Neuro-Symbolic Integration Brings Causal and Reliable Reasoning Proofs *

Sen Yang [1 2 3] **Xin Li** [2 3 †] **Leyang Cui** **Lidong Bing** [2 3] **Wai Lam** [1]

[1] The Chinese University of Hong Kong

[2] DAMO Academy, Alibaba Group

[3] Hupan Lab, 310023, Hangzhou, China

senyang.stu@gmail.com {xinting.lx, l.bing}@alibaba-inc.com

neal19951101@gmail.com wlam@se.cuhk.edu.hk

Abstract

Though prompting LLMs with various reasoning structures produces reasoning proofs along with answers, these proofs are not ensured to be causal and reliable due to the inherent defects of LLMs. Tracking such deficiencies, we present a neuro-symbolic integration method, in which a neural LLM is used to represent the knowledge of the problem while an LLM-free symbolic solver is adopted to do deliberative reasoning using the knowledge. Specifically, our customized meta-interpreters allow the production of reasoning proofs and support flexible search strategies. These reasoning proofs are ensured to be causal and reliable because of the deterministic executing nature of the symbolic solvers. Empirically, on ProofWriter, our method surpasses the CoT baseline by nearly double in accuracy and more than triple in proof similarity. On GSM8K, our method also shows accuracy improvements and nearly doubled proof similarity. Our code is released at `https://github.com/DAMO-NLP-SG/CaRing`.

1 Introduction

Large language models (LLMs), like LLaMA (Touvron et al., 2023) and GPT-4 (OpenAI, 2023), are shown to be effective on several reasoning tasks but still struggle with structurally complex reasoning problems, such as logical reasoning (Tafjord et al., 2021) and arithmetic reasoning (Cobbe et al., 2021; Ribeiro et al., 2023). To tap into the potential of LLMs for better complex reasoning, existing works primarily focus on iteratively prompting LLMs to search over reasoning structures such as chains (e.g., CoT) (Wei et al., 2022; Wang et al., 2023; Zhou et al., 2023), trees (e.g., Tree-of-Thoughts) (Yao et al., 2023; Long, 2023), and graphs (e.g., Graph-of-Thoughts) (Besta et al., 2023; Zhang et al., 2023).

Despite the effectiveness of such methods over various complex reasoning problems, it is observed that they often give correct results with erroneous intermediate steps (Ye and Durrett, 2022; Saparov and He, 2023; Ribeiro et al., 2023). Specifically, Ribeiro et al. (2023) showed that even though prompting GPT-3 with structured intermediate steps yields an average accuracy of 33.84% on five complex reasoning datasets, the average similarity between the predicted and the gold reasoning proofs is merely 0.72% This discrepancy between reasoning accuracy and reasoning proof similarity raises pressing concerns about the reliability and causality of the underlying reasoning process in LLMs.

The discrepancies identified in the reasoning capabilities of LLMs underscore their limitations in emulating human-like deliberate reasoning. One natural solution could be adopting an LLM-free deliberative reasoning engine. Inspired by the seminal work of Kowalski (1979), which argued that a problem-solving algorithm benefits from separating the *logic* component (i.e., the knowledge which can be used to solve the problem) and the *control* component (i.e., the problem-solving strategy with which the knowledge can be used), we propose a neuro-symbolic approach consisting of two components: (1) LLM-based symbolic representation generator (SymGen; §3.1), which translates natural languages into formal knowledge representations that can be used for symbolic inference; (2) LLM-free symbolic inference engine (SymInfer; §3.2), which performs deliberative reasoning by executing the symbolic representationsThe execution strategy is implemented with our customized meta-interpreters, allowing (i) tracing of the reasoning process (§3.2.1); (ii) adoption of various search strategies (§3.2.2). Moreover, by putting LLMs under quarantine during deliberative reasoning, our approach produces reasoning traces that are strictly causal and immune from hallucinations.

* This work was supported by Alibaba Group through the Alibaba Innovative Research (AIR) Program (TA2217728).
† XL is the corresponding author.

To demonstrate its effectiveness in producing better reasoning proofs, we evaluate our CARING (Causal and Reliable Reasoning) on three reasoning datasets with reasoning proof annotations, including two logical reasoning datasets, ProofWriter (Tafjord et al., 2021) and PrOntoQA (Saparov and He, 2023), and one arithmetic reasoning dataset, GSM8K (Ribeiro et al., 2023). CARING consistently outperforms the baselines in terms of both accuracy and reasoning graph similarity. Specifically, our results demonstrate that CARING excels not only in the two logical reasoning datasets derived from logical formulations translated into natural languages, but also achieves enhanced accuracy and superior reasoning proof quality in the GSM8K dataset, which is constructed by human experts and reflects the flexibility of natural languages.

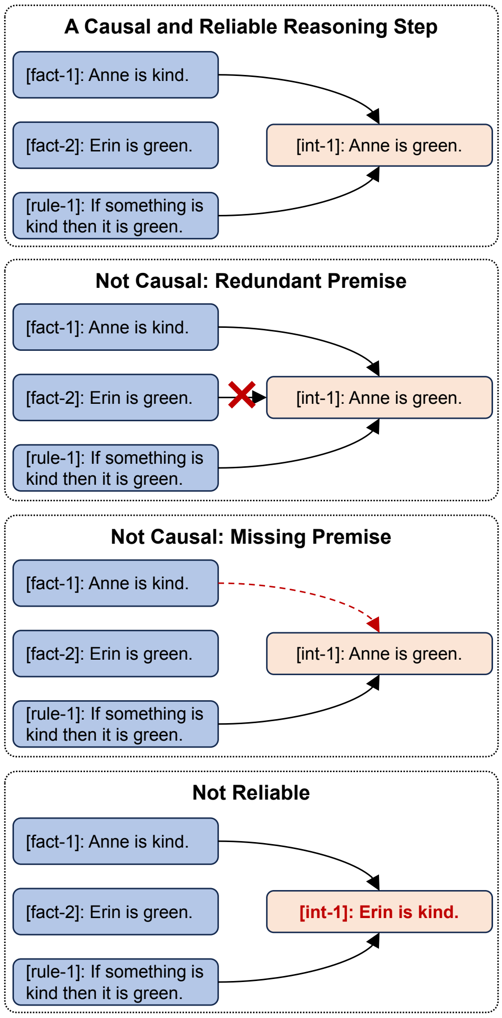

Figure 1: Illustrations of how causality and reliability play important roles in reasoning. LLMs may be (i) non-causal by selecting redundant premises or ignoring relevant ones and (ii) non-reliable by hallucinating incorrect contents during inference.

2 Related Work

2.1 Explainable Complex Reasoning

The Chain-of-Thought prompting method, which found out reasoning with LLMs benefits from generating intermediate steps, has sparked a recent trend in how to better do reasoning while remaining explainable. Several works investigated using other reasoning structures, such as trees (Yao et al., 2023; Long, 2023) and graphs (Besta et al., 2023; Zhang et al., 2023). These approaches have shown improved performance, particularly in complex reasoning tasks where the processes involved are often more intricate than simple linear chains. However, despite the alignment of their reasoning proof structures with the gold-standard proofs, these methods still face challenges in ensuring causality and reliability. This limitation stems from their reliance on LLMs for deliberative reasoning, which are prone to hallucinations and may compromise causality.

Some other recent works adopted a less structured manner (Tafjord et al., 2022; Creswell et al., 2023; Kazemi et al., 2023). For example, Selection-Inference (Creswell et al., 2023) divides the reasoning process into two phases: The Selection phase selects the premises that might be relevant for the next round of inference; The Inference phase conducts a single reasoning step with the selected knowledge fragments.

2.2 Neuro-symbolic Reasoning

Neuro-symbolic systems attempt to leverage the strengths of both neural networks and symbolic reasoning (Andreas et al., 2016; Neelakantan et al., 2017; Hudson and Manning, 2019; Gupta et al., 2020; Nye et al., 2021). This includes the use of neural networks for pattern recognition and learning from unstructured data, integrated with symbolic systems for rule-based reasoning and knowledge representation. Despite significant progress, neuro-symbolic reasoning faces challenges, notably in scalability and the efficient integration of learning and reasoning components.

Recent advancements in neuro-symbolic research, particularly in reasoning over text, have utilized LLMs to encapsulate knowledge from unstructured human languages, as noted in Lyu et al. (2023); Pan et al. (2023). These methods typically translate natural language into symbolic representations for subsequent execution-based reasoning.

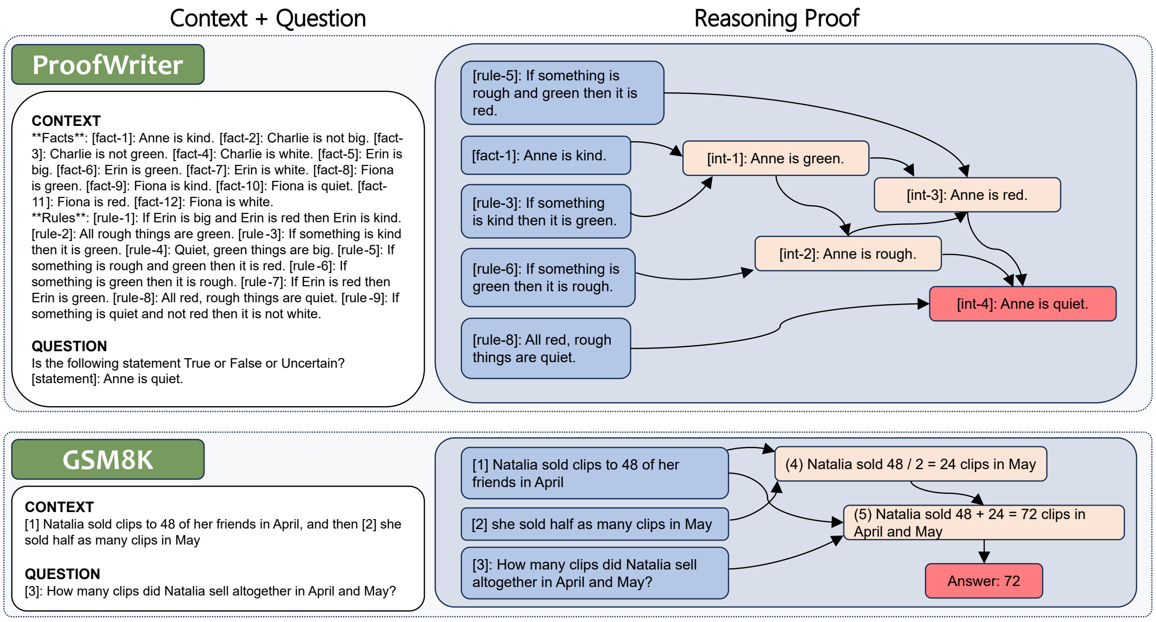

Figure 2: Two examples of complex/structured reasoning problems from ProofWriter and GSM8K, respectively. The reasoning proofs in such problems formulate directed acyclic graphs (DAGs) in a multi-step and multi-premise manner.

However, they have not fully explored the capabilities of symbolic solvers in generating detailed reasoning proofs. In contrast, our approach leverages customized meta-interpreters in conjunction with symbolic solvers to uncover and articulate the underlying reasoning proofs. This not only enhances the transparency of automatic reasoning systems but also simplifies the process for humans to verify their correctness and safety.

3 CaRing

The problems we focus on are featured with structured or complex reasoning. As depicted in Figure 2, these problems typically necessitate multi-step and multi-premise reasoning over a directed acyclic graph (DAG), where individual nodes signify distinct knowledge fragments and directed edges denote reasoning steps. Each reasoning step uses existing knowledge to infer new relevant knowledge. Numerous knowledge fragments are often aggregated to infer a new one, which we denote as "multi-premise". The solver usually performs multiple such steps to reach an ultimate goal, which we denote as "multi-step". This entire reasoning process naturally composes a DAG.

We are interested in providing accurate answers along with causal and reliable explanations for such reasoning problems. This motivates our investigation of LLM-free deliberative reasoning engines. The seminal work of Kowalski (1979) proposed that *Algorithm* = *Logic* + *Control*, where *logic* refers to the knowledge which can be used to solve the problem and *control* refers to the problem-solving strategy in which the knowledge can be used. They further proved that an algorithm benefits from separating the *logic* component and the *control* component. Inspired by this, we present CaRing (Causal and Reliable Reasoning), a modular approach consisting of two components:

- SymGen: LLM-based symbolic representation generator (§3.1), which translates natural languages into formal symbolic knowledge representations that can be used for symbolic inference. A major difference between previous work and our method is that we only use LLMs to represent knowledge but not to do deliberative reasoning.

- SymInfer: LLM-free symbolic inference engine (§3.2), which performs deliberative reasoning by executing the symbolic representations provided by SymGen. By implementing customized meta-interpreters, SymInfer supports (i) causal and reliable tracing of the reasoning process (§3.2.1); (ii) various search strategies, such as Depth-First Search (DFS) and Iterative Deepening Search (IDS) (§3.2.2).

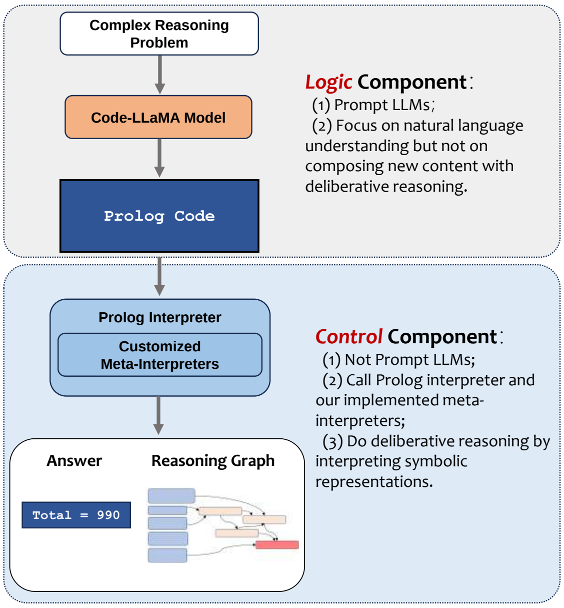

Figure 3: Illustration of our CaRing framework, consisting of a *Logic* component and a *Control* component.

The execution-based tracing approach of SymInfer guarantees both causality and reliability. Under the principle of **Causality**, the inference of a new knowledge piece is strictly linked to those existing fragments that are relevant, ensuring precise and limited attribution. This implies that a causal relationship is established only when the preceding event (at the base of the edge) directly influences the subsequent event (at the apex). Regarding **Reliability**, the content within each newly inferred node is the result of a deterministic process, safeguarding it from the kinds of erroneous hallucinations often encountered in outputs from LLMs.

3.1 SymGen: Symbolic Representation Generator

To represent *logic* (i.e., the knowledge which can be used to solve the problem), we adopt a popular logic programming language, Prolog (Colmerauer and Roussel, 1996). Prolog is a declarative programming language, in which *logic* is expressed as relations (called Facts and Rules), with several examples shown in Table 1. A computation is initiated by running a query over these relations. We will delve into the computation of Prolog in §3.2.

Though LLMs are prone to hallucinate erroneous facts when composing new knowledge, they are shown to be powerful at understanding natural languages and directly translating them into other formats (Ye and Durrett, 2022; Saparov and He, 2023). To utilize such a strong point while avoiding the defect, we only use LLMs to translate natural languages into Prolog representations but not to do deliberative reasoning. Specifically, we few-shot prompt LLMs with several human-written in-context demonstrations, each containing a problem and corresponding Prolog representations, which are later used for symbolic inference.

Natural Language	Prolog Code
Fiona is green.	`green(fiona).`
All red, rough things are quiet.	`quiet(X) :-` `    red(X), rough(X).`
Tina makes $18.00 an hour.	`wage(18.00).`

(she is eligible for overtime,) which is paid by your hourly wage + 1/2 your hourly wage.	`overtime_wage(W) :-`
	`    wage(W1),`
	`    W is 1.5 * W1.`

Table 1: Examples of natural languages and their Prolog representations. It can be seen that the Prolog code is highly declarative, so the LLM in SymGen is only required to do straightforward natural language understanding and translation but not reasoning. In other words, the LLM does not need to infer new knowledge, thus avoiding hallucination as much as possible.

3.2 SymInfer: Symbolic Inference Engine

We use SymInfer to produce answers and reasoning traces by executing the aforementioned symbolic representations. Since we adopt Prolog to represent knowledge, our symbolic inference engine is naturally instantiated with Prolog interpreters. By default, the SWI-Prolog (Wielemaker et al., 2012) interpreter adopts the Depth-First Search (DFS) backtracking strategy and does not yield reasoning proofs. We implement customized Prolog-based meta-interpreters to achieve two goals: (i) To produce reasoning proofs; (ii) To adopt better search algorithms other than DFS.

3.2.1 Reasoning Tracer

We implement a Prolog meta-interpreter to show the reasoning proofs:

```
% Define the operator for proofs
:- op(750, xfy, =>).

% Proof tree generation
mi_tree(true, true).
mi_tree((A,B), (TA,TB)) :-
        mi_tree(A, TA),
        mi_tree(B, TB).
mi_tree(G, builtin(G)) :-
        predicate_property(G, built-in
    ),
```

```prolog
        !,
        call(G).
mi_tree(g(G), TBody => G) :-
        mi_clause(G, Body),
        mi_tree(Body, TBody).
```

3.2.2 Search Strategy

The default search strategy of Prolog is DFS, which may lead to infinite loops. For example, given the knowledge base:

```prolog
parent_of(X, Y) :- offspring_of(Y, X).
offspring_of(X, Y) :- parent_of(Y, X).
parent_of(X, Y) :- mother_of(X, Y).
parent_of(X, Y) :- father_of(X, Y).
mother_of(jack, anna).
```

and a query `?- parent_of(jack, Who).` , the backtracking process would repeat over the first two lines without resorting to other lines due to DFS. To address this issue, we adopt Iterative Deepening Search (IDS), in which the backtracking process performs a series of depth-limited searches, each with an increasing depth limit. This leverages the strengths of both Breadth-First Search (BFS) and DFS. Our Prolog meta-interpreter for IDS is implemented as:

```prolog
% Depth-limited meta-interpreter with
    proof tree generation
mi_limit(true, true, N, N).
mi_limit((A,B), (TA,TB), N0, N) :-
        mi_limit(A, TA, N0, N1),
        mi_limit(B, TB, N1, N).
mi_limit(g(G), TBody => G, N0, N) :-
        N0 #> 0,
        N1 #= N0 - 1,
        mi_clause(G, Body),
        mi_limit(Body, TBody, N1, N).

% Iterative deepening with proof tree
    generation
mi_id(Goal, Proof) :-
        length(_, N),
        mi_limit(Goal, Proof, N, _).

% Iterative deepening with maximum
    depth with proof tree generation
mi_id_limit(Goal, Proof, MaxDepth) :-
        between(1, MaxDepth, N),
        mi_limit(Goal, Proof, N, _).
```

In practice, other search strategies can also be implemented according to the nature of the target problems, such as Uniform-Cost Search (UCS) and Beam Search.

4 Experiments

We briefly introduce our experimental settings in §4.1 and show the experiment results in §4.2.

4.1 Experimental Settings

We present our experimental settings in this section, including our implementation details of the two components (§4.1.1), a brief introduction of the adopted datasets (§4.1.2) and the baselines (§4.1.3).

4.1.1 Implementation

SymGen We adopt the Code-LLaMA (Rozière et al., 2023) family as the base LLMs to translate natural languages into Prolog representations. Our prompting paradigm is in a pure few-shot in-context-learning (ICL) prompting style, without detailed human-written instructions. Each ICL demonstration comprises a question and a piece of Prolog code.

SymInfer We adopt SWI-Prolog (Wielemaker et al., 2012) and PySwip[1] packages to implement the symbolic inference engine. We set the maximum depth to be 20 for Iterative Deepening Search and the number of generated reasoning paths to 20.

4.1.2 Datasets

We evaluate CaRing on three popular complex reasoning datasets, including two logical reasoning datasets (ProofWriter (Tafjord et al., 2021) and PrOntoQA (Saparov and He, 2023)) and one arithmetic dataset (GSM8K (Cobbe et al., 2021; Ribeiro et al., 2023)).

ProofWriter ProofWriter (Tafjord et al., 2021) is a commonly-used logical reasoning dataset. It contains many small rulebases of facts and rules, expressed in English. Each rulebase has a set of questions (English statements) which can either be proven true or false using proofs of various depths, or the answer is "Unknown" (in open-world setting, OWA) or assumed negative (in closed-world setting, CWA). The proofs can naturally be represented as directed acyclic graphs (DAGs). The dataset is divided into several sub-sets according to maximum proof depth, namely $\{0, \leq 1, \leq 2, \leq 3, \leq 5\}$. We adopt the most difficult ≤ 5-depth sub-set in the OWA setting, which contains 482 rulebases and 10,190 questions.

PrOntoQA PrOntoQA (Saparov and He, 2023) is a synthetic question answering dataset designed for diagnosing the logical reasoning ability of LLMs. Each example aims to validate the feasibility of a statement given a context. Among various sub-sets, we follow Pan et al. (2023) to adopt

[1] https://github.com/yuce/pyswip

the most difficult depth-5 *fictional characters* subset, which contains 500 statement-context pairs. Similar to ProofWriter, the proofs provided by the dataset can be naturally represented as DAGs.

GSM8K GSM8K (Cobbe et al., 2021) is a multi-step arithmetic reasoning dataset composed of high-quality grade school math word problems. We adopt the version released by Ribeiro et al. (2023), whose test set contains 270 questions annotated with structured reasoning proofs in the format of DAGs.

4.1.3 Baselines

All baselines prompt the LLMs with few-shot in-context-learning (ICL) demonstrations.

Direct Direct prompting prompt LLMs with answer-only ICL demonstrations. For example, a demonstration of GSM8K looks like:

````
**Problem No.1**
```
sent-1: Jesse and Mia are competing in a week long race.
sent-2: They have one week to run 30 miles.
sent-3: On the first three days Jesse averages (2/3) of a mile.
sent-4: On day four she runs 10 miles.
sent-5: Mia averages 3 miles a day over the first 4 days.
sent-6: What is the average of their average that they have
        to run over the final three days?
```

*Answer*:
#### 6
````

Chain-of-Thought (CoT) CoT prompting (Wei et al., 2022) prompts LLMs with ICL demonstrations that contain both intermediate reasoning steps and answers. An example demonstration of GSM8K looks like:

````
**Problem No.1**
```
sent-1: Jesse and Mia are competing in a week long race.
sent-2: They have one week to run 30 miles.
sent-3: On the first three days Jesse averages (2/3) of a mile.
sent-4: On day four she runs 10 miles.
sent-5: Mia averages 3 miles a day over the first 4 days.
sent-6: What is the average of their average that they have
        to run over the final three days?
```

*Reasoning Process*:
sent3 -> int1: Jesse runs 2 miles in the first three days
        because 3 x (2/3) = 2;
int1 & sent2 & sent4 -> int2: Jesse has 18 miles left to
        run because 30 - 10 - 2 = 18;
int2 & sent6 -> int3: Jesse has to run an average of 6 miles
        a day because 18 / 3 = 6;
sent5 -> int4: Mia runs 12 miles over the first four days
        because 4 x 3 = 12;
int4 & sent2 -> int5: She has 18 miles left to run because
        30 - 12 = 18;
int5 & sent6 -> int6: She has to run six miles a day
        because 18 / 3 = 6;
int3 & int6 -> int7: The total they both have to run is
        12 miles a day;
int7 & sent6 -> int8: The average they have to run per
        day on average is 6 miles because 12 / 2 = 6;
int8 & sent6 -> int9: The answer is 6;
#### 6
````

		Acc	Proof Sim	
			All	**Correct**
7B	Direct	41.78	–	–
	Direct (3-Shot)	43.32	–	–
	CoT	40.95	11.27	17.20
	CoT (3-shot)	42.58	11.52	21.58
	Ours	92.43	75.85	**86.68**
13B	Direct	43.44	–	–
	Direct (3-shot)	44.31	–	–
	CoT	45.88	16.16	27.32
	CoT (3-shot)	54.70	23.18	32.48
	Ours	96.16	80.74	86.34
34B	Direct	44.00	–	–
	Direct (3-shot)	45.93	–	–
	CoT	52.32	15.08	26.30
	CoT (3-shot)	56.50	24.12	34.61
	Ours	**98.11**	**83.17**	85.65

Table 2: Results on ProofWriter. "All" and "Correct" refer to "on all instances" and "on correctly-predicted instances", respectively. "Proof Sim" refers to "Proof Graph Similarity" while "Proof EM" means "Proof Graph Exact Match". The default setting is 2-shot. We additionally conduct 3-shot experiments for baselines to include all types of labels in the in-context demonstrations because this dataset contains three labels: {`true`, `false`, `uncertain`}. We do not conduct 3-shot experiments for our method because it is not sensitive to the number of labels due to its reasoning-by-execution nature.

4.1.4 Evaluation Metrics

Following Ribeiro et al. (2023), we evaluate both the answers and the generated reasoning proofs.

Answer Accuracy Answer accuracy measures a model's ability to predict the correct answer. A prediction is deemed correct if it is (i) the same as the gold option for multi-choice problems and (ii) the same integer as the gold answer for arithmetic reasoning problems. This metric is the upper bound for other metrics since a reasoning graph would be marked as incorrect without evaluation if the answer is marked as incorrect. We report this metric for all datasets.

Reasoning Graph Similarity As shown in Figure 2, the problems that we are interested in naturally compose reasoning graphs in the format of

	Acc (%)		
	7B	**13B**	**34B**
Direct	51.40	56.00	58.20
CoT	52.00	61.00	82.80
Ours	98.80	99.40	**100.00**

Table 3: Results on PrOntoQA across different model sizes. The default setting is 2-shot.

		Acc	Proof Sim	
			All	**Correct**
7B	Direct	4.44	–	–
	CoT	13.70	4.99	36.39
	Ours	12.22	6.57	53.72
13B	Direct	5.55	–	–
	CoT	15.56	5.76	37.03
	Ours	21.48	11.66	**54.26**
34B	Direct	8.15	–	–
	CoT	35.19	13.04	37.07
	Ours	**42.22**	**22.91**	54.25

Table 4: Results on GSM8K. The default setting is 5-shot.

directed acyclic graphs (DAGs). Reasoning graph similarity $\mathrm{sim}(\mathcal{G}_g, \mathcal{G}_p)$ measures the graph similarity between the gold and the predicted reasoning graphs. We follow Ribeiro et al. (2023) to adopt the graph edit distance function $\delta(\mathcal{G}_g, \mathcal{G}_p)$. This function quantifies the graph edit distance by determining the minimum number of operations required over nodes and edges to transform one graph into the other, thereby enabling a comparison of $\mathcal{G}_g$ and $\mathcal{G}_p$ based on their structural similarities. The reasoning graph similarity is normalized to $[0, 1]$ as:

$$\mathrm{sim}(\mathcal{G}_p, \mathcal{G}_g) = 1 - \frac{\delta(\mathcal{G}_p, \mathcal{G}_g)}{\max\{|N_p| + |E_p|, |N_g| + |E_g|\}} \tag{1}$$

where $|N_p|$ and $|E_p|$ denote the count of nodes and edges, respectively, within the predicted reasoning graph. A similar notation applies to $|N_g|$ and $|E_g|$, which represent the number of nodes and edges in the gold graph. Note that the reasoning graph similarity is set to zero if the predicted answer is incorrect. We report this metric for ProofWriter and GSM8K.

4.1.5 Inference Cost

All baselines as well as CaRing prompt the LLM only once per instance. Though the decoding outputs of CaRing are the longest among all, the extra cost is trivial with the help of highly optimized LLM inference packages like vLLM (Kwon et al., 2023).

4.2 Main Results

Tables 2, 3 and 4 show the experimental results on ProofWriter, PrOntoQA and GSM8K, respectively.

ProofWriter The results on ProofWriter are presented in Table 2. CaRing demonstrates notable improvements over existing baselines, particularly in the aspect of reasoning proof similarity. Utilizing the largest 34B model, CaRing achieves a remarkable accuracy rate of 98.11% and a reasoning proof similarity of 83.17%, significantly surpassing the leading baseline by nearly double in accuracy (98.11% vs. 56.50%) and more than triple in proof similarity (83.17% vs. 24.12%).

PrOntoQA The results on PrOntoQA are presented in Table 3. CaRing achieves full accuracy with the 34B model. This is predictable since PrOntoQA was constructed with fairly simple languages, only involving sentences like "*XX is/are YY*".

GSM8K The results on GSM8K are presented in Table 4. This dataset is much more challenging than the previous two logical reasoning datasets for CaRing since it is generally believed that symbolic languages are restricted by their limited expressiveness and cannot properly handle the ambiguity in real-world human languages. Surprisingly, with the 34B model, CaRing outperforms the strongest baseline with a large margin and almost doubles the reasoning proof similarity (22.91% vs. 13.04%). We attribute such improvements to the generalization ability of LLMs.

5 Conclusion

This paper addresses the erroneous reasoning proof problem of relying solely on LLMs for complex reasoning. Specifically, we develop a neuro-symbolic method to produce high-quality reasoning proofs for complex reasoning problems. By implementing customized meta-interpreters for executing Prolog representations and putting LLMs under quarantine during the reasoning phase, our method ensures the reasoning proofs are strictly causal and reliable. We conduct experiments on two synthetic logical reasoning datasets and one human-written arithmetic reasoning dataset. Experimental results demonstrate our method achieves significant improvements in terms of both answer accuracy and reasoning proof similarity.

Acknowledgements

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 39–48. IEEE Computer Society.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2023. Graph of thoughts: Solving elaborate problems with large language models.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *ArXiv preprint*, abs/2110.14168.

Alain Colmerauer and Philippe Roussel. 1996. *The Birth of Prolog*, page 331–367. Association for Computing Machinery, New York, NY, USA.

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2023. Selection-inference: Exploiting large language models for interpretable logical reasoning. In *The Eleventh International Conference on Learning Representations*.

Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner. 2020. Neural module networks for reasoning over text. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Drew A. Hudson and Christopher D. Manning. 2019. Learning by abstraction: The neural state machine. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5901–5914.

Mehran Kazemi, Najoung Kim, Deepti Bhatia, Xin Xu, and Deepak Ramachandran. 2023. LAMBADA: Backward chaining for automated reasoning in natural language. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6547–6568, Toronto, Canada. Association for Computational Linguistics.

Robert Kowalski. 1979. Algorithm = logic + control. *Commun. ACM*, 22(7):424–436.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Jieyi Long. 2023. Large language model guided tree-of-thought.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. *ArXiv preprint*, abs/2301.13379.

Arvind Neelakantan, Quoc V. Le, Martín Abadi, Andrew McCallum, and Dario Amodei. 2017. Learning a natural language interface with neural programmer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Maxwell Nye, Michael Henry Tessler, Joshua B. Tenenbaum, and Brenden M. Lake. 2021. Improving coherence and consistency in neural sequence models with dual-system, neuro-symbolic reasoning.

OpenAI. 2023. Gpt-4 technical report.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. 2023. Logic-LM: empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the 2023 Conference on Empirical Methods in Natural Language Processing (Findings of EMNLP)*, Singapore.

Danilo Neves Ribeiro, Shen Wang, Xiaofei Ma, Henghui Zhu, Rui Dong, Deguang Kong, Juliette Burger, Anjelica Ramos, zhiheng huang, William Yang Wang, George Karypis, Bing Xiang, and Dan Roth. 2023. STREET: A MULTI-TASK STRUCTURED REASONING AND EXPLANATION BENCHMARK. In *The Eleventh International Conference on Learning Representations*.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2023. Code llama: Open foundation models for code.

Abulhair Saparov and He He. 2023. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations*.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.

Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. 2022. Entailer: Answering questions with faithful and truthful chains of reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2078–2093, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Jan Wielemaker, Tom Schrijvers, Markus Triska, and Torbjörn Lager. 2012. SWI-Prolog. *Theory and Practice of Logic Programming*, 12(1-2):67–96.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate problem solving with large language models.

Xi Ye and Greg Durrett. 2022. The unreliability of explanations in few-shot prompting for textual reasoning. In *Advances in Neural Information Processing Systems*.

Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew Chi-Chih Yao. 2023. Cumulative reasoning with large language models. *ArXiv preprint*, abs/2308.04371.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

A Example Appendix

This is an appendix.