# LLM-BASED TYPED HYPERRESOLUTION FOR COMMONSENSE REASONING WITH KNOWLEDGE BASES

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Large language models (LLM) are being increasingly applied to tasks requiring commonsense reasoning. Despite their outstanding potential, the reasoning process of LLMs is prone to errors and hallucinations that hinder their applicability, especially in high-stakes scenarios. Several works have attempted to enhance commonsense reasoning performance of LLMs by (i) using prompting styles that elicit more accurate reasoning, (ii) utilizing the LLM as a semantic parser for a symbolic reasoner, or (iii) enforcing the LLM to simulate a logical inference rule. However, all these solutions have critical limitations: they are unable to leverage the internal commonsense knowledge of the LLM in tandem with an axiomatic knowledge base, they lack a mechanism to reliably repair erroneous inference steps, and their application is restricted to small knowledge bases that fit the context limit of the LLM. In this work, we present LLM-based Typed Hyperresolution (LLM-TH), a logical commonsense reasoning framework that leverages *"theory resolution"*, a concept from classical logical inference which enables integrating LLMs into the *"resolution"* inference rule, thus mitigating reasoning errors and hallucinations and enabling verification of the reasoning procedure. LLM-TH is also equipped with a mechanism for repairing erroneous inference steps supported by theoretical guarantees. Using *"Hyperresolution"* and *"Typed inference"* schemes, we show that LLM-TH can efficiently reason over large knowledge bases consisting of tens of thousands of rules with arbitrary predicate arities. Our experiments on three diverse language-based reasoning tasks—preference reasoning, multi-domain deductive reasoning, and geographical question answering—showcase that LLM-TH, using merely a BART 406M parameter NLI entailment model, significantly reduces reasoning errors compared to baselines using Llama3-70B, Gemini1.5-Flash, GPT-3.5-Turbo, and Mixtral-46.7B.

## 1 INTRODUCTION

The breakthrough in Large Language Models (LLMs) has significantly impacted AI research, paving the way for deploying AI-powered systems in various tasks and applications. This huge impact is primarily due to the outstanding performance of LLMs in tasks that require substantial reasoning skills (Chang et al., 2024; Plaat et al., 2024). LLMs have also acquired commonsense understanding, a critical element for interacting with the real world (Zhao et al., 2024; Valmeekam et al., 2024). However, reasoning performance of LLMs is not infallible. They commonly show reasoning errors and make hallucinations–generating incorrect outputs that seem valid–which hinders their reliable deployment, particularly in high-risk tasks (Tonmoy et al., 2024; Zhang et al., 2023b).

To overcome these challenges in LLM-based reasoning, several approaches have been proposed in the literature that can be broadly categorized into three groups: (i) Using prompting styles that can elicit more accurate reasoning from the LLM (Wei et al., 2022; Kojima et al., 2022; Zhou et al., 2022) or augmenting the prompt by retrieved information (Lewis et al., 2020b), (ii) using the LLM to translate natural language problem and knowledge bases (KB) for a symbolic logical solver (Olausson et al., 2023; Pan et al., 2023), and (iii) using the LLM to emulate a logical inference rule to solve the reasoning problem (Kazemi et al., 2023; Lee & Hwang, 2024).

These works have notably advanced the logical reasoning performance of LLMs; yet, they are all hindered by a number of important limitations: (a) Their application is limited to small KBs that can

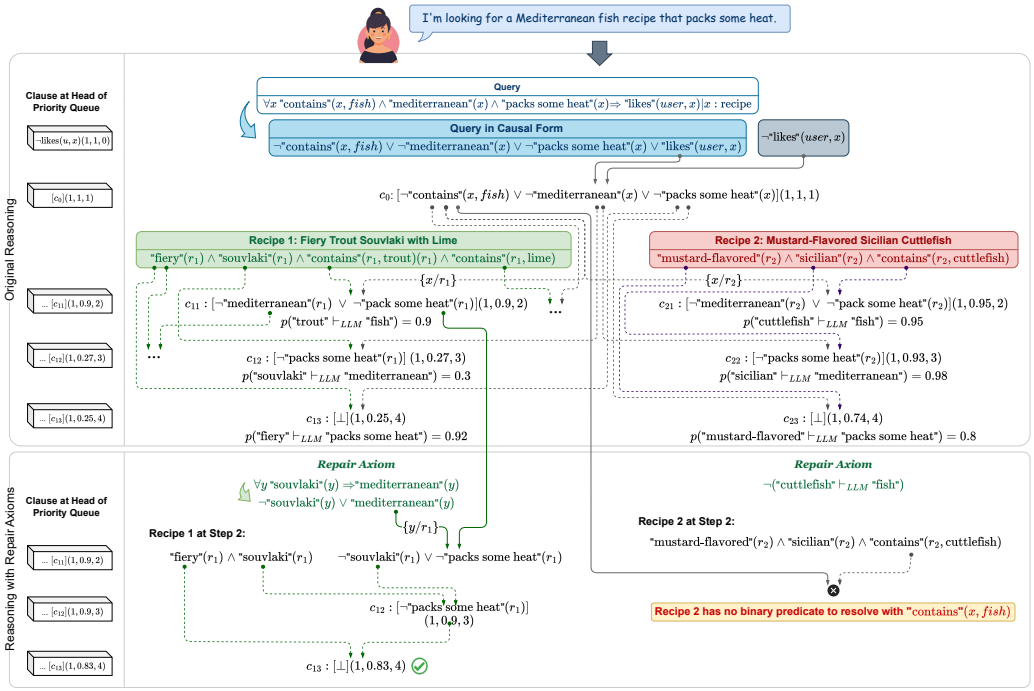


Figure 1: Workflow of LLM-TH shown with a preference reasoning example. Top: Using LLM-based typed hyperresolution to compute proof scores for each recipe option to entail user query. Negated query is the first active clause, and each resolvent is assigned a priority tuple: (*type entailment score*, *predicate entailment score*, *proof length score*) and pushed to the priority queue (only the foremost clause is shown for priority queue of recipe 1 which becomes the next active clause). LLM makes two mistakes: assigning a low score to *"Souvlaki"* entailing *"Mediterranean"* and a high score to *"cuttlefish"* entailing *"fish"*. Bottom: Both kinds of mistakes can be fixed after the insertion of repair axioms, resulting in the correct recommendation of Recipe 1.

fit into the context limit of the LLM, and are not scalable to perform reasoning on practical KBs such as the widely-used Knowledge Graphs (KG) containing thousands of facts and axioms. (b) They are restricted to perform reasoning on a complete KB containing all rules required to solve the problem. However, assuming access to such KB is typically unrealistic in practical use cases, thus calling for the necessity of a methodology to leverage the internal commonsense knowledge of the LLM in the reasoning process. (c) All steps involved in the reasoning process are not transparent and thus, the correctness of the final answer cannot be determined by inspecting the reasoning process. (d) Upon observation of a reasoning error, they do not provide any reliable framework to fix the error and ensure it will not occur in the future.

In this work, we aim to address these limitations by making the following contributions:

- We introduce LLM-based Typed Hyperersolution (LLM-TH), a framework for efficient complex logical commonsense reasoning (Singh et al., 2021; Ismayilzada et al., 2023; He et al., 2021), with KBs containing predicates of arbitrary artities, that facilitates the incorporation of the internal commonsense knowledge of LLMs in the reasoning process. LLM-TH is founded on *"theory resolution"* (Stickel, 1985; Baumgartner, 1992), a concept from classical logical reasoning that allows for the incorporation of specialized theorem provers into the resolution inference rule. (Section 3.1)
- We equip LLM-TH with a mechanism for incorporating the type information of the variables and constants in the problem domain to prune the proof search space and terminate the exploration of reasoning paths that are unlikely to succeed at very early stages. Also, using hyperresolution, an extension of resolution that enables combining clauses to perform several resolution steps simultaneously, we make LLM-TH an efficient and scalable

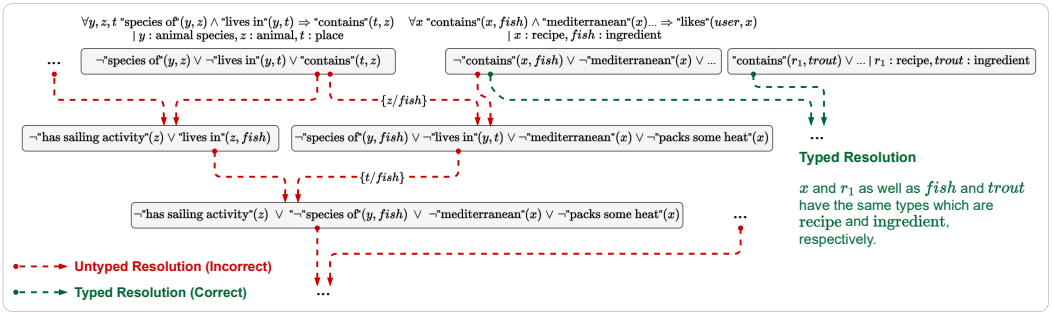


Figure 2: An example of typing mechanism of LLM-TH. The binary predicate *"contains"* is common among clauses from different domains. Left: Application of resolution rule without considering types leads to resolving a literal about *"recipes"* with complementary literals from domains such as *"animals"*. Each incorrect resolution results in a new branch that will be explored but leads to failure, making the process inefficient. Right: in typed resolution, only literals with consistent variable and constant types will be unified, therefore pruning the search space and enhancing efficiency.

reasoning framework for logical commonsense reasoning with LLMs. We show that LLM-TH is easily scalable to KBs consisting of tens of thousands of rules. (Section 5.2.1)

- We show that by providing access to the exact axioms and facts used at every reasoning step, LLM-TH results in a verifiable and faithful reasoning performance. Furthermore, we propose a reliable repair methodology for missed inferences and incorrect reasoning steps due to LLM hallucinations and missed inferences, and provide theoretical proof that it reliably fixes reasoning errors. (Section 4)
- We show that by using the theory hyperresolution framework, LLM-TH is able to leverage the internal commonsense knowledge of the LLM to compensate for KB incompleteness and perform accurate and reliable reasoning. (Section 5.2.2)
- We experiment with LLM-TH on three different tasks involving commonsense reasoning: preference reasoning, multi-domain deductive reasoning, and geographical QA, showcasing the superiority of LLM-TH in terms of answer and reasoning accuracy over Chain of Thought (Kojima et al., 2022; Wei et al., 2022) and retrieval augmented generation (RAG) (Lewis et al., 2020b) baselines using orders of magnitude larger LLMs. (Section 5)

## 2 Related Works

**Eliciting Stronger Reasoning from LLMs** As LLMs scale, they exhibit emergent behaviors such as the capability of solving problems that involve reasoning (Chang et al., 2024; Huang & Chang, 2022). However, their reasoning performance often suffers from errors and incorporating hallucinated facts in their judgments (Tonmoy et al., 2024; Zhang et al., 2023b). Several works have shown that with certain prompting techniques such as Chain of Thought (CoT) (Wei et al., 2022; Kojima et al., 2022), Tree of thought (Yao et al., 2024), Selection-Inference (Creswell et al., 2022), Self-consistency (Wang et al., 2022), Least to most prompting (Zhou et al., 2022), etc., more accurate reasoning can be elicited from LLMs. Retrieval Augmented Generation (RAG) (Lewis et al., 2020b) has also been noted as an effective approach in reducing reasoning errors and hallucinations by including relevant content retrieved from a KB in the prompt to condition LLM's reasoning on dependable information. These methods have made significant progress in advancing the reasoning behavior of LLMs, but even applying them does not guarantee an accurate reasoning behavior from the LLM. Furthermore, since the LLM is entirely in charge of doing the reasoning, there is no control over the reasoning process and its correctness cannot be verified (Shanahan, 2024; Pan et al., 2023). Moreover, the performance of these methods has been shown to degrade when being applied to long-horizon (Dziri et al., 2024) and out-of-domain (Saparov et al., 2024) reasoning tasks as well as problems involving negation (Anil et al., 2022) and contraposition (Zhang et al., 2024).

**Formal Reasoning with LLMs** To offer more control over the reasoning process, two groups of work have been proposed for performing formal reasoning with LLMs: (i) *Semantic parsing* meth-

ods remove the LLM from reasoning and only use it to translate the problem into a symbolic format and delegate the reasoning task to a symbolic solver (Pan et al., 2023; Olausson et al., 2023). (ii) Works enforcing the LLM to emulate an established logical inference rule such as backchaining (Kazemi et al., 2023; Lee & Hwang, 2024). While these groups of works make significant progress in mitigating hallucinations, they both rely on the existing rules in the KB and have no particular mechanism to leverage the rich commonsense knowledge of the LLM in their reasoning. Although recent works (Toroghi et al., 2024) have used resolution inference rule for logical LLM-based reasoning, they are restricted to unary predicates and can only do reasoning over small KBs.

## 3 METHODOLOGY

We first review the (hyper)resolution rule for inference in first-order logic using a function and equality-free first-order logical (FOL) syntax in clausal normal form (Chang & Lee, 2014).[1] We then proceed to introduce the concept of Theory resolution that leverages external sources of knowledge (such as LLMs) to augment the resolution process. Next, we introduce the concept of typed resolution that will be critical for improving the efficiency of inference in our work by prioritizing inference with compatible types and reducing proof length. With all of these ingredients in place, we conclude with our LLM Theory Hyperresolution algorithm (LLM-TH) along with a crucial Lemma that ensures the correctness of *repair axioms* that can be used to override incorrect LLM inferences.

**Resolution Rule and Hyperresolution** *Resolution* is a sound and complete inference rule which is widely used in logical reasoning. From two premise clauses containing complementary literals, resolution rule derives a *resolvent* clause by canceling (resolving) the complementary literals, e.g.,

$$\frac{A(x) \vee B(x,y) \quad \neg B(w,z) \vee C(z)}{A(x) \vee C(y)}\ , \tag{1}$$

under the unification $\theta = \{x/w, y/z\}$. Repeated application of the resolution rule will either result in a contradiction, e.g., deriving both $A(x)$ and $\neg A(x)$ indicating an inconsistent clause set, or reaching a point where no further resolutions are possible.

The efficiency of the repeated application of the resolution rule can be substantially improved by *hyeperresolution* (Robinson, 1965), an extension of resolution that enables combining multiple resolution steps in one inference step. Concretely, it resolves positive literals with all possible matching negative clauses simultaneously, e.g.,

$$\frac{A(x) \vee B_1(x,y) ... \vee B_n(x,y) \quad C(z) \vee \neg B_1(w,z) \vee ... \vee \neg B_n(w,z)}{A(x) \vee C(y)}\ , \theta = \{x/w, y/z\}. \tag{2}$$

**LLM-based Theory Resolution** Application of the resolution rule is originally restricted to clauses with complementary literals that share identical predicates. Theory resolution (Stickel, 1985; Baumgartner, 1992) relaxes this condition and broadens the applicability of the resolution rule by integrating special-purpose theories into resolution. Based on theory resolution, given two clauses $c_1 = A(x) \vee B(x,y)$ and $c_2 = \neg C(w,z) \vee D(z)$, if a theorem prover $T$ identifies $B(x,y)$ and $\neg C(w,z)$ under unification $\theta = \{x/w, y/z\}$ to be unsatisfiable (i.e., $\forall x \forall y B(x,y) \wedge \neg C(x,y) \vdash_T \perp$), the clauses can be resolved despite lacking complimentary literals with identical predicates:

$$\frac{A(x) \vee B(x,y) \quad \neg C(w,z) \vee D(z)}{A(x) \vee D(y)}\ , \theta = \{x/w, y/z\}. \tag{3}$$

In this work, we use an LLM as the theory that identifies the unsatisfiable natural language predicates to perform reasoning via theory resolution. Translating natural language to symbolic form, as semantic parsing methods do, is substantially restricted. For example, they map *"packs some heat"* and being *"spicy"* to completely different symbolic predicates. Therefore, a symbolic reasoner is unable to discern their entailment relationship unless given explicit axioms.

Using LLM-based theory resolution, we can integrate the LLM's commonsense knowledge into the reasoning process to find entailments between predicates and constants without requiring explicit

[1] We refer the reader to Chapters 7-9 of Russell & Norvig(2016) for an in-depth overview of these topics.

axioms. We do this in an extended version of FOL in which predicates, functions, and constants are no longer symbols, but natural language text. In this logical system, which we call *natural language logic*, the unsatisfiability condition in theory resolution reduces to natural language entailment. In other words, if an LLM identifies a natural language predicate $B$ to entail predicate $D$, i.e., $B(x) \vdash_{LLM} D(x)$, and therefore, $B(x) \wedge \neg D(x) \vdash_{LLM} \bot$, then literals $B(x)$ and $D(x)$ can be resolved. For instance, given clauses $c_1 =$*"packs some heat"*$(x)$ and $c_2 = \neg$*"spicy"* $(x) \vee Q(x)$, in which $Q(x)$ is another literal with a natural language predicate, since the LLM identifies the natural language entailment *"packs some heat"* $\vdash_{LLM}$ *"spicy"*, a theory resolution step can be performed as

$$\frac{\text{"packs some heat"}(x) \quad \neg\text{"spicy"}(x) \vee Q(x)}{Q(x)} \,. \tag{4}$$

**LLM-based Typed Theory Resolution** Although resolution is one of the most widely used inference rules in logical reasoning with the key properties of being sound and complete, its application on real-world large-scale knowledge bases is computationally expensive. One of the key challenges is that the space of possible resolutions—the combinations of literals and clauses that can potentially be unified and resolved—can become enormous, and grows exponentially during the resolution process, leading to inefficiencies in finding contradictions or valid derivations. This challenge is often addressed by choosing particular orderings that first explore resolution steps that are more likely to lead to valid proof (Baumgartner, 1992; Sanner & McIlraith, 2006).

In addition to proposing an ordering strategy which is explained in 3.1, we introduce the notion of typing into theory resolution which considerably prunes the space of possible resolutions. Many of the created resolvents during the resolution process, each opening a new search branch, are created by unifying variables and constants with inconsistent ontological types. For instance, a literal *"small"(x)* in a clause about *vehicles* can be resolved with a literal *"small"(y)* which is about *animals*. However, the search branch created by this resolvent is very unlikely to result in a valid proof as it is created by a semantically invalid unification, and in future steps, the proof will be stopped as no further resolutions could be made. By associating variables and constants with their ontological types and considering type consistency in unification, we can significantly prune the space of allowable resolutions, by preventing the exploration of type inconsistent branches from the beginning.

Variable types can be introduced into an FOL sentence as unary predicates, e.g., *"animal"*$(x)$. Considering a universally quantified sentence in FOL as

$$\forall x \forall y \;\; H(x) \wedge T(y) \wedge A(x,y) \implies R(x,y), \tag{5}$$

where $H(.)$ and $T(.)$ are unary predicates indicating types of their corresponding variables, we equivalently write the above sentence in the *typed* FOL as

$$\forall x \forall y \;\; A(x,y) \implies R(x,y) | x:H, y:T, \tag{6}$$

where, $x : H$ and $y : T$ indicate that $x$ and $y$ are of type $H$ and $T$ respectively. This clause can be equivalently written in the clausal form as

$$\forall x \forall y \;\; \neg A(x,y) \vee R(x,y) | x:H, y:T, \tag{7}$$

Next, consider another clause in typed FOL as

$$\forall w \forall z \;\; \neg R(w,z) \vee S(w,z) | w:M, z:N. \tag{8}$$

By introducing the unifier $\theta = \{w/x, z/y\}$, we can perform *typed resolution* between the two clauses 7 and 8 as

$$\frac{\neg A(x,y) \vee R(x,y) | x:H, y:T \quad \neg R(w,z) \vee S(w,z) | w:M, z:N}{\neg A(x,y) \vee S(x,y) | x:C, y:D} \,, \tag{9}$$

where $C$ and $D$, the types of resolvent variables are $C \equiv H \sqcap M$ and $D \equiv T \sqcap N$ using the following lemma which is proved in Appendix A. Here, $\sqcap$ indicates unary type intersection (conjunction).

**Lemma 1.** *Resolving two disjunctive clauses $c_1$ and $c_2$ that include complimentary literals $l(x_1,...,x_n)|x_1 : T_1,...,x_n : T_n$ and $\neg l(y_1,...,y_n)|y_1 : H_1,...,y_n : H_n$ under the unifier $\theta = \{x_1/y_1,...,x_2/y_2\}$ creates a resolvent $c_{res}$ with type variables $x_1 : T_1 \sqcap H_1,...,x_n : T_n \sqcap T_n$.*

Typed resolution can be directly extended to typed theory resolution in our natural language-based logical system by resolving literals $B(x_1,...,x_n)|x_1 : T_1,...,x_n : T_n$ and $D(y_1,...,y_n)|y_1 : H_1,...,H_n$ if the LLM identifies the natural language predicate $B$ to entail $D$, i.e., $B(x_1,...,x_n) \vdash_{LLM} D(y_1,...,y_n)$ if their unified variables have consistent types. In the next section, we elaborate on how type consistency is checked in our LLM-TH framework.

**Algorithm 1** LLM-TH Algorithm

**Input:** $\mathcal{K}$, $q$, *max_proofs*, *max_iters*, $\mathcal{I}$, $\mathcal{F}$
*proofs* $\leftarrow \emptyset$
$PQ \leftarrow \emptyset$ // $PQ$ is an initially empty priority queue.
$PQ.push(\neg q, (1, 1, 0))$ // Negation of $q$ has priority $(1, 1, 0)$, $PQ$ is ordered by Equation 15
$i \leftarrow 0$
**while** $PQ \neq \emptyset \wedge |proofs| \leq max_proofs \wedge i \leq max_iters$ **do**
  $c \leftarrow PQ.pop()$
  **if** $c = \perp$ **then**
    *proofs* $\leftarrow$ *proofs* $\cup \{(\mathcal{A}_c, (\rho^t(c), \rho^e(c), \rho^l(c))\}$ //$\mathcal{A}_c$ is the set of ancestors of $c$, found by backtracking up to $\neg q$
  **else**
    $\beta_c \leftarrow$ candidate clauses in $\mathcal{K}$ with similar arity and different polarity to $c$
    **for** $c_{\text{target}} \in \beta_c$ **do**
      Perform hyperresolution to compute resolvent $c_{\text{res}}$ of $c$ and $c_{\text{target}}$ using Equation 2
      $PQ.push(c_{\text{res}}, (\rho^t(c_{\text{res}}), \rho^e(c_{\text{res}}), \rho^l(c_{\text{res}}))$ // *cf. Equations 11, 13 and 14*
  $i$++
**Output:** *proofs*

### 3.1 LLM-TH Algorithm

This section presents LLM-TH, our proposed algorithm for efficient logical commonsense reasoning, which is based on theory resolution (Stickel, 1985; Baumgartner, 1992) extended to LLM theory resolution with predicates of arbitrary arity, hyperresolution, and simultaneous type inference. Its workflow is shown in a worked example in Figures 1 and 2, and formalized in Algorithm 1.

**Problem Definition** Let $\mathcal{Q}$ denote a set of queries and $\mathcal{K}$ represent a knowledge base (KB) comprising a set of axioms and facts, a set of intended repair axioms $\mathcal{I}$, and a set of forbidden repair axioms $\mathcal{F}$, all expressed in natural language logic and clausal form with arbitrary predicate arities. For each query $q \in \mathcal{Q}$, LLM-TH identifies a set of proofs, denoted as *proofs*. Each proof $f \in$ *proofs* consists of a subset of clauses in $\mathcal{K}$, and is assigned a priority score $\rho$.

**Algorithm** To prove $\mathcal{K}$ entails the query q via resolution, we need to show that repeatedly using the resolution rule on $\mathcal{K} \wedge \neg q$ leads to a contradiction, and is thus unsatisfiable. Following the backward chaining paradigm that offers efficiency benefits (Poole & Mackworth, 2010), we pick $\neg q$ as the first active clause to be resolved with a clause from $\mathcal{K}$. At each step, given an active clause $c = \bigvee_{i=1}^{|c|} l_i$ where each $l_i$ is a literal of arbitrary arity, any clause $c_{target} \in \mathcal{K}$ as $c_{target} = \bigvee_{i=1}^{|c_{target}|} l_{target_i}$ is considered a *candidate clause* to be theory resolved with $c$, yielding the resolvent $c_{res}$, if at least one $(l_i, l_{target_i})$ pair can be formed where $l_i$ and $l_{target_i}$ have equal arities and different polarities. One can lift this polarity restriction for theory resolutions to allow implicit negation in the language of the predicate (e.g.,"*not spicy*") in contrast to explicit logical negation (e.g., $\neg$"*spicy*"). As $\mathcal{K}$ is often large and expands further with new resolvents being derived as resolution advances, efficiency is a key desideratum, which LLM-TH achieves by prioritizing candidate clauses based on two criteria: (i) type consistency and (ii) predicate entailment between the active and candidate clauses.

*Restricting the Space of Resolutions with Typing:* The first mechanism used by LLM-TH to improve efficiency is the use of typed theory resolution to restrict the space of allowable resolutions. In typed theory resolution, the types of variables in the resolvent clause are determined by the conjunction of the types of variables in their parent clause, hence LLM-TH prioritizes clauses with variable types that align with the active clause to encourage valid types. For example, if two candidate clauses are considered to be resolved with an active clause of variable type *"Animal"*, LLM-TH prioritizes a candidate clause with a unifying variable type *"Mammal"* over one with type *"Vehicle"*.

LLM-TH leverages the commonsense knowledge of the LLM to obtain the probability of entailment between the variable types. Formally, for the pair of literals with equal arities $(l, l_{target})$, denoting the set of argument types of $l$ and $l_{target}$ as $T = \{t_i\}$ and $T' = \{t'_i\}$ respectively, the plausibility of unifying each of their variables can be obtained by calculating the entailment probabilities between

each $t_i$ and $t'_i$. Since entailment is an asymmetric relation and its direction is unknown, we need to calculate both $t_i \vdash_{LLM} t'_i$ and $t'_i \vdash_{LLM} t_i$ to obtain the type consistency score. The average of type entailment scores for arguments determines $\rho^{type}(c_{res})$, the overall type priority score for $c_{res}$.

$$\rho^{type}(c_{res}) = \frac{1}{|T_i|}\Sigma_i \left(\max\{p(t_i \vdash_{LLM} t'_i), p(t'_i \vdash_{LLM} t_i)\}\right). \tag{10}$$

Since the main objective is to find the most plausible proofs, i.e., the sequences of most plausible theory resolution steps, we define the first entry of our priority score for each $c_{res}$ as the overall type consistency score of all resolution steps beginning from $\neg q$ that led to its derivation. Let $\mathcal{P}_{c_{res}}$ be the set of parent clauses of $c_{res}$; the proof type consistency score of $c_{res}$ is inductively defined as

$$\rho^t(c_{res}) = \left(\prod_{c' \in \mathcal{P}_{c_{res}}} \rho^t(c')\right) \cdot \rho^{type}_{c_{res}}. \tag{11}$$

*Resolution Ordering based on Predicate Entailment:* Assigning the type consistency scores prunes the resolution search space ensuring that only clauses with compatible argument types will be considered for resolution. To further enhance efficiency, LLM-TH prioritizes the remaining clauses based on their potential for being part of a plausible proof considering their predicate entailment. As explained, in our LLM-based theory resolution framework, LLM entailment is used to identify unsatisfiability of clauses. Therefore, the greater probability the LLM assigns to $l_{target}$ entailing $l$, the more plausible it becomes to resolve $l$ and $l_{target}$. Therefore we define the plausibility of a theory resolution step between $c$ and $c_{target}$ by resolving literals $l$ and $l_{target}$ generating $c_{res}$, denoted by $\rho^{entail}_{c_{res}}$

$$\rho^{entail}_{c_{res}} = \begin{cases} 1 & \text{if } (l_{target} \vdash l) \in \mathcal{I}, \\ 0 & \text{if } \neg(l_{target} \vdash l) \in \mathcal{F}, \\ p(l_{target} \vdash_{LLM} l) & \text{otherwise .} \end{cases} \tag{12}$$

These plausibility scores can help us prioritize the resolvent clauses based on their predicate entailment. For example, in the scenario depicted in Figure 1, resolving *"Sicilian"* with *"Mediterranean"* results in a higher entailment score compared to resolving *"Mustard-flavored"* with *"Mediterranean"*. Hence, it is prudent to prioritize the former resolvent, as it is more likely to contribute to the final proof. As the definition of the overall proof type consistency score, we can compute the overall predicate entailment score inductively to obtain the second entry of our priority score as

$$\rho^e(c_{res}) = \left(\prod_{c' \in \mathcal{P}_{c_{res}}} \rho^e(c')\right) \cdot \rho^{entail}_{c_{res}}. \tag{13}$$

Ultimately, among equally plausible proofs, we prioritize shorter ones that circumvent unnecessary steps. We define the proof length score, a third priority score that accounts for this preference. The proof length score of $c_{res}$ is derived inductively from the maximum length of its parent clauses as

$$\rho^l(c_{res}) = 1 + \max_{c' \in \mathcal{P}_{c_{res}}} \rho^l(c'). \tag{14}$$

Each resolvent $c_{res}$ is assigned the priority tuple $(\rho^t(c_{res}), \rho^e(c_{res}), \rho^l(c_{res}))$ and then pushed to the priority queue $PQ$, in which the order of clauses is specified as

$$\begin{aligned} c_1 \preceq c_2 \iff & [\rho^t(c_1) > \rho^t(c_2)] \vee [(\rho^t(c_1) = \rho^t(c_2)) \wedge (\rho^e(c_1) > \rho^e(c_2))] \\ & \vee [(\rho^t(c_1) = \rho^t(c_2)) \wedge (\rho^e(c_1) = \rho^e(c_2)) \wedge (\rho^l(c_1) < \rho^l(c_2)]. \end{aligned} \tag{15}$$

By applying this prioritization scheme, the type consistency priority score first applies a hard filter to avoid exploration of resolvents with invalid types, and the predicate entailment and length priorities together enable an efficient inference via LLM-based theory resolution. These efficiency enhancements enable LLM-TH to be applied to large-scale KBs. Furthermore, it is able to reason over incomplete KBs by benefiting from the commonsense knowledge of the LLM to fill in the missing axioms by identifying entailing predicates in the theory resolution process.

At the start of each iteration of LLM-TH, the clause with the highest rank in $PQ$ becomes the active clause. When a resolution step yields a contradiction, the proof and its respective priority score are added to the set of found proofs by backtracking the ancestor clauses. The algorithm

terminates when either a certain number of proofs are found or the maximum number of iterations is exceeded. Notably, LLM-TH is not limited to proving a single query; instead, it finds a set of proofs and assigns a strength score to each. This feature enables it to evaluate the likelihood of each query being entailed, which is critical for tasks requiring ranking, such as answering multiple-choice questions. Furthermore, LLM-TH can reason on incomplete KBs by using the LLM's commonsense knowledge to fill in the missing axioms by identifying entailed predicates via theory resolution.

## 4 Fixing Erroneous Resolutions in LLM-TH

LLM-TH enables the verification of the reasoning process by providing access to atomic resolution steps. Therefore, if an incorrect theory resolution step is performed due to an erroneous entailment probability assigned by the LLM, the source of failure can be easily identified. Here, we show that by adding a repair rule, LLM-TH can recover from its mistake and fix the reasoning. In the example provided in Figure 1, the LLM's mistake in assigning a low entailment score for *"Souvlaki"* to entail *"Meditteranean"* leads to a missed inference. Also, incorrectly considering *"cuttlefish"* to entail being a *"fish"* leads to incorrectly resolving these literals. However, introducing the correct axiom $\forall y \text{\textit{"Souvlaki"}}(y) \implies \text{\textit{"Meditteranean"}}(y)$ to $\mathcal{I}$ and introducing $\neg(\text{\textit{"cuttlefish"}} \vdash_{LLM} \text{\textit{"fish"}})$ to $\mathcal{F}$ fix these mistakes. We formalize this property here and prove it in Appendix B.

**Proposition 1.** *Consider proof $P_c^{\phi}$ using axiom $\phi$ that derives clause $c$. For any incorrect LLM reasoning axiom $\phi$, a Repair Axiom $\phi'$ can be inserted such that $P_c^{\phi'}$ will be produced before $P_c^{\phi}$.*

## 5 Experiments

We empirically evaluate LLM-TH[2] on three different tasks representing commonsense reasoning with KBs on different datasets and compare it against variations of four different baselines to compare them from different aspects by answering the following questions:

- **RQ1-Scalability:** How does the reasoning performance of LLM-TH compare to baselines when reasoning with large, but complete knowledge bases?
- **RQ2-Reasoning with incomplete KBs:** How effectively do LLM-TH and the baselines use the LLM's commonsense knowledge to compensate for the incompleteness of the KB?
- **RQ3-Efficiency:** How is the efficiency of LLM-TH influenced by typed hyperresolution?

### 5.1 Tasks and Datasets Description

- **Preference reasoning:** An exemplar commonsense reasoning task is providing recommendations using natural language statements of user preferences and restrictions. For this task, we use Recipe-MPR (Zhang et al., 2023a), a dataset consisting of 500 queries, e.g., *"I want French food, but I'm on a budget"* and five-way recipe options. This dataset covers a broad range of commonsense reasoning skills such as temporal and analogical reasoning.
- **Multi-domain Deductive reasoning:** Since established datasets for logical commonsense reasoning with LLMs, e.g., ProntoQA (Saparov & He, 2022) and COPA-SSE (Brassard et al., 2022), have small KBs with less than 20 facts and axioms per query, we find them insufficient for evaluating the reasoning capability on large KBs. Thus, following the approach in Saparov & He (2022), we create a deductive reasoning dataset using manually written commonsense axioms and ground facts sampled from Wikidata knowledge graph (Vrandečić & Krötzsch, 2014). This dataset contains more than 32k rules about five different domains: Biological entities, foods, vehicles, drugs and diseases, and sports, and 1000 queries that answering them requires 2 to 7 reasoning steps. We release this dataset to encourage research on LLM-based commonsense reasoning on large-scale KBs.
- **Geographical QA:** Using the same approach for generating the multi-domain deductive reasoning dataset, we create a KB about geographical entities, e.g., cities, deserts, museums, etc. containing more than 12k rules and 500 queries which we also release.

[2]https://anonymous.4open.science/r/typed-logic-release-476D/

Table 1: Reasoning performance of methods across the three datasets on complete KBs. The preference reasoning dataset lacks an explicit rule base, making RAG-based baselines irrelevant.

| Method | Preference Reasoning | | | Deductive Reasoning | | | Geographical QA | | |
|---|---|---|---|---|---|---|---|---|---|
| | Accuracy | RS Macro | RS Micro | Accuracy | RS Macro | RS Micro | Accuracy | RS Macro | RS Micro |
| **GPT-3.5 Turbo** | | | | | | | | | |
| Zero-Shot CoT | 0.86±0.04 | 0.60 | 0.80 | 0.69±0.02 | 0.45 | 0.48 | 0.71±0.03 | 0.60 | 0.77 |
| Few-Shot CoT | 0.87±0.02 | 0.65 | 0.81 | 0.65±0.03 | 0.45 | 0.48 | 0.82±0.02 | 0.50 | 0.53 |
| RAG + Zero-Shot CoT | NA | NA | NA | 0.68±0.02 | 0.85 | 0.92 | 0.74±0.01 | 0.80 | 0.88 |
| RAG + Few-Shot CoT | NA | NA | NA | 0.69±0.02 | 0.65 | 0.84 | 0.83±0.02 | 0.75 | 0.84 |
| **Gemini-1.5-Flash** | | | | | | | | | |
| Zero-Shot CoT | 0.84±0.04 | 0.60 | 0.75 | 0.60±0.01 | 0.40 | 0.81 | 0.78±0.03 | 0.20 | 0.51 |
| Few-Shot CoT | 0.86±0.02 | 0.55 | 0.79 | 0.66±0.05 | 0.20 | 0.71 | 0.79±0.02 | 0.25 | 0.54 |
| RAG + Zero-Shot CoT | NA | NA | NA | 0.78±0.02 | 0.85 | 0.93 | 0.79±0.03 | 0.50 | 0.72 |
| RAG + Few-Shot CoT | NA | NA | NA | 0.86±0.04 | 0.45 | 0.72 | 0.78±0.03 | 0.25 | 0.47 |
| **Llama3 70B** | | | | | | | | | |
| Zero-Shot CoT | 0.87±0.01 | 0.55 | 0.80 | 0.80±0.03 | 0.15 | 0.77 | 0.78±0.01 | 0.25 | 0.57 |
| Few-Shot CoT | **0.91±0.01** | 0.70 | 0.84 | 0.78± 0.02 | 0.55 | 0.58 | 0.87±0.02 | 0.45 | 0.45 |
| RAG + Zero-Shot CoT | NA | NA | NA | 0.78±0.01 | 0.50 | 0.81 | 0.87±0.030 | 0.40 | 0.65 |
| RAG + Few-Shot CoT | NA | NA | NA | 0.80±0.02 | 0.75 | 0.80 | 0.91±0.02 | 0.65 | 0.71 |
| **Mixtral 46.7B** | | | | | | | | | |
| Zero-Shot CoT | 0.79±0.03 | 0.60 | 0.84 | 0.59±0.02 | 0.30 | 0.66 | 0.71±0.02 | 0.50 | 0.70 |
| Few-Shot CoT | 0.74±0.02 | 0.65 | 0.83 | 0.67±0.01 | 0.45 | 0.53 | 0.80±0.01 | 0.45 | 0.49 |
| RAG + Zero-Shot CoT | NA | NA | NA | 0.65±0.02 | 0.65 | 0.81 | 0.66±0.07 | 0.25 | 0.51 |
| RAG + Few-Shot CoT | NA | NA | NA | 0.46±0.03 | 0.30 | 0.43 | 0.70±0.06 | 0.50 | 0.65 |
| **VERA** (T5 5B) | 0.86 | NA | NA | 0.76 | NA | NA | 0.68 | NA | NA |
| **LLM-TH** (BART 406M) | 0.84 | **0.90** | **0.94** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** | **1.00** |

### 5.2 Baselines and Evaluation

Existing formal reasoning methods with LLMs, i.e., semantic parsing methods and methods emulating inference rules, suffer from two limitations that make them inapplicable to our studied datasets: (i) they cannot leverage the internal commonsense knowledge of the LLM and only rely on an explicit and complete rule base, which Recipe-MPR lacks. (ii) They are limited to small KBs that fit in the LLM context size, but our studied KBs are much larger. We use established methodologies for eliciting more faithful reasoning from the LLMs as our comparison baselines: (a) zero-shot CoT (Kojima et al., 2022), (b) few-shot CoT (Wei et al., 2022), and (c) RAG (Lewis et al., 2020b) using a dense retriever (Song et al., 2020) to find relevant rules from the KB and prompting the LLM with zero-shot CoT and (d) few-shot CoT. We also compare against (e) VERA (Liu et al., 2023), a method for estimating plausibility of commonsense statements. VERA estimates a score in the range of $(0, 1)$, and we consider a score greater (smaller) than $0.5$ as a True (False) prediction. Except VERA which uses T5 (Raffel et al., 2020), we use a number of common LLMs for other baselines: (1) Gemini 1.5-flash, (2) Llama3 (70B), (3) Mixtral (56.7B), and (4) GPT3.5 Turbo, while using BART large (Lewis et al., 2020a)[3] (406 M) and RoBERTa large (Liu et al., 2019)[4] (365 M) tuned on MNLI (Williams et al., 2018) dataset to obtain entailment probabilities for LLM-TH. We also use pyDatalog[5] for performing hyperresolution for grounding on the KB facts and use Gemini 1.5-flash to convert natural language axioms to clausal natural language logic form.

We evaluate the reasoning performance based on both, (1) the correctness of the final answer, measured by *accuracy*, and (2) the correctness of the reasoning process by evaluating proofs using the *reasoning score* (RS) (Kazemi et al., 2023) metric which is manually calculated for 20 randomly chosen responses in which the final answer was correct. RS is typically assessed as a binary decision based on whether the predicted proof is entirely aligned with the ground truth proof (Kazemi et al., 2023; Lee & Hwang, 2024), which leads to both a single erroneous step and wholly flawed reasoning being penalized equally. To achieve a more granular evaluation of the proofs, we calculate both the conventional *macro RS* and following the idea of Min et al. (2023), we propose a metric which we call *micro RS*. Let $P$ be a provided proof and $P^*$ a valid ground truth proof. Using the indicator function $\mathbb{I}$, we define the micro RS for each query as $RS_{Micro} = \frac{1}{|P|}\sum_{p \in P} \mathbb{I}(p \in P^*)$.

[3]https://huggingface.co/facebook/bart-large-mnli

[4]https://huggingface.co/FacebookAI/roberta-large-mnli

[5]https://pypi.org/project/pyDatalog/

Table 2: Reasoning performance on incomplete KBs. Numbers in parenthesis indicate the difference with the method's performance on a complete KB (Table 1) with ↓ (↑) showing a decrease (increase).

| Method | Deductive Reasoning | | Geographical QA | |
|---|---|---|---|---|
| | Zero-Shot CoT | Few-Shot CoT | Zero-Shot CoT | Few-Shot CoT |
| GPT-3.5-Turbo | 0.65±0.01 (0.03↓) | 0.54±0.04 (0.15↓) | 0.60±0.02 (0.14↓) | 0.64±0.03 (0.19↓) |
| Gemini-1.5-Flash | 0.73±0.02 (0.05↓) | 0.73±0.01 (0.13↓) | 0.68±0.02 (0.11↓) | 0.66±0.02 (0.11↓) |
| Llama3 70B | 0.77±0.02 (0.01↓) | 0.77±0.01 (0.03↓) | 0.66±0.02 (0.21↓) | 0.66±0.02 (0.25↓) |
| Mixtral 46.7B | 0.50±0.04 (0.14↓) | 0.48±0.02 (0.03↑) | 0.53±0.03 (0.12↓) | 0.58±0.03 (0.13↓) |
| LLM-TH (using BART 406M) | **0.97** | - | **0.95** | - |
| LLM-TH (using RoBERTa 356M) | **0.96** | - | **0.92** | - |

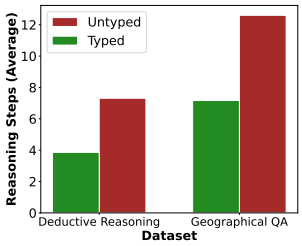


Figure 3: Influence of typing on the efficiency of the inference algorithm.

**5.2.1 RQ1: Reasoning Performance on Complete KB** Results of the reasoning performance of different methods on complete KBs are provided in Table 1. On the preference reasoning task, which has a small KB, although zero-shot and few-shot CoT with a large LLM such as Llama 3 (70B) yield superior accuracy, LLM-TH outperforms zero-shot and few-shot CoT with Mixtral and is competitive with zero-shot CoT using GPT3.5 and Gemini despite using a much smaller LLM. For the reasoning scores, LLM-TH exhibits a more correct and faithful reasoning process than all other methods. On this task, since the dataset does not contain an explicit KB, RAG-based baselines reduce to zero-shot and few-shot CoT. On larger KBs of deductive reasoning and geographical QA tasks, the limitations of existing LLM-based methods are revealed as none of them compare to the accuracy of LLM-TH. Furthermore, they all obtain imperfect reasoning scores, reflecting their hallucinations and reasoning errors. On these datasets, LLM-TH performs standard resolution which is a sound and complete inference rule, as reflected in the results. While complete KBs are impractical in real-world use cases, results of this experiment highlight that existing baselines, as opposed to LLM-TH, exhibit reasoning failures on large scale KBs *even* when they are complete.

**5.2.2 RQ2: Reasoning Performance on Incomplete KBs** To enable reasoning over practical KBs, leveraging the commonsense reasoning ability of the LLM to compensate for KB incompleteness is essential. To assess this capability, in our experiments on deductive reasoning and geographical QA datasets that have explicit KBs, we simulate KB incompleteness by omitting one of the rules used in the proof of each query, to test whether the LLM can use its commonsense knowledge to deduce, e.g., *"paying taxes"* implies *"earning revenue"*. Since few-shot and zero-shot CoT methods rely solely on the internal LLM knowledge, they are irrelevant to this RQ that examines the role of KB incompleteness. Hence, we compare LLM-TH against variations of RAG with zero-shot and few-shot CoT prompting. Results in Table 2 show that using the theory hyperresolution framework, LLM-TH is able to achieve significantly higher accuracies compared to the RAG-based baselines that clearly struggle with incompleteness compared to RAG results for complete KBs in Table 1.

**5.2.3 RQ3: Influence of Typing on Efficiency** To verify the efficiency enhancement offered by introducing type information to the hyperresolution framework of LLM-TH, we perform an ablation experiment on the complete KB, by comparing the average number of reasoning steps that LLM-TH takes to find the answers with its untyped variant that does not consider types in prioritization. In summary, the results of this experiment shown in Figure 3 indicate that typed hyperresolution reduces the proof length by half, which translates to a significantly reduced search space.

## 6 Conclusion

We proposed LLM-TH for logical commonsense reasoning with large and incomplete KBs. Using theory resolution, LLM-TH integrates LLM commonsense knowledge into the resolution inference rule to enable reasoning over incomplete KBs with arbitrary predicates. LLM-TH shows strong performance: it matches or outperforms baselines that use orders of magnitude larger LLMs; its use of an LLM-based typed hyperresolution approach yields high efficiency gains; and its transparency and repairability establish it as a solution for factual and correct reasoning on large-scale KBs. In summary, LLM-TH holds promise to significantly reduce hallucinations in LLM-based reasoning.

**Ethics Statement:** By introducing LLM-TH, we tried to enhance the transparency and increasing control over the reasoning process of LLM-based logical commonsense reasoning. However, drawing logically valid conclusions does not necessarily mean that all axioms, rules, and the internal commonsense knowledge of the LLM which are leveraged in the reasoning process follow ethical requirements. A responsible and credible usage of LLM-TH, like any other reasoning framework, requires careful considerations and assessments of the knowledge base, the underlying LLM, and the user-defined axioms to ensure desired unbiased and ethical performance.

**Reproducibility Statement:** We release all our code and data in the supplementary materials, also accessible on the LLM-TH anonymous repository[6]. We also explain the experimental setup and dataset descriptions in Section 5, and include all prompts utilized for the LLM usage in Appendix E, as well as in the supplementary materials.

## REFERENCES

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models. *Advances in Neural Information Processing Systems*, 35:38546–38556, 2022.

Marco Baroni, Armand Joulin, Allan Jabri, German Kruszewski, Angeliki Lazaridou, Klemen Simonic, and Tomas Mikolov. Commai: Evaluating the first steps towards a useful general ai. *arXiv preprint arXiv:1701.08954*, 2017.

Peter Baumgartner. An ordered theory resolution calculus. In *Logic Programming and Automated Reasoning: International Conference LPAR'92 St. Petersburg, Russia, July 15–20, 1992 Proceedings 3*, pp. 119–130. Springer, 1992.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Ana Brassard, Benjamin Heinzerling, Pride Kavumba, and Kentaro Inui. Copa-sse: Semi-structured explanations for commonsense reasoning. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pp. 3994–4000, 2022.

Chin-Liang Chang and Richard Char-Tung Lee. *Symbolic logic and mechanical theorem proving*. Academic press, 2014.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45, 2024.

Antonia Creswell, Murray Shanahan, and Irina Higgins. Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint arXiv:2205.09712*, 2022.

Ernest Davis and Gary Marcus. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103, 2015.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36, 2024.

Tianqing Fang, Zeming Chen, Yangqiu Song, and Antoine Bosselut. Complex reasoning over logical queries on commonsense knowledge graphs. *arXiv preprint arXiv:2403.07398*, 2024.

Weinan He, Canming Huang, Yongmei Liu, and Xiaodan Zhu. Winologic: A zero-shot logic-based diagnostic dataset for winograd schema challenge. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 3779–3789, 2021.

[6]https://anonymous.4open.science/r/typed-logic-release-476D/

Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*, 2023.

Mete Ismayilzada, Debjit Paul, Syrielle Montariol, Mor Geva, and Antoine Bosselut. Crow: Benchmarking commonsense reasoning in real-world tasks. *arXiv preprint arXiv:2310.15239*, 2023.

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

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

Stefanie Krause and Frieder Stolzenburg. Commonsense reasoning and explainable artificial intelligence using large language models. In *European Conference on Artificial Intelligence*, pp. 302–319. Springer, 2023.

Jinu Lee and Wonseok Hwang. Symba: Symbolic backward chaining for multi-step natural language reasoning. *arXiv preprint arXiv:2402.12806*, 2024.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7871–7880, 2020a.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33: 9459–9474, 2020b.

Hugo Liu and Push Singh. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.

Jiacheng Liu, Wenya Wang, Dianzhuo Wang, Noah A Smith, Yejin Choi, and Hannaneh Hajishirzi. Vera: A general-purpose plausibility estimation model for commonsense statements. *arXiv preprint arXiv:2305.03695*, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12076–12100, 2023.

Robert C Moore. *The role of logic in knowledge representation and commonsense reasoning*. SRI International. Artificial Intelligence Center, 1982.

Theo Olausson, Alex Gu, Ben Lipkin, Cedegao Zhang, Armando Solar-Lezama, Joshua Tenenbaum, and Roger Levy. Linc: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5153–5176, 2023.

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

Mete Ismayilzada Debjit Paul, Syrielle Montariol, Mor GevaO, and Antoine Bosselut. Crow: Benchmarking commonsense reasoning in real-world tasks.

Aske Plaat, Annie Wong, Suzan Verberne, Joost Broekens, Niki van Stein, and Thomas Back. Reasoning with large language models, a survey. *arXiv preprint arXiv:2407.11511*, 2024.

David L Poole and Alan K Mackworth. *Artificial Intelligence: foundations of computational agents*. Cambridge University Press, 2010.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

John Alan Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM (JACM)*, 12(1):23–41, 1965.

Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson, 2016.

Scott Sanner and Sheila A McIlraith. An ordered theory resolution calculus for hybrid reasoning in first-order extensions of description logic. In *KR*, pp. 100–111, 2006.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*, 2019.

Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations*, 2022.

Abulhair Saparov, Richard Yuanzhe Pang, Vishakh Padmakumar, Nitish Joshi, Mehran Kazemi, Najoung Kim, and He He. Testing the general deductive reasoning capacity of large language models using ood examples. *Advances in Neural Information Processing Systems*, 36, 2024.

Murray Shanahan. Talking about large language models. *Communications of the ACM*, 67(2):68–79, 2024.

Ke Shen and Mayank Kejriwal. An experimental study measuring the generalization of fine-tuned language representation models across commonsense reasoning benchmarks. *Expert Systems*, 40 (5):e13243, 2023.

Shikhar Singh, Nuan Wen, Yu Hou, Pegah Alipoormolabashi, Te-Lin Wu, Xuezhe Ma, and Nanyun Peng. Com2sense: A commonsense reasoning benchmark with complementary sentences. *arXiv preprint arXiv:2106.00969*, 2021.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet: Masked and permuted pre-training for language understanding. *Advances in neural information processing systems*, 33: 16857–16867, 2020.

Mark E Stickel. Automated deduction by theory resolution. *Journal of Automated Reasoning*, 1(4): 333–355, 1985.

SM Tonmoy, SM Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv preprint arXiv:2401.01313*, 2024.

Armin Toroghi, Willis Guo, Ali Pesaranghader, and Scott Sanner. Verifiable, debuggable, and repairable commonsense logical reasoning via LLM-based theory resolution. In *The 2024 Conference on Empirical Methods in Natural Language Processing*, 2024.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change. *Advances in Neural Information Processing Systems*, 36, 2024.

Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2022.

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

Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, 2018.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Haochen Zhang, Anton Korikov, Parsa Farinneya, Mohammad Mahdi Abdollah Pour, Manasa Bharadwaj, Ali Pesaranghader, Xi Yu Huang, Yi Xin Lok, Zhaoqi Wang, Nathan Jones, et al. Recipe-mpr: A test collection for evaluating multi-aspect preference-based natural language retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2744–2753, 2023a.

Yanfang Zhang, Yiliu Sun, Yibing Zhan, Dapeng Tao, Dacheng Tao, and Chen Gong. Large language models as an indirect reasoner: Contrapositive and contradiction for automated reasoning. *arXiv preprint arXiv:2402.03667*, 2024.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. Siren's song in the ai ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023b.

Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for large-scale task planning. *Advances in Neural Information Processing Systems*, 36, 2024.

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

## A Proof of Lemma 1

**Lemma 1.** *Resolving two disjunctive clauses $c_1$ and $c_2$ that include complimentary literals $l(x_1, ..., x_n)|x_1 : T_1, ..., x_n : T_n$ and $\neg l(y_1, ..., y_n)|y_1 : H_1, ..., y_n : H_n$ under the unifier $\theta = \{x_1/y_1, ..., x_2/y_2\}$ creates a resolvent $c_{res}$ with type variables $x_1 : T_1 \sqcap H_1, ..., x_n : T_n \sqcap T_n$.*

*Proof.* Assume $c_1$ to be $A(x_1, ..., x_n) \vee l(x_1, ..., x_n)|x_1 : T_1, ..., x_n : T_n$ and $c_2$ to be $\neg l(y_1, ..., y_n) \vee B(y_1, ..., y_n)|y_1 : H_1, ..., y_n : H_n$. Following 5 and 6, we can rewrite these clauses in implication form FOL by adding type predicates as

$$\forall x_1, ..., \forall x_n \bigwedge_{i=1}^{n} T_i(x_i) \wedge \neg A(x_1, ..., x_n) \implies l(x_1, ..., x_n), \tag{16}$$

$$\forall y_1, ..., \forall y_n \bigwedge_{i=1}^{n} H_i(y_i) \wedge l(y_1, ..., y_n) \implies B(y_1, ..., y_n), \tag{17}$$

Converting these clauses to the disjunctive form yields

$$\forall x_1, ..., \forall x_n \bigvee_{i=1}^{n} \neg T_i(x_i) \vee A(x_1, ..., x_n) \vee l(x_1, ..., x_n), \tag{18}$$

$$\forall y_1, ..., \forall y_n \bigvee_{i=1}^{n} \neg H_i(y_i) \vee \neg l(y_1, ..., y_n) \vee B(y_1, ..., y_n). \tag{19}$$

Now, we can resolve $l(x_1, ..., x_n)$ with $\neg l(y_1, ..., y_n)$ under the unifier $\theta = \{x_1/y_1, ..., x_2/y_2\}$ as

$$\frac{\bigvee_{i=1}^{n} \neg T_i(x_i) \vee A(x_1, ..., x_n) \vee l(x_1, ..., x_n) \quad \bigvee_{i=1}^{n} \neg H_i(x_i) \vee \neg l(y_1, ..., y_n) \vee B(y_1, ..., y_n)}{\bigvee_{i=1}^{n} \neg T_i(x_i) \bigvee_{i=1}^{n} \neg H_i(x_i) \vee A(x_1, ..., x_n) \vee B(x_1, ..., x_n)}, \tag{20}$$

which can be rewritten as

$$\forall x_1, ..., \forall x_n \bigwedge_{i=1}^{n} T_i(x_i) \wedge H_i(x_i) \implies A(x_1, ..., x_n) \vee B(x_1, ..., x_n). \tag{21}$$

Therefore, the unary type predicates for each $x_i$ becomes the conjunction of the types from their parent clauses, which in our typed FOL notation, can be equivalently written as

$$\forall x_1, ..., \forall x_n A(x_1, ..., x_n) \vee B(x_1, ..., x_n) | x_1 : T_1 \sqcap H_1, ..., x_n : T_n \sqcap H_n. \tag{22}$$

□

## B Proof of Repairability of LLM-TH

**Proposition 1.** *Consider proof $P_c^{\phi}$ using axiom $\phi$ that derives clause $c$. For any incorrect LLM reasoning axiom $\phi$, a Repair Axiom $\phi'$ can be inserted such that $P_c^{\phi'}$ will be produced before $P_c^{\phi}$.*

*Proof.* A proof $P_c^{\phi} = P_c \cup \{\phi\}$ can be viewed as the combined set of clauses $P_c$ and $\phi$ that derive clause $c$. An incorrect reasoning reasoning step $\phi$ can be either a missed inference due to the LLM mistakenly assigning a low priority to a resolution, or an incorrect resolution assigned a high priority due to an LLM misbelief.

We begin the proof for missed inference case, for which the repair axiom $\phi' \in \mathcal{I}$ is available. We can obtain the proof score $\rho^e(P_c^{\phi})$ of clause $c$ by inductively unrolling Equation 13 for $\rho^e(c)$ over all ancestor clauses $P_c^{\phi}$ that derive it. This yields a simple product form: $\rho^e(P_c^{\phi}) = \rho_{\phi}^{entail} \cdot \prod_{c' \in P_c} \rho_{c'}^{entail}$. Now, comparing two different derivations $P_c^{\phi}$ and $P_c^{\phi'}$ of $c$, we can easily show that $\rho^e(P_c^{\phi'}) > \rho^e(P_c^{\phi})$ since $\frac{\rho^e(P_c^{\phi'})}{\rho^e(P_c^{\phi}))} = \frac{\rho_{\phi'}^{entail} \cdot \prod_{c' \in P_c} \rho_{c'}^{entail}}{\rho_{\phi}^{entail} \cdot \prod_{c' \in P_c} \rho_{c'}^{entail}} = \frac{\rho_{\phi'}^{entail}}{\rho_{\phi}^{entail}} > 1$ given that the explicit Repair Axiom has $\rho_{\phi'}^{entail} = 1$ (following Equation 13) while the LLM entailment score $\rho_{\phi}^{entail} < 1$. Hence, the proof $P_c^{\phi'}$ containing the Repair Axiom $\phi'$ will always be given precedence over $P_c^{\phi}$ according to the total ordering of Equation 15 used to prioritize proofs in the LLM-TH Algorithm 1.

Correctness of the proposition for case $\phi$ is an incorrect resolution that $\phi' \in \mathcal{F}$ forbids it is obvious as in presence of $\phi'$, the priority score assigned to the resolvent clause $c$, i.e., $\rho_{\phi}^{entail}$ will be set to 0 (following Equation 13). As a result, the resolvent clause is excluded from any proof by being assigned the lowest priority. □

# C EXPLANATION OF THE LLM-TH ALGORITHM

In this section, we provide a detailed explanation of the LLM-TH algorithm, presented in Algorithm 1.

The algorithm takes as inputs the knowledge base ($\mathcal{K}$), the query ($q$), a computational budget defined by the maximum number of iterations (*max_iters*), the maximum number of found proofs shown by *max_proofs*, and two optional sets of repair axioms: (i) the set of intended resolutions denoted by $\mathcal{I}$, and (ii) the set of forbidden resolutions shown by $\mathcal{F}$.

As explained in Section 3.1, LLM-TH is not limited to finding a single proof to the query, but rather it can find a number of proofs with different plausibilities, a capability that is necessary for answering multiple-choice questions. For example, in the preference reasoning example provided in Figure 1, it finds a proof for each of the candidate recipes with a plausibility score assigned to each proof, which is used to rank the recipe options.

The algorithm initializes an empty set, *proofs*, to store discovered proofs, an iteration counter $i$, and an initially empty priority queue $PQ$. $PQ$ stores resolvent clauses and orders them according to their priority scores, using Equation 15. In the backward chaining approach used in LLM-TH, the first clause stored in $PQ$ is $\neg q$ with the priority of $(1, 1, 0)$. The algorithm's main loop executes as long as the priority queue ($PQ$) is not empty, the maximum proof limit is not reached, and the iteration cap is not exceeded. During each iteration, the clause $c$ with the highest priority in $PQ$ is dequeued and designated as the active clause. In the initial iteration, since $PQ$ contains only $\neg q$, this clause becomes the first active clause.

If the active clause is $\perp$, the previous resolution step has resulted in a contradiction. The proof can then be reconstructed by tracing the ancestors of $c$, starting from its parent set $\mathcal{P}_c$ and continuing up to the initial clause $\neg q$. Once a proof is found, it is added to the set *proofs*. However, if $c \neq \perp$, the algorithm proceeds by selecting clauses to resolve with $c$. It first identifies $\beta_c$, the set of candidate clauses in $\mathcal{K}$ that are resolvable with $c$. A clause must meet two conditions to be resolvable with $c$ in a theory resolution step following Equation 3: (i) it must share the same arity as $c$, and (ii) it must have a different polarity from $c$. After forming $\beta_c$, LLM-TH iterates through all clauses $c_{\text{target}} \in \beta_c$ and resolves each with $c$ via hyperresolution, as described in Equation 2, to obtain $c_{\text{res}}$. It then forms the priority tuple using Equations 11 , 13 , and 14 , and adds $c_{\text{res}}$ to $PQ$ based on this priority.

The algorithm continues until either the maximum number of proofs is found, $PQ$ becomes empty, or the maximum allowed number of iterations is reached.

# D COMMONSENSE REASONING

The general conception that humans have about the world and how it works, often referred to as commonsense knowledge, and the ability to reason about it, known as commonsense reasoning, are essential capabilities that average human beings possess. In order to be able to interact with the real world and humans, AI agents are also required to obtain this knowledge (Baroni et al., 2017; Shen & Kejriwal, 2023). Therefore, commonsense reasoning has been a central problem in AI over the past decades (Moore, 1982; Liu & Singh, 2004; Davis & Marcus, 2015). With the advent of language models, and particularly, transformer-based architectures, models were developed that exhibited a decent performance on some commonsense reasoning tasks such as physical and social reasoning (Bisk et al., 2020; Sap et al., 2019). Further improvements were obtained by scaled language models, i.e., LLMs, as they exhibited an outstanding capability of commonsense understanding and reasoning across various evaluation benchmarks (Zhao et al., 2024; Krause & Stolzenburg, 2023).

These improvements paved the way for inspecting new aspects of commonsense reasoning, such as reasoning on problems involving logical inference, a vein of work referred to as *complex commonsense reasoning* (Fang et al., 2024; He et al., 2021). The introduction of this line of work shed light on important limitations of LLMs in challenging commonsense reasoning tasks. Although LLMs have shown outstanding reasoning performance on simple commonsense reasoning benchmarks, they struggle with more challenging tasks, such as commonsense reasoning in real-world use cases (Paul et al.).

These remaining challenges and the risk of reasoning errors restrict the deployability of LLMs in real-world applications, especially in high-stakes scenarios. Despite the extensive amount of existing work on commonsense reasoning with LLMs, to the best of our knowledge, no work has targeted the problem of combining commonsense reasoning with reasoning over factual KBs. This problem is particularly important since LLMs show factual hallucinations when they lack domain-specific knowledge(Huang et al., 2023; Tonmoy et al., 2024). In this work, we aim to take complex logical commonsense reasoning task a step further, by proposing a methodology that leverages both commonsense reasoning knowledge of the LLM, as well as the factual knowledge of the KB.

## E PROMPTS USED FOR LANGUAGE MODELS

We provide the prompts that we used for the LLMs in the experiments of this paper. They are also included in our repository along with the implementation code and data.

**Prompt for Preference Reasoning Task (Zero-shot and Few-shot CoT)**

```
 Task: You will be given a query that asks for a recipe and five
 options that you have to choose from. Think step by step. First
 state your reason for your choice and then say: "Therefore, the
 selected recipe is <recipe id>.
Query: {{QUERY}}
[Examples if few-shot:]
```

**Prompt for Multi-domain Deductive Reasoning Task (Zero-shot and Few-shot CoT)**

```
Task: You will be given a query about some knowledge graph
entities in the form of a first order logic predicate that is
either True or False (for example, "CanHoldIn(Apple, Basket)"
which means one can hold an apple in a basket). Your task is to
identify whether the answer to this query is "True" or "False"
and also provide a proof of the answer. First, state your proof
mentioning the rules you used and then say: "Therefore, the
answer is True" or "Therefore, the answer is False". Think step
by step.
Query: {{QUERY}}
[Examples if few-shot:]
```

**Prompt for Multi-domain Deductive Reasoning (RAG with Zero-shot and Few-shot CoT)**

```
Task: You will be given a query about some knowledge graph
entities in the form of a first order logic predicate that is
either True or False (for example, "CanHoldIn(Apple, Basket)"
which means one can hold an apple in a basket) and a Knowledge
Base containing a set of rules that will help you identify the
answer. Your task is to identify whether the answer to the query
is "True" or "False" and also provide a proof of the query using
the knowledge base. First state your proof mentioning the rules
you used and then say: "Therefore, the answer is True" or
"Therefore, the answer is False". Think step by step.
Query: {{QUERY}}
KB: {{KB}}
[Examples if few-shot:]
```

**Prompt for Geographical QA Task (Zero-shot and Few-shot CoT)**

```
Task: You will be given a query about geographical entities in
the form of a first order logic predicate that is either True or
False. Your task is to identify whether the answer to the query
is "True" or "False" and also provide a proof of the query. First
state your proof mentioning the rules you used and then say:
"Therefore, the answer is True" or "Therefore, the answer is
False". Think step by step.
```

```
Query: {{QUERY}}
[Examples if few-shot:]
```

**Prompt for Geographical QA Task (RAG with Zero-shot and Few-shot CoT)**

```
Task: You will be given a query about geographical entities in
the form of a first order logic predicate that is either True or
False, and a Knowledge Base containing a set of rules that will
help you identify the answer. Your task is to identify whether
the answer to the query is "True" or "False" and also provide a
proof of the query using the knowledge base. First state your
proof mentioning the rules you used and then say: "Therefore, the
answer is True" or "Therefore, the answer is False". Think step
by step.
Query: {{QUERY}}
KB: {{KB}}
[Examples if few-shot:]
```

# F ANECDOTAL EXAMPLES

To offer more insight into the responses and proofs provided by LLM-TH and the comparison baselines, this section presents anecdotal examples illustrating each model's performance on the evaluated tasks. Since Llama3-70B shows the best overall performance for our comparison baselines in the experimental results provided in Section 5, we show the exemplar outputs from this LLM. Specifically, we showcase the outputs from the following models:

- LLM-TRes
- Zero-shot Chain of Thought prompting
- Few-shot Chain of Thought prompting
- RAG + Zero-shot Chain of Thought prompting
- RAG + Few-shot Chain of Thought prompting

We apply these models to the Deductive reasoning and Geographical QA tasks studied in RQ1 and RQ2 and provide anecdotal examples to elucidate their capabilities and limitations. In each anecdotal example, we present the query, the set of relevant facts and axioms from the KB, and the ground truth proof, followed by the response that each model provides. We also indicate correct proof steps with green color and highlight incorrect ones in red. It is noteworthy that since RQ2 requires access to the KB and studies the model's ability to compensate for KB incompleteness, the only relevant baselines are RAG-based models.

Examples of the experiments on the complete KB are provided in section F.1, and examples of performance on the incomplete KB are detailed in section F.2.

## F.1 EXPERIMENTS ON THE COMPLETE KB

### F.1.1 Deductive Reasoning

**Query**: Does Sulfadiazine disinfect Rhodospirillaceae?

**Relevant Rules from the KB**:

Facts:

(1) *"Subclass of"*(*"Rhodospirillaceae"*"*Rhodospirillaceae bacteria"*),

(2) *"Gram Negative"*(*"Rhodospirillaceae bacteria"*)

(3) *"Treats Gram Negative"*(*"Sulfadiazine"*)

Axioms:

(4)

$$\forall A, B, C, \text{“Subclass of”}(A, B) \wedge \text{“Gram Negative”}(B) \wedge \text{“Treats Gram Negative”}(C) \implies \text{“Unresistant To”}(A, C) | \text{A: “Bacteria”}, \text{B: “Taxon”}, \text{C: “Antibiotic”}$$

(5)

$$\forall A, B, \text{“Unresistant To”}(A, B) \implies \text{“Can be Used to Treat”}(A, B) | \text{A: “Bacteria”}, \text{“B: Antibiotic”}$$

(6)

$$\forall A, B, \text{“Can be Used to Treat”}(A, B) \implies \text{“Can be Used to Kill”}(A, B) | \text{A: “Bacteria”}, \text{“B: Antibiotic”}$$

(7) $\forall A, B$, *"Can be Used to Kill"*$(A, B) \implies$ *"Disinfects"*$(A, B)|$*A: "Bacteria", "B: Antibiotic"*

**Ground Truth Proof**:

1- Applying (1), (2), (3) to (4) with the following substitution:

$\theta = \{A/$*"Rhodospirillaceae"*$, B/$*"Rhodospirillaceae bacteria"*$, C/$*"Sulfadiazine"*$\}$ yields:

*"Unresistant To"("Rhodospirillaceae", "Sulfadiazine").*

2- Applying *"Unresistant To"("Rhodospirillaceae", "Sulfadiazine")* to (5)

with $\theta = \{A/$*"Rhodospirillaceae"*$, B/$*"Sulfadiazine"*$\}$ yields:

*"Can be Used to Treat"("Rhodospirillaceae", "Sulfadiazine")*

3-Applying *"Can be Used to Treat"("Rhodospirillaceae", "Sulfadiazine")* to (6)

with $\theta = \{A/$*"Rhodospirillaceae"*$, B/$*"Sulfadiazine"*$\}$ yields:

*"Can be Used to Kill"("Rhodospirillaceae", "Sulfadiazine")*

4- Applying *"Can be Used to Kill"("Rhodospirillaceae", "Sulfadiazine")* to (7)

with $\theta = \{A/$*"Rhodospirillaceae"*$, B/$*"Sulfadiazine"*$\}$ yields:

5- *"Disinfects"("Rhodospirillaceae", "Sulfadiazine")*.

Therefore, the answer to the query is True.

**LLM-TH:**

query: *"Disinfects"("Rhodospirillaceae", "Sulfadiazine")*:

Negated query: ¬*"Disinfects"("Rhodospirillaceae","Sulfadiazine")*

Note: In the following sequent calculus formulae, we use "*R*" to refer to "*Rhodospirillaceae*", "*S*" to represent "*Sulfadiazine*", "*GN*" to show "*Gram Negative*", "*TGN*", to denote "*Treats Gram Negative*", and "*SO*" to show "*Subclass Of*" due to space limitations.

Step 1-

$$\frac{\neg\text{"Disinfects"("R","S")} \quad \neg\text{"Can be Used to Kill"(A,B)} \vee \text{"Disinfects"(A,B)}}{\neg\text{"Can be Used to Kill"("R","S")}}.$$

Under unification $\theta = \{A/$*"Rhodospirillaceae"*$, B/$*"Sulfadiazine"*$\}$.

Step 2-

$$\frac{\neg\text{"Can be Used to Kill"("R","S")} \quad \text{"Can be Used to Kill"(A,B)} \vee \neg\text{"Can be Used to Treat"(A,B)}}{\neg\text{"Can be Used to Treat"("R","S")"}}.$$

Under unification $\theta = \{A/$*"Rhodospirillaceae"*$, B/$*"Sulfadiazine"*$\}$.

Step 3-

$$\frac{\neg\text{"Can be Used to Treat"("R","S")} \quad \text{"Can be Used to Treat"(A,B)} \vee \neg\text{"Unresistant To"(A,B)}}{\neg\text{"Unresistant To"("R","S")}}.$$

Step 4-

$$\frac{\neg\text{"Unresistant To"("R","S")} \quad \text{"Unresistant To"(A,B)} \vee \neg\text{"SO"(A,B)} \vee \neg\text{"GN"(B)} \vee \neg\text{"TGN"(C)}}{\neg\text{"SO"("R",B)} \vee \neg\text{"GN"(B)} \vee \neg\text{"TGN"("S")}}$$

Under unification $\theta = \{A/$*"Rhodospirillaceae"*$, C/$*"Sulfadiazine"*$\}$.

Step 5-

$$\frac{\neg\text{“SO”}(\text{“R”},B) \vee \neg\text{“GN”}(B) \vee \neg\text{“TGN”}(\text{“S”}) \quad \text{“SO”}(\text{“R”, “R bacteria”}) \wedge \text{“GN”}(\text{“R bacteria”}) \wedge \text{“TGN”}(\text{“S”})}{\perp}.$$

Proof score: $\rho_{\text{query}} = (1, 1, 5)$

* * *

**Zero-shot CoT:**

To prove the answer to the query, I will use the following rules from the knowledge base:
1. Rhodospirillaceae is a type of Bacteria. (IsA(Rhodospirillaceae, Bacteria))

2. Sulfadiazine is a type of Antibiotic. (IsA(sulfadiazine, Antibiotic))

3. Antibiotics disinfect Bacteria. (Disinfects(Antibiotic, Bacteria))
Using modus ponens on rules 1, 2, and 3, I can infer that: Disinfects(sulfadiazine, Rhodospirillaceae)
Therefore, the answer is True.

**Few-shot CoT:**

1.Sulfadiazine is an antibacterial drug.

2. Rhodospirillaceae is a family of bacteria.

3. Sulfadiazine is not typically used as a disinfectant, but rather as an antibiotic to treat bacterial infections.

Therefore, the answer is False.

**RAG + Zero-shot CoT:**

1. SubClassOf(Rhodospirillaceae, Rhodospirillaceae bacteria) (KB)

2. GramNegative(Rhodospirillaceae bacteria) (KB)

3. ∀ A:bacteria, B:Taxon, C:antibiotic, SubClassOf(A, B) && GramNegative(B) && TreatsGramNegative(C) $\implies$ UnResistantTo(A, C) (KB)

4. UnResistantTo(Rhodospirillaceae, sulfadiazine) (inferred from 1, 2, and 3, with A = Rhodospirillaceae, B = Rhodospirillaceae bacteria, and C = sulfadiazine)

5. ∀ A:bacteria B:antibiotic, UnResistantTo(A, B) $\implies$ CanBeUsedToTreat(A, B) (KB)

6. CanBeUsedToTreat(Rhodospirillaceae, sulfadiazine) (inferred from 4 and 5)

7. ∀ A:bacteria B:antibiotic, CanBeUsedToTreat(A, B) $\implies$ CanBeUsedToKill(A, B) (KB)

8. CanBeUsedToKill(Rhodospirillaceae, sulfadiazine) (inferred from 6 and 7)

9. ∀ A:bacteria B:antibiotic, CanBeUsedToKill(A, B) $\implies$ Disinfects(A, B) (KB)

10. Disinfects(Rhodospirillaceae, sulfadiazine) (inferred from 8 and 9)

Therefore, the answer is True.

**RAG + Few-shot CoT:**

1. Rhodospirillaceae is a gram negative bacteria,

2. sulfadiazine treats gram negative bacteria,

3. and sulfadiazine can be used to treat gram negative bacteria, therefore sulfadiazine can be used to treat Rhodospirillaceae.

Therefore, the answer is True.

**Discussion:**

These anecdotal examples demonstrate that on a complete knowledge base (KB), LLM-TH can successfully prove the query by resolving complementary clauses, without requiring the LLM to generate any additional axioms.
An analysis of the proofs generated by the baseline methods reveals limitations in LLM reasoning, particularly in zero-shot and few-shot settings. For instance, the zero-shot CoT baseline asserts that "Antibiotics disinfect bacteria"; however, this statement is overly broad, as not all antibiotics can disinfect all types of bacteria. While the final answer in this case is correct, there is a flaw in one of the reasoning steps. Also, the few-shot CoT baseline produces an incorrect final answer due to an error in reasoning. In contrast, the RAG-based baselines, utilizing both prompting styles, correctly prove the query by leveraging retrieved facts from the KB.

#### F.1.2 Geographical QA

**Query**: Can I drive between Biyabanak and Ryam Frolovskiy?

**Relevant Rules from the KB**:

Facts:

(1) *"Located in"*(*"Biyabanak"*, *"Iran"*),

(2) *"Located in"*(*"Ryam Frolovskiy"*, *"Russia"*),

(3) *"Share Land Borders"*(*"Iran"*, *"Azerbaijan"*)

(4) *"Share Land Borders"*(*"Azerbaijan"*, *"Russia"*)

Axioms:

(5)

$$\forall A, B, \textit{"Share Land Borders"}(A, B) \implies \textit{"Land Connected"}(A, B) | A : \textit{country}, B : \textit{country}$$

(6)

$$\forall A, B, C, \textit{"Land Connected"}(A, B) \land \textit{"Land Connected"}(B, C) \implies \textit{"Land Connected"}(A, C) | A : \textit{location}, B : \textit{location}, C : \textit{location}$$

(7)

$$\forall A, B, C, D, \textit{"Located in"}(A, C) \land \textit{"Located in"}(B, D) \land \textit{"Land Connected"}(C, D) \implies \textit{"Can Drive Between"}(A, B) | A : \textit{location}, B : \textit{location}, C : \textit{location}, D : \textit{location}$$

**Ground Truth Proof**:

1- Applying (3) to (5) with the following substitution:

$\theta = \{A/\textit{"Iran"}, B/\textit{"Azerbaijan"}\}$ yields:

(8) *"Land Connected"*(*"Iran"*, *"Azerbaijan"*).

2- Applying (3) to (5) with the following substitution:

$\theta = \{A/\textit{"Azerbaijan"}, B/\textit{"Russia"}\}$ yields:

(9) *"Land Connected"*(*"Azerbaijan"*, *"Russia"*).

3- Applying (8), (9) to (6) with the following substitution:

$\theta = \{A/\textit{"Iran"}, B/\textit{"Azerbaijan"}, C/\textit{"Russia"}\}$ yields:

(10) *"Land Connected"("Iran","Russia").*

4- Applying (1), (2), (10) to (7) with the following substitution:

$\theta = \{A/\text{“Biyabanak”}, B/\text{“Ryam Frolovskiy”}, C/\text{“Iran”}, D/\text{“Russia”}\}$ yields:

(11) *“Can Drive Between”(“Biyabanak”, “Ryam Frolovskiy”)*.

Therefore, the answer to the query is True.

---

**LLM-TH:**

Proof for query *“Can Drive Between”(“Biyabanak”, “Ryam Frolovskiy”)*

Negated query: ¬*“Can Drive Between”(“Biyabanak”, “Ryam Frolovskiy”)*

Note: In the following sequent calculus formulae, we use “*Bi*” to refer to “*Biyabanak*”, “*RF*” to represent “*Ryam Frolovskiy*”, “*Az*“ to refer to ‘*Azerbaijan*”, “*CDB*” to show “*Can Drive Between*”, “*LI*” to show “*Located In*”, “*LC*” to show “*Land Connected*”, and *“SLB”* to show “*“Share Land Borders”*” due to space limitations.

Step 1-

$$\frac{\neg\text{“CDB”(“Bi”,“RF”)} \quad \text{“CDB”(A,B)} \vee \neg\text{“LI”(A,C)} \vee \neg\text{“LI”(B,D)} \vee \neg\text{“LC”(C,D)}}{\neg\text{“LI(“Bi”,C)”} \vee \neg\text{“LI”(“RF”,D)} \vee \neg\text{“LC”(C,D)}} .$$

Under unification $\theta = \{A/\text{“Biyabanak”}, C/\text{“Ryam Frolovskiy”}\}$.

Step 2-

$$\frac{\neg\text{“LI”(“Bi”,C)} \vee \neg\text{“LI”(“RF”,D)} \vee \neg\text{“LC”(C,D)} \quad \text{“LI”(“Bi”,“Iran”)} \wedge \text{“LI”(“RF”,“Russia”)}}{\neg\text{“LC”(“Iran”,“Russia”)}} .$$

Under unification $\theta = \{C/\text{“Iran”}, D/\text{“Russia”}\}$.

Step 3-

$$\frac{\neg\text{“LC”(“Iran”,“Russia”)} \quad \text{“LC”(A,C)} \vee \neg\text{“LC”(A,B)} \vee \neg\text{“LC”(B,C)}}{\neg\text{“LC”(“Iran”,B)} \vee \neg\text{“LC”(B,“Russia”)}} .$$

Under unification $\theta = \{A/\text{“Iran”}, C/\text{“Russia”}\}$.

Step 4-

$$\frac{\neg\text{“LC”(“Iran”,B)} \quad \text{“LC”(“Iran”,B)} \vee \neg\text{“SLB”(“Iran”,B)}}{\neg\text{“SLB”(“Iran”,B)}} .$$

Step 5-

$$\frac{\neg\text{“LC”(B,“Russia”)} \quad \text{“LC”(B,“Russia”)} \vee \neg\text{“SLB”(B,“Russia”)}}{\neg\text{“SLB”(B,“Russia”)}} .$$

Step 6-

$$\frac{\neg\text{“SLB”(“Iran”, B)} \vee \neg\text{“SLB”(B, “Russia”)} \quad \text{“SLB”(“Iran”,“Az”)} \wedge \text{“SLB”(“Az”,“Russia”)}}{\perp} .$$

Under unification $\theta = \{B/\text{“Az”}\}$.

Proof score: $\rho_{\text{query}} = (1, 1, 6)$

* * *

---

**Zero-shot CoT:**

To answer this query, I will use the following rules from the knowledge base:
1. CanDriveBetween(x, y) $\implies$ Connected(x, y)

2. Connected(Biyabanak, Ryam Frolovskiy) $\implies$ False (this is a fact in the knowledge base)

Using rule 1, I can infer that CanDriveBetween(Biyabanak, Ryam Frolovskiy) implies Connected(Biyabanak, Ryam Frolovskiy).
However, according to rule 2, Connected(Biyabanak, Ryam Frolovskiy) is False.
Therefore, the answer is False.

**Few-shot CoT:**

1.Biyabanak is a desert in Iran,

2. and Ryam Frolovskiy is an island in the Caspian Sea, which is not connected to Iran by land.

Therefore, the answer is False.

**RAG + Zero-shot CoT:**

1. LocatedIn(Biyabanak, Iran) (Given in KB)

2. PartOf(Iran, Iran) ( Rule: $\forall$ A:place or location B:place or location, LocatedIn(A, B) $\implies$ PartOf(A, B) )

3. ShareLandBorders(Iran, Azerbaijan) (Given in KB)

4. LandConnected(Iran, Azerbaijan) ( Rule: $\forall$ A:country B:country, ShareLandBorders(A, B) $\implies$ LandConnected(A, B) )

5. LocatedIn(Ryam Frolovskiy, Azerbaijan) (Assuming this is true, since it's not given in KB)

6. PartOf(Ryam Frolovskiy, Azerbaijan) ( Rule: $\forall$ A:place or location B:place or location, LocatedIn(A, B) $\implies$ PartOf(A, B) )

7. CanDriveBetween(Biyabanak, Ryam Frolovskiy) ( Rule: $\forall$ A, B, C, D, LocatedIn(A, C) && LocatedIn(B, D) && LandConnected(C, D) $\implies$ CanDriveBetween(A, B)|A:location, B:location, C:location, D:location )

Therefore, the answer is True.

**RAG + Few-shot CoT:**

1. Since Biyabanak is a desert in Iran,

2. and Ryam Frolovskiy is a place in Russia,

3.and Iran and Russia are land connected (although not explicitly stated in the provided KB, but it can be inferred from the general knowledge that Iran and Russia share a border), then it is possible to drive between Biyabanak and Ryam Frolovskiy.

Using the rule: $\forall$ A, B, C, D , LocatedIn(A, ?C) && LocatedIn(B, ?D) && LandConnected(?C, ?D) $\implies$ CanDriveBetween(A, B) With A = Biyabanak, B = Ryam Frolovskiy, C = Iran, D = Russia

Therefore, the answer is True.

**Discussion:**

Similar to the deductive reasoning experiment, LLM-TH produces the correct proof on a complete knowledge base (KB) without requiring the LLM to surface commonsense axioms from it.
This example also highlights the limitations of the LLM-based baselines. For instance, the zero-shot CoT baseline incorrectly claims that *Biyabanak* and *Ryam Frolovskiy* are not connected, leading to an incorrect final answer. Similarly, the few-shot CoT baseline provides an inaccurate statement about *Ryam Frolovskiy*, describing it as an island in the *Caspian sea* when it is, in fact, a forest in *Russia*.
The RAG-based baselines further reveal challenges in the retrieval stage. For example, the relevant fact about the location of *Ryam Frolovskiy* is not retrieved, prompting the LLM to make assumptions in the zero-shot setting. Although the final answer in this case is correct, the reasoning steps contain errors and deviate from the correct proof. In the few-shot prompting setting, while the final answer is also correct, it is based on an inaccurate statement (in fact, Iran and Russia do not share a land border).

## F.2 EXPERIMENTS ON THE INCOMPLETE KB

### F.2.1 Deductive Reasoning

**Query**: Does Sulfadiazine resolve an infection caused by Rhodospirillaceae?

**Relevant Rules from the KB**:

Facts:

(1) *"Subclass of"*(*"Rhodospirillaceae"*"*Rhodospirillaceae bacteria"*),

(2) *"Gram Negative"*(*"Rhodospirillaceae bacteria"*)

(3) *"Treats Gram Negative"*(*"Sulfadiazine"*)

Axioms:

(4)

$$\forall A, B, C, \text{"Subclass of"}(A, B) \wedge \text{"Gram Negative"}(B) \wedge \text{"Treats Gram Negative"}(C) \implies \text{"Unresistant To"}(A, C) | A\text{: "Bacteria"}, B\text{: "Taxon"}, C\text{: "Antibiotic"}$$

(5)

$$\forall A, B, \text{"Unresistant To"}(A, B) \implies \text{"Can be Used to Treat"}(A, B) | A\text{: "Bacteria"}, \text{"B: Antibiotic"}$$

(6)

$$\forall A, B, \text{"Can be Used to Treat"}(A, B) \implies \text{"Can be Used to Kill"}(A, B) | A\text{: "Bacteria"}, \text{"B: Antibiotic"}$$

(7) $\forall A, B, \text{"Can be Used to Kill"}(A, B) \implies \text{"Disinfects"}(A, B) | A\text{: "Bacteria"}, \text{"B: Antibiotic"}$

**Ground Truth Proof**:

1- Applying (1), (2), (3) to (4) with the following substitution:

$\theta = \{A/\text{"Rhodospirillaceae"}, B/\text{"Rhodospirillaceae bacteria"}, C/\text{"Sulfadiazine"}\}$ yields:

*"Unresistant To"*(*"Rhodospirillaceae"*, *"Sulfadiazine")*.

2- Applying *"Unresistant To"*(*"Rhodospirillaceae"*, *"Sulfadiazine")* to (5)

with $\theta = \{A/\text{"Rhodospirillaceae"}, B/\text{"Sulfadiazine"}\}$ yields:

*"Can be Used to Treat"*(*"Rhodospirillaceae"*, *"Sulfadiazine")*

3-Applying *"Can be Used to Treat"*(*"Rhodospirillaceae"*, *"Sulfadiazine")* to (6)

with $\theta = \{A/\text{"Rhodospirillaceae"}, B/\text{"Sulfadiazine"}\}$ yields:

*"Can be Used to Kill"*(*"Rhodospirillaceae"*, *"Sulfadiazine"*)

4- Applying *"Can be Used to Kill"*(*"Rhodospirillaceae"*, *"Sulfadiazine"*) to (7)

with $\theta = \{A/\text{"Rhodospirillaceae"}, B/\text{"Sulfadiazine"}\}$ yields:

5- *"Disinfects"*(*"Rhodospirillaceae"*, *"Sulfadiazine"*).

6- Using the commonsense axiom (absent from the KB) that:

$$\forall A, B, \textit{“Disinfects”}(A, B) \implies \textit{“Resolves Infection Caused By”}(A, B) | \textit{A: “Bacteria”}, \textit{“B: Antibiotic”}$$

Therefore, the answer to the query is True.

---

**LLM-TH:**

Proof for query *“Resolves Infection Caused By”*(*“Rhodospirillaceae”*, *“Sulfadiazine”*):

Negated query: ¬*“Resolves Infection Caused By”(“Rhodospirillaceae”,“Sulfadiazine”)*
Note: In the following sequent calculus formulae, we use “*R*” to refer to “*Rhodospirillaceae*”, “*S*” to represent “*Sulfadiazine*”, “*GN*” to show “*Gram Negative*”, “*TGN*”, to denote “*Treats Gram Negative*”, “*SO*” to show “*Subclass Of*” due to space limitations.

Step 1-

$$\frac{\neg\textit{“Resolves Infection Caused By”(“R”,“S”)} \quad \textit{“Disinfects(A,B)”} \vee \neg\textit{“Can be Used to Kill(A,B)”}}{\neg\textit{“Can be Used to Kill”(“R”,“S”)}}.$$

Under unification $\theta = \{A/\textit{“Rhodospirillaceae”}, B/\textit{“Sulfadiazine”}\}$.

In this step, by theory resolving ¬*“Resolves Infection Caused By”(“R”,“S”)* with *“Disinfects”(A,B)*, the LLM successfully surfaced the commonsense axiom:

$$\forall A, B, \textit{“Disinfects”}(A, B) \implies \textit{“Resolves Infection Caused By”}(A, B) | \textit{A: “Bacteria”}, \textit{“B: Antibiotic”}.$$

Step 2-

$$\frac{\neg\textit{“Can be Used to Kill”(“R”,“S”)} \quad \textit{“Can be Used to Kill”(A,B)} \vee \neg\textit{“Can be Used to Treat”(A,B)}}{\neg\textit{“Can be Used to Treat(“R”,“S”)”}}.$$

Under unification $\theta = \{A/\textit{“Rhodospirillaceae”}, B/\textit{“Sulfadiazine”}\}$.

Step 3-

$$\frac{\neg\textit{“Can be Used to Treat”(“R”,“S”)} \quad \textit{“Can be Used to Treat”(A,B)} \vee \neg\textit{“Unresistant To”(A,B)}}{\neg\textit{“Unresistant To(“R”,“S”)”}}.$$

Step 4-

$$\frac{\neg\textit{“Unresistant To”(“R”,“S”)} \quad \textit{“Unresistant To”(A,B)} \vee \neg\textit{“SO”(A,B)} \vee \neg\textit{“GN”(B)} \vee \neg\textit{“TGN”(C)}}{\neg\textit{“SO”(“R”,B)} \vee \neg\textit{“GN”(B)} \vee \neg\textit{“TGN”(“S”)}}$$

Under unification $\theta = \{A/\textit{“Rhodospirillaceae”}, C/\textit{“Sulfadiazine”}\}$.

Step 5-

$$\frac{\neg\textit{“SO”(“R”,B)} \vee \neg\textit{“GN”(B)} \vee \neg\textit{“TGN”(“S”)} \quad \textit{“SO”(“R”, “R bacteria”)} \wedge \textit{“GN”(“R bacteria”)} \wedge \textit{“TGN”(“S”)}}{\perp}.$$

Therefore, the query is proven to be True with the Proof score: $\rho_{\text{query}} = (1, 0.981, 5)$

* * *

---

**RAG + Zero-shot CoT:**

1. SubClassOf(Rhodospirillaceae, Rhodospirillaceae bacteria) (from KB)

2. GramNegative(Rhodospirillaceae bacteria) (from KB)

3. ∀A, B, C, SubClassOf(A, B) ∧ GramNegative(B) ∧ TreatsGramNegative(C) $\implies$ $UnResistantTo(A, C)$ (from KB)

4. SubClassOf(Rhodospirillaceae, Rhodospirillaceae bacteria) $\implies$ PartOf(Rhodospirillaceae, Rhodospirillaceae bacteria) (from KB, using the rule ∀A, B, SubClassOf(A, B) $\implies$ PartOf(A, B)|A: animal, B: Taxon)

5. PartOf(Rhodospirillaceae, Rhodospirillaceae bacteria)∧GramNegative(Rhodospirillaceae bacteria)∧ TreatsGramNegative(sulfadiazine) $\implies$ UnResistantTo(Rhodospirillaceae, sulfadiazine) (from 2, 4, and the assumption that sulfadiazine treats Gram-Negative bacteria)

6. ∀A, B, UnResistantTo(A, B) $\implies$ CanBeUsedToTreat(A, B)|A:bacteria, B:antibiotic (from KB)

7. UnResistantTo(Rhodospirillaceae, sulfadiazine) $\implies$ CanBeUsedToTreat(Rhodospirillaceae, sulfadiazine) (from 5 and 6)

8. ∀A, B, CanBeUsedToTreat(A, B) $\implies$ CanBeUsedToKill(A, B)|A: bacteria, B: antibiotic (from KB)

9. CanBeUsedToTreat(Rhodospirillaceae, sulfadiazine) $\implies$ CanBeUsedToKill(Rhodospirillaceae, sulfadiazine) (from 7 and 8)

10. ∀A, B, CanBeUsedToKill(A, B) $\implies$ Disinfects(A, B)|A: bacteria, B: antibiotic (from KB)

11. CanBeUsedToKill(Rhodospirillaceae, sulfadiazine) $\implies$ Disinfects(Rhodospirillaceae, sulfadiazine) (from 9 and 10) Therefore, the answer is True.

---

**RAG + Few-shot CoT:**

1- Rhodospirillaceae is a gram-negative bacteria,

2- and sulfadiazine treats gram-negative bacterias. Therefore, the answer is True.

---

**Discussion:**

This example demonstrates how LLM-TH can perform commonsense reasoning on an incomplete KB. Since no complementary literal for *"Resolves Infection Caused By"* is found, LLM-TH invokes the LLM to perform a theory resolution step. In this step, it correctly selects *"Disinfects"* as the clause with the highest priority for theory resolution, enabling it to provide the correct proof.
In this example, all baselines are also able to produce the correct proof. However, the RAG+zero-shot CoT baseline exhibits minor errors, such as selecting incorrect KB rules in step 4 and neglecting KB facts in step 5, instead basing its reasoning on an assumption.

---

### F.2.2 Geographical QA

---

**Query**: Can I take a road trip between Ryam Frolovskiy and Biyabanak?

---

**Relevant Rules from the KB**:

Facts:

(1) *"Located in"*(*"Biyabanak"*, *"Iran"*),

(2) *"Located in"*(*"Ryam Frolovskiy"*, *"Russia"*),

(3) *"Share Land Borders"*(*"Iran"*, *"Azerbaijan"*)

(4) *"Share Land Borders"*(*"Azerbaijan"*, *"Russia"*)

Axioms:

(5)

$$\forall A, B, \text{“Share Land Borders”}(A, B) \implies \text{“Land Connected”}(A, B) | A : \text{country}, B : \text{country}$$

(6)

$$\forall A, B, C, \text{“Land Connected”}(A, B) \wedge \text{“Land Connected”}(B, C) \implies \text{“Land Connected”}(A, C) | A : \text{location}, B : \text{location}, C : \text{location}$$

(7)

$$\forall A, B, C, D, \text{“Located in”}(A, C) \wedge \text{“Located in”}(B, D) \wedge \text{“Land Connected”}(C, D) \implies \text{“Can Drive Between”}(A, B) | A : \text{location}, B : \text{location}, C : \text{location}, D : \text{location}$$

---

**Ground Truth Proof**:

1- Applying (3) to (5) with the following substitution:

$\theta = \{A/\text{“Iran”}, B/\text{“Azerbaijan”}\}$ yields:

(8) *“Land Connected”*(*“Iran”*, *“Azerbaijan”*).

2- Applying (3) to (5) with the following substitution:

$\theta = \{A/\text{“Azerbaijan”}, B/\text{“Russia”}\}$ yields:

(9) *“Land Connected”*(*“Azerbaijan”*, *“Russia”*).

3- Applying (8), (9) to (6) with the following substitution:

$\theta = \{A/\text{“Iran”}, B/\text{“Azerbaijan”}, C/\text{“Russia”}\}$ yields:

(10) *“Land Connected”(“Iran”,“Russia”).*

4- Applying (1), (2), (10) to (7) with the following substitution:

$\theta = \{A/\text{“Biyabanak”}, B/\text{“Ryam Frolovskiy”}, C/\text{“Iran”}, D/\text{“Russia”}\}$ yields:

(11) *“Can Drive Between”*(*“Biyabanak”*, *“Ryam Frolovskiy”*).

5- Using the commonsense axiom (absent from the KB) that:

$$\forall A, B, \text{“Can Drive Between”}(A, B) \implies \text{“Can Take Road Trip Between”}(A, B) | \text{A: “location”}, \text{“B: location”}$$

Therefore, the answer to the query is True.

---

**LLM-TH:**

Proof for query *“Can Drive Between”*(*“Biyabanak”*, *“Ryam Frolovskiy”*)

Negated query: $\neg$*“Can Drive Between”*(*“Biyabanak”*, *“Ryam Frolovskiy”*)

Note: In the following sequent calculus formulae, we use “*Bi*” to refer to “*Biyabanak*”, “*RF*” to represent “*Ryam Frolovskiy*”, “*Az*“ to refer to ‘*Azerbaijan*”, “*CTRTP*” to show “*Can Take Road Trip Between*”, “*CDB*” to show “*Can Drive Between*”, “*LI*” to show “*Located In*”, “*LC*” to show “*Land Connected*”, and *“SLB”* to show *““Share Land Borders””* due to space limitations.

Step 1-

$$\frac{\neg\text{“CTRTB”(“Bi”,“RF”)} \quad \text{“CDB”(A,B)} \vee \neg\text{“LI”(A,C)} \vee \neg\text{“LI”(B,D)} \vee \neg\text{“LC”(C,D)}}{\neg\text{“LI(“Bi”,C)”} \vee \neg\text{“LI”(“RF”,D)} \vee \neg\text{“LC”(C,D)}} .$$

Under unification $\theta = \{A/\text{“Biyabanak”}, C/\text{“Ryam Frolovskiy”}\}$.

In this step, by theory resolving $\neg$*"Can Take Road Trip Between"("Bi", "RF")* with *"Can Drive Between"(A,B)*, the LLM successfully surfaced the commonsense axiom:

$$\forall A, B, \text{"Can Drive Between"}(A, B) \implies \text{"Can Take Road Trip Between"}(A, B) | \text{A: "location", "B: location"}.$$

Step 2-

$$\frac{\neg\text{"LI"("Bi",C)} \vee \neg\text{"LI"("RF",D)} \vee \neg\text{"LC"(C,D)} \quad \text{"LI"("Bi","Iran")} \wedge \text{"LI"("RF","Russia")}}{\neg\text{"LC"("Iran","Russia")}}.$$

Under unification $\theta = \{C/\text{"Iran"}, D/\text{"Russia"}\}$.

Step 3-

$$\frac{\neg\text{"LC"("Iran","Russia")} \quad \text{"LC"(A,C)} \vee \neg\text{"LC"(A,B)} \vee \neg\text{"LC"(B,C)}}{\neg\text{"LC"("Iran",B)} \vee \neg\text{"LC"(B,"Russia")}}.$$

Under unification $\theta = \{A/\text{"Iran"}, C/\text{"Russia"}\}$.

Step 4-

$$\frac{\neg\text{"LC"("Iran",B)} \quad \text{"LC"("Iran",B)} \vee \neg\text{"SLB"("Iran",B)}}{\neg\text{"SLB"("Iran",B)}}.$$

Step 5-

$$\frac{\neg\text{"LC"(B,"Russia")} \quad \text{"LC"(B,"Russia")} \vee \neg\text{"SLB"(B,"Russia")}}{\neg\text{"SLB"(B,"Russia")}}.$$

Step 6-

$$\frac{\neg\text{"SLB"("Iran", B)} \vee \neg\text{"SLB"(B, "Russia")} \quad \text{"SLB"("Iran","Az")} \wedge \text{"SLB"("Az","Russia")}}{\perp}.$$

Under unification $\theta = \{B/\text{"Az"}\}$.

Therefore, the query is proven to be True with the proof score: $\rho_{\text{query}} = (1, 0.994, 6)$

* * *

---

**RAG + Zero-shot CoT:**

1- From the knowledge base, we have: *PartOf(Biyabanak, Iran)* (from the given KB)

2- PartOf(Ryam Frolovskiy, Belarus) (not mentioned in the KB, but assuming it's true for the sake of the query)

3- Using the rule:

"$\forall A, B, C$, *PartOf(A, B)* $\wedge$ *PartOf(B, C)* $\implies$ *PartOf(A, C)*$|$*A: location*, *B: location*, *C: location*", We can infer: *PartOf(Biyabanak, Asia)* (since Iran is part of Asia)

4- and *PartOf(Ryam Frolovskiy, Europe)* (since Belarus is part of Europe)

5- Using the rule: $\forall$*A, B, C, LandConnected(A, B)* $\wedge$ *LandConnected(B, C)* $\implies$ *LandConnected(A, C)*$|$*A:country*, *B:country*, *C:country*, we can infer: *Not LandConnected(Iran, Belarus)* (since Iran is in Asia and Belarus is in Europe, and they are not land-connected)

6- Using the rule:"$\forall A, B, C, D$, *LocatedIn(A, C)* $\wedge$ *LocatedIn(B, D)* $\wedge$ *LandConnected(C, D)* $\implies$ *CanDriveBetween(A, B)|A: location, B: location, C: country, D: country*", we cannot infer *CanDriveBetween(Biyabanak, Ryam Frolovskiy)* since Iran and Belarus are not land-connected. Therefore, the answer is False.

---

**RAG + Few-shot CoT:**

1- To determine if you can take a road trip between Biyabanak and Ryam Frolovskiy, we need to check if Biyabanak and Ryam Frolovskiy are located in countries that are land connected.
2- From the KB, we know that Biyabanak is part of Iran,

3- and Ryam Frolovskiy is not mentioned in the KB, but since it is not mentioned, we will assume it is in Russia (based on the name).

4- From the KB, we also know that Iran and Russia are land connected (rule 1).

5- Using the rule "$\forall A, B, C, D$, *LocatedIn(A, C)* $\wedge$ *LocatedIn(B, D)* $\wedge$ *LandConnected(C, D)* $\implies$ *CanDriveBetween(A, B)|A: location, B: location, C: country, D: country*", we can infer that you can take a road trip between Biyabanak and Ryam Frolovskiy. Therefore, the answer is True.

---

**Discussion:**

This example further illustrates the successful application of LLM-based theory resolution by LLM-TH, where a theory resolution step leverages the LLM's commonsense knowledge to address KB incompleteness. Specifically, in Step 1, the LLM surfaces a commonsense axiom to theory resolve *¬Can Drive Between"* with *¬Can Take Road Trip Between"*.
This example also highlights the limitations of the RAG-based baselines. Due to a failure in retrieving facts about *Ryam Frolovskiy*, the RAG+Zero-shot CoT baseline makes an incorrect assumption about its location. Furthermore, subsequent reasoning steps are flawed, resulting in a wrong answer. Although the RAG+Few-shot CoT baseline provides the correct final answer, its reasoning process is not entirely reliable. Notably, because the LLM lacks access to a fact about the location of *Ryam Frolovskiy*, it makes an assumption about its location—which happens to be correct in this case. However, it also incorrectly references a rule in the KB that states *Iran* and *Russia* are land connected, even though no such rule exists in the KB.
These examples underscore the limited reliability of existing LLM-based baselines when combining commonsense reasoning with factual information.

---