning and filtering. For each question–answer pair, we use GPT-4o (Hurst et al., 2024) to generate an explicit reasoning graph, which captures the logical dependencies leading to the correct answer. The resulting dataset is organized in the format of {*Question*, *Graph Reasoning*, *Label*}. More information is shown in Appendix B.

Evaluation Benchmarks. Our work primarily targets general-domain question answering. To evaluate the model's performance in this setting, we adopt several widely used benchmarks, including the LogiQA test set (Liu et al., 2020), AIW, AIW+ (Nezhurina et al., 2024), and AR-LSAT (Wang et al., 2022). In addition, to evaluate the cross-domain generalization of our method, which is trained on general-domain data, we further evaluate it on MedQA (Jin et al., 2020) and MathQA (Amini et al., 2019). Details of the benchmarks are provided in Appendix A.

4.2 Baselines

We compare our proposed framework with three categories of baselines, including proprietary LLMs, open-source LLMs, and graph-based reasoning methods. For proprietary LLMs, we include GPT-4o (Hurst et al., 2024), GPT-5.1 (OpenAI, 2025), Claude-3.5-Haiku (Anthropic, 2024), and Gemini-2.5-Pro (Google, 2025), which serve as strong closed-source baselines with advanced reasoning capabilities. For open-source LLMs, we evaluate a series of models with varying scales, including LLaMA-3.2-3B, LLaMA-3.1-8B, LLaMA-3.3-70B (Dubey et al., 2024), Qwen2.5-7B, and Qwen2.5-72B (Yang et al., 2024a). For graph-based reasoning methods, we compare with Reasoning with Graphs (RwG) (Han et al., 2025), which relies on externally pre-extracted graphs to enhance model reasoning. We also include Graph-of-Thoughts (GoT) (Yao et al., 2024), which builds graph-like reasoning structures through iterative thought generation and aggregation.

4.3 Implementation Details

We perform supervised fine-tuning of LLaMA-3.3-70B [2] using LoRA (Hu et al., 2022), keeping the base model frozen. Models are trained for 1-3 epochs with early stopping (patience=1), using a batch size of 4 and learning rate 6×10^{-5}. Gradient accumulation is employed to effectively increase the batch size. Maximum generation length is set to 1024. The checkpoint with the best validation performance is selected for evaluation. Experiments are conducted on 8 NVIDIA A100 40GB GPUs. Full hyperparameter details are provided in the Appendix G. We use accuracy (Acc) as the evaluation metric for our experiments, following previous methods (Cao, 2024; Han et al., 2025).

[2] https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct

5 Results and Analysis

5.1 Main Results

Table 1 reports the accuracy of different models across general-domain and specialized-domain QA benchmarks. Overall, SGR-LLaMA3.3-70B achieves an average accuracy of 61.03%, which is competitive with the proprietary GPT-4o model (61.52%) and substantially outperforms comparable open-source models. Compared with its base model LLaMA-3.3-70B, SGR improves the overall average accuracy from 43.29% to 61.03%, yielding a gain of 17.74 percentage points. We note that Gemini-2.5-Pro achieves substantially stronger performance on the four benchmarks where it participates, with an average accuracy of 82.77%. Nevertheless, SGR provides an open and reproducible graph-structured reasoning approach that consistently improves its open-source base model.

In the general-domain setting, including LogiQA, AIW, and AR-LSAT, SGR-LLaMA3.3-70B achieves an average accuracy of 53.05%, surpassing GPT-4o (46.09%) and all evaluated open-source LLMs. Compared with its base model, SGR improves the general-domain average from 38.27% to 53.05%, corresponding to a gain of 14.78 percentage points. Compared with graph-based baselines, SGR outperforms RwG-LLaMA3.1-70B by 18.76 percentage points on average over the shared general-domain benchmarks. It also exceeds GoT-LLaMA3.3-70B by 12.32 percentage points, suggesting that direct self-graph reasoning is more effective than multi-step graph construction in these general-domain QA tasks. This may be because multi-step graph construction can accumulate errors when intermediate thoughts or edges are incorrectly generated.

To further evaluate the generalization capability of SGR, we conduct experiments on specialized-

Method	General Domain			Specialized Domain		Overall Avg. (%)
	LogiQA	AIW	AR-LSAT	MedQA	MathQA	
Proprietary LLMs						
GPT-4o (Hurst et al., 2024)	74.01	32.50	31.75	88.29	81.05	61.52
GPT-5.1 (OpenAI, 2025)	76.34	57.00	33.33	89.55	39.09	59.06
Claude-3.5-Haiku (Anthropic, 2024)	65.97	2.50	29.41	76.36	79.36	50.72
Gemini-2.5-Pro* (Google, 2025)	85.75	76.00	96.22	-	73.10	-
Open-source LLMs						
LLaMA-3.2-3B (Dubey et al., 2024)	41.28	1.70	20.00	49.57	29.21	28.35
LLaMA-3.1-8B (Dubey et al., 2024)	49.17	5.00	26.96	55.22	29.39	33.15
LLaMA-3.3-70B (Dubey et al., 2024)	64.01	19.50	31.30	63.55	38.09	43.29
Qwen2.5-7B (Yang et al., 2024a)	34.10	5.00	17.39	59.54	38.29	30.96
Qwen2.5-72B (Yang et al., 2024a)	**76.91**	5.00	**34.78**	74.42	52.60	49.00
Specialized Graph-based Methods						
RwG-LLaMA3.1-70B† (Han et al., 2025)	59.13	12.00	31.73	-	-	-
RwG-Claude-3-sonnet† (Han et al., 2025)	45.16	2.60	30.86	-	-	-
GoT-LLaMA3.3-70B (Besta et al., 2024)	61.69	30.50	30.00	70.38	**70.02**	52.52
Ours						
SGR-Llama3.3-70B	69.91	**57.50**	31.74	**78.81**	67.17	**61.03**

Table 1: Accuracy of models across all benchmarks, covering both general-domain and specialized-domain question answering. Methods are categorized into proprietary LLMs, open-source LLMs, and specialized reasoning approaches. The best results among non-proprietary LLMs are highlighted in **bold**, and the second-best are underlined. The red and blue indicate the average performance of our SGR method and its base model, respectively. Asterisk-marked Gemini results are omitted for medical-domain questions due to policy restrictions. Dagger-marked RwG results are taken from the original publication due to reproduction issues; therefore, MedQA and MathQA are excluded.

domain tasks, including MedQA and MathQA. SGR-LLaMA3.3-70B achieves 78.81% on MedQA and 67.17% on MathQA, reaching an average accuracy of 72.99%. This improves over the base model by 22.17 percentage points on the two specialized-domain benchmarks. GoT performs particularly well on MathQA, achieving 70.02%, which suggests that multi-step thought graph construction can be beneficial when the reasoning process is relatively well-structured. However, its performance is less stable on general-domain QA, where reasoning dependencies are often more ambiguous and intermediate errors may accumulate across multiple generation steps. In contrast, SGR maintains strong performance across both general-domain and specialized-domain tasks, indicating that explicit self-generated reasoning graphs provide a structured and checkable reasoning representation that generalizes beyond the training domain.

5.2 Ablation Studies

Effectiveness of self-graph reasoning on different model scales. To further assess the effectiveness of SGR across model scales, we conduct ablation studies by comparing base models with their self-graph reasoning variants. We evaluate LLaMA-3.1-8B and LLaMA-3.3-70B, with results summarized in Figure 4. The results show that, for the 8B model, SGR brings only marginal gains on MathQA and MedQA, and even leads to performance degradation on more complex general-domain QA benchmarks. This suggests that when the base model has limited reasoning capacity, the requirement to generate an explicit graph may introduce additional structural constraints and gener-

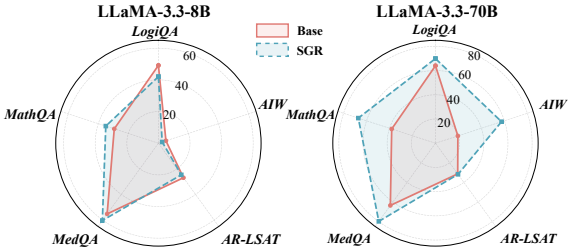


Figure 4: Comparison of accuracy between the base models and our SGR across all five benchmarks for LLaMA-3.3-8B and LLaMA-3.3-70B.

Methods	LogiQA	AIW	AR-LSAT	Avg
DA-Prompt	64.01	19.50	31.30	38.27
CoT-Prompt	64.69	31.00	29.13	41.61
DA-SFT	69.85	26.50	**33.04**	43.13
CoT-SFT	67.88	35.00	27.83	43.57
SGR-Prompt (ours)	64.49	40.00	32.17	45.55
SGR-SFT (ours)	**69.91**	**57.50**	31.74	**53.05**

Table 2: Accuracy comparison of different reasoning settings based on LLaMA-3.3-70B on general-domain QA benchmarks. DA, CoT, and SGR denote direct answering, linear reasoning, and Self-Graph Reasoning, respectively. Prompt indicates inference with prompting only, while SFT denotes supervised fine-tuning.

ation burden. Instead of helping the model organize reasoning, the graph format may increase the difficulty of task solving.

In contrast, the 70B model demonstrates substantial and consistent improvements across all benchmarks, indicating that stronger models are better able to leverage graph structures to organize intermediate reasoning and make dependencies explicit. Overall, these findings suggest that the benefits of SGR depend on the model's underlying reasoning capacity. Stronger LLMs are better suited for graph-structured reasoning, and the impact of SGR becomes more pronounced as model size increases. We expect that applying SGR to even larger or more capable LLMs may yield further improvements.

Effect of graph-structured reasoning. To disentangle the effect of graph-structured reasoning from that of supervised fine-tuning, we compare DA, CoT, and SGR under both prompting-only and SFT settings using the same base model, LLaMA-3.3-70B. For CoT-SFT, we convert the same graph-reasoning training data into linear CoT-style traces, so that CoT-SFT and SGR-SFT use the same questions, answers, and data scale. As shown in Table 2, SGR-SFT achieves the best average performance, showing that supervised training brings substantial performance gains. Meanwhile, SGR-Prompt also achieves competitive results and outperforms both DA-Prompt and CoT-Prompt on average. This suggests that explicitly representing intermediate reasoning as a graph provides useful structure even without fine-tuning. Moreover, under the same SFT setting, SGR-SFT outperforms DA-SFT and CoT-SFT by 9.92 and 9.48 percentage points, respectively. This indicates that the improvement is not solely due to supervised fine-tuning, but also benefits from the graph representation. More importantly, SGR explicitly represents the dependencies among intermediate steps, making it easier to inspect how the final answer is derived.

Methods	LogiQA	AIW	AR-LSAT	Avg
Best-of-N Sampling	67.55	21.50	29.57	39.54
Self-Consistency	**70.64**	54.26	**32.61**	52.50
SGR (ours)	69.91	**57.50**	31.74	**53.05**

Table 3: Accuracy comparison of multi-sample reasoning strategies based on LLaMA-3.3-70B on general-domain QA benchmarks. Best-of-N Sampling and Self-Consistency use three sampled generations per instance.

Comparison with multi-sample reasoning strategies. We further compare SGR with widely adopted multi-sample reasoning strategies, including Best-of-N Sampling and Self-Consistency, using LLaMA-3.3-70B under the same evaluation setting. As shown in Table 3, SGR achieves the best average performance across the three general-domain QA benchmarks, reaching 53.05%, compared with 52.50% for Self-Consistency and 39.54% for Best-of-N Sampling. Notably, Best-of-N Sampling and Self-Consistency require multiple sampled generations at inference time, whereas SGR produces a structured reasoning graph and final answer in a single inference pass, avoiding the additional cost of repeated sampling.

5.3 Case studies

We conduct a case study on general-domain QA to illustrate how SGR represents the reasoning process as an explicit self-generated graph, as shown in Figure 5. Unlike linear reasoning paradigms such as CoT, which present reasoning as a sequential textual trace, SGR constructs a structured reasoning graph before producing the final answer. This graph provides a clearer and more checkable representation of how intermediate reasoning steps support the final prediction.

For the given question, the model generates a reasoning graph consisting of 9 nodes and 8 directed edges, where each node corresponds to an intermediate reasoning step and each edge encodes a dependency between reasoning units. This structured organization makes the reasoning process easier to follow and inspect. Specifically, the branching node *"Identify siblings (brothers & sisters)"* separates the interpretation of the two sibling types, forming distinct reasoning branches for brother-related and sister-related information. This decom-

Question Alice has 4 brothers and she also has 3 sisters. How many sisters does one of Alice's brother have?

Actual Output

<reasoning>
<step>Alice exists -> leads to -> Consider family members related to her</step>
<step>Consider family members related to her -> leads to -> Identify siblings (brothers & sisters)</step>
<step>**Identify siblings (brothers & sisters)** -> leads to -> Note number of brothers = 4</step>
<step>**Identify siblings (brothers & sisters)** -> leads to -> Note number of sisters = 3 + Alice herself for sibling count from brother perspective</step>
<step>Note number of brothers = 4 -> leads to -> Each brother is part of this group but question focuses on sister count per brother</step>
<step>Note number of sisters = 3 + Alice herself for sibling count from brother perspective -> leads to -> Total sisters seen by each brother includes Alice now counted among them</step>
<step>Total sisters seen by each brother includes Alice now counted among them -> leads to -> **Calculate total sisters including Alice from brother's view = 3 actual sisters + 1 (Alice)**</step>
<step>Calculate total sisters including Alice from brother's view = 3 actual sisters + 1 (Alice) -> leads to -> **Conclusion that each brother sees 4 sisters**</step>
</reasoning>
<answer>4</answer>

Reasoning Graph Visualization

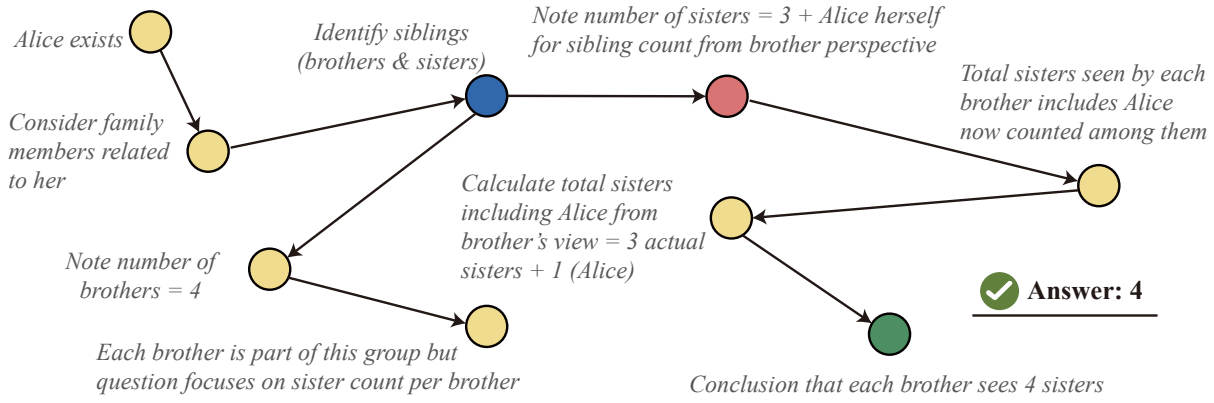


Figure 5: A case study in general-domain QA, illustrating the model's actual output and its reasoning graph visualization. The blue nodes denote branch nodes, the red nodes denote key nodes, and the green node denotes the final decision node.

position makes it easier to examine how each subproblem contributes to the final answer, rather than mixing different reasoning units within a single linear trace. A key step appears in the node *"Note number of sisters = 3 + Alice herself for sibling count from brother perspective"*, which explicitly captures the perspective shift required by the question. Since the question asks how many sisters one of Alice's brothers has, Alice herself must also be counted as a sister from the brother's perspective. By representing this transition as an independent node, the graph makes this otherwise implicit reasoning step directly visible and easier to verify.

Finally, SGR aggregates the relevant information through the computation node *"Calculate total sisters including Alice from brother's view = 3 actual sisters + 1 (Alice)"* and reaches the decision node *"Conclusion that each brother sees 4 sisters."*, leading to the correct answer: 4.

6 Conclusion

In this work, we introduced Self-Graph Reasoning (SGR), a framework that guides LLMs to generate explicit reasoning graphs before producing final answers. SGR represents reasoning units and their dependencies as a graph, providing a structured and clear representation of how intermediate steps support the final prediction. We also constructed a 10K-instance graph-reasoning dataset for structured supervision. Experiments across five QA benchmarks show that SGR improves its base model by 17.74% on average and achieves competitive performance among non-proprietary baselines. These findings suggest that self-generated reasoning graphs can serve as an effective and inspectable intermediate representation for LLM reasoning.

Limitations

Training Data. As an initial exploration of self-graph reasoning, our current study constructs a medium-scale graph-reasoning dataset with approximately 10K training instances. While this dataset is sufficient to examine the feasibility of SGR, it may not fully cover the diversity of reasoning patterns required by more complex QA tasks. Future work should explore larger-scale graph-reasoning datasets and compare different teacher models, including open-source teachers, to better understand how teacher choice affects SGR training.

Base Model Scale. As discussed in the paper, the effectiveness of SGR is inherently tied to the capability of the base models. Due to computational constraints, our experiments are conducted on a 70B model. Applying SGR to larger-scale models may further improve reasoning consistency and overall performance.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)*, pages 2357–2367.

Anthropic. 2024. Claude 3.5 haiku. https://www.anthropic.com/news/3-5-models-and-computer-use. Retrieved December 12, 2025.

Iván Arcuschin, Jett Janiak, Robert Krzyzanowski, Senthooran Rajamanoharan, Neel Nanda, and Arthur Conmy. 2025. Chain-of-thought reasoning in the wild is not always faithful. *arXiv preprint arXiv:2503.08679*.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, and 1 others. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and 1 others. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 17682–17690.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Lang Cao. 2024. Graphreason: Enhancing reasoning capabilities of large language models through a graph-based verification approach. In *Proceedings of the 2nd Workshop on Natural Language Reasoning and Structured Explanations (@ ACL 2024)*, pages 1–12.

Yingjian Chen, Haoran Liu, Yinhong Liu, Jinxiang Xie, Rui Yang, Han Yuan, Yanran Fu, Peng Yuan Zhou, Qingyu Chen, James Caverlee, and 1 others. 2025. Graphcheck: Breaking long-term text barriers with extracted knowledge graph-powered fact-checking. *arXiv preprint arXiv:2502.16514*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Google. 2025. Gemini-2.5-pro. https://docs.cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-5-pro. Retrieved December 12, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Haoyu Han, Yaochen Xie, Hui Liu, Xianfeng Tang, Sreyashi Nag, William Headden, Yang Li, Chen Luo, Shuiwang Ji, Qi He, and 1 others. 2025. Reasoning with graphs: Structuring implicit knowledge to enhance llms reasoning. *arXiv preprint arXiv:2501.07845*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Bowen Jin, Chulin Xie, Jiawei Zhang, Kashob Kumar Roy, Yu Zhang, Zheng Li, Ruirui Li, Xianfeng Tang, Suhang Wang, Yu Meng, and 1 others. 2024. Graph chain-of-thought: Augmenting large language models by reasoning on graphs. *arXiv preprint arXiv:2404.07103*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2020. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *arXiv preprint arXiv:2009.13081*.

Zixuan Ke, Fangkai Jiao, Yifei Ming, Xuan-Phi Nguyen, Austin Xu, Do Xuan Long, Minzhi Li, Chengwei Qin, Peifeng Wang, Silvio Savarese, and 1 others. 2025. A survey of frontiers in llm reasoning: Inference scaling, learning to reason, and agentic systems. *arXiv preprint arXiv:2504.09037*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, and 1 others. 2023. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*.

Jihyung Lee and Gary Geunbae Lee. 2025. Structured reasoning and answer verification: Enhancing question answering system accuracy and explainability. *Knowledge-Based Systems*, 311:113091.

Jindong Li, Yali Fu, Li Fan, Jiahong Liu, Yao Shu, Chengwei Qin, Menglin Yang, Irwin King, and Rex Ying. 2025. Implicit reasoning in large language models: A comprehensive survey. *arXiv preprint arXiv:2509.02350*.

Haoran Liu, Amir Tahmasbi, Ehtesham Sam Haque, and Purak Jain. 2025. Llms for customized marketing content generation and evaluation at scale. *arXiv preprint arXiv:2506.17863*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2023. Reasoning on graphs: Faithful and interpretable large language model reasoning. *arXiv preprint arXiv:2310.01061*.

Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Yuan-Fang Li, Chen Gong, and Shirui Pan. 2024. Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models. *arXiv preprint arXiv:2410.13080*.

Marianna Nezhurina, Lucia Cipolina-Kun, Mehdi Cherti, and Jenia Jitsev. 2024. Alice in wonderland: Simple tasks showing complete reasoning breakdown in state-of-the-art large language models. *arXiv preprint arXiv:2406.02061*.

OpenAI. 2025. Gpt5.1. https://openai.com/ja-JP/index/introducing-gpt-5/. Retrieved December 12, 2025.

Avinash Patil and Aryan Jadon. 2025. Advancing reasoning in large language models: Promising methods and approaches. *arXiv preprint arXiv:2502.03671*.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M Ni, Heung-Yeung Shum, and Jian Guo. 2023. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. *arXiv preprint arXiv:2307.07697*.

Maria Teleki, Vedangi Bengali, Xiangjue Dong, Sai Tejas Janjur, Haoran Liu, Tian Liu, Cong Wang, Ting Liu, Yin Zhang, Frank Shipman, and 1 others. 2025. A survey on llms for story generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 13954–13966.

Yijun Tian, Huan Song, Zichen Wang, Haozhu Wang, Ziqing Hu, Fang Wang, Nitesh V Chawla, and Panpan Xu. 2024. Graph neural prompting with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19080–19088.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Preprint*, arXiv:2305.04388.

Siyuan Wang, Zhongkun Liu, Wanjun Zhong, Ming Zhou, Zhongyu Wei, Zhumin Chen, and Nan Duan. 2022. From lsat: The progress and challenges of complex reasoning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Zezhong Wang, Xingshan Zeng, Weiwen Liu, Yufei Wang, Liangyou Li, Yasheng Wang, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. 2025. Chain-of-probe: Examining the necessity and accuracy of CoT step-by-step. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 2586–2606, Albuquerque, New Mexico. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yilin Wen, Zifeng Wang, and Jimeng Sun. 2024. Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models. In *Proceedings*

of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10370–10388.

Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. 2024. Faithful logical reasoning via symbolic chain-of-thought. *arXiv preprint arXiv:2405.18357*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024a. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Rui Yang, Qingcheng Zeng, Keen You, Yujie Qiao, Lucas Huang, Chia-Chun Hsieh, Benjamin Rosand, Jeremy Goldwasser, Amisha Dave, Tiarnan Keenan, and 1 others. 2024b. Ascle—a python natural language processing toolkit for medical text generation: development and evaluation study. *Journal of Medical Internet Research*, 26:e60601.

Yao Yao, Zuchao Li, and Hai Zhao. 2024. Got: Effective graph-of-thought reasoning in language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2901–2921.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *Preprint*, arXiv:2210.03493.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and 1 others. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A Benchmark Details

A.1 General Domain Benchmarks

LogiQA. The LogiQA dataset (Liu et al., 2020) is a multiple-choice reading comprehension benchmark designed to evaluate logical reasoning in natural language understanding. Each instance consists of a context, a question, and four candidate answers, only one of which is logically correct. The questions are sourced from real-world examination materials, such as national civil service and postgraduate entrance tests, ensuring a high level of reasoning complexity. We use the training set of LogiQA2.0 to construct our graph reasoning dataset and the test set as a benchmark for evaluation.

AIW. The AIW dataset (Nezhurina et al., 2024) is a benchmark of reasoning problems. Each problem typically presents a simple natural language scenario, such as familial relationships with variables, and requires the model to derive a logically correct answer from basic commonsense logic. Unlike traditional multiple-choice datasets, AIW focuses on minimal text examples that expose reasoning breakdowns in state-of-the-art models, making it a useful diagnostic for evaluating core inferential competence.

AR-LSAT. The AR-LSAT dataset (Wang et al., 2022) is a large-scale benchmark constructed from the analytical reasoning (logic games) section of the official LSAT (Law School Admission Test). It is designed to evaluate the formal reasoning and deductive inference abilities of language models. Each problem describes a context and requires the model to select the correct answer. We use the test set as a benchmark for evaluation.

A.2 Specialized Domain Benchmarks

MedQA. The MedQA dataset (Jin et al., 2020) is a large-scale medical question answering benchmark designed to evaluate the clinical knowledge and reasoning abilities of language models. Each question is a multiple-choice item sourced from real-world medical licensing examinations. The dataset covers a wide range of medical domains, requiring models to integrate factual recall with domain-specific reasoning.

MathQA. The MathQA dataset (Amini et al., 2019) is a large-scale benchmark for evaluating the mathematical reasoning and problem-solving abilities of language models. It is gathered by using a new representation language to annotate over the AQuA-RAT dataset, covering a wide range of mathematical domains. Each question requires the model to translate a natural language description into a formal reasoning process and compute the correct numerical answer.

Dataset	Size	Question Length
General Domain		
LogiQA	1572	930.57
AIW	200	98.79
AR-LSAT	230	892.36
Specialized Domain		
MedQA	1273	877.57
MathQA	2985	216.76

Table 4: Statistics of Benchmark Datasets. We report the size of each benchmark and the average text length of questions.

A.3 Preprocessing for Benchmarks

To facilitate a unified evaluation across different benchmarks, we standardize all datasets into a consistent {question, label} format. For datasets originally containing a separate *context* field, we concatenate them with the question text to form a single *question* input. The details are summarized in Table 4.

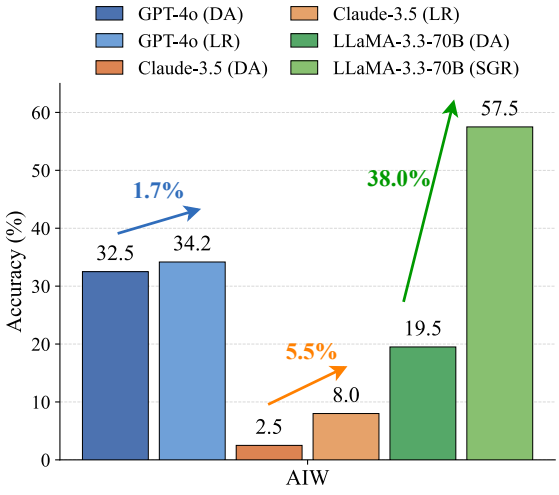


Figure 6: Comparison of accuracy under different reasoning paradigms on the AIW dataset. We evaluate three paradigms: Direct Answering (DA), Linear Reasoning (LR), and our Self-Graph Reasoning (SGR) across GPT-4o, Claude-3.5-haiku, and LLaMA-3.3-70B.

B Our Graph Reasoning Dataset

We construct our graph-based reasoning dataset based on the training set of LogiQA 2.0. The origi-

Method	General Domain			Specialized Domain		Overall Avg. (%)
	LogiQA	AIW	AR-LSAT	MedQA	MathQA	
LLaMA-3.1-8B	49.17	5.0	26.96	55.22	29.39	33.15
SGR-LLaMA-3.1-8B	42.05	2.50	24.68	60.36	35.03	32.92
SGR-LLaMA-3.1-8B w/ GRPO	56.68	5.5	24.35	59.47	36.25	36.45

Table 5: Accuracy of the base LLaMA-3.1-8B model, our SGR version, and the GRPO-fine-tuned version across all benchmarks.

Model	Cost ($)
GPT-4o	~26
Claude-3.5-haiku	~10
Gemini-2.5-Pro	~24
GPT-4o CoT	~80
Claude-3.5-haiku CoT	~32
SGR-Llama3.3-70B (Ours)	~33.6

Table 6: Comparison of the cost of our method with other LLMs on the LogiQA test set.

nal LogiQA 2.0 training corpus contains a total of 12,547 samples. After reasoning graph construction and data cleaning, we retain 9,869 high-quality samples for our graph reasoning benchmark. Examples of the constructed graph reasoning data is shown in Figure 8.

C Comparison with Existing Paradigms.

To further demonstrate the effectiveness of our self-graph reasoning (SGR) compared with existing reasoning paradigms, we conduct comparative experiments on the AIW dataset, which requires strong reasoning ability. Specifically, we evaluate three paradigms: direct answering, linear reasoning (CoT), and our SGR approach.

For CoT, we apply standard CoT prompting to both GPT-4o and Claude-3.5-Haiku. For comparison, we train LLaMA-3.3-70B with our SGR framework as the representative implementation of our method. The results are presented in Figure 6. The experimental results show that, compared to direct answering, GPT-4o achieves a 1.7% gain and Claude-3.5-Haiku achieves a 5.5% gain with CoT reasoning. In contrast, our SGR-LLaMA-3.3-70B achieves a remarkable 38% improvement, surpassing both proprietary models. These results demonstrate the effectiveness of graph-structured reasoning, which enables clearer and logically consistent reasoning processes.

D Analysis of Computational Cost

We compared the computational cost of our proposed SGR-LLaMA-3-70B model with several LLMs on the LogiQA benchmark. For our locally deployed model, the cost was estimated at $0.8 per GPU hour, requiring approximately 42 GPU hours in total. As shown in Table 6, our method incurs a total cost of about $33.6, which is substantially lower than that of GPT-4o (CoT) (approximately $80), while achieving comparable performance and offering a more interpretable, graph-structured reasoning process. Given that our model can be deployed locally with a moderate computational budget, these results highlight the efficiency and scalability of the SGR framework for reasoning tasks.

E GRPO Fine-Tuning

To mitigate the potential dilution of supervision on the final answer caused by intermediate reasoning steps, we perform an additional round of GRPO fine-tuning following standard SFT. This stage strengthens both the structured reasoning format and the accuracy of the final answer through two complementary reward functions. The first reward evaluates whether the model's output strictly adheres to the predefined structured reasoning–answer template, while the second reward assesses whether the content within the <*answer*> tag matches the label y_i.

We conduct GRPO fine-tuning based on the LLaMA-3.3-8B, using a batch size of 8, a maximum of 6000 steps, and a learning rate of 5×10^{-6}. This post-SFT training phase encourages the model to preserve structured reasoning while maximizing the correctness of the final answer. Results in Table 5 indicate that the additional GRPO fine-

Method	LogiQA	AIW	MedQA
Qwen3-8B	73.20	43.00	66.80
Qwen3-8B-thinking	80.40	80.00	75.80
GPT-5.1	76.34	57.00	89.55
SGR-LLaMA-3.3-70B	69.91	57.50	79.81

Table 7: Comparison of accuracy for our Self-Graph Reasoning (SGR) framework against recent reasoning LLMs, including Qwen-3-8B-thinking and other LLMs, on the LogiQA, AIW, and MedQA datasets.

tuning substantially enhances the model's answer accuracy, particularly on the in-domain LogiQA dataset, where it achieves a 7.51% improvement over the base model. This demonstrates that GRPO effectively mitigates the performance degradation associated with the limited capacity of small-scale LLMs. However, we also observe that the reasoning process of the GRPO-tuned model becomes overly concise, occasionally omitting intermediate reasoning steps. These findings suggest an inherent trade-off between answer accuracy and reasoning completeness in small-scale language models.

F Comparison with Reasoning LLMs

To assess the effectiveness of our Self-Graph Reasoning (SGR) framework, we compare its performance against the recent reasoning LLM Qwen-3-8B-thinking (Yang et al., 2025), evaluated on the LogiQA, AIW, and MedQA datasets, as shown in Table 7. Qwen-3B-thinking achieves strong results across all three benchmarks, even outperforming the proprietary GPT-5.1 model. This may be partly due to the inclusion of these public benchmarks in its pretraining or instruction-tuning data, which likely provides prior exposure to the evaluation distributions. In contrast, our SGR-LLaMA-3.3-70B model is fine-tuned with only a small amount of graph data and without any benchmark-specific priors, yet it still achieves competitive accuracy.

G Detail of Hyperparameter.

We list all the parameters used for SGR-Llama3.3 70B, as shown in Table 8. This includes configuration details such as batch size, learning rate, LoRA, and optimizer settings.

H Prompts

We provide the prompts used in our experiments. To generate candidate reasoning graphs, the model is prompted as illustrated in Figure 9. For integrating candidate graphs into an optimal reasoning graph, the model is prompted as shown in Figure 10. A standard question-answering prompt is constructed for the baseline LLMs, as depicted in Figure 11. Within our Self-Graph Reasoning (SGR) framework, the model is prompted as illustrated in Figure 12.

Hyperparameter	Value
seed	42
batch_size	8
num_epochs	1–3
learning_rate	5×10^{-6}
weight_decay	0.01
grad_steps	4
warmup	0.05
early_stop_patience	1
lora_r	4, 8
lora_alpha	8, 16
lora_dropout	0.1
lora_target_modules	q_proj, v_proj
max_txt_len	1024
max_new_tokens	2048

Table 8: Hyperparameters.

I Additional Case Study

To further illustrate the difference between linear reasoning and graph reasoning, we provide an additional case study in Figure 7. The example compares the reasoning traces generated by the two paradigms on the same question. While linear reasoning presents the reasoning process as a sequential textual trace, graph reasoning explicitly organizes intermediate reasoning units as nodes and their dependencies as edges. This makes the support relations among different reasoning steps more visible and easier to inspect. As shown in Figure 7, linear reasoning may contain useful intermediate information, but its long textual form makes it difficult to identify how each step contributes to the final answer. In contrast, the graph reasoning output separates key claims into structured nodes and connects them through directed dependencies, providing a clearer view of the reasoning path and the final decision.

Social self-organization refers to a social organization that is established spontaneously, develops independently, acts independently, has a certain scale, exists in the form of non-governmental organizations, and gradually moves from disorder to order.
According to the above definition, which of the following options doesn't belong to social self-organization?
Options:
A. A software enterprise in a city regularly organizes entrepreneurs to have business exchanges
B. A group of students get to know each other through QQ group and participate in respecting the elderly together every weekend
C. Every Saturday night, young people dance hip-hop in the square of a municipal park
D. A group of young people set up animal shelters to adopt all kinds of abandoned pets

Linear reasoning

To determine which option doesn't belong to social self-organization, let's analyze each option against the definition provided: Social self-organization must be established spontaneously, develop independently, act independently, have a certain scale, exist in the form of non-governmental organizations, and move from disorder to order.
-Option A: A software enterprise organizing business exchanges could be seen as a form of self-organization but might not fit perfectly if it's primarily driven by commercial interests rather than spontaneous, independent social action.
- Option B: A group of students organizing through a QQ group to participate in community service fits the definition of social self-organization as it's spontaneous, independent, and non-governmental.
- Option C: Young people dancing hip-hop in a park could be considered a form of social self-organization as it's a spontaneous gathering, though its scale and impact might vary.
- Option D: A group setting up animal shelters is a clear example of social self-organization as it's a non-governmental effort that develops independently to address a social issue.
Considering the definition, Option A seems less likely to fit because it involves a commercial enterprise, which might not align with the non-governmental, spontaneous nature emphasized in the definition of social self-organization.
Answer: A

Graph reasoning

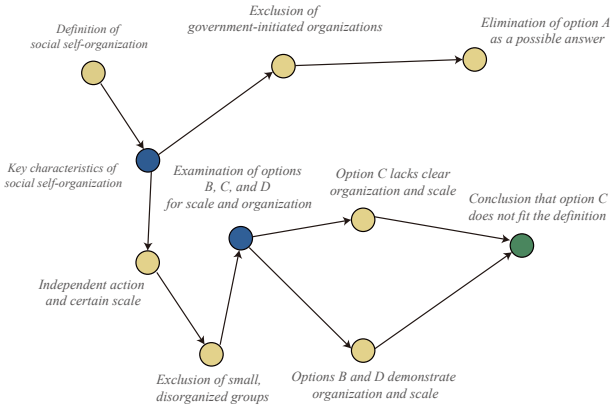


Figure 7: Additional case study comparing linear reasoning and graph reasoning. Linear reasoning presents the reasoning process as a sequential textual trace, while graph reasoning represents intermediate reasoning units and their dependencies as an explicit graph, making the reasoning process more structured and easier to inspect.

	LogiQA		AIW	
Criterion	Mean	≥ 3.5 (%)	Mean	≥ 3.5 (%)
Node correctness	3.64	64	4.00	68
Edge validity	3.36	48	4.12	68
Reasoning relevance	4.28	84	4.68	96
Reasoning completeness	3.24	52	4.00	68
Verifiability	3.56	56	4.28	76
Overall	**3.62**	**60.8**	**4.22**	**75.2**

Table 9: Human ratings of 50 sampled SGR-generated graphs. Threshold columns show the percentage scoring at least 3.5; Overall is the mean across the five criteria.

J Human Evaluation of Reasoning Graphs

Figures 5 and 7 show examples of SGR-generated graphs. We also evaluated 50 randomly sampled outputs from the LogiQA and AIW test sets. Table 9 reports human ratings for five graph-quality criteria.

AIW scores were higher on all five criteria. On LogiQA, edge validity (3.36) and reasoning completeness (3.24) had the lowest mean scores.

Question	Reasoning Path	Label
In a magic show, from the seven magicians - G.H.K.L.N.P and Q, select 6 to perform, perform into two teams: 1 team and 2 team. Each team by the front, middle and after three positions, playing the magician just each occupies a position, the selection and position arrangement of the magician must meet the following conditions :(1) if the arrangement of G or H play, they must be in the front. (2) If K is to play, he must be in the middle. (3) If L is to play, he must be in team 1. (4) Neither P nor K can be on the same team as N. (5) P cannot be in the same team as Q. (6) If H is in team 2, Q is in the middle position of team 1. If H is in team 2, which of the followings is listed as an acceptable arrangement for team 1? Options: A. Front: L Middle: Q Rear: N B. Front: G Middle: K Rear: N C. Front: L Middle: Q Rear: G D. Front: Q Middle: K Rear: L	["H is in team 2", "leads to", "Q is in the middle of team 1"] ["Q is in the middle of team 1", "leads to", "Evaluate possible positions for L"] ["Evaluate possible positions for L", "leads to", "L must be in team 1"] ["L must be in team 1", "leads to", "L can be in the front or rear of team 1"] ["L can be in the front or rear of team 1", "leads to", "Consider L in the front"] ["L can be in the front or rear of team 1", "leads to", "Consider L in the rear"] ["Consider L in the front", "leads to", "Evaluate rear position for team 1"] ["Consider L in the rear", "leads to", "Evaluate front position for team 1"] ["Evaluate rear position for team 1", "leads to", "N can be at the rear"] ["Evaluate front position for team 1", "leads to", "N can be in the front"] ["N can be at the rear", "leads to", "Front: L, Middle: Q, Rear: N"] ["N can be in the front", "leads to", "Front: N, Middle: Q, Rear: L"] ["Front: L, Middle: Q, Rear: N", "leads to", "Option A is valid"] ["Front: N, Middle: Q, Rear: L", "leads to", "Check validity based on other rules"] ["Check validity based on other rules", "leads to", "Option A satisfies all rules"] ["Option A is valid", "leads to", "Conclusion: Option A is an acceptable arrangement for team 1"]	A
Among people who have a history of chronic trouble falling asleep, some rely only on sleeping pills to help them fall asleep, and others practice behavior modification techniques and do not take sleeping pills. Those who rely only on behavior modification fall asleep more quickly than do those who rely only on sleeping pills, so behavior modification is more effective than are sleeping pills in helping people to fall asleep. Which one of the following, if true, most weakens the argument? Options: A. The people who are the most likely to take sleeping pills rather than practice behavior modification techniques are those who have previously had the most trouble falling asleep. B. People who do not take sleeping pills spend at least as many total hours asleep each night as do the people who take sleeping pills. C. Most people who have trouble falling asleep and who use behavior modification techniques fall asleep more slowly than do most people who have no trouble falling asleep. D. The people who are the most likely to practice behavior modification techniques rather than take sleeping pills are those who prefer not to use drugs if other treatments are available.	["Some people have trouble falling asleep", "leads to", "Some people use sleeping pills"] ["Some people have trouble falling asleep", "leads to", "Some people use behavior modification"] ["Some people use sleeping pills", "leads to", "People using pills fall asleep slower than those using behavior modification"] ["Some people use behavior modification", "leads to", "People using behavior modification fall asleep faster"] ["People using pills fall asleep slower than those using behavior modification", "leads to", "Behavior modification is more effective than pills"] ["Behavior modification is more effective than pills", "leads to", "Behavior modification helps people fall asleep faster than pills"] ["Some people use sleeping pills", "leads to", "People using pills may have had more trouble falling asleep initially"] ["People using pills may have had more trouble falling asleep initially", "leads to", "Initial severity of insomnia affects falling asleep speed"] ["Initial severity of insomnia affects falling asleep speed", "leads to", "Behavior modification may not inherently be more effective"] ["Behavior modification may not inherently be more effective", "leads to", "Option A weakens the argument"] ["People using pills may have had more trouble falling asleep initially", "leads to", "People with severe insomnia prefer sleeping pills"] ["People with severe insomnia prefer sleeping pills", "leads to", "Severity affects effectiveness comparison"] ["Severity affects effectiveness comparison", "leads to", "Ignoring severity difference weakens argument"] ["Ignoring severity difference weakens argument", "leads to", "Option A is correct"]	A
The so-called memory representation refers to the image of things perceived by people in the human brain. Which of the following activities is designed for memory content that does not belong to memory representation? Options: A. Tom still remembers her eating Quanjude roast duck in China. B. Lao Li often recalls the hard life of his childhood. C. The beautiful sound of the piano echoed in her ears. D. Xiao Hong can recite the multiplication formula skillfully.	["Memory representation is the image of things perceived", "leads to", "Memory representation involves sensory or emotional experiences"] ["Memory representation involves sensory or emotional experiences", "leads to", "Memory representation is not abstract knowledge"] ["Memory representation involves sensory or emotional experiences", "leads to", "Tom remembers eating roast duck is a sensory experience"] ["Memory representation involves sensory or emotional experiences", "leads to", "Lao Li recalls hard life is a sensory experience"] ["Memory representation involves sensory or emotional experiences", "leads to", "The sound of the piano is a sensory experience"] ["Memory representation is not abstract knowledge", "leads to", "Reciting the multiplication formula is abstract knowledge"] ["Tom remembers eating roast duck is a sensory experience", "leads to", "Option A relates to memory representation"] ["Lao Li recalls hard life is a sensory experience", "leads to", "Option B relates to memory representation"] ["The sound of the piano is a sensory experience", "leads to", "Option C relates to memory representation"] ["Reciting the multiplication formula is abstract knowledge", "leads to", "Option D does not relate to memory representation"] ["Option A relates to memory representation", "leads to", "Option A is not the answer"] ["Option B relates to memory representation", "leads to", "Option B is not the answer"] ["Option C relates to memory representation", "leads to", "Option C is not the answer"] ["Option D does not relate to memory representation", "leads to", "Option D is the answer"]	D

Figure 8: Samples of our graph reasoning data. Each sample consists of a question, its corresponding reasoning graph, and the correct answer (label).

Prompt for candidate reasoning graph generation

Question: **{question}**
Answer: **{answer}**

Construct a **concise reasoning graph** that explores the key reasoning paths leading to the correct answer.
The graph should capture multiple hypotheses or branches, but keep it **short and high-quality**: only 3-5 key steps plus the final conclusion.

Granularity rules:
- Each node must be an **atomic proposition**: one subject + one predicate (+ optional object/value).
- Split sentences by meaning: if a claim can be divided into smaller independent claims, make multiple nodes.
- No pronouns; always use the same canonical entity names.
- Each node should be concise and only include essential information.
- Avoid redundancy: do not repeat the same fact multiple times.

Graph rules:
- Start from the initial facts given in the question.
- Allow **branching** into multiple reasoning directions only when alternative possibilities exist.
- Branches should converge to the final conclusion that supports the correct answer.
- Each edge represents a **small, justified inference step**, formatted as ["premise", "leads to", "next step"].
- Keep the reasoning graph **connected from start to conclusion**.

Output format (strictly):
<reasoning>
["node1", "leads to", "node2"]
["node1", "leads to", "node3"]
["node2", "leads to", "node4"]
...
["nodeN", "leads to", "conclusion"]
</reasoning>

<answer>
A/B/C/D
</answer>

Constraints:
- Output only the <reasoning> graph and the <answer> block.
- Ensure reasoning is **concise**, **atomic**, and **high-quality**.
- Represent multiple branches when necessary, but limit the graph to key steps (3-5 + final conclusion).

Figure 9: Prompt for candidate reasoning graph generation.

Prompt for candidate reasoning graph integration

Question: **{question}**
Answer: **{answer}**

You are given three reasoning paths that attempt to explain the answer to this question.
Your task is to integrate them into a **single optimal reasoning graph** that includes multiple possible branches but ultimately converges on the correct answer.

Reasoning 1: **{reasoning1}**
Reasoning 2: **{reasoning2}**
Reasoning 3: **{reasoning3}**

Integration rules:
1. If two entities are similar or co-refer, merge them into one, keeping the more meaningful/specific name.
2. Each node must be an **atomic proposition** (one subject + one predicate + optional object/value).
3. If a sentence expresses multiple independent claims, split it into multiple nodes.
4. Each node must represent exactly one indivisible fact/idea.
5. Each edge must represent a **small justified inference step**, not a leap.
6. When evaluating options, base judgments only on explicit prior nodes.
7. At least one node must branch into ≥2 alternative reasoning directions before convergence.
8. All branches must remain connected and converge toward the final conclusion.

Output format:
<reasoning>
["thinking1", "leads to", "thinking2"]
["thinking1", "leads to", "thinking3"]
["thinking2", "leads to", "thinking4"]
["thinking3", "leads to", "thinking5"]
...
["thinkingN", "leads to", "conclusion"]
</reasoning>

<answer>
A/B/C/D
</answer>

Constraints:
- Output only the <reasoning> graph and the <answer> block, nothing else.
- All nodes must be atomic, unique, and quoted strings.
- Ensure at least one explicit branching before convergence to the conclusion.

Figure 10: Prompt for candidate reasoning graph integration.

Prompt for the baseline LLM

Question: **{question}**
Please provide the correct answer in the format: <answer>...</answer>
For example, if choosing option B, the output is: <answer> B </answer>.

Figure 11: Prompt for the baseline LLMs.

Prompt used in our Self-Graph Reasoning (SGR) framework

Question: **{question}**
Instruction:
1. Carefully think step by step to determine the correct answer.
2. Show your graph reasoning in a <reasoning> block.
3. Each <step> must be one edge: source node -> leads to -> target node.
4. After reasoning, always provide the final answer in <answer> tag.
5. Format strictly as:
<reasoning>
<step>node A -> leads to -> node B</step>
...
</reasoning>
<answer>...</answer>
5. Do NOT generate anything outside <reasoning> and <answer>.

Figure 12: Prompt used in our Self-Graph Reasoning (SGR) framework.