

# Enhancing Zero-Shot Chain-of-Thought Reasoning in Large Language Models through Logic

Xufeng Zhao, Mengdi Li, Wenhao Lu, Cornelius Weber,  
Jae Hee Lee, Kun Chu, and Stefan Wermter

University of Hamburg

{xufeng.zhao, wenhao.lu, cornelius.weber, jae.hee.lee, kun.chu,  
stefan.wermter}@uni-hamburg.de, mengdi.li@studium.uni-hamburg.de

## Abstract

Recent advancements in large language models have showcased their remarkable generalizability across various domains. However, their reasoning abilities still have significant room for improvement, especially when confronted with scenarios requiring multi-step reasoning. Although large language models possess extensive knowledge, their behavior, particularly in terms of reasoning, often fails to effectively utilize this knowledge to establish a coherent thinking paradigm. Generative language models sometimes show hallucinations as their reasoning procedures are unconstrained by logical principles. Aiming to improve the zero-shot chain-of-thought reasoning ability of large language models, we propose Logical Chain-of-Thought (LogiCoT), a neurosymbolic framework which leverages principles from symbolic logic to verify and revise the reasoning processes accordingly. Experimental evaluations conducted on language tasks in diverse domains, including arithmetic, commonsense, symbolic, causal inference, and social problems, demonstrate the efficacy of the enhanced reasoning paradigm by logic.

**Keywords:** Large Language Models, Reasoning, Chain-of-Thought, Logic

## 1. Introduction

Large language models (LLMs) are expected to be omniscient because of their extraordinary ability to deal with tasks requiring knowledge of common sense or even specialized field knowledge. The success has been established in numerous fields extending beyond the realm of language processing (Bubeck et al., 2023; Yao et al., 2023b; Ahn et al., 2022; Zhao et al., 2023).

However, one major problem residing in generative LLMs yet to be solved is their tendency to hallucinate wrong statements in a confident style (Bang et al., 2023). A quick example can be found by asking a non-internet-based LLM about very recent news – it will too easily make up facts without hesitation.

An educated human with expertise in logical reasoning can systematically examine words before coming to a conclusion. Unlike logical reasoning by humans, the logical incompetence of the deduction by LLMs makes their decisions untrustworthy. LLMs may have all the logical concepts and tricks available but fail to actively utilize them in an organized manner, which brings the demand for expert guidance. Principles in logic well-adapted by humans can also benefit the reasoning ability of language models.

Take a simple logic question as an example: “If Tom plays football outside, then John will also join to play; if John plays football, then Mary won’t go outside. Known Mary is outside. Is Tom playing football?” Nine out of ten answers from ChatGPT<sup>1</sup>

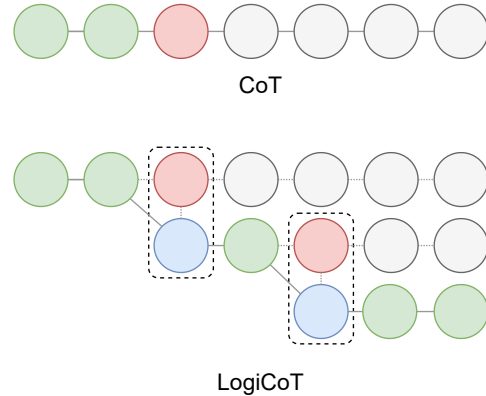


Figure 1: An overview of CoT (chain-of-thought prompting, Wei et al., 2022) and LogiCoT. In CoT, the failure of entailment (●) makes the rest of the deduction untrustworthy (●), consequently impeding the overall success of the deduction. In contrast, LogiCoT is designed to think-verify-revise: it adopts those who pass the verification (●) and revise (●) those who do not, thereby effectively improving the overall reasoning capability.

will conclude that “we cannot conclude whether Tom is playing football or not”. However, with the help of the knowledge in logic provided to ChatGPT that the contrapositive holds the exact same truth value with the original proposition, we may put it another way to prompt ChatGPT to “use contrapositive”. Then it deduces correctly: “... Using the contrapositive of the first statement, if John does not join

<sup>1</sup><https://openai.com/blog/chatgpt>

to play (which we have deduced), then it implies that Tom does not play football outside. Therefore, based on the given information and the contrapositives, it can be deduced that Tom is not playing football.” There is no newly introduced knowledge but a prompt of using *contrapositive*, a special variational expression of the original premise. While the concepts of logic are not new to a large language model, the model initially struggles to incorporate them. Compared to randomly sampling for diverse statements, the one derived from logical equivalence works effectively as it could be expressed quite differently in natural language and may result in a totally different deduction.

Motivated by the reasoning process in logic, we propose **Logical Chain-of-Thought** (LogiCoT) to further expand the zero-shot reasoning ability of LLMs, which not only lets the LLM *think step by step* but also verify, step by step, according to the guidance via the principle of *Reductio ad Absurdum*, and revise the reasoning chain if necessary to guarantee a sound inference (see Fig. 1 for an overview).

## 2. Related Work

In order to unleash the power of a pre-trained generative language model, the quality of the prompts to interact plays an important role. Summarizing known works, the reasoning procedure benefits from a prompt that guides it to possess the following properties:

- **Relevance** The generative model can be easily distracted by irrelevant words in the prompt. A pre-selection of context helps the correctness of reasoning (Creswell et al., 2022; Creswell and Shanahan, 2022; Ling et al., 2023).
- **Decomposition** An automatic decomposition of a tough question improves the reasoning reliability, which has been evidenced by the success of Least-to-Most (Zhou et al., 2023), Zero-shot-CoT (Kojima et al., 2022) and many prompting techniques (Yao et al., 2023a; Kojima et al., 2022; Wei et al., 2022).
- **Verification/Grounding** External functions, e.g. a third-party calculator for mathematical problems (Schick et al., 2023), external information acquisition from Wikipedia (Yao et al., 2023b), or an affordance evaluation function in robotics (Ahn et al., 2022), can ground the generation to be meaningful. This verification can be triggered under a specified condition or be applied to the reasoning process (Lightman et al., 2023; Ling et al., 2023; Li et al., 2023).
- **Diversity** The collective intelligence from a set of reasoning paths (typically, sampling  $N$

times) helps produce a final answer that is consistent among these variants. Despite the surging  $N$ -times cost, this ensemble approach has been widely adopted to combine with other techniques for higher accuracy (Li et al., 2023; Ling et al., 2023; Yao et al., 2023a; Zheng et al., 2023).

- **Revision** Revision (or refinement) can be regarded as a special *diversity* but is conditioned on the previous generation as hints. It re-examines the words with an extra focus on the quality in terms of, for example, validity and conciseness (Madaan et al., 2023; Zheng et al., 2023; Welleck et al., 2022).

**Chain-of-Thought Prompting.** Prior works show that LLMs have the corresponding power for complex tasks but require a proper strategy to unleash, e.g. human-in-the-loop (Ouyang et al., 2022) alignment tuning and Chain-of-Thought prompting (CoT) (Wei et al., 2022). In order to generate a chain of thoughts that decomposes the original problem into several small parts which a language model can easily handle, CoT creates few-shot exemplars of a detailed reasoning path to let the model follow. Least-to-most (Zhou et al., 2023) explicitly prompts the LLM to divide complex questions into sub-problems and conquer them one by one. Moreover, zero-shot-CoT (Kojima et al., 2022) showcases the impressive effectiveness of simply attaching the sentence “Let’s think step by step.” before any zero-shot reasoning trace starts. We build our approach under a zero-shot setting and integrate zero-shot-CoT as a baseline to improve with. While existing CoT-based methods focus on encouraging the reasoning step to be concrete but lack supervision of their faithfulness, we propose a step-by-step verification mechanism. A very recent study of Ling et al. (2023) also addresses this credential concern with double-checking and has reached a positive improvement, which also emphasizes the benefit of per-step verification. However, while their work anticipates autonomous detection of errors by just prompting “Double-check the reasoning process...”, our work is motivated from a logical perspective and empowers the language model to argue different possibilities. Moreover, our method not only suggests verification but also introduces revision of the suspected reasoning steps.

When posed with a question, the careful selection of relevant facts to ingest is equally critical in preventing the language model from becoming distracted or potentially misinformed, which might result in hallucinations. This consideration of *relevance* is even crucial when the context becomes very long. Previous works typically resort to a language model to evaluate the relevance of facts and infer with the ones contributing to an intermediate

reasoning step (Creswell et al., 2022; Ling et al., 2023). Our verification of each reasoning step is conducted by prompting a language model to find relevant premises to deduct from.

**Variational Reasoning.** A single reasoning trace may be biased. In order to produce a set of reasoning candidates, previous works resort to generating samples several times (Wang et al., 2023) with the same prompt, or create diverse prompts in the beginning for variants (Li et al., 2023). However, this approach is costly and inefficient since many of the reasoning steps are non-controversial so not requiring duplicates. Our method avoids unnecessary checking and only revises reasoning steps deemed implausible, resulting in a reasoning chain growing only when required. It costs more than generating one chain because of the verification and possible revision but is more efficient than a naive ensemble. Besides, LogiCoT can be combined with an ensemble approach to produce a set of verified chains, further increasing the confidence for later majority voting required by the ensemble.

Compared to the ensemble-based method that independently samples diverse variants, revision produces a special diversity. It is an iterative generating process conditioned on previous content. Many previous works actually benefit from this manner though not explicitly mentioned. For example, Progressive-Hint Prompting (Zheng et al., 2023) generates consistent answers by progressively guiding the LLM with hints of accumulated possible answers. It repeats generation again and again until the answer is deemed consistent with the previous. Other works generate content conditioned not only on the previous content but also on extra feedback (Madaan et al., 2023). To obtain a revision with high quality, this guiding feedback should be specific and actionable. Our work takes advantage of this property and revises for those reasoning steps that fail to pass the verification, during which the post hoc explanation (Jung et al., 2022) will act as a constructive revision suggestion.

**Neurosymbolic AI.** Neurosymbolic AI combines neural networks with symbolic representations and reasoning techniques. Its success stems from the ability to leverage symbolic (structured) knowledge to enhance learning or reasoning (Sarker et al., 2021; d’Avila Garcez and Lamb, 2020; Nye et al., 2021). Unlike the end-to-end black-box framework, it is more interpretable and explainable because of the transparency of the symbolic framework.

There exist works that adopt concepts from symbolic logic (Agler, 2012) to establish a reliable reasoning path (Creswell et al., 2022; Jung et al., 2022). To solve binary question-answering problems, Jung et al. (2022) propose to generate a post hoc explanation graph for a statement and compute the relative relations to formulate a symbolic logic ex-

pression. The truth of the statement is thereby assigned by solving the satisfiability problem of this symbolic expression. The LogiCoT framework employs a neurosymbolic methodology, leveraging logical rules and post hoc arguments to enhance error detection.

### 3. Methodology

As demonstrated in the contraposition example presented in the introduction, when known logical rules are applied to achieve an identical transformation (equivalent in logic but markedly distinct in natural language expression), it affords the LLMs the chance to engage in reasoning from an alternative perspective.

A challenge is that the language model has to identify the inherent logical structures first to know whether certain prior knowledge can be effectively applied. Moreover, transforming everything from the real world into a symbolic expression is unrealistic. The applicable scenario is limited because questions in many reasoning fields beyond logic, e.g. mathematics problem solving, can hardly be expressed in symbolic logic. Nevertheless, there is promise in incorporating concepts from logic that contribute to the process of argument proof in order to construct a neurosymbolic framework (d’Avila Garcez and Lamb, 2020; Creswell et al., 2022) that facilitates a *causal* reasoning trace, i.e. the premises and leading thoughts entail the thoughts behind. Continuing with the success of “let the model talk”, e.g. “let’s think step by step” in zero-shot-CoT (Kojima et al., 2022), we further propose to guide the conversation with logic for more systematic exploration instead of counting on its recklessness.

#### 3.1. Reductio ad Absurdum

When given an argument generated by an LLM, it is difficult for the language model to recognize errors (i.e. to prove falseness) through free double-checking by itself. This is also the case in the field of logic. Many propositions pose challenges when it comes to direct deductive reasoning. One commonly employed technique to establish a claim is known as *reductio ad absurdum* (reduction to absurdity), which involves an initial assumption and consequent derivation of absurdity or contradiction. Let  $P$  and  $Q$  denote two propositions. The relation between a premise and its conclusion can be expressed as  $P \vdash Q$ . Here “ $\vdash$ ” is a syntactic turnstile which means  $Q$  is a syntactic consequence of  $P$  (Agler, 2012), i.e. there exists a proof that claims the conclusion  $Q$  given the premise  $P$ . In order to prove  $Q$  by the mean of *reductio ad absurdum*, let us assume its negation  $\neg Q$  is valid and then check

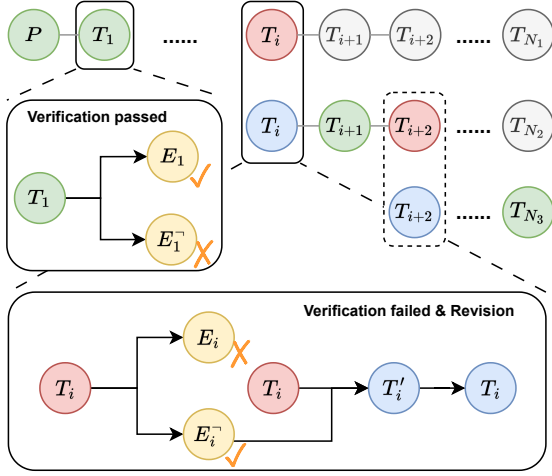


Figure 2: A diagram demonstrating the *think-verify-revision* loop of LogiCoT. The two zoom-in boxes exhibit the processes when a thought passes (top-left) and fails (bottom) the verification respectively. A thought passing the verification is kept in the reasoning trace, while a thought failing the verification is revised and a new chain of thought is generated based on the revision. The meaning of the symbols in this figure is introduced in Sec. 3.2 and Sec. 3.3.

the contradiction<sup>2</sup> of the *conjunctive proposition*

$$C = P \wedge \neg Q, \quad (1)$$

where “ $\wedge$ ” is a binary *conjunction operator*, meaning the truth of the conjunction requires the truth of both sides. Upon the contradiction of the co-existence of the  $P$  and  $\neg Q$ ,  $P \vdash Q$  is thus proved true, and then we can claim the validation of the conclusion  $Q$  given the premise  $P$ .

Many logic principles, e.g. the contraposition mentioned in the introduction section (see Appendix A for a proof), can be derived by deductions following this rule. This thinking paradigm helps humans check arguments carefully before composing a conclusion. As we will demonstrate later, the reasoning ability of LLMs can also be improved by benefiting from this paradigm.

### 3.2. Logical Chain-of-Thought (LogiCoT)

There is a lot of evidence confirming that a series of coherent explanations helps an LLM to unleash its reasoning power (Wei et al., 2022; Kojima et al., 2022; Zhou et al., 2023), while some discouragement on its utterance, e.g. prompts like “just tell me the result without any explanation”, catastrophically hinders the reveal of wisdom. So we continue with this success of having an explicit reasoning process.

<sup>2</sup>A proposition is considered contradictory if and only if it is false under every valuation.

A typical  $N$ -step reasoning trace can be expressed as  $\{P, T_1, \dots, T_N\}$ , where  $P$  is the known premise and  $T_i$  is the  $i$ -th step of thoughts. Usually,  $T_N$  concludes the thoughts and answers the specified question.

Unfortunately, LLMs hallucinate. LLMs usually generate content autoregressively, which means the generation of  $T_i$  is based on the former content  $\{P, \dots, T_{i-1}\}$ . Errors in  $T_i$  will propagate and gradually influence  $T_{i'}$  for increasing  $i' > i$ , making the successive deductions and ultimately the final conclusion untrustworthy. Therefore, we propose a verification loop to double-check each reasoning step. Following Eq. 1, this double-check procedure unrolls by checking the validity of  $P, \dots, T_{i-1} \vdash T_i$ , i.e. the contradiction of

$$C_i = P \wedge T_1 \wedge \dots \wedge T_{i-1} \wedge \neg T_i, \quad (2)$$

once  $T_{<i}$  passed the verification. If any step  $T_i$  fails the verification, this implies that the premises and previously verified thoughts  $T_{<i}$  do not entail  $T_i$ . In this case,  $T_{\geq i}$  needs to be revised.

To apply the *introduction of negation* ( $\neg I$ ) on  $T_i$ , a straightforward way is to format  $\neg T_i$  as “It is false to say  $T_i$ ” or to give to the LLM an instruction of “Negate  $T_i$ ”.

Finally, the LLM has to identify contradictions in  $C_i$  (Eq. 2). To obtain a more accurate result, we draw inspiration from the chain-of-thought approach and ask the model to *compose* a post hoc explanation  $E_i^-$  of  $\neg T_i$  (Jung et al., 2022) and check the validity of  $C'_i = C_i \wedge E_i^-$  instead. We call this approach *composing* LogiCoT.

Considering that a logical error in a text generated by an LLM is hard to spot by the LLM itself (which is the case in the *composing* approach) we additionally propose to alleviate the difficulty in verifying  $T_i$  by generating a pair of post hoc explanations  $E_i$  and  $E_i^-$  of  $T_i$  and  $\neg T_i$  respectively, and let the LLM decide between  $T_i \wedge E_i$  and  $\neg T_i \wedge E_i^-$  and *adopt* one of the two. We call this approach *adopting* LogiCoT or just LogiCoT.<sup>3</sup>

### 3.3. Chain Growth

Upon the suspect of a step  $T_i$ , LogiCoT drops all of the trailing thoughts  $T_{>i}$  and branches out for revision  $T'_i$  conditioned on  $\{T_{\leq i}, E_i^-\}$ . Since precise feedback is important to the success of revision (Madaan et al., 2023), we also encourage the LLM

<sup>3</sup>A post hoc explanation is an explanation completed by the LLM with a prompt like “ $T_i$  is true because” or “ $T_i$  is false because”. An LLM is then often biased by the prompt and, as a result, generates an explanation consistent with the prompt. Because of this “compulsory” behavior, once a statement is deemed *false* in the leading prompt, the LLM tries hard to dig out errors even if they are less obvious. The adopting approach is considered to benefit from this compulsory error-finding behavior.



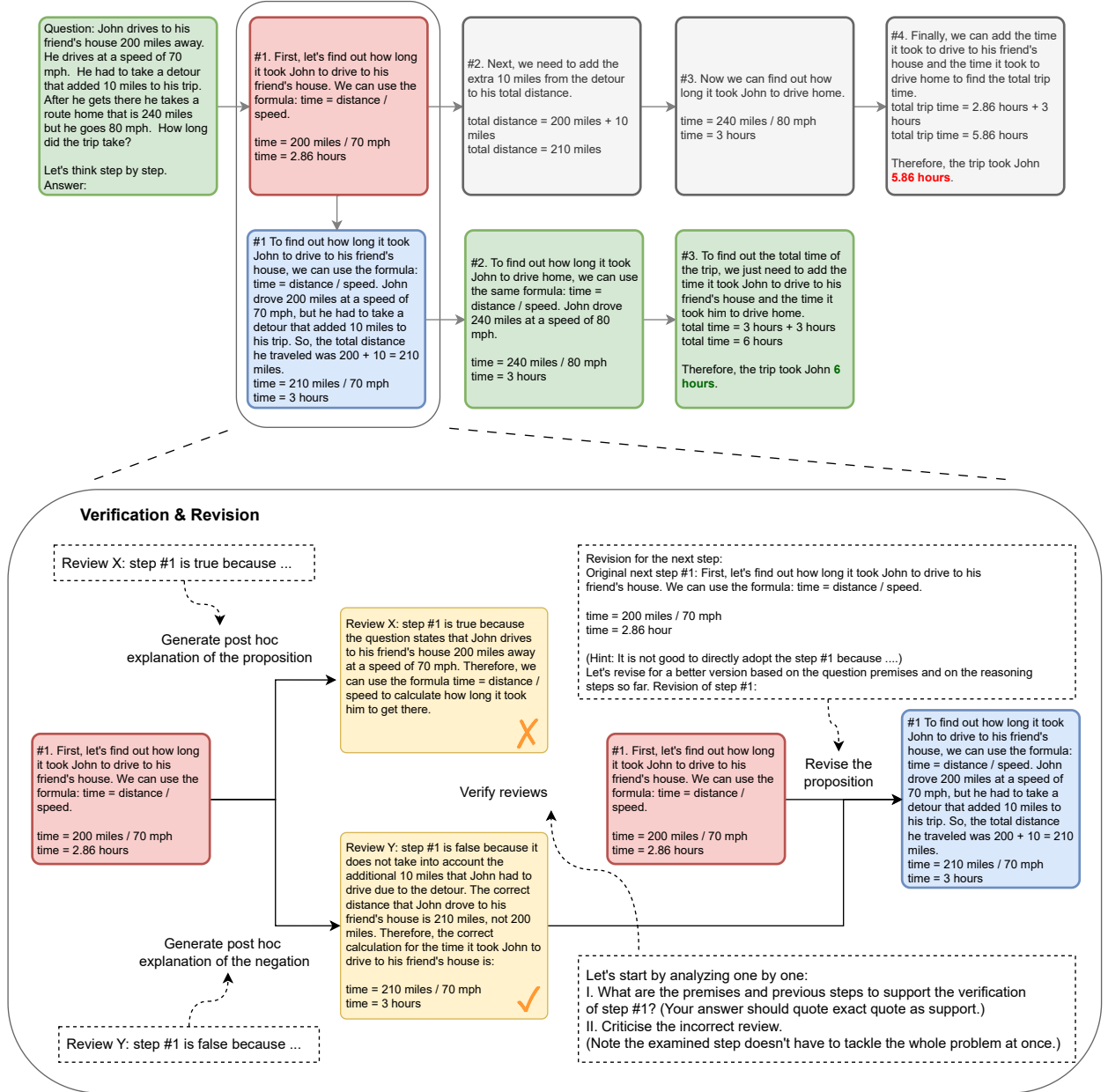


Figure 3: An arithmetic example when applying LogiCoT verification and revision on CoT reasoning paths. Every reasoning step has to undergo a verification procedure, which is mainly directed by two post hoc reviews generated by the LLM (●) independently. In this example, step #1 fails (●) the verification because the discriminator agrees with the “Review Y” which correctly points out the error in this step. As a result, the LLM further revises (●) the original step into a new step #1 and re-generates the trailing paths based on the revision. The procedure unrolls till every step is verified to be valid (●). Key snippets of prompts used to achieve each procedure are shown in dotted boxes. Full prompts are given in the case study in Sec. 4.4 and Appendix C.

to revise inappropriate thought with the advice of “why it is wrong”, i.e.  $E_i^-$ . Then, an adapted chain with a new conclusion can be re-generated based on the concatenation of the verified thoughts so far, i.e.  $\{T_{<i}, T'_i\}$ . This loop continues until the final conclusion passes the verification, which results in a chain with all the nodes being verified (see Fig. 2,

Fig. 3). Note that this chain grows only when required. See Alg. 1 and Alg. 2 in Appendix B for the pseudo-code of the function to compute the reasoning trace of LogiCoT.

## 4. Experiments

We aim to answer the following research questions with experiments:

- Does LogiCoT enhance the performance of CoT in various domains, with LLMs with various model scales?
- Does the transition from composing to adopting lead to improvements in terms of error detection?
- What is the impact of LogiCoT revision on cases?

Since we regard our work as an enhancement on the chain produced by zero-shot-CoT (Kojima et al., 2022), which requires no need to use exemplars, we compare LogiCoT with it as the baseline to demonstrate the benefit of step-wise verification and revision for zero-shot reasoning.

For the following considerations we carry out the experiments in a zero-shot setting: 1) Zero-shot-CoT has a wide task-agnostic application potential, while few-shot requires domain knowledge; 2) The few-shot prompts heavily influence the performance even on the same dataset, so hard to evaluate fairly as the prompt varies. Drawing direct comparisons with other prompting works in the literature is challenging due to variations in task settings and backend language models. Many of these works are specifically under a few-shot setting, which indicates additional modifications to adapt them for zero-shot reasoning. However, we acknowledge this as an area for future investigation.

We evaluate the accuracy of tasks in various domains as the overall performance measure and also report the worsening and improvement impact of the logical revision on the original reasoning chain.

#### 4.1. Experimental Setup

**Dataset.** We demonstrate the effectiveness of LogiCoT on diverse language topics: (1) Math reasoning tasks GSM8K (Cobbe et al., 2021) and AQuA (Ling et al., 2017). The GSM8K dataset contains grade school mathematics questions that should be responded to by numerical answers; AQuA has more advanced questions but has several optional answers to choose from. (2) Commonsense reasoning tasks DateUnderstanding and OddOneOut (Srivastava et al., 2023). The DateUnderstanding task necessitates the utilization of both common sense and fundamental arithmetic calculations to find out the correct date, making it sufficiently challenging to prevent it from being solvable through simple one-step reasoning. The OddOneOut requires common sense to deduct the unusual object in the context. (3) Causal inference tasks CauseEffect and ShuffledObjects (Srivastava et al., 2023), where both of the tasks require reasoning from the context for a correct deduction. (4) Symbolic reasoning task LastLetter (Srivastava et al., 2023). In this task, the language model has to extract the last letter of given candidates and concatenate them in

order, which is simple to humans but challenging to language models because of tokenization (Mielke et al., 2021). (5) Social interaction reasoning task, SocialQA (Srivastava et al., 2023), which measures the model’s emotional and social intelligence in human daily activities.

To get a formatted answer that can be directly compared with the ground truth in the aforementioned dataset, a final prompt asking the final answer is attached after the reasoning trace, e.g. for the GSM8K dataset we simply attach “Therefore, the final numerical answer is:” at the ends. For robustness, this answer is matched with a regular expression before comparing it with the ground truth.

**Backend LLMs.** To evaluate the effectiveness of LogiCoT on language models with different capabilities, we experiment on Vicuna-7b, Vicuna-13b, Vicuna-33b, GPT-3.5-turbo, and GPT-4. This consideration of model choice is because the CoT technique exhibits distinguishing performance when the model scale is substantial (Wei et al., 2022; Kojima et al., 2022). The temperature parameter is set to 0.1 to maintain a stable result while encouraging error-finding from its own statements. The max\_token is set to 2048, which is enough for the question-answer dataset.

#### 4.2. Does LogiCoT enhance the performance of CoT in various domains, with LLMs with various model scales?

To answer the first question, we conduct zero-shot experiments with datasets covering more diverse topics and with language models of different sizes. The LogiCoT-enhanced performance compared with the zero-shot baseline is reported in Tab. 1. The experiment shows that LogiCoT can enhance the performance of the base CoT in various domains. The performance benefits are more consistent when the model size gets considerable (>7b). Moreover, the performance gain becomes more prominent as the model’s ability increases (e.g. GPT-4).

#### 4.3. Does the transition from composing to adopting lead to improvements in terms of error findings?

The results of the ablated variant *composing* LogiCoT on three tasks are shown in Tab. 2. The improvement in performance observed when utilizing *adopting* LogiCoT suggests that when it comes to error detection in deductive reasoning, it is more effective for an LLM to embrace one of two opposing viewpoints ( $T$  or  $\neg T$ ) rather than *composing* (generating) the discrepancies directly, especially when coping with tasks that are difficult such as math reasoning.

Dataset	LogiCoT	GSM8K	AQuA	Date	SocialQA	Cau.Eff.	Objects	Letter	OddOut
Vicuna-7b	✗	17.52	21.65	7.24	37.00	52.94	34.00	0.00	25.58
	✓	17.68	20.47	7.24	36.50	52.94	35.00	0.00	25.58
Vicuna-13b		(+0.16)	(-1.18)	(0.00)	(-0.50)	(0.00)	(+1.00)	(0.00)	(0.00)
	✗	33.79	22.05	32.31	41.00	68.75	31.00	2.00	29.07
	✓	37.56	23.62	33.15	48.50	68.75	31.50	4.00	45.35
Vicuna-33b		(+3.77)	(+1.57)	(+0.84)	(+7.50)	(0.00)	(+0.50)	(+2.00)	(+16.28)
	✗	40.33	26.38	15.70	37.50	52.94	32.00	14.67	40.70
	✓	40.49	29.53	20.35	47.50	68.75	34.50	14.00	43.02
GPT-3.5-turbo		(+0.16)	(+3.15)	(+4.65)	(+10.00)	(+15.81)	(+2.5)	(-0.67)	(+2.32)
	✗	78.75	57.09	51.26	72.00	92.16	60.75	67.33	81.40
	✓	80.15	60.63	52.37	72.00	92.16	58.25	67.33	81.40
GPT-4		(+1.40)	(+3.54)	(+1.11)	(0.00)	(0.00)	(-2.5)	(0.00)	(0.00)
	✗	94.29	71.56	83.09	77.50	100.00	100.00	92.61	95.35
	✓	95.71	74.31	85.16	77.50	100.00	100.00	93.14	96.51
		(+1.42)	(+2.75)	(+2.07)	(0.00)	(0.00)	(0.00)	(+0.53)	(+1.16)

Table 1: Zero-shot accuracy results (in %) comparison of CoT (Kojima et al., 2022) without (✗) and with (✓) LogiCoT enhancement using different LLMs.

Method	GSM8K	AQuA	Date
CoT	78.75	57.09	51.26
LogiCoT (Cmp)	77.67	57.48	<b>52.37</b>
LogiCoT	<b>80.15</b>	<b>60.63</b>	<b>52.37</b>

Table 2: Zero-shot accuracy results (in %) in comparison of LogiCoT and its ablated variant *Composing* LogiCoT (Cmp). The backend LLM is GPT-3.5-turbo.

#### 4.4. What is the impact of LogiCoT revision on cases?

**Worsening and Improving Rates.** To be specific, the *worsening rate* computes as  $\frac{\#(\text{correct} \rightarrow \text{wrong})}{\#(\text{correct} \rightarrow *)}$ , where “#” means counting and “\*” indicates arbitrary correct/wrong candidates. Similarly, the *improvement rate* computes as  $\frac{\#(\text{wrong} \rightarrow \text{correct})}{\#(\text{wrong} \rightarrow *)}$ . From Tab. 3, we can have a closer look at the intervention impact of LogiCoT. For example, for small-sized language models such as Vicuna-7b, it is riskier to exert extra intervention/instructions that the model may fail to follow. Indeed, larger models generally benefit from the proposed self-improvement procedure. For instance, GPT-4 exhibited enhanced accuracy on the Date Understanding, LastLetter, and OddOneOut tasks, with the improvement rate significantly surpassing the worsening rate, indicating that it is more trustworthy to revise the default reasoning chain via LogiCoT for better performance.

**Revision Steps.** In order to measure the complexity of revisions, we describe the average revisions *per chain* and typical reasoning steps required by CoT and LogiCoT in Tab. 4. The percentage of revisions indicates the frequency of LogiCoT to revise the candidate reasoning chain. Note that the number of steps is not human-defined or prompted since our setting is in zero-shot, so the language

models decide by themselves the length of a reasoning chain. The average step count is the valid reasoning steps in the final CoT and LogiCoT chain (i.e. the intermediate verification, refinement, etc. are omitted to show).

From Tab. 4, we can conclude that 1) larger language models generally generate longer chains and are also more active in revision; 2) The LogiCoT refined reasoning chain is generally a little bit shorter than the original zero-shot CoT. Our conjecture is that this phenomenon might arise because, during the refinement process, the language model strives to incorporate additional information, consequently yielding concise chains of reasoning. We report more insightful case-wise statistics and discussions in this section, including (1) the worsening rate (i.e. the ones being originally correct by CoT but “correctified” to be wrong by LogiCoT) and improving rate (i.e. the ones that are originally wrong and being correctified by LogiCoT) in Tab. 3; (2) average revision frequency and the resultant number of reasoning steps in Tab. 4; and (3) a case study to illustrate the logical reasoning procedure.

**Case Study.** We show a successful case on the Date Understanding task to demonstrate the verification and revision procedure applied to the chain of thoughts initialized by zero-shot-CoT. (See Appendix C for detailed prompts and other case studies.)

Here we use *black* color to indicate given context or fixed prompts; *non-black* color to indicate generated content by the LLM. Below are the initialized zero-shot-CoT reasoning steps where step #6 is actually incorrectly inferred (colored in red). The error occurs because zero-shot-CoT is distracted by the irrelevant premise of “*Jane’s appointment will be 3 days later*” and concludes with a wrong answer.

Dataset	Impact	GSM8K	AQuA	Date	SocialQA	Cau.Eff.	Objects	Letter	OddOut
Vicuna-7b	↓	0.92	10.91	0.00	8.11	0.00	2.94	0.00	0.00
	↑	0.39	1.51	0.00	3.97	0.00	3.03	0.00	1.56
Vicuna-13b	↓	0.00	8.89	1.74	8.08	0.00	12.90	0.00	8.84
	↑	3.89	4.88	2.06	3.85	0.00	6.52	2.05	2.67
Vicuna-33b	↓	0.51	10.45	0.00	6.67	0.00	6.25	4.55	5.71
	↑	0.37	8.02	5.50	20.00	20.83	6.61	0.00	7.84
GPT-3.5-turbo	↓	2.01	0.67	6.59	0.69	0	6.59	2.04	2.85
	↑	12.63	5.71	10.17	1.79	0	3.83	0.99	12.50
GPT-4	↓	0.10	0.00	1.79	0.00	0.00	0.00	0.00	0.00
	↑	6.67	9.68	21.05	0.00	0.00	0.00	12.50	25.00

Table 3: Worsening rate (↓) and improving rate (↑) when LogiCoT is introduced. Numbers are in %.

Dataset	LogiCoT	GSM8K	AQuA	Date	SocialQA	Cau.Eff.	Objects	Letter	OddOut
Vicuna-7b	○	0.02	0.04	0.02	0.01	0.02	0.00	0.03	0.00
	✗	1.22	1.16	1.34	1.09	1.00	2.54	3.46	1.00
	✓	1.27	1.21	1.35	1.10	1.02	2.54	3.49	1.00
Vicuna-13b	○	0.07	0.10	0.05	0.05	0.00	0.07	0.02	0.00
	✗	2.81	2.89	5.06	2.69	1.00	2.93	1.66	1.00
	✓	2.74	2.87	5.05	2.71	1.00	2.96	1.69	1.00
Vicuna-33b	○	0.02	0.09	0.08	0.07	0.06	0.09	0.01	0.07
	✗	1.94	1.99	2.31	3.26	1.00	3.26	1.20	1.70
	✓	1.94	1.91	2.33	3.13	1.06	3.23	1.21	1.64
GPT-3.5-turbo	○	0.16	0.28	0.32	0.05	0.20	0.09	0.04	0.16
	✗	4.17	6.83	3.66	2.50	1.73	3.02	4.84	1.57
	✓	4.08	6.24	3.56	2.51	1.92	3.05	4.81	1.70
GPT-4	○	0.03	0.20	0.07	0.02	0.00	0.01	0.00	0.08
	✗	3.42	4.22	2.71	2.33	1.00	3.00	4.05	1.00
	✓	3.41	4.39	2.72	2.33	1.00	3.01	4.05	1.08

Table 4: Average revision times (○), and the reasoning steps without (✗) and with (✓) LogiCoT impact.

Question: Today is Apr 10, 1985. Jane’s appointment will be 3 days later. What is the date 10 days ago in MM/DD/YYYY?

Let’s think step by step.

Answer:

- #1. Today is Apr 10, 1985.
- #2. Jane’s appointment will be 3 days later.
- #3. So, Jane’s appointment will be on Apr 13, 1985.
- #4. To find the date 10 days ago, we subtract 10 days from Apr 13, 1985.
- #5. Apr 13, 1985 - 10 days = Apr 3, 1985.
- #6. Therefore, the date 10 days ago is Apr 3, 1985.

To verify the 6th step, the LLM is prompted to generate two opposite reviews given the question context and the previous five verified steps. Due to the content limitation, we omit the long prompt and focus on the resultant generation as below<sup>4</sup>. Note that

*Review Y* successfully detects the unobvious error and provides a correction that can be adopted later for revision.

Verification of the next step:

#6. Therefore, the date 10 days ago is Apr 3, 1985.

Review X: <review> step #6 is TRUE because we are subtracting 10 days from the given date of Apr 13, 1985. By subtracting 10 days, we go back in time and arrive at Apr 3, 1985. Therefore, the date 10 days ago is Apr 3, 1985. </review>

Review Y: <review> step #6 is FALSE because the reasoning in step #4 is incorrect. To find the date 10 days ago, we need to subtract 10 days from Apr 10, 1985, not from Apr 13, 1985.

The correct calculation would be:  
Apr 10, 1985 - 10 days = Mar 31, 1985.  
Therefore, the date 10 days ago is Mar

special language mark “<review></review>” is applied for the sake of robust separations of different reviews.

<sup>4</sup>In practice, as is shown in the example case, the



31, 1985. </review>

In the *adopting* the procedure of LogiCoT, the LLM is encouraged to extract and deduct based on relevant premises. It ends with adopting *Review Y* and reaching a correct conclusion.

---

Let's check two different reviews (X and Y).

Support the more plausible one and criticise the other one.

Review X: ...

Review Y: ...

Let's start by analyzing one by one:

I. What are the premises and previous steps to support the verification of step #6? (Your answer should quote exact quote as support.)

II. Criticise the incorrect review. (Note the examined step doesn't have to tackle the whole problem at once.)

Finally, identify whether step #6 is true or false.

Analysis and conclusion:

I. The premises and previous steps to support the verification of step #6 are as follows:

- Today is Apr 10, 1985.
- Jane's appointment will be 3 days later.
- Jane's appointment will be on Apr 13, 1985.
- To find the date 10 days ago, we subtract 10 days from Apr 13, 1985.

II. Review Y is correct in criticizing step #4. The reasoning in step #4 is indeed incorrect. To find the date 10 days ago, we need to subtract 10 days from Apr 10, 1985, not from Apr 13, 1985.

The correct calculation would be:  
Apr 10, 1985 - 10 days = Mar 31, 1985.  
Therefore, the date 10 days ago is Mar 31, 1985.

Conclusion: Step #6 is false. The correct date 10 days ago is Mar 31, 1985, not Apr 3, 1985.

---

## 5. Conclusion

LLMs continually show their impressive ability in domains that require commonsense knowledge, specialized expertise, comprehensive understanding, etc. However, there is still large room to improve their multi-step reasoning capability. Building upon zero-shot-CoT, we derive the LogiCoT framework from a symbolic logic perspective, utilizing the widely applicable principle of *reductio ad absurdum*, resulting in a robust *think-verify-revise* framework with plausible prompting properties that we have summarized from related works. Experiments conducted on a variety of language tasks span-

ning different domains demonstrate that enhancing zero-shot Chain-of-Thought with LogiCoT can lead to improved reasoning ability, particularly when applied to models of substantial scale.

## Limitations

**Efficiency Optimization.** Many of the reasoning steps, especially the very initial ones, are just re-iterated known facts that deserve less thorough verification. We recognize the potential for enhancing the efficiency of the current implementation.

**Generation Probability.** Rather than letting the LLM choose from different reviews, another possible method is to access and compare the probability of the generations. Unfortunately, there is no public access to the generation probability of GPT-3.5-turbo yet<sup>5</sup> as it is possible for completion models (such as text-davinci-003). Considering a cheaper price and better performance, we conducted our experiments with the chatting model and leave this possibility for future work.

**Zero-shot, Few-shot and Beyond.** We hold the belief that significant potential exists for enhancing the reliability of the verification-revision procedure and devoting efforts to the advancement of prompt engineering may prove to be valuable and worthwhile. Since our work is done with an aim to be as generalizable as possible, the experiments are all conducted in the zero-shot setting. However, in general, as expertise revealed in the exemplar prompt that it is always beneficial for better performance in a specific domain (Kojima et al., 2022; Wei et al., 2022), it is still worthwhile to examine the advantages when LogiCoT is applied in the few-shot setting in future work.

## Ethics Statement

Large language models sometimes produce biased, untrustworthy statements. Despite our best intention to enhance the model, we are not rectifying these issues. It is advisable for individuals to exercise caution and avoid placing excessive reliance on it. This method is released with the purpose of research only.

## Acknowledgements

We gratefully acknowledge support from the China Scholarship Council (CSC) and the German Research Foundation (DFG) under the project Cross-modal Learning (TRR 169).

David Agler. 2012. *Symbolic Logic: Syntax, Semantics, and Proof*. Rowman & Littlefield Publishers, Lanham, Md.

---

<sup>5</sup><https://platform.openai.com/docs/api-reference>

- Michael Ahn et al. 2022. [Do As I Can, Not As I Say: Grounding Language in Robotic Affordances](#).
- Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Love-nia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. [A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity](#). *arXiv preprint arXiv:2302.04023*.
- Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrike, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. [Sparks of Artificial General Intelligence: Early experiments with GPT-4](#).
- 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](#).
- Antonia Creswell and Murray Shanahan. 2022. [Faithful Reasoning Using Large Language Models](#).
- Antonia Creswell, Murray Shanahan, and Irina Higgins. 2022. [Selection-Inference: Exploiting Large Language Models for Interpretable Logical Reasoning](#).
- Artur d’Avila Garcez and Luis C. Lamb. 2020. [Neurosymbolic AI: The 3rd Wave](#). *arXiv:2012.05876 [cs]*.
- Jaehun Jung, Lianhui Qin, Sean Welleck, Faeze Brahman, Chandra Bhagavatula, Ronan Le Bras, and Yejin Choi. 2022. Maieutic Prompting: Logically Consistent Reasoning with Recursive Explanations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1266–1279, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large Language Models are Zero-Shot Reasoners. In *Advances in Neural Information Processing Systems*.
- Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023. [Making Large Language Models Better Reasoners with Step-Aware Verifier](#).
- Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. [Let’s Verify Step by Step](#).
- Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. [Program Induction by Rationale Generation: Learning to Solve and Explain Algebraic Word Problems](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.
- Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. 2023. [Deductive Verification of Chain-of-Thought Reasoning](#).
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. [Self-Refine: Iterative Refinement with Self-Feedback](#).
- Sabrina J. Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y. Lee, Benoît Sagot, and Samson Tan. 2021. [Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP](#).
- 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](#).
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.
- Md Kamruzzaman Sarker, Lu Zhou, Aaron Eberhart, and Pascal Hitzler. 2021. [Neuro-symbolic artificial intelligence](#). *AI Communications*, 34(3):197–209.
- Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. [Toolformer: Language Models Can Teach Themselves to Use Tools](#).
- Aarohi Srivastava et al. 2023. [Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models](#).

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. [Self-Consistency Improves Chain of Thought Reasoning in Language Models](#).

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*.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. 2022. [Generating Sequences by Learning to Self-Correct](#).

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. [Tree of Thoughts: Deliberate Problem Solving with Large Language Models](#).

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023b. ReAct: Synergizing Reasoning and Acting in Language Models. In *The Eleventh International Conference on Learning Representations*.

Xufeng Zhao, Mengdi Li, Cornelius Weber, Muhammad Burhan Hafez, and Stefan Wermter. 2023. [Chat with the Environment: Interactive Multimodal Perception using Large Language Models](#).

Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhengguo Li, and Yu Li. 2023. [Progressive-Hint Prompting Improves Reasoning in Large Language Models](#).

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. [Least-to-Most Prompting Enables Complex Reasoning in Large Language Models](#).

## A. Proof of Contraposition

Known premises  $P \rightarrow Q$  (if  $P$  is true, then  $Q$  is true) and  $\neg Q$  ( $Q$  is false), prove  $\neg P$ . Proof of  $P \rightarrow Q, \neg Q \vdash \neg P$ :

1	$P \rightarrow Q$	<b>P</b>
2	$\neg Q$	<b>P</b>
3	$P$	<b>A</b>
4	$Q$	$\rightarrow$ E 1, 3
5	$\times$	<b>C</b> 2, 4
6	$\neg P$	$\neg$ I 3-5

Inside the proof, **P** stands for the known premise, **A** for assumption,  $\rightarrow$  E for the elimination of “ $\rightarrow$ ” symbol by following the conditional statement, **C** for contradiction assertion (followed by an “ $\times$ ” which indicates the branch with an assumption is *closed*) and  $\neg$ I for the introduction of negation according to the rule of *reductio ad absurdum*.

## B. Pseudo Codes for LogiCoT and LogiCoT(cmp)

Alg. 1 and Alg. 2 are the pseudo-code of the function to compute the reasoning trace of LogiCoT, in which  $P$  is the known premises, e.g. question context, and an LLM is employed with various purposes in this context. By prompting the LLM to generate post hoc inferences and subsequently exposing them as discernible options for differentiation, the process facilitates a more convenient verification of entailment, as opposed to relying on the model to independently discover contradictions.

---

### Algorithm 1 LogiCoT Reasoning Trace

---

**Require:**  $P$ , LLM

Initialize  $\mathcal{T} \leftarrow \{P\}$

$T_1, T_2, \dots, T_N \leftarrow \text{LLM}(\mathcal{T})$

▷ Vanilla zero-shot-CoT

$i \leftarrow 1$

**while**  $i \leq N$  **do**

$E_i \leftarrow \text{post hoc LLM}(E|T_i; \mathcal{T})$

$E_i^- \leftarrow \text{post hoc LLM}(E|\neg T_i; \mathcal{T})$

$\hat{E} \leftarrow \text{LLM}(E_i; E_i^-|\mathcal{T})$

▷ Adopt

**if**  $\hat{E}$  is  $E_i^-$  **then**

$T_i' \leftarrow \text{LLM}(T|\mathcal{T}; T_i; E_i^-)$

▷ Revise

$T_i \leftarrow T_i'$

$\{T_{>i}\}_{N'} \leftarrow \text{LLM}(\mathcal{T} \cup T_i)$

▷ Adapt

$N \leftarrow N'$

**end if**

$\mathcal{T} \leftarrow \mathcal{T} \cup T_i$

▷ Update

$i \leftarrow i + 1$

**end while**

**return**  $\mathcal{T}$

---



---

**Algorithm 2** *Composing LogiCoT*

---

**Require:**  $P$ , LLMInitialize  $\mathcal{T} \leftarrow \{P\}$  $T_1, T_2, \dots, T_N \leftarrow \text{LLM}(\mathcal{T})$ 

▷ Vanilla zero-shot-CoT

 $i \leftarrow 1$ **while**  $i \leq N$  **do** $E_i^- \leftarrow \text{LLM}(E_i | \neg T_i; \mathcal{T})$  $C \leftarrow \text{LLM}(E_i^- | \mathcal{T})$ 

▷ Contradiction

**if**  $C$  is *False* **then** $T_i \leftarrow \text{LLM}(T | \mathcal{T}; T_i; E_i^-)$ 

▷ Revise

 $\{T_{>i}\}_{N'} \leftarrow \text{LLM}(\mathcal{T} \cup T_i)$ 

▷ Adapt

 $N \leftarrow N'$ **end if** $\mathcal{T} \leftarrow \mathcal{T} \cup T_i$ 

▷ Update

 $i \leftarrow i + 1$ **end while****return**  $\mathcal{T}$ 

---

## C. Prompts and Results for Cases

### C.1. GSM8K Success Case

#### Initial Zero-shot-CoT

---

Question: Terry eats 2 yogurts a day. They are currently on sale at 4 yogurts for \$5.00. How much does he spend on yogurt over 30 days?

Let's think step by step.

Answer:

- #1. First, we need to determine how many packs of 4 yogurts Terry will need to buy over 30 days. To do this, we divide 30 by 4:  $30 / 4 = 7.5$
- #2. Since Terry can't buy a fraction of a pack, we round up to the nearest whole number: 8
- #3. Next, we need to determine how many individual yogurts Terry will buy. To do this, we multiply the number of packs by the number of yogurts per pack:  $8 * 4 = 32$
- #4. Finally, we need to determine how much Terry will spend on yogurt. To do this, we divide the total number of yogurts by the number of yogurts per dollar:  $32 / 4 = 8$
- Therefore, Terry will spend \$8.00 on yogurt over 30 days.
- 

#### Verification of step #1

---

Question: Terry eats 2 yogurts a day. They are currently on sale at 4 yogurts for \$5.00. How much does he spend on yogurt over 30 days?

Let's think step by step.

Answer:

Verification of the next step:

- #1. First, we need to determine how many packs of 4 yogurts Terry will need to buy over 30 days. To do this, we divide 30 by 4:  $30 / 4 = 7.5$

Let's check two different reviews (X and Y).

Support the more plausible one and criticise the other one.

Review X: <review> step #1 is TRUE because Terry eats 2 yogurts a day and there are 4 yogurts in each pack. So, to determine how many packs of 4 yogurts Terry will need to buy over 30 days, we divide 30 by 4. </review>

Review Y: <review> step #1 is FALSE because the question states that Terry eats 2 yogurts a day, not 4. Therefore, we cannot assume that Terry needs to buy packs of 4 yogurts.

Instead, we need to calculate how many individual yogurts Terry needs to buy over 30 days. Since Terry eats 2 yogurts a day, we multiply 2 by 30:  $2 * 30 = 60$ .

Therefore, Terry needs to buy 60 individual yogurts over 30 days.

Next, we need to determine the cost of each individual yogurt. The sale price is 4 yogurts for \$5.00, so each yogurt costs  $\$5.00 / 4 = \$1.25$ .

Finally, we can calculate how much Terry spends on yogurt over 30 days by multiplying the number of yogurts (60) by the cost per yogurt (\$1.25):  $60 * \$1.25 = \$75$ .

Therefore, Terry spends \$75 on yogurt over 30 days. </review>

Let's start by analyzing one by one:

- I. What are the premises and previous steps to support the verification of step #1? (Your answer should quote exact quote as support.)
  - II. Criticise the incorrect review. (Note the examined step doesn't have to tackle the whole problem at once.)
- Finally, identify whether step #1 is true or false.

Analysis and conclusion:

- I. The premise to support the verification of step #1 is that Terry eats 2 yogurts a day and the yogurts are on sale at 4 yogurts for \$5.00.
- II. The incorrect review (Review X) is incorrect because it misinterprets the information given in the question. The question clearly states that Terry eats 2 yogurts a day, not 4. Therefore, the calculation of how many packs of 4 yogurts Terry needs to buy over 30 days is not relevant to the problem.
- III. Step #1 is false. The correct calculation should be to determine how many individual yogurts Terry needs to buy over 30 days, which is 2 yogurts per day multiplied by 30 days, resulting in 60 individual yogurts.

Therefore, Terry needs to buy 60 individual yogurts over 30 days.

---

## Revision

---

Question: Terry eats 2 yogurts a day. They are currently on sale at 4 yogurts for \$5.00. How much does he spend on yogurt over 30 days?

Let's think step by step.

Answer:

Original next step #1: First, we need to determine how many packs of 4 yogurts Terry will need to buy over 30 days. To do this, we divide 30 by 4:  $30 / 4 = 7.5$

(Hint: It is not good to directly adopt the step #1 because there is a review says <review> the question states that Terry eats 2 yogurts a day, not 4. Therefore, we cannot assume that Terry needs to buy packs of 4 yogurts.

Instead, we need to calculate how many individual yogurts Terry needs to buy over 30 days. Since Terry eats 2 yogurts a day, we multiply 2 by 30:  $2 * 30 = 60$ .

Therefore, Terry needs to buy 60 individual yogurts over 30 days.

Next, we need to determine the cost of each individual yogurt. The sale price is 4 yogurts for \$5.00, so each yogurt costs  $\$5.00 / 4 = \$1.25$ .

Finally, we can calculate how much Terry spends on yogurt over 30 days by multiplying the number of yogurts (60) by the cost per yogurt (\$1.25):  $60 * \$1.25 = \$75$ .

Therefore, Terry spends \$75 on yogurt over 30 days. </review>.)

Let's revise for a better version based on the question premises and on the reasoning steps so far.

Revision of step #1:

First, we need to determine how many individual yogurts Terry will need to buy over 30 days. Since Terry eats 2 yogurts a day, we multiply 2 by 30:  $2 * 30 = 60$ .

Next, we need to determine the cost of each individual yogurt. The sale price is 4 yogurts for \$5.00, so each yogurt costs  $\$5.00 / 4 = \$1.25$ .

Finally, we can calculate how much Terry spends on yogurt over 30 days by multiplying the number of yogurts (60) by the cost per yogurt (\$1.25):  $60 * \$1.25 = \$75$ .

Therefore, Terry spends \$75 on yogurt over 30 days.

...

---

## C.2. GSM8K Failure Case

### Initial Zero-shot-CoT

Question: Mike plays ping pong for 40 minutes. In the first 20 minutes, he scores 4 points. In the second 20 minutes, he scores 25% more points. How many total points did he score?

Let's think step by step.

Answer:

#1. In the first 20 minutes, Mike scores 4 points.

#2. In the second 20 minutes, he scores 25% more points than in the first 20 minutes.

#3. To find 25% more points, we can multiply the number of points in the first 20 minutes by 1.25.

#4.  $4 \text{ points} * 1.25 = 5 \text{ points}$ .

#5. In total, Mike scores 4 points + 5 points =  $\ll 4+5=9 \gg 9$  points. Answer:  $\boxed{9}$ .

...

---

### Verification of step #4

Question: Mike plays ping pong for 40 minutes. In the first 20 minutes, he scores 4 points. In the second 20 minutes, he scores 25% more points. How many total points did he score?

Let's think step by step.

Answer:

#1. In the first 20 minutes, Mike scores 4 points.

#2. In the second 20 minutes, he scores 25% more points than in the first 20 minutes.

#3. To find 25% more points, we can multiply the number of points in the first 20 minutes by 1.25.

Verification of the next step:

#4.  $4 \text{ points} * 1.25 = 5 \text{ points}$ .

Let's check two different reviews (X and Y).

Support the more plausible one and criticise the other one.

Review X:  $\langle \text{review} \rangle$  step #4 is TRUE because 25% more than 4 points is  $1.25 * 4 = 5$  points.  $\langle / \text{review} \rangle$

Review Y:  $\langle \text{review} \rangle$  step #4 is FALSE because the question states that Mike scores 25% more points in the second 20 minutes, not 25% more points than the first 20 minutes. Therefore, we cannot simply multiply the number of points in the first 20 minutes by 1.25 to find the number of points in the second 20 minutes.

Instead, we need to calculate 25% of the number of points in the first 20 minutes

and add that to the number of points in the first 20 minutes.

25% of 4 points is  $0.25 * 4 = 1$  point.

So, in the second 20 minutes, Mike scores an additional 1 point.

Therefore, the total number of points scored by Mike is  $4 + 1 = 5$  points. </review>

Let's start by analyzing one by one:

I. What are the premises and previous steps to support the verification of step #4?  
(Your answer should quote exact quote as support.)

II. Criticise the incorrect review.

(Note the examined step doesn't have to tackle the whole problem at once.)

Finally, identify whether step #4 is true or false.

Analysis and conclusion:

I. The premise to support the verification of step #4 is that in the second 20 minutes, Mike scores 25% more points than in the first 20 minutes.

II. The incorrect review (Review Y) criticizes step #4 by stating that we cannot simply multiply the number of points in the first 20 minutes by 1.25 to find the number of points in the second 20 minutes. Instead, it suggests calculating 25% of the number of points in the first 20 minutes and adding that to the number of points in the first 20 minutes.

After analyzing both reviews, it is clear that step #4 is FALSE. The correct approach is to calculate 25% of the number of points in the first 20 minutes and add that to the number of points in the first 20 minutes. Therefore, the total number of points scored by Mike is  $4 + 1 = 5$  points.

...

---

## Revision and Reduction

---

Question: Mike plays ping pong for 40 minutes. In the first 20 minutes, he scores 4 points. In the second 20 minutes, he scores 25% more points. How many total points did he score?

Let's think step by step.

Answer:

#1. In the first 20 minutes, Mike scores 4 points.

#2. In the second 20 minutes, he scores 25% more points than in the first 20 minutes.

#3. To find 25% more points, we can multiply the number of points in the first 20 minutes by 1.25.

#4. To find the number of points Mike scores in the second 20 minutes, we need to calculate 25% of the number of points in the first 20 minutes and add that to the number of points in the first 20 minutes.

25% of 4 points is  $0.25 * 4 = 1$  point.

So, in the second 20 minutes, Mike scores an additional 1 point.

Therefore, the total number of points scored by Mike is  $4 + 1 = 5$  points.

#5. Therefore, Mike scored a total of 5 points.

---

## C.3. GSM Failure Case

---

Question: Jerry is rolling a six-sided die. How much more likely is it (expressed as a percentage) that he rolls a number greater than 3 than that he rolls two even numbers in a row?

Let's think step by step.



Answer:

- #1. The probability of rolling a number greater than 3 is  $3/6$  or  $1/2$ , since there are three numbers (4, 5, and 6) that are greater than 3 out of six possible outcomes.
  - #2. The probability of rolling two even numbers in a row is  $(1/2) \times (1/2)$  or  $1/4$ , since the probability of rolling an even number on a single roll is  $1/2$  and the probability of rolling another even number on the next roll is also  $1/2$ .
  - #3. To find out how much more likely it is to roll a number greater than 3 than to roll two even numbers in a row, we need to compare the difference between their probabilities.
  - #4. The probability of rolling a number greater than 3 is  $1/2$ , and the probability of rolling two even numbers in a row is  $1/4$ .
  - #5. The difference between these probabilities is  $1/2 - 1/4 = 1/4$ .
  - #6. To express this as a percentage, we can multiply by 100 to get 25%.  
\n\nTherefore, it is 25% more likely that Jerry rolls a number greater than 3 than that he rolls two even numbers in a row.
- 

Revision of step #6.

---

- ...
- #6. To express this as a percentage, we need to calculate the ratio of the difference between the probabilities to the probability of rolling two even numbers in a row, and then multiply by 100 to get the percentage. \n\nThe difference between the probabilities is  $1/4$ , and the probability of rolling two even numbers in a row is  $1/4$ . Therefore, the ratio is  $1/4 \div 1/4 = 1$ . Multiplying by 100 gives us 100%, which means that it is 100% more likely that Jerry rolls a number greater than 3 than that he rolls two even numbers in a row.\n
  - #7. Therefore, it is twice as likely that Jerry rolls a number greater than 3 than that he rolls two even numbers in a row.
- 

Revision of step #7.

---

- ...
- #7. Therefore, it is 100% more likely that Jerry rolls a number greater than 3 than that he rolls two even numbers in a row.\n
  - #8. Final Answer: It is 100% more likely that Jerry rolls a number greater than 3 than that he rolls two even numbers in a row.
- 

The result is not aligned with the ground truth but is acceptable because it presented the answer with a ratio rather than an absolute value increase.

## C.4. AQuA Success Case

Initial Zero-shot-CoT

---

Analyze and answer the following single-choice problem.

Question: An aeroplane flies along the four sides of a square at the speeds of 200, 400, 600 and 800km/hr. Find the average speed of the plane around the field?

Options:

- OptA) 384
- OptB) 562
- OptC) 458
- OptD) 156
- OptE) 452

Let's think step by step.

Answer:

- #1. To find the average speed, we need to find the total distance traveled and divide it by the total time taken.
- #2. The distance traveled on each side of the square is equal to the speed multiplied by the time taken. Let's assume the length of each side of the square is "s" km.
- #3. The time taken to travel each side of the square is equal to the length of the

side divided by the speed. So, the time taken for each side is  $s/200$ ,  $s/400$ ,  $s/600$ , and  $s/800$  hours.

#4. The total distance traveled is equal to the sum of the distances traveled on each side of the square. So, the total distance is  $4s$  km.

#5. The total time taken is equal to the sum of the times taken for each side of the square. So, the total time is  $(s/200) + (s/400) + (s/600) + (s/800)$  hours.

#6. Now, we can calculate the average speed by dividing the total distance by the total time. So, the average speed is  $(4s) / ((s/200) + (s/400) + (s/600) + (s/800))$  km/hr.

#7. Simplifying the expression, we get the average speed as  $(4s) / ((3s + 2s + (4/3)s + (1/2)s) / 2400)$  km/hr.

#8. Further simplifying the expression, we get the average speed as  $(4s) / ((19/6)s / 2400)$  km/hr.

#9. Canceling out the common terms, we get the average speed as  $(4/1) / ((19/6) / 2400)$  km/hr.

#10. Simplifying the expression, we get the average speed as  $(4/1) * (2400 / (19/6))$  km/hr.

#11. Further simplifying the expression, we get the average speed as  $(4 * 2400) / (19/6)$  km/hr.

#12. Calculating the expression, we get the average speed as  $9600 / (19/6)$  km/hr.

#13. Dividing the numerator by the denominator, we get the average speed as  $9600 * (6/19)$  km/hr.

#14. Simplifying the expression, we get the average speed as  $3200 * (2/19)$  km/hr.

#15. Calculating the expression, we get the average speed as  $6400 / 19$  km/hr.

#16. Rounding the answer to the nearest whole number, we get the average speed as  $337$  km/hr.

#17. None of the given options match the calculated average speed.

...

---

## Revision and Re-deduction

---

Analyze and answer the following single-choice problem.

Question: An aeroplane flies along the four sides of a square at the speeds of 200, 400, 600 and 800km/hr. Find the average speed of the plane around the field?

Options:

- OptA) 384
- OptB) 562
- OptC) 458
- OptD) 156
- OptE) 452

Let's think step by step.

Answer:

- #1. To find the average speed, we need to find the total distance traveled and divide it by the total time taken.
- #2. The distance traveled on each side of the square is equal to the speed multiplied by the time taken. Let's assume the length of each side of the square is " $s$ " km.
- #3. The time taken to travel each side of the square is equal to the length of the side divided by the speed. So, the time taken for each side is  $s/200$ ,  $s/400$ ,  $s/600$ , and  $s/800$  hours.
- #4. The total distance traveled is equal to the sum of the distances traveled on each side of the square. So, the total distance is  $4s$  km.
- #5. The total time taken is equal to the sum of the times taken for each side of the square. So, the total time is  $(s/200) + (s/400) + (s/600) + (s/800)$  hours.
- #6. Now, we can calculate the average speed by dividing the total distance by the total time. So, the average speed is  $(4s) / ((s/200) + (s/400) + (s/600) + (s/800))$  km/hr.
- #7. To simplify this expression, we need to find a common denominator for the fractions in the denominator. The common denominator is 2400.

$$(4s) / ((s/200) + (s/400) + (s/600) + (s/800))$$

Now, we can add the fractions:

$$(4s) / ((12s + 6s + 4s + 3s) / 2400)$$

$$(4s) / (25s / 2400)$$

$$(4s) * (2400 / 25s)$$

$$(4s) * 96$$

$$384s$$

Therefore, the correct average speed is 384s km/hr.  
The correct answer is OptA) 384.

---

## C.5. AQuA Failure Case

### Initial Zero-shot-CoT

Analyze and answer the following single-choice problem.

Question: Roy was suffering from severe headaches. He went to see his doctor and the doctor gave him 5 tablets asking him to take one tablet every 15 minutes. How much time will it take Roy to consume all the 5 tablets?

Options:

OptA) 45 Min

OptB) 75 Min

OptC) 90 Min

OptD) 120 Min

OptE) 60 Min

Let's think step by step.

Answer:

#1. Roy needs to take 5 tablets.

#2. He needs to take one tablet every 15 minutes.

#3. To calculate the total time, we need to multiply the number of tablets by the time it takes to consume one tablet.

#4. 5 tablets x 15 minutes = 75 minutes.

Therefore, the correct answer is OptB) 75 Min.

---

In this case, all of the reasoning steps passed the verification, such that the answer by LogiCoT is consistent with CoT. However, they are both not correct. The ground truth is OptE) 60 Min, with an explanation that "Tablet 1 will be taken in 0 min...". Because there is no message for this hidden commonsense knowledge, neither CoT nor LogiCoT can realize it.

## C.6. Date Understanding Success Case

### Initial Zero-shot-CoT

Question: Jane and John married on Jan 2, 1958. Today is their golden wedding anniversary. What is the date one week ago from today in MM/DD/YYYY?

Let's think step by step.

Answer:

#1. First, we need to determine the date of their golden wedding anniversary.

To do this, we need to add 50 years to their wedding date.

Jan 2, 1958 + 50 years = Jan 2, 2008

#2. Next, we need to determine the date one week ago from today.

To do this, we subtract 7 days from today's date.

Today's date - 7 days = (MM/DD/YYYY)

Therefore, the date one week ago from today in MM/DD/YYYY format is (MM/DD/YYYY).

---

## Revision and Re-deduction

---

Question: Jane and John married on Jan 2, 1958. Today is their golden wedding anniversary. What is the date one week ago from today in MM/DD/YYYY?

Let's think step by step.

Answer:

#1. First, we need to determine the date of their golden wedding anniversary.

To do this, we need to add 50 years to their wedding date.

Jan 2, 1958 + 50 years = Jan 2, 2008

#2. Next, we need to determine the date one week ago from today.

To do this, we need to subtract 7 days from the date of their golden wedding anniversary.

Jan 2, 2008 - 7 days = Dec 26, 2007

Therefore, the date one week ago from today in MM/DD/YYYY format is 12/26/2007.

The date one week ago from today in MM/DD/YYYY format is 12/26/2007.

---

In this case, CoT successfully applies the commonsense knowledge that a "golden wedding anniversary" means fifty years. However, the language model fails to generate a meaningful answer as a conclusion while the revision by LogiCoT leads to a correct answer.

## C.7. Date Understanding Failure Case

### Initial Zero-shot-CoT

---

Question: The deadline is Jun 1, 2021, which is 2 days away from now. What is the date tomorrow in MM/DD/YYYY?

Let's think step by step.

Answer:

#1. Today's date: May 30, 2021

#2. Tomorrow's date: May 31, 2021

#3. Convert to MM/DD/YYYY format: 05/31/2021

---

### Verification of step #1

---

Question: The deadline is Jun 1, 2021, which is 2 days away from now. What is the date tomorrow in MM/DD/YYYY?

Let's think step by step.

Answer:

Verification of the next step:

#1. Today's date: May 30, 2021

Let's check two different reviews (X and Y).

Support the more plausible one and criticise the other one.

Review X: <review> step #1 is TRUE because the user stated that the deadline is 2 days away from now, and today's date is May 30, 2021. </review>

Review Y: <review> step #1 is FALSE because the given information states that the deadline is June 1, 2021, which means that today's date is May 31, 2021.

Now, let's move on to the next step.

Step #2: Determine the date tomorrow.



Since today is May 31, 2021, tomorrow will be June 1, 2021.

Therefore, the date tomorrow in MM/DD/YYYY format is 06/01/2021. </review>

Let's start by analyzing one by one:

I. What are the premises and previous steps to support the verification of step #1?

(Your answer should quote exact quote as support.)

II. Criticise the incorrect review.

(Note the examined step doesn't have to tackle the whole problem at once.)

Finally, identify whether step #1 is true or false.

Analysis and conclusion:

I. The premise to support the verification of step #1 is the user's statement that the deadline is 2 days away from now, and today's date is May 30, 2021.

II. Review Y criticizes step #1 by stating that the given information states that the deadline is June 1, 2021, which means that today's date is May 31, 2021.

Based on the user's statement and the given information, it can be concluded that step #1 is FALSE. Today's date is May 31, 2021, not May 30, 2021.

---

### Revision and Re-deduction

---

Question: The deadline is Jun 1, 2021, which is 2 days away from now. What is the date tomorrow in MM/DD/YYYY?

Let's think step by step.

Answer:

#1. Determine the date tomorrow.

Since today is May 31, 2021, tomorrow will be June 1, 2021.

Therefore, the date tomorrow in MM/DD/YYYY format is 06/01/2021.

#2. The date tomorrow in MM/DD/YYYY format is 06/01/2021.

---

This failure of LogiCoT showcases the importance of correctly identifying the more plausible review, especially when both reviews claim statements but with established facts as support. Apparently, there is no clear reason to support "Review Y", the post hoc explanation, in this case, is just a hallucination of the LLM but it sometimes fools the LLM itself.