Non-Interactive Symbolic-Aided Chain-of-Thought for Logical Reasoning

Phuong Minh Nguyen, Tien Huu Dang, Naoya Inoue

Non-Interactive Symbolic-Aided Chain-of-Thought for Logical Reasoning

Phuong Minh Nguyen and **Tien Huu Dang** and **Naoya Inoue**

Japan Advanced Institute of Science and Technology
{phuongnm,tiendh,naoya-i}@jaist.ac.jp
Correspondence: phuongnm@jaist.ac.jp

Abstract

This work introduces Symbolic-Aided Chain-of-Thought (CoT), an improved approach to standard CoT, for logical reasoning in large language models (LLMs). The key idea is to integrate lightweight symbolic representations into few-shot prompts, structuring the inference steps with a consistent strategy to make reasoning patterns more explicit within a non-interactive reasoning process. By incorporating these symbolic structures, Symbolic-Aided CoT preserves the generalizability of standard prompting techniques while enhancing the transparency, interpretability, and analyzability of LLM logical reasoning. Extensive experiments on four well-known logical reasoning benchmarks—ProofWriter, FOLIO, ProntoQA, and LogicalDeduction, which cover diverse reasoning tasks and scenarios—demonstrate the effectiveness of the proposed approach, particularly in complex reasoning tasks that require navigating multiple constraints or rules. Notably, Symbolic-Aided CoT consistently improves LLMs' reasoning capabilities across various model sizes and significantly outperforms conventional CoT on three out of four datasets, ProofWriter, ProntoQA, and LogicalDeduction.

1 Introduction

In recent years, pre-trained Large Language Models (LLMs) have achieved exceptional success across a wide spectrum of Natural Language Processing (NLP) tasks (Wei et al., 2022; Shin and Van Durme, 2022; Dubey et al., 2024; Yang et al., 2025). As a result, LLMs have become a central paradigm in NLP research and applications. Their impressive performance is largely attributed to their ability to perform few-shot in-context learning—the mechanism by which models infer solutions based solely on the format and structure of the input prompt, without requiring gradient computations (Brown et al., 2020; Garg et al., 2022; Wei et al., 2022).
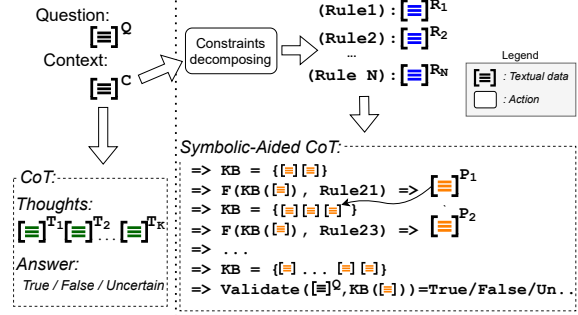

Figure 1: Comparison between standard CoT and Symbolic-Aided CoT for logical reasoning tasks.

Notably, as model size grows, prompting methods that leverage intermediate reasoning steps consistently surpass standard input–output prompting methods. This reasoning strategy, known as Chain-of-Thought prompting (CoT; Wei et al. (2022)), relies on explicitly modeling the reasoning process. For in-context learning, the CoT reasoning technique has demonstrated compelling results across a variety of NLP tasks (Wei et al., 2022; Zhou et al., 2023). Despite recent advancements, applying LLMs to logical reasoning tasks still faces several critical challenges, including conflicts between pretrained knowledge and counterfactual assumptions (Ortu et al., 2024), failures on cyclic inference graphs (Zhang et al., 2025), and planning errors during the solving process (Ye et al., 2023).

To address these issues, various strategies have been proposed, which can be broadly categorized into two main groups: (1) designing an external symbolic solver, which delegates the actual reasoning process to an automated theorem prover (Ye et al., 2023; Pan et al., 2023) or programming languages (Gao et al., 2023; Xu et al., 2024a); and (2) constructing a framework that systematically decomposes complex tasks into subtasks—such as rule selection, premise inference, and scoring—to enhance the overall reliability of the system (Zhang

et al., 2025; Feng et al., 2024; Sun et al., 2024; Xu et al., 2024b). Although the first approaches potentially achieve remarkable performance, they typically require powerful LLMs (Ye et al., 2023; Gao et al., 2023) such as GPT-4 (Achiam et al., 2023) or additional extensive pre-training phase (Xu et al., 2024a; Feng et al., 2024) for successful parsing from problem description to the logical forms (*e.g.,* First-order logics - FOL).

In this study, we focus on the second approach, which aims to improve LLM logical reasoning *without relying on any external symbolic reasoner or programming language*. Building on insights from recent works (Sun et al., 2024; Qi et al., 2025; Zhang et al., 2025), we target the challenge of designing a mechanism to systematically decompose complex logical reasoning tasks into simpler subtasks that can be solved by the inherent understanding ability of LLMs in a single inference pass. Closely related to our work, Xu et al. (2024a); Feng et al. (2024) recently introduced LLM-based frameworks that leverage first-order logic—*a strict formal language* with well-defined syntax and semantics—to support faithful logical reasoning. However, previous studies (Sun et al., 2024; Xu et al., 2024b; Zhang et al., 2025) primarily rely on self-refinement or interactive (multi-turn) generation, where each turn solves a subtask and its output is passed to the next. They overlook non-interactive reasoning, in which the LLM performs the entire reasoning process without any assistance from external modules or subprocesses. To address this gap, we explore *non-interactive (single-turn) reasoning generation*, allowing a more direct and fair comparison with CoT prompting.

We introduce Symbolic-Aided CoT, a novel variant of CoT (Wei et al., 2022), designed to leverage symbolic representations to enhance the non-interactive logical inference capabilities of LLMs (Figure 1). In conventional CoT, intermediate reasoning steps are provided in few-shot demonstrations as unstructured text, enabling LLMs to approximate the logical reasoning process. However, relying solely on textual descriptions for complex reasoning introduces ambiguity, as the inherent vagueness of natural language limits LLMs' ability to generalize precise reasoning steps. Our core idea is to integrate *lightweight* symbolic structures into few-shot prompts, making the inference steps more transparent and structured, while simultaneously strengthening the induction-head behavior of LLMs (Olsson et al., 2022). Specifically, our Symbolic-Aided CoT prompting framework explicitly outlines essential reasoning substeps: *rule matching*—selecting constraint rules that align with the current state of inference, *new premise inference*—applying the selected rules to generate new premises, and *knowledge base (KB) updating*—appending the newly inferred premises to the KB. By incorporating these symbolic structures, our method preserves the flexibility and general applicability of standard prompting techniques while enhancing both the interpretability and analyzability of LLM reasoning behavior. Empirical evaluations across four reasoning QA benchmarks, ProofWriter, FOLIO, ProntoQA, and LogicalDeduction, demonstrate the effectiveness of our approach, particularly in scenarios involving complex reasoning that requires navigating multiple rules and constraints. Remarkably, our Symbolic-Aided CoT, when applied with open-source LLMs (*e.g.,* Qwen3), achieves performance comparable to that of a complex multi-turn reasoning framework (Sun et al., 2024) built on the powerful GPT-4 model, particularly on the ProofWriter, ProntoQA, and LogicalDeduction datasets.

The remainder of this paper is organized into five sections. Section 2 provides an overview of logical reasoning tasks and compares our approach with prior studies. Section 3 presents the details of our proposed framework and its variants. Section 4 describes the experimental setup and reports the results, with key findings discussed in Section 5. Finally, the conclusions summarize our contributions and outline directions for future work are presented in Section 6.

2 Background and Related Work

2.1 Background

Notation and problem formulation. Logical reasoning is a fundamental NLP task within the Question Answering domain (Weston et al., 2015; Tafjord et al., 2021). In this task, machine learning models are required to answer a question based on a context containing multiple logical conditions or constraints. Formally, we denote the list of rules (or constraints) provided in the context as $\mathcal{R} = \{r_i\}_{0 \leq i < N}$. Given a question (Q), the correct answer (A) must be derived from knowledge supported by a subset of rules $\mathcal{R}^* \subset \mathcal{R}$. To address the challenges posed by logical reasoning, prior research largely falls into two overarching directions:

utilizing *external symbolic solvers* and developing *LLM-based logical solvers*.

In the first approach, the main idea is to leverage LLMs to translate textual descriptions of constraints and the question into formal logical formulas, which are then passed to an explicit symbolic solver (*e.g.,* the Z3 theorem prover[1]) to derive the final answer. Formally, all constraints are aggregated to construct a logical program: $\mathcal{F} = \{\text{LLM}^{\text{lang2logic}}(r_i)\}_{r_i \in \mathcal{R}}$. Similarly, the question is also transformed to the logical form $f^q = \text{LLM}^{\text{lang2logic}}(Q)$. Then, a symbolic reasoner is reasoned over the transformed logical forms to yield the final answer: $\text{SymbolicReasoner}(\mathcal{F}, f^q)$.

In the second approach, the original logical reasoning problem is decomposed into a sequence of subtasks. Each subtask is solved individually, and the process iterates over multiple turns until a specified stopping condition is satisfied, ultimately producing the final result: $\text{Loop}([\text{LLM}^{\text{subtask}_t}(\mathcal{R}, Q)]_t)$ where subtask$_t$ may represent an arbitrary LLM-based unit function such as *rule matching*, *rule inference*, or *new premise scoring*, among others. These sub-tasks are typically carefully designed and arranged with sequential dependencies, with the aim of mitigating hallucinations.

2.2 Related Works

Building on the success of LLMs across a wide range of NLP tasks (Chung et al., 2024; Dubey et al., 2024; Yang et al., 2025), logical reasoning has emerged as a particularly important area of study, serving as a key benchmark for evaluating both the reasoning capabilities and the overall intelligence of these models (Zhou et al., 2023; Feng et al., 2024). As aforementioned, we categorize recent work into two main approaches:

Utilizing external symbolic solvers. In this line of work, the main challenge lies in transforming the description of constraints into logical formulas, $\text{LLM}^{\text{lang2logic}}$, effectively functioning as a semantic parser. More specifically, Yao et al. (2023) introduced a method to enrich CoT prompting by leveraging results from external APIs, invoking functions to retrieve supplementary information. Building on this idea, Gao et al. (2023) proposed an approach that augments the intermediate reasoning steps of CoT reasoning through a runtime environment (*e.g.,* a Python interpreter), which has proven

particularly effective for mathematical reasoning tasks. Further, Pan et al. (2023); Ye et al. (2023); Olausson et al. (2023) enhanced the performance of logical reasoning by translating all constraints in a sample into logical forms, completing a logical program, and solving it using an independent symbolic reasoner. In addition, Xu et al. (2024a) strengthened the logical-form parsing process of the open-source LLMs by leveraging fine-tuning on a large-scale, curated dataset. In contrast, our method does not rely on any external symbolic solver; instead, it integrates symbolic syntax directly into the reasoning process, aiming to enable the LLM itself to reason as a symbolic solver.

LLM-based logical solvers. This approach leverages LLMs directly by designing frameworks that decompose complex reasoning tasks into smaller, manageable subtasks—such as rule selection, premise derivation, and scoring—to enhance overall reasoning robustness (Zhang et al., 2025; Feng et al., 2024; Sun et al., 2024). In particular, Sun et al. (2024); Xu et al. (2024a) introduced a framework that enables LLMs to uncover hidden premises and integrates scoring components through a multi-turn reasoning process. Similarly, Zhang et al. (2025) proposed a framework that incrementally refines reasoning via three subcomponents: Proposer, Verifier, and Reporter. Finally, Feng et al. (2024) presented LoGiPT, a method that enhances LLMs' ability to function as logical solvers by learning the reasoning process step by step through additional training on large-scale data collected from the reasoning traces of external symbolic solvers. Compared to our work, LoGiPT similarly performs step-by-step reasoning as a logical solver and can produce a proof tree at the end; however, *our method achieves competitive performance without the need for any additional training.*

3 Methodology

In this section, we present our *Symbolic-Aided CoT* method, its variants, and the motivation behind it in comparison with baseline prompting techniques: *Standard* - which directly provides an answer without any reasoning - and *CoT* (Wei et al., 2022) - which produces an answer accompanied by step-by-step reasoning. All of these prompting methods are augmented with a hard-selected few-shot examples included in the prompt (Qi et al., 2025). All prompts are directly fed forward through the LLM to obtain the final predicted

---

[1]`https://github.com/Z3Prover/z3`

answer in a single turn:

$$A^{\text{out}} = \text{LLMs}(\text{prompting}(\mathcal{R}, Q)) \qquad (1)$$

In this setup, the LLM is solely responsible for generating the desired answer given the contextual input. The prompting component consists of a few-shot template designed to help the LLM understand the task description while leveraging its own knowledge to reason over the list of provided rules in the contextual information. For clarity, the templates for *Standard* and *CoT* prompting are shown in the first two rows of Table 1.

Table 1: Template of input and output for prompting techniques: *Standard* and *CoT* and *Symbolic-Aided CoT*.

Standard (Input)	Context: *[[All constraints, $\mathcal{R}$]]* Question: *[[Content of the question, Q]]* Options: A) True B) False C) Uncertain
Standard (Output)	The correct option is: { "answer": *[[A]]* }
CoT (Input)	Context: *[[All constraints, $\mathcal{R}$]]* Question: *[[Content of the question, Q]]* Options: A) True B) False C) Uncertain

CoT (Output)	The correct option is:
	{
	"reasoning": *[[reasoning content]]* ,
	"answer": *[[A]]*
	}

Symbolic-Aided CoT (Input)	### Let us define F as a function that infers new premises based on a given list of facts and rules. Using these facts and rules, provide a reasoning path that leads to one of the values of a Validate function: True, False, or Uncertain.
	
	### Example1: Given list of facts and rules:
	
	  # (Rule*[[i]]*): *[[content of $r_i \in \mathcal{R}$]]* $\cdots$
	
	  # (Question): *[[content of the question, Q]]*

Symbolic-Aided CoT (Output)	# (Answer): Start from the object and their condition mentioned in the question to collect relevant facts: # KB = { } => F(KB(*[[premises in KB]]*), Rule*[[i']]*) => *[[inferred premises]]* # KB = {*[[KB values for each reasoning steps]]*} # validate the question with the current inferred premise => Validate(Question=*[[Q]]*, KB(*[[selected premise]]*)) = *[[A]]*.

3.1 Symbolic-Aided CoT

We formulate logical reasoning tasks into three fundamental sub-tasks, namely, reasoning operators: *rule matching*, *rule inference*, and *knowledge base updating*. Previous frameworks, such as DetermLR (Sun et al., 2024) and CR (Zhang et al., 2025), were also built on carefully designed unit operators, integrating them with procedural programming to process the outputs of LLMs. The key difference between our Symbolic-Aided CoT and these approaches is that Symbolic-Aided CoT is conceived entirely as an LLM-driven program.

In our design, the LLM is expected to learn the flow of the logical reasoning process from a few-shot demonstration. To this end, the LLM has full visibility of all sub-reasoning steps and autonomously decides which step to execute next. The overview of our Symbolic-Aided CoT is presented in the third row of Table 1, which illustrates the instruction text, list of rules, question, and reasoning-flow examples. For a clearer explanation, we elaborate on the two gray blocks shown in this table, which pertain to rule tagging and reasoning operators in our Symbolic-Aided CoT.

Rule tagging. In preparing the prompt input, we first segment the contextual information into a list of rules by splitting it into individual sentences using the NLTK toolkit[2]. Each sentence is then tagged with its order index (*e.g.,* Rule5 for the fifth sentence), allowing the LLM to track and reference the reasoning steps. Here, we assume that the LLMs can link each rule's content to its corresponding tag and reference this symbol appropriately in the reasoning flow in subsequent steps.

Reasoning operators. This demonstration serves as the core example that enables LLMs to learn, in context, the pattern for solving logical reasoning tasks. We use a set of symbols, similar to those in programming languages, to represent the inference flow (see Figure 2). At each reasoning step, the LLM selects the relevant rules and premises from the current knowledge base (KB) to infer new premises. Each newly inferred premise is then appended to the KB for use in the next inference step. A breadth-first search strategy is applied to traverse the nodes (premises), as illustrated in Figure 2. Inspired by how humans solve such tasks, we maintain a KB state to prevent cyclical inference loops: if a newly inferred premise already exists in the KB, it is not added again. All patterns of rule selection, inference, and KB updating are implicitly conveyed within the demonstrations provided in the few-shot prompts, allowing the LLM to internalize these reasoning steps.

4 Experiment and Analysis

In this section, we present a detailed description of our experiments, together with the results and analyses, to assess the effectiveness of our Symbolic-Aided CoT prompting in comparison to standard CoT and prior methods.
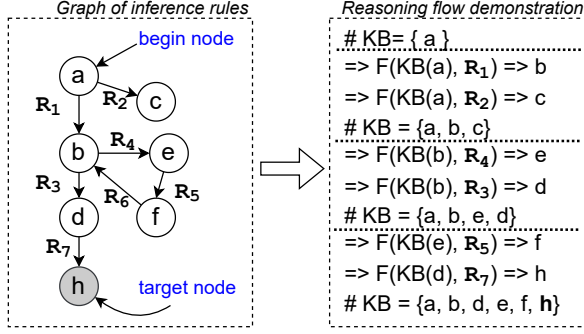
---

[2] https://www.nltk.org/

Figure 2: **Left**: Graphical model of inference rules. **Right**: Reasoning flows in the Symbolic-Aided CoT demonstrations.

4.1 Experimental Setup

Datasets. We conducted experiments on four well-known benchmark datasets about the logical reasoning task: (1) **ProofWriter** (Tafjord et al., 2021) - we use the subset under the open-world assumption, where each sample has three possible answer options: true, false, or unknown. Following Pan et al. (2023), we evaluate on the subset with the longest reasoning depth (5 hops), which contains 600 cases. (2) **FOLIO** (Han et al., 2024) is a challenging, expert-curated dataset for logical reasoning that contains rules closely aligned with real-world knowledge. Following the setup of previous work (Sun et al., 2024), we evaluate our method on a subset of this dataset comprising 204 examples. (3) **ProntoQA** introduced by Saparov and He (2023) - similar to the ProofWriter dataset, we also choose the hardest subset of this data with 5-hop reasoning across 500 samples for the evaluation, following previous works (Sun et al., 2024; Qi et al., 2025); (4) **LogicalDeduction** (Srivastava et al., 2023) is a dataset for logically identifying the order of objects given a list of description constraints. We follow the previous setting from Sun et al. (2024), using 300 evaluation samples containing all subsets of three, five, and seven objects (the greater the number of objects, the more complex the logical reasoning required to determine their order).

Evaluation metric. In order to evaluate system performance, we use Accuracy as the metric, which is standard and allows for direct and fair comparison with previous works (Pan et al., 2023; Sun et al., 2024; Qi et al., 2025).

Setting. We conducted our experiments primarily on open-source LLMs, including Llama-3.1-8B-Instruct (Grattafiori et al., 2024) and the Qwen-3 models (Yang et al., 2025). Specifically, we aim to evaluate the effectiveness of our Symbolic-Aided CoT compared to standard CoT and standard prompting on these LLMs. In addition, we performed extensive experiments on the Qwen-3 models across various sizes—1.7B, 4B, 8B, and 32B—to further assess the scalability and effectiveness of our proposed prompting method. Finally, for comparison with previous approaches, we also conducted extensive experiments on a powerful closed-source LLM, GPT-4 (Achiam et al., 2023), to evaluate our proposed method. For open-source models, we use greedy decoding to generate the answer *i.e.,* the token with the maximum logit is picked.

4.2 Main Results

For the main results across all benchmark datasets, ProofWriter, FOLIO, and ProntoQA, we present the performance of our Symbolic-Aided CoT compared with Standard Prompting, CoT Prompting, and previous methods in Table 2.

Overall performance. These results demonstrate the clear superiority of our proposed method over the standard CoT on three datasets, ProofWriter, ProntoQA, and LogicalDeduction. Notably, on the ProofWriter dataset, Symbolic-Aided CoT significantly outperforms CoT, achieving improvements of 15% and 21%, 22% on the Qwen-3-8B, Qwen-3-14B, and Llama3.1-8B-Instruct models, respectively. In addition, the improvement is clearly observed on the LogicalDeduction and ProntoQA datasets across three open-source LLMs. These findings further highlight that the degree of improvement varies across different LLMs. The effectiveness of Symbolic-Aided CoT largely depends on each model's ability to understand logical relationships and recognize logical matching patterns embedded within the few-shot prompts. Moreover, our method is simple yet effective, achieving competitive performance compared to prior works such as Logic-LM (Pan et al., 2023) and DetermLR (Sun et al., 2024), even when those methods are supported by a more powerful GPT-4 model.

On the FOLIO dataset, the results show that the Symbolic-Aided approach has a weakness compared to CoT prompting, especially with the Qwen-14B LLM. We found that the FOLIO dataset

338

is specifically designed by experts to cover various aspects of factual knowledge, which allows the CoT prompting technique to leverage this advantage (leaking factual knowledge) when solving questions. For example, in a question about the tennis player Djokovic, CoT prompting tends to use external knowledge such as *"Djokovic is famous and is an athlete"* , which is not provided in the set of facts in the context, to support the inference flow. In contrast, our Symbolic-Aided CoT approach relies strictly on the inference rules given in the context.

Finally, we evaluate our proposed method in the setting that uses GPT-4 as the backbone LLM for the reasoning task (last row of Table 2). Compared to the SymbCoT framework (Xu et al., 2024a), our method achieves superior performance on ProntoQA but lower performance on other datasets. This difference can be attributed to SymbCoT's use of complex interactive reasoning sub-steps—such as translator, planner, solver, and verifier—each supported by carefully designed prompts tailored to the specific sub-step and logical reasoning task. Overall, our method surpasses the performance of previous methods on the ProntoQA dataset and achieves remarkable results on the ProofWriter, FOLIO, and LogicalDeduction datasets. These results demonstrate the robustness of our approach, even in the stringent setting that uses only non-interactive inference without the support of an external solver or multi-step inferences, such as the DetermLR (Sun et al., 2024) or LogicLM (Pan et al., 2023) approaches.

Impact of model size on performance. To assess the effectiveness of our Symbolic-Aided CoT across different model sizes, we conducted experiments using various Qwen LLMs on ProofWriter (Figure 3) and LogicalDeduction (Figure 4) datasets. These results demonstrate that our method consistently outperforms both CoT and standard prompting across model sizes. Furthermore, our approach appears to encourage LLMs to more explicitly articulate the underlying logical reasoning patterns, even in small-scale models. For example, on the ProofWriter dataset, Qwen3-8B achieves performance comparable to that of the 32B model. On the LogicalDeduction dataset, Qwen3-8B attains 86.9% of the performance of the 32B model. We argue that our Symbolic-Aided CoT decomposes the original complex logical reasoning tasks into sub-reasoning operations–such as selecting rules, generating new

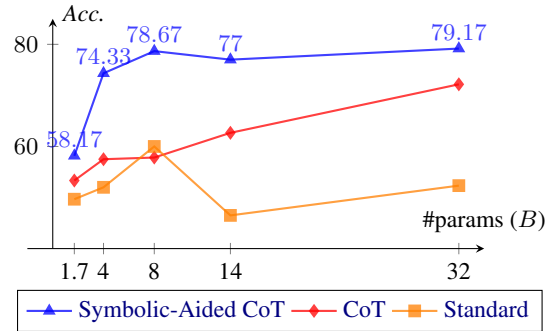premises, and extending KB premises–that can be effectively addressed by smaller language models.

Figure 3: Performance across different model sizes of Qwen-3 with three prompting techniques on the ProofWriter dataset.
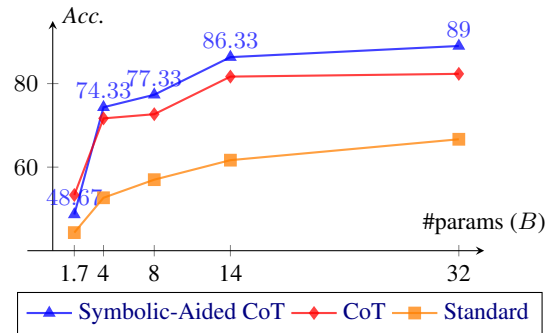

Figure 4: Performance across different model sizes of Qwen-3 with three prompting techniques on the LogicalDeduction dataset.

Ablation studies. For evaluating the contribution of each sub-component in our Symbolic-Aided CoT prompting, we conduct two ablation studies: (1) removing the KB-tracking variables (SymbolA.CoT$^{-\text{KB tracking}}$), which removes the text segment *"#KB = [[KB values for each reasoning step]]"* in Table 1, and (2) removing the symbolic Validate function (SymbolA.CoT$^{-\text{Validate}}$), which removes the text segment *"Validate(Question=[[Q]], KB([[selected premise]]))"* in Table 1. The ablation results (shown in Figure 5) indicate that KB-tracking variables play an important role in the reasoning process, helping LLMs avoid loops in the conferencing process. Furthermore, KB-tracking intuitively provides additional features to the hidden representation of premises, allowing the model to distinguish inferred premises from conditional premises in the constraint rules. In another aspect, the symbolic Validate function in our Symbolic-Aided CoT helps LLMs refer back to the original question

339

Table 2: Performance comparison among different methods. *E.Solver* refers to the system supported by an external symbolic solver module. *ICL* stands for in-context learning, and *Supervised FT* stands for the supervised fine-tuning approach. The **best** is marked.

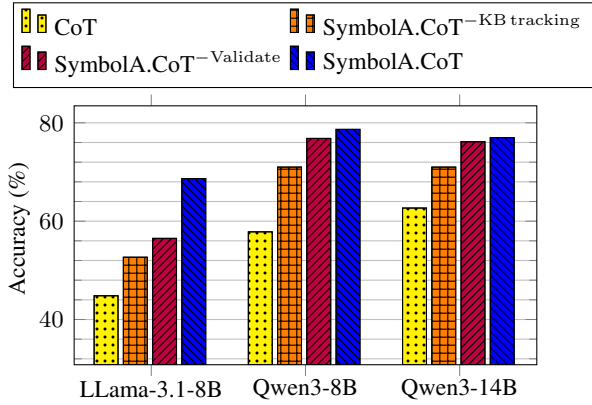| Methods	Learning paradigm	Interaction mode	ProofWriter	FOLIO	ProntoQA	L.Deduction
Llama3.1-8B-Instruct						
Fine-tuned ID (Qi et al., 2025)	Supervised FT	Non-Interactive	71.67	70.00	—	—
Standard	ICL few-shot	Non-Interactive	36.83	53.92	51.80	40.67
CoT	ICL few-shot	Non-Interactive	44.83	**56.86**	74.00	58.00
Symbolic-Aided CoT (ours)	ICL few-shot	Non-Interactive	**68.67**	55.88	**89.00**	**59.33**
Qwen3-8B						
Standard	ICL few-shot	Non-Interactive	60.00	62.25	80.80	57.00
CoT	ICL few-shot	Non-Interactive	57.83	**66.67**	95.80	72.67
Symbolic-Aided CoT (ours)	ICL few-shot	Non-Interactive	**78.67**	65.69	**97.20**	**77.33**
Qwen3-14B						
Standard	ICL few-shot	Non-Interactive	46.50	67.16	77.80	61.67
CoT	ICL few-shot	Non-Interactive	62.67	**74.02**	97.20	81.67
Symbolic-Aided CoT (ours)	ICL few-shot	Non-Interactive	**77.00**	65.20	**97.80**	**86.33**
GPT-4						
CoT (Sun et al., 2024)	ICL few-shot	Non-Interactive	67.41	67.65	91.00	73.33
Logic-LM (Pan et al., 2023)	ICL few-shot	Interactive +E.Solver	79.66	78.92	83.20	87.63
DetermLR (Sun et al., 2024)	ICL few-shot	Interactive +Programming	79.17	75.49	98.60	85.00
SymbCoT (Xu et al., 2024a)	ICL few-shot	Interactive +Programming	**82.50**	**83.33**	99.60	**93.00**
Symbolic-Aided CoT (ours)	ICL few-shot	Non-Interactive	77.09	74.51	**100.00**	86.33

Figure 5: Ablation study on the Symbolic-Aided CoT (SymbolA.CoT).

in the context to select the appropriate premise for logical matching and producing the final answer.

4.3 Result Analysis

Confusing ratio. Here we report the confusion matrices (Figure 6) of the answers in the ProofWriter dataset, generated by the Qwen-8B model. For both methods, the original CoT and our Symbolic-Aided CoT, the recall score for False questions is the highest, followed by True and Uncertain. This is due to the complex nature of the logical reasoning task, which involves multi-hop reasoning steps; reasoning paths leading to wrong conclusions are typically more numerous than those leading to correct ones. Comparing our Symbolic-Aided CoT to the original CoT, our method shows improvement across all three question types. The

main improvement comes from reducing confusion in Uncertain questions, decreasing misclassification as True or False. We argue that, through symbolic injection, our method encourages clearer logical patterns and structure, thereby enhancing the logical reasoning ability of LLMs.
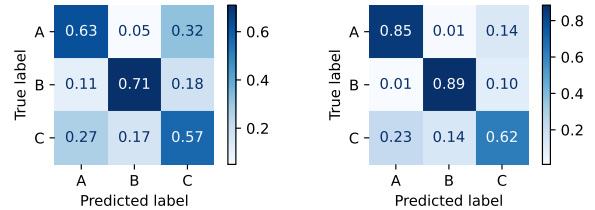

Figure 6: Result comparison with confusion matrices between our Symbolic-Aided CoT (right) and the original CoT (left). The labels A, B, and C refer to the answers True, False, and Uncertain, respectively.

Semantic representation of symbolic tokens. Here, Figure 7 visualizes the semantic representation of symbolic tokens using principal component analysis (PCA) based on the last layer's output hidden states of the LLM Qwen-3-14B. This experiment aims to analyze, at a low level, how LLMs understand symbolic tokens in our Symbolic-Aided CoT prompting. We found that LLMs can clearly distinguish the meaning of symbolic tokens (purple data points) from sample content tokens in our proposed prompting method. This is because these logical tokens play the role of structuring the inference flow (latent reasoning pattern) of LLMs, which is separate from the content words in facts

and rules. Through few-shot in-context learning, these tokens are represented in a distinct semantic space. Via the self-attention mechanism, logical tokens are paired with content tokens to yield features specific to reasoning operators (such as matching rules or inferring new premises). This suggests that LLMs can uncover the hidden patterns of logical reasoning operators implied by the symbolic tokens within few-shot learning.
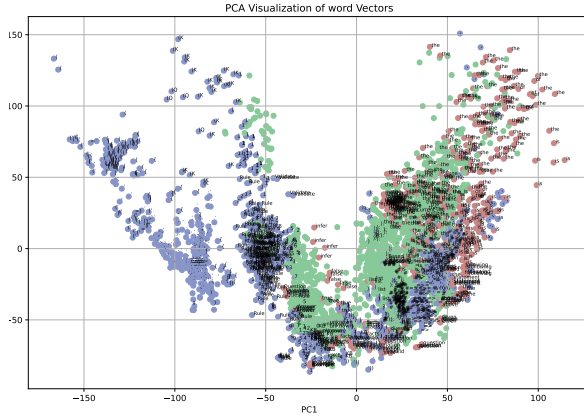

Figure 7: Visualization of last-layer word hidden states from `Qwen3-8B`, with dimensionality reduced via PCA, in the Symbolic-Aided CoT setting on the ProofWriter dataset. The purple, brown, and green data points represent the embeddings of logical symbols (*e.g.,* "=>", "KB"), instruction tokens (*e.g.,* "Let," "us," "define"), and sample content tokens (*e.g.,* "cat," "mouse"), respectively.

Case studies. Based on case studies of incorrect predictions, we identified several improvement scenarios exhibited by Symbolic-Aided CoT compared to the standard CoT prompting technique (Table 3 in Appendix A): (1) *Hallucinated Inference Rules:* LLMs often generate inference rules that are either fabricated, logically invalid, or misaligned with the intended knowledge base (first row in Table 3). This phenomenon is caused by the counterfactual or pre-trained knowledge embedded within LLMs can conflict with in-context rules or premises provided at inference time. This undermines the assumption that the model will reason strictly within the given context or constraints; (2) *Unstoppable Inference Flow:* The reasoning process lacks a clear halting condition (third row in Table 3). The model continues generating premises without a mechanism to determine when inference should stop, leading to uncontrolled or incoherent inference chains. This highlights the need to explicitly track and manage the state of the knowledge

base (KB) during reasoning; (3) *Failure on Cyclic Inference Graphs:* When the inference space forms a cyclic graph, LLMs often fail—either entering infinite reasoning loops or struggling to resolve the cycle (second and third rows in Table 3). These models lack the structural awareness to detect and handle loops in reasoning chains; (4) *Rule Matching Errors:* LLMs frequently fail to apply inference rules correctly due to poor condition matching (fourth row in Table 3). The model may skip necessary preconditions or generate incorrect intermediate steps, breaking the logical flow of multi-step reasoning.

5 Discussion

Logical reasoning tasks have attracted numerous research works recently (Pan et al., 2023; Ye et al., 2023; Olausson et al., 2023; Sun et al., 2024; Zhang et al., 2025), especially following the massive success of LLMs. Unlike previous methods, by proposing Symbolic-Aided CoT, we primarily aim to enhance the logical reasoning ability of LLMs rather than simply build a system to improve performance on logical reasoning tasks. For example, frameworks such as Logic-LM (Pan et al., 2023) and SatLM (Ye et al., 2023) use LLMs only to translate logical problems into inputs for explicit symbolic reasoners. Frameworks like CR (Zhang et al., 2025) and DetermLR (Sun et al., 2024) leverage LLMs to perform small constituent logical reasoning steps, rather than directly evaluating the LLM's logical reasoning ability on the entire problem.

The experimental results show that our Symbolic-Aided CoT prompting technique is reliable and effectively improves the logical reasoning ability of LLMs, even for small model sizes. Our prompting method is simple, yet effective and flexible, allowing customization for any logical reasoning task. It can also yield proof trees that facilitate explanation and enhance transferability.

6 Conclusion

In this work, we introduced Symbolic-Aided CoT, a novel prompting technique for non-interactive logical reasoning, which achieves superior performance on well-known benchmark datasets—most notably ProofWriter, ProntoQA, and LogicalDeduction. Our method is deliberately simple to preserve generalizability and shows strong potential for extension to other reasoning tasks. For future work, Symbolic-Aided CoT, grounded in structural

characteristics, could be combined with mechanisms for refining the latent semantic vector space, thereby further improving the faithfulness and reliability of LLMs' reasoning capabilities.

Acknowledgments

Limitations

We discuss the following limitations and future works: We have evaluated the proposed method on four widely used logical reasoning benchmarks. However, they are mostly synthetic; incorporating real-world datasets or diverse reasoning tasks (*e.g.,* commonsense reasoning) would strengthen claims of generalizability. Relying solely on automatic metrics like accuracy overlooks qualitative aspects; integrating human evaluations to assess reasoning faithfulness and interpretability would offer a more holistic validation. Future research could also explore the method's robustness to adversarial perturbations, sensitivity to prompts, and scalability to longer reasoning chains.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, and 16 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Jiazhan Feng, Ruochen Xu, Junheng Hao, Hiteshi Sharma, Yelong Shen, Dongyan Zhao, and Weizhu Chen. 2024. Language models can be deductive solvers. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4026–4042, Mexico City, Mexico. Association for Computational Linguistics.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: Program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10764–10799. PMLR.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. 2022. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598.

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

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alexander Wardle-Solano, Hannah Szabó, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, and 16 others. 2024. FOLIO: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22017–22031, Miami, Florida, USA. Association for Computational Linguistics.

Theo X. Olausson, Alex Gu, Ben Lipkin, Cedegao E. Zhang, Armando Solar-Lezama, Joshua B. Tenenbaum, and Roger P. Levy. 2023. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, and 7 others. 2022. In-context learning and induction heads. *Transformer Circuits*

Thread. Https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Francesco Ortu, Zhijing Jin, Diego Doimo, Mrinmaya Sachan, Alberto Cazzaniga, and Bernhard Schölkopf. 2024. Competition of mechanisms: Tracing how language models handle facts and counterfactuals. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8420–8436, Bangkok, Thailand. Association for Computational Linguistics.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. 2023. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3806–3824, Singapore. Association for Computational Linguistics.

Chengwen Qi, Ren Ma, Bowen Li, He Du, Binyuan Hui, Jinwang Wu, Yuanjun Laili, and Conghui He. 2025. Large language models meet symbolic provers for logical reasoning evaluation. In *The Thirteenth International Conference on Learning Representations*.

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*.

Richard Shin and Benjamin Van Durme. 2022. Few-shot semantic parsing with language models trained on code. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5417–5425, Seattle, United States. Association for Computational Linguistics.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, and 431 others. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*. Featured Certification.

Hongda Sun, Weikai Xu, Wei Liu, Jian Luan, Bin Wang, Shuo Shang, Ji-Rong Wen, and Rui Yan. 2024. DetermLR: Augmenting LLM-based logical reasoning from indeterminacy to determinacy. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9828–9862, Bangkok, Thailand. Association for Computational Linguistics.

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.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed 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*, volume 35, pages 24824–24837. Curran Associates, Inc.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart Van Merriënboer, Armand Joulin, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*.

Fangzhi Xu, Zhiyong Wu, Qiushi Sun, Siyu Ren, Fei Yuan, Shuai Yuan, Qika Lin, Yu Qiao, and Jun Liu. 2024a. Symbol-LLM: Towards foundational symbol-centric interface for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13091–13116, Bangkok, Thailand. Association for Computational Linguistics.

Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. 2024b. Faithful logical reasoning via symbolic chain-of-thought. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13326–13365, Bangkok, Thailand. Association for Computational Linguistics.

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*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Xi Ye, Qiaochu Chen, Isil Dillig, and Greg Durrett. 2023. SatLM: Satisfiability-aided language models using declarative prompting. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew C Yao. 2025. Cumulative reasoning with large language models. *Transactions on Machine Learning Research*.

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 Improving examples

In this section, we present detailed improvement examples from the ProofWriter dataset for our

Symbolic-Aided CoT method compared to the CoT method, in the setting using the `Qwen3-8B` model (Table 3).

Table 3: Improving examples from the ProofWriter dataset for our Symbolic-Aided CoT method compared to the CoT method. The red text refers to incorrect logical reasoning or hallucinated content, and the green text refers to the final premises related to the target question.

Id	Context, Question, and Answer	CoT's output + [Error Type]	Symbolic-Aided CoT's output + [Error Type]
ProofWriter-_AttNeg-OWA-D5-596_Q6	### Given list of facts and rules: # (Rule1): Anne is not rough. # (Rule2): Bob is blue. # (Rule3): Erin is not furry. # (Rule4): Erin is red. # (Rule5): Gary is big. # (Rule6): Gary is not cold. # (Rule7): Gary is furry. # (Rule8): If something is blue then it is rough. # (Rule9): Red things are rough. # (Rule10): If Gary is rough then Gary is not blue. # (Rule11): Rough things are red. # (Rule12): Big things are quiet. # (Rule13): All cold things are big. # (Rule14): If something is red then it is big. [...] # (Question): Based on the above information, is the following statement true, false, or unknown? Erin is not quiet. # (Answer): **False**	Erin is red. Red things are rough. If something is rough then it is not quiet. So Erin is not quiet. *(1) Hallucinated Inference Rules*	Start from the object and their condition mentioned in the question to collect relevant facts: Erin, is not quiet # KB = { } => Rule3 = 'Erin is not furry' => Rule4 = 'Erin is red' # KB = {Erin is not furry, Erin is red} => F(KB('Erin is red'), Rule9) => 'Erin is rough' # KB = {Erin is not furry, Erin is red, Erin is rough} => F(KB('Erin is red'), Rule11) => 'Erin is red' (already in KB) # KB = {Erin is not furry, Erin is red, Erin is rough} => F(KB('Erin is red'), Rule14) => 'Erin is big' # KB = {Erin is not furry, Erin is red, Erin is rough, Erin is big} => F(KB('Erin is big'), Rule12) => 'Erin is quiet' # KB = {Erin is not furry, Erin is red, Erin is rough, Erin is big, Erin is quiet} # valid the question with current inferred premies => Validate(Question='Erin is not quiet', KB('Erin is quiet'))=**False**.
ProofWriter-_RelNoneg-OWA-D5-356_Q4	### Given list of facts and rules: # (Rule1): The cow eats the mouse.[...] # (Rule15): The squirrel likes the cow. # (Rule16): The squirrel likes the mouse. # (Rule17): If something likes the cow then it visits the mouse. # (Rule18): If something likes the cow then the cow is green. # (Rule19): If something eats the squirrel and the squirrel likes the rabbit then it eats the rabbit. # (Rule20): If something is cold then it likes the cow. # (Rule21): If something visits the squirrel then the squirrel likes the rabbit. # (Rule22): If the cow likes the rabbit then the cow is cold.[...] # (Question): Based on the above information, is the following statement true, false, or unknown? The squirrel does not visit the mouse. # (Answer): **False**	The cow likes the rabbit. If the cow likes the rabbit then the cow is cold. The cow is cold. The cow likes the squirrel. If something is cold then it likes the cow. The cow likes the cow. The cow visits the mouse. If something likes the cow then it visits the mouse. The cow visits the mouse. The squirrel likes the cow. If something visits the squirrel then the squirrel likes the rabbit. The squirrel likes the rabbit. The squirrel eats the rabbit. If something likes the rabbit and it visits the mouse then the mouse visits the squirrel [...] *(3) Failure on Cyclic Inference Graph* *(4) Rule Matching Errors*	Start from the object and their condition mentioned in the question to collect relevant facts: The squirrel, does not visit the mouse # KB = { } => Rule1 = 'The cow eats the mouse' # KB = {cow eats mouse, [...], squirrel is rough, squirrel likes cow, squirrel likes mouse} => F(KB('squirrel likes cow'), Rule17) => 'squirrel visits mouse' # KB = {cow eats mouse, [...] squirrel is rough, squirrel likes cow, squirrel likes mouse, squirrel visits mouse} # valid the question with current inferred premies => Validate(Question='The squirrel does not visit the mouse', KB('squirrel visits mouse')) = **False**.
ProofWriter-_RelNeg-OWA-D5-688_Q22	### Given list of facts and rules:[...] # (Rule6): The lion needs the mouse. # (Rule7): The mouse is round. # (Rule8): The mouse likes the lion. # (Rule9): The mouse needs the lion.[...] # (Rule15): If something likes the cat and it does not visit the cat then it visits the lion. # (Rule16): If the lion is green and the lion visits the mouse then the mouse is red. # (Rule17): All red things are nice. # (Rule18): If something likes the lion then it is red. # (Rule19): If the mouse visits the cat and the mouse needs the squirrel then the mouse does not like the cat. # (Rule20): Nice things are big. # (Rule21): If something is big then it visits the squirrel. # (Rule22): If the mouse visits the squirrel then the mouse does not visit the lion. # (Question): Based on the above information, is the following statement true, false, or unknown? The lion likes the mouse. # (Answer):": **Uncertain**	The mouse likes the lion. The mouse needs the lion. The mouse visits the cat. The mouse does not visit the lion. The mouse is round. The mouse is red. The mouse is nice. The mouse is big. The mouse is nice. The mouse is big. The mouse visits the squirrel. The mouse does not visit the lion. The mouse is red. The mouse is nice. The mouse is big. The mouse is nice. The mouse is big. The mouse is nice. The mouse is big. [...] *(2) Unstoppable Inference Flow* *(3) Failure on Cyclic Inference Graphs*	Start from the object and their condition mentioned in the question to collect relevant facts: The lion, likes the mouse # KB = { } [...] # KB = {The lion needs the mouse, The mouse likes the lion, The mouse is red, The mouse is nice} => F(KB('The mouse is nice'), Rule20) => 'The mouse is big' # KB = {The lion needs the mouse, The mouse likes the lion, The mouse is red, The mouse is nice, The mouse is big} => F(KB('The mouse is big'), Rule21) => 'The mouse visits the squirrel' # KB = {The lion needs the mouse, The mouse likes the lion, The mouse is red, The mouse is nice, The mouse is big, The mouse visits the squirrel} => F(KB('The mouse visits the squirrel'), Rule22) => 'The mouse does not visit the lion'[...] # valid the question with current inferred premies => Validate(Question='The lion likes the mouse', KB('The mouse likes the lion')) =**Uncertain**.
ProofWriter-_AttNoneg-OWA-D5-245_Q19	### Given list of facts and rules:[...] # (Rule4): Erin is nice.[...] # (Rule11): Furry people are smart. # (Rule12): If someone is smart and nice then they are round. # (Rule13): Cold people are red. # (Rule14): If someone is quiet then they are nice. # (Rule15): All red people are furry. # (Rule16): All smart, furry people are nice. # (Question): Based on the above information, is the following statement true, false, or unknown? Erin is not quiet. # (Answer) : **Uncertain**	Erin is nice. If someone is nice then they are quiet. So Erin is quiet. Therefore, Erin is quiet, so the statement 'Erin is not quiet' is false. *(4) Rule Matching Errors*	Start from the object and their condition mentioned in the question to collect relevant facts: Erin, is not quiet # KB = { } => Rule4 = 'Erin is nice' # KB = {Erin is nice} => F(KB('Erin is nice'), Rule14) => 'Erin is quiet' # KB = {Erin is nice, Erin is quiet} # valid the question with current infered premies => Validate(Question='Erin is not quiet', KB('Erin is quiet')) = **False**. *(4) Rule Matching Errors*