Can Large Language Models Understand DL-Lite Ontologies? An Empirical Study

Anonymous ACL submission

Abstract

Large language models (LLMs) have shown significant achievements in solving a wide range of tasks. Recently, LLMs' capability to store, retrieve and infer with symbolic knowledge has drawn a great deal of attention, showing their potential to understand structured information. However, it is not yet known whether LLMs can understand Description Logic (DL) ontologies. In this work, we empirically analyze the LLMs' capability of understanding DL-Lite ontologies covering 6 representative tasks from syntactic and semantic aspects. With extensive experiments, we demonstrate both the effectiveness and limitations of LLMs in understanding DL-Lite ontologies. We find that LLMs can understand formal syntax and model-theoretic semantics of concepts and roles. However, LLMs struggle with understanding TBox NI transitivity and handling ontologies with large ABoxes. We hope that our experiments and analyses provide more insights into LLMs and inspire to build more faithful knowledge engineering solutions.

1 Introduction

Large Language Models (LLMs) (Brown et al., 2020a; OpenAI, 2023; Touvron et al., 2023) have showcased remarkable proficiency in understanding textual data and revolutionized the field of natural language processing. Recent studies suggest that LLMs possess adaptability to store, retrieve and infer with symbolic knowledge such as knowledge graphs (KGs) (Mruthyunjaya et al., 2023; Feng et al., 2023), sparking interest in their potential for understanding structured information. However, LLMs' capacity in understanding more complex symbolic knowledge, Description Logic (DL) ontologies, remains unexplored.

Compared with KGs, DL ontologies have more fined-grained knowledge representation with formal syntax and model-theoretic semantics. For syntax, while most KGs generally only support atomic entities like $PhdStudent$, DL ontologies can support various constructors and compound concepts such as $\neg PhdStudent \sqcap \exists HasStudentID$. For semantics, DL ontologies have model-theoretic semantics. For example, the above complex concept can be interpreted as the set of individuals who are not PhD students but do have a student ID. Further, DL ontologies efficiently support logical reasoning such as $R_1 \sqsubseteq R_2, C \sqsubseteq \neg\exists R_2^- \rightarrow \exists R_1^- \sqsubseteq \neg C$. Understanding a DL ontology goes beyond just the capabilities of storage, retrieval, and inference, but involves a deeper comprehension of its formal syntax and semantic interpretations.

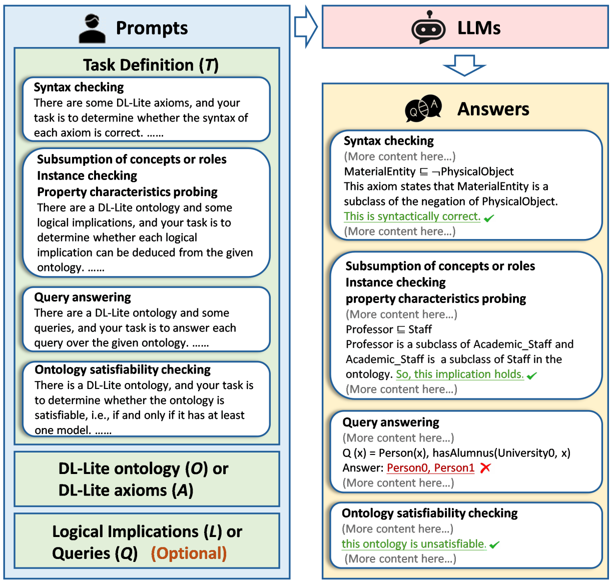


Figure 1: Illustration and examples of evaluation tasks.

While the necessity for more detailed investigations for LLMs' capacity in understanding DL ontologies is clear, a comprehensively evaluation presents a challenge. Most related works focus on LLMs' capacity to capturing patterns in KGs (Mruthyunjaya et al., 2023; Feng et al., 2023), far away from indicating that LLMs possess the ability to understand DL ontologies. Even though many endeavors study whether LLMs can do logical reasoning (Wang et al., 2024b; Bao et al., 2024; Luo et al., 2023; Pan et al., 2023), few of them explore LLMs' capacity with DL services. DL is primarily focused on representing and reasoning about the hierarchical relationships and properties of concepts within a domain, distinguishing it from other logics by its emphasis on structured, formal ontology. This research gap highlights the significance and challenges in comprehensively evaluating whether LLMs can understand DL ontologies.

In this study, we investigate how effectively LLMs can understand DL-Lite ontologies, a member of the DL ontology family known for simplicity and efficient reasoning. We present an evaluation framework that comprehensively assesses LLMs' capability to understand DL-Lite ontologies in two aspects, respectively, whether LLMs can grasp the formal representations (the syntactic aspect) and whether LLMs can understand the semantic interpretations of ontologies and effectively utilize them (the semantic aspect). For the syntactic aspect, we investigate whether LLMs can comprehend the structural rules, valid statements, and expressions of DL-Lite through syntax checking. For the semantic aspect, we first investigate whether LLMs can understand the semantics of concepts and roles from two aspects, intension and extension, by subsumption of concepts or roles and instance checking respectively. Additionally, we probe property characteristics in DL-Lite ontologies, such as inverse roles and functional roles. Further, we conduct query answering and ontology satisfiability checking to evaluate whether LLMs can understand the semantics of the whole ontologies. Figure 1 gives an illustration of these tasks.

Through extensive experiments, we find that:

• LLMs possess capacity for understanding DL-Lite syntax (Section 4.1).

• LLMs can understand the semantics of concepts, roles (Section 4.2.1) and some property characteristics (Section 4.2.2).

• LLMs fail to understand some TBox NI transitivity rules, thus LLMs' capability for subsumption of concepts or roles is limited (Section 4.2.1).

• LLMs fail to handle ontologies with large scale ABoxes, thus LLMs' capability for instance checking and query answering is limited (Section 4.2.1, Section 4.2.3).

• LLMs can perform ontology satisfiability checking with DL-Lite ontologies but struggle with detecting inconsistency in complex ontologies (Section 4.2.4).

To the best of our knowledge, this is the first study to conduct comprehensive evaluations about whether LLMs can understand DL-Lite ontologies. Overall, our work contributes to a better understanding of LLMs' behaviors and inspires to build more faithful knowledge engineering solutions.

2 Related Work

2.1 LLMs for Syntax Understanding

With the arrival of LLMs, some works focus on using LLMs to translate natural language into formal language to reduce labor in real-world applications. For example, Fill et al. (2023) use ChatGPT to generate entity relation (ER) diagrams for conceptual modeling and Yang et al. (2023) present a fine-tuned LLaMA-7B model to translate natural language into first-order logic (FOL). Mateiu and Groza (2023) convert natural language sentences into OWL Functional Syntax, showing LLMs' prospect of ontology engineering. However, there is a significant difference in syntax between DL and other formal languages like ER, FOL and OWL, and few works study whether LLMs can understand DL syntax.

2.2 LLMs for Semantics Understanding

Some studies, like (Mruthyunjaya et al., 2023; Feng et al., 2023), focus on LLMs' capacity of matching up to knowledge that presents in KGs, but such kind of factual knowledge is not the main focus of DL ontology. Shani et al. (2023) analyze how well LLMs capture concepts and their structures, showing evidence that LLMs can understand conceptual knowledge, but DL ontologies support more automated reasoning than just conceptual taxonomies. Further, recent works conduct evaluations on how effectively LLMs can capture logic and perform logical reasoning (Wang et al., 2024b; Bao et al., 2024; Luo et al., 2023; Pan et al., 2023; Chen et al., 2023). However, none of them study LLMs' capacity in understanding DL semantics. Focusing on representation and reasoning with structured, formal ontology, DL provides formal semantics based on model theory and strikes a balance between expressiveness and computational tractability , making differences with other logics.

Additionally, some works study LLMs acting as knowledge bases (Heinzerling and Inui, 2021), which focus on LLMs' capacity for storing and retrieving knowledge. In contrast, we conduct an in-depth study of LLMs' understanding of the components (e.g., concepts and roles) in DL ontologies, like how these components get their meanings (from two aspects, extension and intension) and how the meaning of a complex expression depends on its parts (considering various reasoning services).

3 Preliminaries

In this section, we briefly recall some basic notions about DL-Lite ontology (Calvanese et al., 2007, 2009). Particularly, we focus on DL-Lite$_{core}$, DL-Lite$_{\mathcal{F}}$ and DL-Lite$_{\mathcal{R}}$, three members in DL-Lite family, while our evaluation framework can be applied to any other description logics (DLs) such as DL-Lite$_{\mathcal{A}}$, $\mathcal{ALC}$ and $\mathcal{EL}$.

DL-Lite ontology. We start from DL-Lite$_{core}$ concepts and roles, which are defined as follows:

$$B ::= A \mid \exists R \mid \exists R^- \qquad R ::= P \mid P^-$$
$$C ::= B \mid \neg B \mid C_1 \sqcap C_2 \qquad E ::= R \mid \neg R$$

where A denotes an atomic concept, P denotes an atomic role, and P^- denotes the inverse of the atomic role P and $\neg R$ denote the negation of R. We call B, R, C, E a basic concept, a basic role, a general concept and a general role respectively.

A DL-Lite$_{core}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. $\mathcal{T}$ is formed by a finite set of concept inclusion assertions of the form $B \sqsubseteq$ C. $\mathcal{A}$ is formed by a finite set of membership assertions on atomic concepts and on atomic roles, of the form $A(a)$ and $P(a, b)$, where a and b are constants. DL-Lite$_{\mathcal{R}}$ extends DL-Lite$_{\text{core}}$ with role inclusion assertions of the form $R \sqsubseteq E$ and DL-Lite$_{\mathcal{F}}$ extends DL-Lite$_{core}$ with functionality on roles or on their inverses of the form (funct R).

The semantics of DL-Lite is given in a model-theoretic way via interpretations over a fixed infinite domain Δ. Given an interpretation $\mathcal{I}$ and an assertion α, $\mathcal{I} \models \alpha$ means that $\mathcal{I}$ is a model of α. An interpretation is a model of a DL-Lite ontology $\mathcal{O}$, if and only if it is a model for each assertion in $\mathcal{O}$. An ontology $\mathcal{O}$ is satisfiable if it has at least one model. $\mathcal{O}$ logically implies an assertion α, written $\mathcal{O} \models \alpha$, if all models of $\mathcal{O}$ are also models of α.

Reasoning services with DL-Lite ontology. Designed for knowledge representation and efficient reasoning, DL-Lite ontology supports several DL reasoning services (Calvanese et al., 2007):

- Ontology satisfiability checking: given an ontology $\mathcal{O}$, verify whether $\mathcal{O}$ admits at least one model;
- Logical implication of $\mathcal{O}$ assertions, which consists of the following sub-problems:
  - Instance checking: given an ontology $\mathcal{O}$, a concept C and a constant a (resp., a role R and a pair of constants a and b), verify whether $\mathcal{O} \models C(a)$ (resp., $\mathcal{O} \models R(a, b)$)
  - Subsumption of concepts or roles: given a TBox $\mathcal{T}$ and two general concepts C_1 and C_2 (resp., two general roles R_1 and R_2), verify whether $\mathcal{T} \models C_1 \sqsubseteq C_2$ (resp., $\mathcal{T} \models R_1 \sqsubseteq R_2$).
  - Checking functionality - given a TBox $\mathcal{T}$ and a basic role R, verify whether $\mathcal{T} \models$ (funct R).
- Query answering: given an ontology $\mathcal{O}$ and a query q over $\mathcal{O}$, compute the answer set $\mathrm{ans}(q, \mathcal{O})$.

A key characteristic of DL-Lite syntax and semantics is that they are primarily designed for performing these DL reasoning services efficiently. Conducting an extensive evaluation of LLMs for these tasks is beneficial to provide insights into whether LLMs can understand DL-Lite ontologies.

Transitivity rules. For instance checking and subsumption of concepts or roles, we especially focus on deducing logical implications with some reasoning rules. Borrowing the idea of Canonical Interpretation (PI-closure) and Closure of Negative Inclusion Assertions (NI-closure) from (Calvanese et al., 2007, 2009), we collect the reasoning rules in three categories, 2 TBox PI (positive inclusion) transitivity rules, 11 TBox NI (negative inclusion) transitivity rules and 5 ABox transitivity rules. We cover them in Appendix A and there are some examples below:

TBox PI transitivity examples:
$\alpha = C_1 \sqsubseteq C_2, \beta = C_2 \sqsubseteq C_3 \rightarrow \beta_{\text{new}} = C_1 \sqsubseteq C_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_2 \sqsubseteq R_3 \rightarrow \beta_{\text{new}} = R_1 \sqsubseteq R_3$
TBox NI transitivity examples:
$\alpha = C_1 \sqsubseteq C_2, \beta = C_3 \sqsubseteq \neg C_2 \rightarrow \beta_{\text{new}} = C_1 \sqsubseteq \neg C_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = \exists R_2^- \sqsubseteq \neg C \rightarrow \beta_{\text{new}} = \exists R_1^- \sqsubseteq \neg C$
ABox transitivity examples:
$\alpha = C \sqsubseteq \exists R, \beta = C(a) \rightarrow \beta_{\text{new}} = R(a, a_{\text{new}})$
$\alpha = \exists R \sqsubseteq C, \beta = R(a, a') \rightarrow \beta_{\text{new}} = C(a)$

4 Unveiling LLMs' Capabilities in Understanding DL-Lite Ontology

In this section, we comprehensively investigate how effectively LLMs can understand DL-Lite ontologies, especially, grasp the formal representations (syntax) and interpretations of elements in ontologies (semantics). We conduct a series of tasks, including syntax checking, subsumption of concepts or roles, instance checking, query answering, ontology satisfiability checking and property characteristics probing. Figure 2 presents an overview of the evaluation framework for the first three tasks.

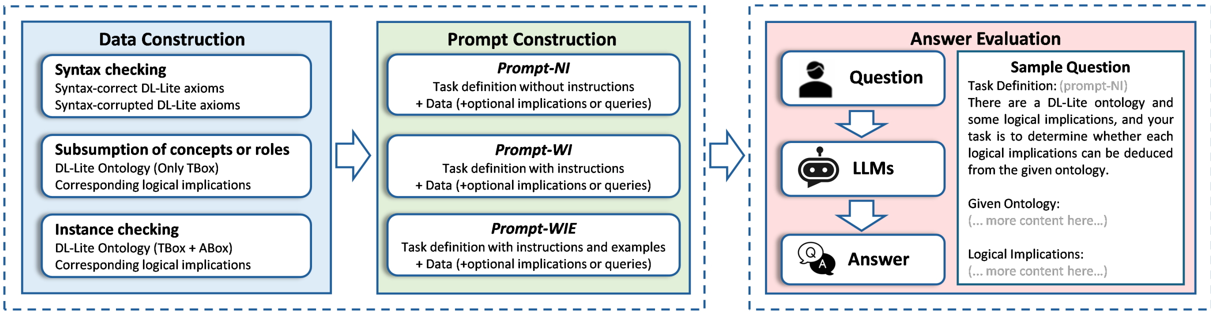


Figure 2: Evaluation pipeline for syntax checking, subsumption of concepts or roles, and instance checking.

Datasets	GO			FMA			MarineTLO			Music			OBI		
Metric	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
GPT3.5–*NI*	66	90	76	100	100	100	96	87	91	100	97	98	100	100	100
GPT3.5–*WI*	66	97	79	68	100	81	100	100	100	83	100	91	83	97	89
GPT3.5–*WIE*	72	93	81	65	100	79	87	87	87	63	100	77	82	90	86
GPT4o–*NI*	100	97	98	86	100	92	100	100	100	100	100	100	100	97	98
GPT4o–*WI*	91	100	95	88	100	94	97	100	98	100	100	100	79	100	88
GPT4o–*WIE*	100	100	100	88	100	94	97	100	98	97	100	98	94	100	97
LLaMA3-8b–*NI*	65	93	77	50	100	67	50	100	67	63	97	76	58	93	71
LLaMA3-8b–*WI*	91	100	95	50	100	67	67	100	80	76	97	85	64	90	75
LLaMA3-8b–*WIE*	67	93	78	58	97	73	63	100	77	78	97	86	71	100	83

Table 1: Performances of LLMs in syntax checking (%).

We collect specified datasets for each task and construct three prompts of binary questions, and test three LLMs, namely, GPT3.5 (Brown et al., 2020b), GPT4o[1] (OpenAI, 2023) and LLaMA3-8B[2] (Touvron et al., 2023). The evaluation pipelines of the other three tasks introduced later are quite similar.

4.1 Can LLMs Understand the Syntax of DL-Lite Ontologies?

An important aspect of how effectively LLMs can understand DL-Lite ontologies is their capacity to comprehend the syntax. In this section, we conduct syntax checking to evaluate LLMs' comprehension of structural rules and the construction of valid statements and expressions in DL-Lite ontologies.

Datasets. We select several commonly used DL ontologies, including Gene Ontology (GO) (Consortium, 2004), Foundational Model of Anatomy (FMA) (Rosse and Mejino Jr, 2008), Ontology for Biomedical Investigations (OBI) (Bandrowski et al., 2016), MarineTLO (Tzitzikas et al., 2016) and the Music Ontology (Raimond and Sandler, 2012). For each DL ontology, we randomly collect 30 DL-Lite axioms. For each collected axiom, we insert random one type of syntax error, such as invalid quantifier (eg. $\exists TeachesTo \rightarrow \exists\exists TeachesTo$) and invalid conjunction (eg. $Professor \sqcap \exists TeachesTo \rightarrow \sqcap\ Professor\ \exists TeachesTo$). We summarize typical syntax errors in DL-Lite in Appendix B. We build 150 correct and 150 corrupted DL-Lite axioms as datasets for syntax checking.

Experimental setup. We utilize binary questions for syntax checking. Generally, the prompts include task description (T) and the input DL-Lite axioms (A). We design three kinds of prompts:
- prompt without any instructions about DL-Lite syntax in T, denoted as *NI* (No Instructions);
- prompt with instructions about DL-Lite syntax in T, denoted as *WI* (With Instructions);
- prompt with instructions about DL-Lite syntax and corresponding examples in T, denoted as *WIE* (With Instructions and Examples).

Figure 1 shows an example and we cover detailed prompts in Appendix C.

Results analysis. In Table 2, we present precision, recall and F1 score of tested LLMs and prompts. Overall, LLMs possess the ability to understand DL-Lite syntax. We find that no matter what kinds of prompts we use, GPT4o achieves good results on all the five data sources. In comparison, LLaMA3-8b shows relatively poor results. To deliver a more in-depth investigation, we conduct analyses for the following questions:

Can instructions or examples benefit LLMs' understanding of DL-Lite syntax? For GPT3.5 and GPT4o, there is little difference among the three prompts, while performances of LLaMA3-8B–*WI* and LLaMA3-8B–*WIE* are significantly better than those of LLaMA3-8B–*NI*. This may be because

[1]https://openai.com/index/hello-gpt-4o/
[2]https://github.com/meta-llama/llama3

GPT3.5 and GPT4o have learned detailed DL-Lite syntax during training but LLaMA3-8B hasn't.

What types of errors do LLMs usually make for syntax checking? In most cases, LLMs achieve high recall and relatively low precision, since LLMs hardly mistake correct axioms, but do sometimes treat incorrect axioms as correct. Especially, we find that LLMs sometimes perform poorly in distinguishing between concepts and roles. For example, they may treat $\exists isConnectedTo \sqsubseteq Organ^{-}$ as syntax-correct, which is incorrect since inverse ($^{-}$) can only be put on roles.

4.2 Can LLMs Understand the Semantics of DL-Lite ontologies?

Another aspect of whether LLMs can understand ontologies is their capacity to comprehend the semantics. Semantics goes beyond the syntactic structure and explores the interpretation and significance of the elements like concepts and roles of the ontology. In this section, we explore the capability of LLMs to understand the semantics of the components within ontology (i.e., concepts and roles) considering instance checking and subsumption of concepts or roles. Additionally, we probe some property characteristics (i.e., inverse roles and functional roles) in DL-Lite ontologies. Further, we conduct query answering and ontology satisfiability checking to explore LLMs' capacity to understand the semantics of the whole ontologies.

4.2.1 Semantics of Concepts and Roles

We evaluate the capacity of LLMs to understand the semantics of concepts and roles from two aspects: extension and intension (Bouaud et al., 1995; Woods, 1975; Formica, 2006; Wang et al., 2024a). The extension of a concept or role refers to the set of individuals or objects that fall under that concept or role (Bouaud et al., 1995; Formica, 2006). For example, the extension of the concept "President of the U.S." would be the set of all individuals considered as U.S. presidents such as "Barack Obama" and "Joe Biden". The intension of a concept or role refers to the characteristics, properties, or conditions that determine whether an individual belongs to that concept or role (Formica, 2006). For example, "President of the U.S." is a "Politician" and "someone who plays a role in federal legislation"[3].

We use instance checking for the former, since it involves determining whether a particular individual belongs to a specified concept within a given ontology. Subsumption of concepts or roles is for the latter, which involves determining whether one concept or role is subsumed by another more general concept or role, reflecting the attributes, characteristics, constraints, and conditions encompassed by the inherent intension.

Data Sources	#T. $B \sqsubseteq C$	#T. $R \sqsubseteq E$	#L. $B \sqsubseteq C$	#L. $R \sqsubseteq E$
VICODI	193	9	195	9
STOCKEXCHANGE	26	0	12	0
UNIVERSITY	36	5	31	9
ADOLENA	100	0	72	0
SEMINTEC	55	0	47	0

Table 2: Statistics about data sources for subsumption of concepts or roles. # denotes "the number of", and T., L. denote TBox and logical implications respectively.

Data Sources	#O. $C(a)$	#O. $R(a,b)$	#L. $C(a)$	#L. $R(a,b)$
UOBM1	2338	0	478	0
UOBM2	1389	0	278	0
UOBM3	678	0	136	0
UOBM4	576	0	113	0
UOBM5	466	0	93	0

Table 3: Statistics about data sources for instance checking. # denotes "the number of", and O., L. denote ontology and logical implications respectively.

Datasets. For subsumption of concepts or roles, we use the TBox of existing DL-Lite ontologies. We select 4 DL-Lite$_{\mathcal{R}}$ ontologies, VICODI (Nagypál et al., 2005), STOCKEXCHANGE (Rodriguez-Muro et al., 2008), UNIVERSITY (Guo et al., 2005), ADOLENA (Keet et al., 2008) from (Pérez-Urbina et al., 2009), and SEMINTEC from (Motik and Sattler, 2006) as approximation of DL-Lite ontology. For instance checking, we construct a series of DL-Lite ontologies of varying sizes using the UOBM benchmark (Ma et al., 2006). We select a variant of UOBM ontology, denoted as UOBM0, and derive five additional ontologies with significantly different ABox sizes by randomly removing class assertions from UOBM0, which are labeled as UOBM1, UOBM2, UOBM3, UOBM4 and UOBM5 respectively.

Then we load the ontologies into Protégé [4] and utilize the reasoning engine HermiT (Glimm et al., 2014) to infer logical implications. We cover the details of using Protégé to obtain logical implications in Appendix D. Because there are a large number of logical implications in instance checking, we randomly select a subset for evaluation. Table 2 and Table 3 show the statistical details.

[3] https://en.wikipedia.org/wiki/President_of_the_United_States

[4] https://protege.stanford.edu/

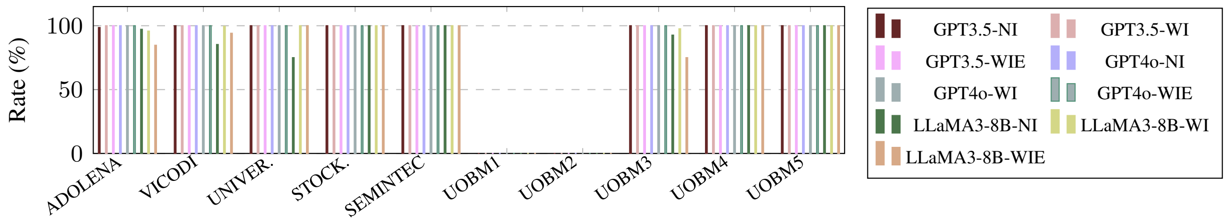


Figure 3: Performances of LLMs in subsumption of concepts or roles and instance checking.

DL-Lite Ontology	Logical Implications
Case 1: TBox = { $C_1 \sqsubseteq C_2$, $C_2 \sqsubseteq \neg C_3$, $C_4 \sqsubseteq \neg C_2$, $R_1 \sqsubseteq R_2$, $\exists R_2 \sqsubseteq \neg C_5$, $C_6 \sqsubseteq \neg \exists R_2$, $R_3 \sqsubseteq R_4$, $\exists R_4^- \sqsubseteq \neg C_7$, $C_8 \sqsubseteq \neg \exists R_4^-$, $R_5 \sqsubseteq R_6$, $R_6 \sqsubseteq \neg R_7$, $R_8 \sqsubseteq \neg R_6$ }	$C_1 \sqsubseteq \neg C_3$, $C_1 \sqsubseteq \neg C_4$, $\exists R_1 \sqsubseteq \neg C_5$, $\exists R_1 \sqsubseteq \neg C_6$, $\exists R_3^- \sqsubseteq \neg C_7$, $\exists R_3^- \sqsubseteq \neg C_8$, $R_5 \sqsubseteq \neg R_7$, $R_5 \sqsubseteq \neg R_8$
Case 2: TBox = { $C_1 \sqsubseteq C_2$, $C_1 \sqsubseteq C_3$, $C_2 \sqsubseteq C_4$, $R_1 \sqsubseteq R_2$, $R_3 \sqsubseteq R_4$, $C_5 \sqsubseteq \exists R_5$, $\exists R_6 \sqsubseteq C_6$, $\exists R_7 \sqsubseteq \exists R_8$}; ABox = { $C_1(a)$, $C_1(b)$, $R_1(c,d)$, $R_3(e,f)$, $C_5(a)$, $R_6(a,k)$, $R_7(g,h)$}	$C_2(a)$, $C_3(a)$, $C_2(b)$, $C_3(b)$, $R_2(c,d)$, $R_4(e,f)$, $C_4(a)$, $C_4(b)$, $R_5(a,\_)$, $C_6(a)$, $R_8(h,\_)$

Table 4: Some ontologies in case study of transitivity rules.

Experimental setup. The prompts include task description (T), input ontology (O, only TBox for subsumption of concepts or roles while TBox + ABox for instance checking) and logical implications (L). We design three kinds of prompts:

• prompt without any instructions about reasoning rules in T, denoted as *NI*;

• prompt with instructions about reasoning rules (TBox PI transitivity, TBox NI transitivity for concept or role subsumption, and ABox transitivity for instance checking) in T, denoted as *WI*;

• prompt with instructions about reasoning rules (same as above) and corresponding examples in T, denoted as *WIE*.

Figure 1 shows examples and we cover detailed prompts in Appendix C. The evaluation metric is the ratio of logical implications that LLMs can deduce to all the logical implications.

Results analysis. The performances of LLMs in subsumption of concepts or roles and instance checking are represented in Figure 3. For subsumption of concepts or roles, we find that LLMs achieve promising results in most cases. However, for instance checking, none of the logical implications can be inferred by LLMs for UOBM1 and UOBM2, even though LLMs achieve good performances for the other three ontologies. This is because the task of subsumption of concepts or roles only requires the input of the TBox which is usually relatively small, while instance checking requires an ontology that includes both the TBox and the ABox where sometimes the ABox can be quite large. We input the TBox and ABox at one prompt and the size of UOBM1 and UOBM2 exceeds the maximum size limit that the selected LLMs can handle. Overall, LLMs perform well in these two tasks when the input ontology is relatively small. More specifically, we analyze the following questions:

How do the size of the ontology and the scale of LLMs affect the understanding of the ontology? The experimental results show that the larger the ontology is, the worse the understanding of LLMs is. For small ontologies, LLMs can achieve almost 100% performance. However, when the size of the ontology exceeds a certain threshold, the performance of LLMs drops to nearly 0%. Similarly, the larger the scale of the LLM is, the better its capacity to understand ontologies is. For instance, the scale of LLaMA3-8B is much smaller than that of GPT-3.5 and GPT-4o, so its performances on several ontologies are significantly worse.

Can LLMs understand the transitivity rules and efficiently apply them in reasoning? For subsumption of concepts or roles and the smaller three ontologies UOBM3, UOBM4, UOBM5 in instance checking in Figure 3, GPT4o can deduce all the implications and GPT3.5, LLaMA3-7b can both deduce most of the logical implications, indicating that LLMs can efficiently perform instance checking and subsumption of concepts or roles when the ontology is not that large.

However, this does not mean that LLMs truly understand and correctly use every transitivity rule because: (1) The used transitivity rules for those logical implications only cover a small part of all the transitivity rules; (2) LLMs may have potential hallucinations about transitivity rules. Thus we conduct a case study. We build five handcrafted DL-Lite ontologies with logical implications for this use where each logical implication can be de-

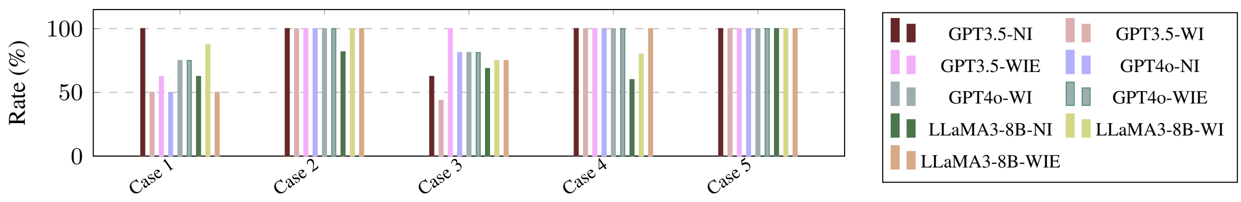


Figure 4: Performances for case study of tranitivity rules.

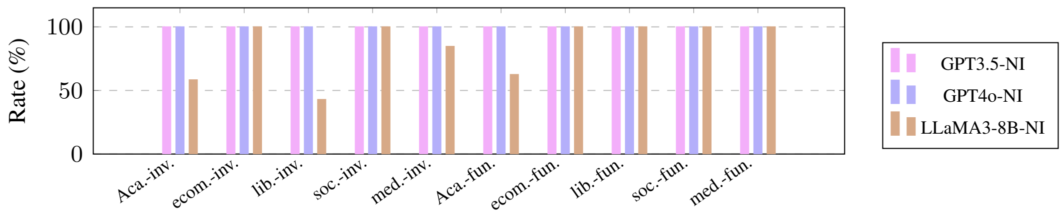


Figure 5: Performances for probing of inverse role property and (inverse) functional role property.

duced by certain kind of transitivity rule and the examples cover all the introduced transitivity rules. Table 6 shows two of them and we cover all of them in Appendix E. We apply the above prompts but add “Give reasons or inferring process for each answer.” to the end of task definition (T). Figure 4 shows the results. LLMs perform well in case 2, case 4 and case 5, but perform poorly in case 1 and case 3, because most logical implications in case 1 and case 3 can only be deduced by TBox NI transitivity, and those in other cases can be deduced by TBox PI transitivity or ABox transitivity. LLMs fail to understand TBox NI transitivity rules well, and instructions or examples have limited effect. We also find LLMs give incorrect explanations to logical implications which can only be deduced by certain TBox NI transitivity rules, indicating that LLMs have hallucinations about TBox NI transitivity rules or possess some incorrect knowledge about TBox NI transitivity.

4.2.2 Property Characteristics Probing

Property characteristics, such as *symmetric property*, *transitive property*, *functional property* and *inverse functional property*, play a significant role in a DL ontology. Some studies have shown evidence that the LLMs have limited knowledge of some property characteristics without external knowledge or instructions such as inverse role property (called "reversal curse" in (Berglund et al., 2023)) and property inheritance (Shani et al., 2023). In this work, especially, we focus on two important property characteristics in DL-Lite, inverse role property and (inverse) functional role property. We set property characteristics probing tasks:

- inverse role probing: Given an ontology O, a role R, its inverse role $P = R^{-}$, and two constants a and b which satisfy $O \models R(a, b)$, verify whether $O \models P(b, a)$.
- (inverse) functional role probing: Given an ontology O, a functional role (funct R) (an inverse functional role (funct R^{-})), and three constants a, b and c which satisfies $O \models R(a, b)$ and $O \models R(a, c)$ (resp. $O \models R(b, a)$ and $O \models R(c, a)$), verify whether $b \equiv c$.

Datasets. We obtain the DL-Lite datasets by extracting and processing existing DL ontologies, namely, Academic Hierarchy (from the University Ontology Benchmark (Ma et al., 2006)), E-Commerce System (from the GoodRelations Ontology (Hepp, 2008)), Library System (from the Dublin Core Metadata (Weibel et al., 1998)), Social Network Relations (from FOAF, Friend of a Friend (Golbeck and Rothstein, 2008)) and Medical Relationships (from SNOMED CT (El-Sappagh et al., 2018)).

For inverse role property probing, we select inverse roles in the ontologies and use them to build logical implications. For example, if $WorksIn$ and $WorksIn^{-}$ exists, we add $Employs$, $Employs \sqsubseteq WorksIn^{-}$, $WorksIn^{-} \sqsubseteq Employs$ to the ontology. If $WorksIn(a, b)$ exists in the ontology, we build the logical implication $Employs(b, a)$. For (inverse) functional role property probing, similarly we select functional roles and build logical implications. For example, if $(funct\ BelongsTo)$ and $BelongsTo(a, b)$ hold, we then add $BelongsTo(a, x)$ to the ontology and build the logical implication $x \equiv b$. Statistical details are covered in Appendix F.

Experimental setup. The prompt is almost the same to prompt-*NI* in instance checking. We add "Give reasons or inferring process." to the end of the task definition. We use GPT4o and the same metric in instance checking for evaluation.

Results analysis. The results in Figure 5 show that LLMs can deduce most of the logical implications. LLMs give reasonable explanations of the deducing process such as "Since BelongsTo(Product1, Category1) is given and BelongsTo is the inverse of Owns, hence Owns(Category1, Product1) can be deduced" and "Given: WorksAt(DrBrown,RegionalHospital) and WorksAt(DrBrown,x3). Since WorksAt is a functional property, DrBrown can only work at one hospital. Thus, x3 must be RegionalHospital to satisfy the functional constraint". LLMs have the potential to understand such logical constraints in DL ontologies, indicating the promising prospects to utlize ontologies to enhance LLMs' inference capacity such as in the scene of "reversal curse" (Berglund et al., 2023).

4.2.3 Query Answering

Query answering over an ontology involves retrieving information that satisfies a given query based on this ontology (Calvanese et al., 2007).

Datasets. We use the Lehigh University Benchmark (LUBM) (Guo et al., 2005) with the given TBox, ABox example and 14 test queries[5].

Experimental setup. We use GPT4o for evaluation. Similar to prompt-*NI* in instance checking, the prompt includes task description (T), input ontology (O) and the query (Q). Because LLMs can't handle large-scale ABox at one time as shown in Section 4.2.1, we cut the ontology into 10 parts and input them in turn.

Results analysis. Test results show that GPT4o fails to give a totally correct answer for each query. For Q3, Q8, Q12, Q13 and Q14, GPT4o can only answer a very small part of all the expected answers. For other queries, GPT4o has hallucinations and answer incorrect answers. LLMs can't memorize and understand large scale factual knowledge and fail to perform query answering well practically.

4.2.4 Ontology Satisfiability Checking

Ontology satisfiability checking is to verify the logical consistency of an ontology by ensuring the existence of at least one model that satisfies its axioms. This process is closely related to the semantic relationships within the ontology because a consistent, semantically meaningful ontology is more likely to be satisfiable and able to provide an accurate representation of the intended domain.

[5]https://swat.cse.lehigh.edu/projects/lubm/

Datasets	economy-inc.			MaasMatch.		
Metric	Precision	Recall	F1-Score	Precision	Recall	F1-Score
GPT3.5–*NI*	100	93.1	96.4	57.6	100	73.1
GPT4o–*NI*	100	89.7	94.5	63.0	76.3	69.0
LLaMA3-8b–*NI*	81.0	58.6	68.0	55.3	55.3	55.3

Table 5: Performances of LLMs in ontology satisfiability checking (%).

Datasets. We build inconsistent DL-Lite ontologies by generating minimal inconsistent subsets (MISs) (Hunter et al., 2008) of existing inconsistent ontologies from (Ji et al., 2014). We choose economy-Inc. and Maa-edas-iasted in (Ji et al., 2014) to generate MISs, because the expressivity of their MISs is close to DL-Lite. We select 29 MISs of economy-Inc. and 38 MISs of Maa-edas-iasted. For each MIS, we randomly delete an axiom to obtain the corresponding consistent ontology.

Experimental setup. The experimental settings are similar to those in syntax checking. We use the prompt-*NI* including task definition (T) and ontology (O). We cover prompts in Appendix C.

Results analysis. From Table 5, we observe that LLMs perform well on economiy-inc., and relatively poor on Maa-edas-iasted, since Maa-edas-iasted is more complex and has more constructors. Overall, LLMs can detect logical inconsistencies in DL-Lite ontologies to some degree. However, this capacity is limited for more complex inconsistent DL ontologies.

5 Conclusion

We have empirically investigated whether LLMs can understand DL-Lite ontologies. Extensive experimental results demonstrated the effectiveness and limitations of LLMs in understanding the syntax and semantics of DL-Lite ontologies. For instance, LLMs possess the ability to understand formal syntax and semantics of concepts, roles and property characteristics. However, LLMs still struggle with understanding TBox NI transitivity rules and handling ontologies with large scale ABoxes.

As future works, we will consider exploring the ability of LLMs to understand ontologies in other lightweight ontology languages, such as $\mathcal{EL}$, and to understand ontologies in intractable ontology languages, such as $\mathcal{ALC}$ and $\mathcal{SHOIQ}$.

Limitations

This work is an empirical study on LLMs' capacity of understanding DL-Lite ontologies, and it has several limitations. Firstly, the size and diversity are limited due to the data sources and costs of LLMs. Secondly, there are various kinds of DLs and we just choose DL-Lite for evaluation. We thus encourage future work to conduct investigations for more DLs. Finally, it still remains unexplored how to improve LLMs' understanding capacity for TBox NI transitivity and large-scale ABox.

References

Anita Bandrowski, Ryan Brinkman, Mathias Brochhausen, Matthew H Brush, Bill Bug, Marcus C Chibucos, Kevin Clancy, Mélanie Courtot, Dirk Derom, Michel Dumontier, et al. 2016. The ontology for biomedical investigations. *PloS one*, 11(4):e0154556.

Guangsheng Bao, Hongbo Zhang, Linyi Yang, Cunxiang Wang, and Yue Zhang. 2024. Llms with chain-of-thought are non-causal reasoners. *arXiv preprint arXiv:2402.16048*.

Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2023. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*.

Jacques Bouaud, Bruno Bachimont, Jean Charlet, and Pierre Zweigenbaum. 1995. Methodological principles for structuring an "ontology". In *Proceedings of the IJCAI'95 Workshop on "Basic Ontological Issues in Knowledge Sharing*, pages 19–25.

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

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

Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, and Riccardo Rosati. 2009. Ontologies and databases: The dl-lite approach. In *Reasoning Web International Summer School*, pages 255–356. Springer.

Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. 2007. Tractable reasoning and efficient query answering in description logics: The dl-lite family. *Journal of Automated reasoning*, 39:385–429.

Meiqi Chen, Yubo Ma, Kaitao Song, Yixin Cao, Yan Zhang, and Dongsheng Li. 2023. Learning to teach large language models logical reasoning. *arXiv preprint arXiv:2310.09158*.

Gene Ontology Consortium. 2004. The gene ontology (go) database and informatics resource. *Nucleic acids research*, 32(suppl_1):D258–D261.

Shaker El-Sappagh, Francesco Franda, Farman Ali, and Kyung-Sup Kwak. 2018. Snomed ct standard ontology based on the ontology for general medical science. *BMC medical informatics and decision making*, 18:1–19.

Chao Feng, Xinyu Zhang, and Zichu Fei. 2023. Knowledge solver: Teaching llms to search for domain knowledge from knowledge graphs. *arXiv preprint arXiv:2309.03118*.

Hans-Georg Fill, Peter Fettke, and Julius Köpke. 2023. Conceptual modeling and large language models: impressions from first experiments with chatgpt. *Enterprise Modelling and Information Systems Architectures (EMISAJ)*, 18:1–15.

Anna Formica. 2006. Ontology-based concept similarity in formal concept analysis. *Information sciences*, 176(18):2624–2641.

Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. 2014. Hermit: an owl 2 reasoner. *Journal of automated reasoning*, 53:245–269.

Jennifer Golbeck and Matthew Rothstein. 2008. Linking social networks on the web with foaf: A semantic web case study. In *AAAI*, volume 8, pages 1138–1143.

Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. 2005. Lubm: A benchmark for owl knowledge base systems. *Journal of Web Semantics*, 3(2-3):158–182.

Benjamin Heinzerling and Kentaro Inui. 2021. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1772–1791, Online. Association for Computational Linguistics.

Martin Hepp. 2008. Goodrelations: An ontology for describing products and services offers on the web. In *Knowledge Engineering: Practice and Patterns: 16th International Conference, EKAW 2008, Acitrezza, Italy, September 29-October 2, 2008. Proceedings 16*, pages 329–346. Springer.

Anthony Hunter, Sébastien Konieczny, et al. 2008. Measuring inconsistency through minimal inconsistent sets. *KR*, 8(358-366):42.

Qiu Ji, Zhiqiang Gao, Zhisheng Huang, and Man Zhu. 2014. Measuring effectiveness of ontology debugging systems. *Knowledge-Based Systems*, 71:169–186.

C Maria Keet, Ronell Alberts, Aurona Gerber, and Gibson Chimamiwa. 2008. Enhancing web portals with ontology-based data access: The case study of south africa's accessibility portal for people with disabilities. In *OWLED*, volume 432.

Man Luo, Shrinidhi Kumbhar, Mihir Parmar, Neeraj Varshney, Pratyay Banerjee, Somak Aditya, Chitta Baral, et al. 2023. Towards logiglue: A brief survey and a benchmark for analyzing logical reasoning capabilities of language models. *arXiv preprint arXiv:2310.00836*.

Li Ma, Yang Yang, Zhaoming Qiu, Guotong Xie, Yue Pan, and Shengping Liu. 2006. Towards a complete owl ontology benchmark. In *The Semantic Web: Research and Applications: 3rd European Semantic Web Conference, ESWC 2006 Budva, Montenegro, June 11-14, 2006 Proceedings 3*, pages 125–139. Springer.

Patricia Mateiu and Adrian Groza. 2023. Ontology engineering with large language models. *2023 25th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pages 226–229.

Boris Motik and Ulrike Sattler. 2006. A comparison of reasoning techniques for querying large description logic aboxes. In *International Conference on Logic for Programming Artificial Intelligence and Reasoning*, pages 227–241. Springer.

Vishwas Mruthyunjaya, Pouya Pezeshkpour, Estevam Hruschka, and Nikita Bhutani. 2023. Rethinking language models as symbolic knowledge graphs. *arXiv preprint arXiv:2308.13676*.

Gábor Nagypál, Richard Deswarte, and Jan Oosthoek. 2005. Applying the semantic web: The vicodi experience in creating visual contextualization for history. *Literary and Linguistic Computing*, 20(3):327–349.

R OpenAI. 2023. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2(5).

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

Héctor Pérez-Urbina, Boris Motik, and Ian Horrocks. 2009. A comparison of query rewriting techniques for dl-lite. *Description Logics*, 477:29.

Yves Raimond and Mark Sandler. 2012. Evaluation of the music ontology framework. In *Extended Semantic Web Conference*, pages 255–269. Springer.

Mariano Rodriguez-Muro, Lina Lubyte, and Diego Calvanese. 2008. Realizing ontology based data access: A plug-in for protégé. In *2008 IEEE 24th International Conference on Data Engineering Workshop*, pages 286–289. IEEE.

Cornelius Rosse and José LV Mejino Jr. 2008. The foundational model of anatomy ontology. In *Anatomy ontologies for bioinformatics: principles and practice*, pages 59–117. Springer.

Chen Shani, Jilles Vreeken, and Dafna Shahaf. 2023. Towards concept-aware large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13158–13170, Singapore. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Yannis Tzitzikas, Carlo Allocca, Chryssoula Bekiari, Yannis Marketakis, Pavlos Fafalios, Martin Doerr, Nikos Minadakis, Theodore Patkos, and Leonardo Candela. 2016. Unifying heterogeneous and distributed information about marine species through the top level ontology marinetlo. *Program*, 50(1):16–40.

Keyu Wang, Guilin Qi, Jiaoyan Chen, and Tianxing Wu. 2024a. Embedding ontologies via incoprorating extensional and intensional knowledge. *arXiv preprint arXiv:2402.01677*.

Siyuan Wang, Zhongyu Wei, Yejin Choi, and Xiang Ren. 2024b. Can llms reason with rules? logic scaffolding for stress-testing and improving llms. *arXiv preprint arXiv:2402.11442*.

Stuart Weibel, John Kunze, Carl Lagoze, and Misha Wolf. 1998. Dublin core metadata for resource discovery. Technical report.

William A Woods. 1975. What's in a link: Foundations for semantic networks. In *Representation and understanding*, pages 35–82. Elsevier.

Yuan Yang, Siheng Xiong, Ali Payani, Ehsan Shareghi, and Faramarz Fekri. 2023. Harnessing the power of large language models for natural language to first-order logic translation. *arXiv preprint arXiv:2305.15541*.

A DL-Lite Transitivity Rules

TBox PI transitivity rules:
$\alpha = C_1 \sqsubseteq C_2, \beta = C_2 \sqsubseteq C_3 \rightarrow \beta_{\text{new}} = C_1 \sqsubseteq C_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_2 \sqsubseteq R_3 \rightarrow \beta_{\text{new}} = R_1 \sqsubseteq R_3$

TBox NI transitivity rules:
$\alpha = C_1 \sqsubseteq C_2, \beta = C_2 \sqsubseteq \neg C_3 \rightarrow \beta_{\text{new}} = C_1 \sqsubseteq \neg C_3$
$\alpha = C_1 \sqsubseteq C_2, \beta = C_3 \sqsubseteq \neg C_2 \rightarrow \beta_{\text{new}} = C_1 \sqsubseteq \neg C_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = \exists R_2 \sqsubseteq \neg C \rightarrow \beta_{\text{new}} = \exists R_1 \sqsubseteq \neg C$
$\alpha = R_1 \sqsubseteq R_2, \beta = C \sqsubseteq \neg \exists R_2 \rightarrow \beta_{\text{new}} = \exists R_1 \sqsubseteq \neg C$
$\alpha = R_1 \sqsubseteq R_2, \beta = C \sqsubseteq \neg \exists R_2^- \rightarrow \beta_{\text{new}} = \exists R_1^- \sqsubseteq \neg C$
$\alpha = R_1 \sqsubseteq R_2, \beta = \exists R_2^- \sqsubseteq \neg C \rightarrow \beta_{\text{new}} = \exists R_1^- \sqsubseteq \neg C$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_2 \sqsubseteq \neg R_3 \rightarrow \beta_{\text{new}} = R_1 \sqsubseteq \neg R_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_3 \sqsubseteq \neg R_2 \rightarrow \beta_{\text{new}} = R_1 \sqsubseteq \neg R_3$
$\alpha = R \sqsubseteq \neg R \rightarrow \beta_{\text{new}_1} = \exists R \sqsubseteq \neg \exists R, \beta_{\text{new}_2} = \exists R^- \sqsubseteq \neg \exists R^-$
$\alpha = \exists R \sqsubseteq \neg \exists R \rightarrow \beta_{\text{new}_1} = R \sqsubseteq \neg R, \beta_{\text{new}_2} = \exists R^- \sqsubseteq \neg \exists R^-$
$\alpha = \exists R^- \sqsubseteq \neg \exists R^- \rightarrow \beta_{\text{new}_1} = R \sqsubseteq \neg R, \beta_{\text{new}_2} = \exists R \sqsubseteq \neg \exists R$

ABox transitivity rules:
$\alpha = C_1 \sqsubseteq C_2, \beta = C_1(a) \rightarrow \beta_{\text{new}} = C_1(a)$
$\alpha = C \sqsubseteq \exists R, \beta = C(a) \rightarrow \beta_{\text{new}} = R(a, a_{\text{new}})$
$\alpha = \exists R \sqsubseteq C, \beta = R(a, a') \rightarrow \beta_{\text{new}} = C(a)$
$\alpha = \exists R_1 \sqsubseteq \exists R_2, \beta = R_1(a, a') \rightarrow \beta_{\text{new}} = R_2(a, a_{\text{new}})$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_1(a, a') \rightarrow \beta_{\text{new}} = R_2(a, a')$

We refer to Section 3.1 in (Calvanese et al., 2007) for detailed illustrations and examples about these transitivity roles.

B Typical DL-Lite Syntax Errors

	Common Syntax Errors in DL-Lite	Examples
Invalid inverse	Inverse operator on a concept	$Professor^-$
	Misplaced inverse operator	$^-TeachesTo$
	Inverse operator on a quantifier	$\exists^-$
Invalid quantifiers	Misplaced quantifiers	$TeachesTo\exists$
	Quantifiers with concept following	$\exists Professor$
	Quantifiers missing role following	$\exists$
	Redundant multiple quantifiers	$\exists\exists TeachesTo$
Invalid negation	Misplaced negation operator	$Professor\neg$
	Negation without anything following	$\neg$
Invalid conjunction	Conjoining incomplete concepts	$Professor\sqcap$
	Conjoining a concept with a role	$Professor \sqcap TeachesTo$
	Conjoining roles directly	$TeachesTo \sqcap HasTutor$
	Missing conjunction operator	$Professor\exists TeachesTo$
	Misplaced conjunction operator	$\sqcap Professor\exists TeachesTo$

C Prompts And Answer Examples

All symbols and constructors in the prompts can be input into LLMs, but only one kind of font can be input into LLMs (Colors and italics are only for display convenience).

Prompt-*NI* for syntax checking:

Task Description:
There are some DL-Lite axioms, and your task is to determine whether the syntax of each of these axioms is correct.

Given DL-Lite Axioms:
MaterialEntity $\sqsubseteq$ ¬PhysicalObject
∃hasPerformer¬ $\sqsubseteq$ MusicalExpression
Investigation $\sqsubseteq$ ∃hasPart
Protocol $\sqsubseteq$ ¬Investigation¬
(··· more context here ···)

Answer:

Prompt-*WI* for syntax checking:

Task Description:
There are some DL-Lite axioms, and your task is to determine whether the syntax of each of these axioms is correct.
DL-Lite$_{core}$ concepts and roles are defined as follows:
$B ::= A \mid \exists R \mid \exists R^{-} \quad R ::= P \mid P^{-}$
$C ::= B \mid \neg B \mid C_1 \sqcap C_2 \quad E ::= R \mid \neg R$
where A denotes an atomic concept, P denotes an atomic role, and P^{-} denotes the inverse of the atomic role P. We call B, R, C, E a basic concept, a basic role, a general concept and a general role respectively.
A DL-Lite$_{core}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. $\mathcal{T}$ is formed by a finite set of concept inclusion assertions of the form $B \sqsubseteq$ C. $\mathcal{A}$ is formed by a finite set of membership assertions on atomic concepts and on atomic roles, of the form $A(a)$ and $P(a, b)$. DL-Lite$_{\mathcal{R}}$ extends DL-Lite$_{\text{core}}$ with role inclusion assertions of the form $R \sqsubseteq E$ and DL-Lite$_{\mathcal{F}}$ extends DL-Lite$_{core}$ with functionality on roles or on their inverses of the form (Funct R).

Given DL-Lite Axioms:
MaterialEntity $\sqsubseteq$ ¬PhysicalObject
∃hasPerformer¬ $\sqsubseteq$ MusicalExpression
Investigation $\sqsubseteq$ ∃hasPart
Protocol $\sqsubseteq$ ¬Investigation¬
(··· more context here ···)

Answer:

Prompt-*WIE* for syntax checking:

Task Description:
There are some DL-Lite axioms, and your task is to determine whether the syntax of each of these axioms is correct.
DL-Lite$_{core}$ concepts and roles are defined as follows:
$B ::= A \mid \exists R \mid \exists R^{-} \quad R ::= P \mid P^{-}$
$C ::= B \mid \neg B \mid C_1 \sqcap C_2 \quad E ::= R \mid \neg R$
where A denotes an atomic concept, P denotes an atomic role, and P^{-} denotes the inverse of the atomic role P. We call B, R, C, E a basic concept, a basic role, a general concept and a general role respectively.
A DL-Lite$_{core}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. $\mathcal{T}$ is formed by a finite set of concept inclusion assertions of the form $B \sqsubseteq$ C. $\mathcal{A}$ is formed by a finite set of membership assertions on atomic concepts and on atomic roles, of the form $A(a)$ and $P(a, b)$. DL-Lite$_{\mathcal{R}}$ extends DL-Lite$_{\text{core}}$ with role inclusion assertions of the form $R \sqsubseteq E$ and DL-Lite$_{\mathcal{F}}$ extends DL-Lite$_{core}$ with functionality on roles or on their inverses of the form (Funct R).
Here are some examples of common syntactic errors:
$^{-}TeachesTo$ is incorrect, for the misplaced inverse operator; $\exists^{-}$ is incorrect, for the inverse operator applied to a quantifier; $\exists Professor$ is incorrect, for the quantifier with a concept following; $\exists$ is incorrect, for the quantifier missing a role following; $\exists\exists TeachesTo$ is incorrect, for the redundant multiple quantifiers; $TeachesTo\exists$ is incorrect, for the misplaced quantifiers; $Professor\neg$ is incorrect, for the misplaced negation operator; $Professor\sqcap$ is incorrect, for conjoining incomplete concepts; $Professor \sqcap TeachesTo$ is incorrect, for conjoining a concept with a role; $TeachesTo \sqcap HasTutor$ is incorrect, for conjoining roles directly; $Professor\exists TeachesTo$ is incorrect, for missing conjunction operator.

Given DL-Lite Axioms:
MaterialEntity $\sqsubseteq$ ¬PhysicalObject
∃hasPerformer¬ $\sqsubseteq$ MusicalExpression

Investigation $\sqsubseteq$ ∃hasPart
Protocol $\sqsubseteq$ ¬Investigation¬
(⋯ more context here ⋯)

Answer:

Sample answer for syntax checking:

(⋯ more context here ⋯)
Now, let's analyze each given DL-Lite axiom for syntactic correctness:

1. MaterialEntity $\sqsubseteq$ ¬ PhysicalObject
This axiom states that MaterialEntity is a subclass of the negation of PhysicalObject.
This is syntactically correct.

2. ∃hasPerformer¬ $\sqsubseteq$ Performance ⊓ MusicalExpression
Here, ∃hasPerformer¬ seems to be intended as an existential quantification, but the syntax ¬ is misplaced. Correct form should be something like $\exists hasPerformer$ $\sqsubseteq$ Performance ⊓ MusicalExpression.
Therefore, this is syntactically incorrect due to the misplacement of the negation.
(⋯ more context here ⋯)

Prompt-*NI* for subsumption of concepts or roles:

Task Description:
There are a DL-Lite ontology and some logical implications, and your task is to determine whether each of these logical implications can be deduced from the given ontology.

Given Ontology :
Ability $\sqsubseteq$ ¬Disability
Ability $\sqsubseteq$ ¬Device
Ability $\sqsubseteq$ ∃isAssistedBy
(⋯ more context here ⋯)

Logical Implications:
Achondroplasia $\sqsubseteq$ PhysicalDisability
Amputation $\sqsubseteq$ PhysicalDisability
AssistiveDevice $\sqsubseteq$ Device
Autism $\sqsubseteq$ MentalDisability
(⋯ more context here ⋯)

Answer:

Prompt-*WI* for subsumption of concepts or roles:

Task Description:
There are a DL-Lite ontology and some logical implications, and your task is to determine whether each of these logical implications can be deduced from the given ontology.
Here, you are provided with some reasoning rules:
$\alpha = C_1 \sqsubseteq C_2, \beta = C_2 \sqsubseteq C_3 \rightarrow \beta_{\text{new}} = C_1 \sqsubseteq C_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_2 \sqsubseteq R_3 \rightarrow \beta_{\text{new}} = R_1 \sqsubseteq R_3$
$\alpha = C_1 \sqsubseteq C_2, \beta = C_2 \sqsubseteq \neg C_3 \rightarrow \beta_{\text{new}} = C_1 \sqsubseteq \neg C_3$
$\alpha = C_1 \sqsubseteq C_2, \beta = C_3 \sqsubseteq \neg C_2 \rightarrow \beta_{new} = C_1 \sqsubseteq \neg C_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = \exists R_2 \sqsubseteq \neg C \rightarrow \beta_{\text{new}} = \exists R_1 \sqsubseteq \neg C$
$\alpha = R_1 \sqsubseteq R_2, \beta = C \sqsubseteq \neg \exists R_2 \rightarrow \beta_{\text{new}} = \exists R_1 \sqsubseteq \neg C$
$\alpha = R_1 \sqsubseteq R_2, \beta = C \sqsubseteq \neg \exists R_2^- \rightarrow \beta_{\text{new}} = \exists R_1^- \sqsubseteq \neg C$
$\alpha = R_1 \sqsubseteq R_2, \beta = \exists R_2^- \sqsubseteq \neg C \rightarrow \beta_{\text{new}} = \exists R_1^- \sqsubseteq \neg C$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_2 \sqsubseteq \neg R_3 \rightarrow \beta_{\text{new}} = R_1 \sqsubseteq \neg R_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_3 \sqsubseteq \neg R_2 \rightarrow \beta_{\text{new}} = R_1 \sqsubseteq \neg R_3$
$one\ of\ the\ assertions\ R \sqsubseteq \neg R, \exists R \sqsubseteq \neg \exists R, \exists R^- \sqsubseteq \neg \exists R^- \rightarrow the\ other\ two$

Given Ontology :
Ability $\sqsubseteq$ ¬Disability
Ability $\sqsubseteq$ ¬Device
Ability $\sqsubseteq$ ∃isAssistedBy
(⋯ more context here ⋯)

Logical Implications:
Achondroplasia $\sqsubseteq$ PhysicalDisability
Amputation $\sqsubseteq$ PhysicalDisability
AssistiveDevice $\sqsubseteq$ Device
Autism $\sqsubseteq$ MentalDisability
(⋯ more context here ⋯)

Answer:

Prompt-*WIE* for subsumption of concepts or roles:

Task Description:
There are a DL-Lite ontology and some logical implications, and your task is to determine whether

each of these logical implications can be deduced from the given ontology.
Here, you are provided with some reasoning rules:
$\alpha = C_1 \sqsubseteq C_2, \beta = C_2 \sqsubseteq C_3 \rightarrow \beta_{\text{new}} = C_1 \sqsubseteq C_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_2 \sqsubseteq R_3 \rightarrow \beta_{\text{new}} = R_1 \sqsubseteq R_3$
$\alpha = C_1 \sqsubseteq C_2, \beta = C_2 \sqsubseteq \neg C_3 \rightarrow \beta_{\text{new}} = C_1 \sqsubseteq \neg C_3$
$\alpha = C_1 \sqsubseteq C_2, \beta = C_3 \sqsubseteq \neg C_2 \rightarrow \beta_{new} = C_1 \sqsubseteq \neg C_3$
$\alpha = R_1 \sqsubseteq R_2, \beta = \exists R_2 \sqsubseteq \neg C \rightarrow \beta_{\text{new}} = \exists R_1 \sqsubseteq \neg C$
$\alpha = R_1 \sqsubseteq R_2, \beta = C \sqsubseteq \neg \exists R_2 \rightarrow \beta_{\text{new}} = \exists R_1 \sqsubseteq \neg C$
(··· more context here ···)
Here are some examples:
If HasParent $\sqsubseteq$ HasAncestor and Mortal $\sqsubseteq$ ¬∃HasAncestor, then ∃HasParent $\sqsubseteq$ ¬Mortal.
If HasParent $\sqsubseteq$ HasAncestor and Immortal $\sqsubseteq$ ¬∃hasAncestor$^-$, then ∃hasParent$^-$ $\sqsubseteq$ ¬Immortal.
If HasParent $\sqsubseteq$ HasAncestor and ∃ hasAncestor$^-$ $\sqsubseteq$ ¬immortal, then ∃hasParent$^-$ $\sqsubseteq$ ¬Immortal. If HasParent $\sqsubseteq$ HasAncestor and HasAncestor $\sqsubseteq$ ¬HasSibling, then HasParent $\sqsubseteq$ ¬HasSibling.
(··· more context here ···)

Given Ontology :
Ability $\sqsubseteq$ ¬Disability
Ability $\sqsubseteq$ ¬Device
Ability $\sqsubseteq$ ∃isAssistedBy
(··· more context here ···)

Logical Implications:
Achondroplasia $\sqsubseteq$ PhysicalDisability
Amputation $\sqsubseteq$ PhysicalDisability
AssistiveDevice $\sqsubseteq$ Device
Autism $\sqsubseteq$ MentalDisability
(··· more context here ···)

Answer:

Sample answer for subsumption of concepts or roles:

(··· more context here ···)
1. Professor $\sqsubseteq$ Staff
Professor is a subclass of Academic_Staff and Academic_Staff is a subclass of Staff in the ontology. So, this implication holds.

2. BatteryPowered_Wheelchair $\sqsubseteq$ Motorised_Wheelchair
There's no direct assertion about BatteryPowered_Wheelchair being a subclass of Motorised_Wheelchair in the TBox. Therefore, this implication cannot be deduced.

(··· more context here ···)

Prompt-*NI* for instance checking:

Task Description:
There are a DL-Lite ontology and some logical implications, and your task is to determine whether each of these logical implications can be deduced from the given ontology.

Given Ontology :
(··· more context here ···)
AssistantProfessor(AssistantProfessor0)
SportsFan(AssistantProfessor0)
(··· more context here ···)

Logical Implications:
Man(AssistantProfessor0)
SportsLover(AssistantProfessor0)
(··· more context here ···)

Answer:

Prompt-*WI* for instance checking:

Task Description:
There are a DL-Lite ontology and some logical implications, and your task is to determine whether each of these logical implications can be deduced from the given ontology.
Here, you are provided with some reasoning rules:
$\alpha = C_1 \sqsubseteq C_2, \beta = C_1(a) \rightarrow \beta_{\text{new}} = C_1(a)$
$\alpha = C \sqsubseteq \exists R, \beta = C(a) \rightarrow \beta_{\text{new}} = R(a, a_{\text{new}})$
$\alpha = \exists R \sqsubseteq C, \beta = R(a, a') \rightarrow \beta_{\text{new}} = C(a)$
$\alpha = \exists R_1 \sqsubseteq \exists R_2, \beta = R_1(a, a') \rightarrow \beta_{\text{new}} = R_2(a, a_{\text{new}})$
$\alpha = R_1 \sqsubseteq R_2, \beta = R_1(a, a') \rightarrow \beta_{\text{new}} = R_2(a, a')$

Given Ontology :
AssistantProfessor(AssistantProfessor0)
SportsFan(AssistantProfessor0)

(··· more context here ···)

Logical Implications:
Man(AssistantProfessor0)
SportsLover(AssistantProfessor0)
(··· more context here ···)

Answer:

Prompt-*WIE* for instance checking:

Task Description:
There are a DL-Lite ontology and some logical implications, and your task is to determine whether each of these logical implications can be deduced from the given ontology.
Here, you are provided with some reasoning rules:
$\alpha = C_1 \sqsubseteq C_2, \beta = C_1(a) \rightarrow \beta_{\text{new}} = C_1(a)$
(··· more context here ···)
Here are examples:
If Human $\sqsubseteq$ Animal and Human(John), then Animal(John)
If Human $\sqsubseteq$ ∃hasParent and Human(John), then hasParent(John, _).
If ∃hasChild $\sqsubseteq$ Parent and HasChild(John, _), then Parent(John).
If ∃hasChild $\sqsubseteq$ ∃hasDescendant and HasChild(John, _), then hasDescendant(John, _).
If HasParent $\sqsubseteq$ HasRelative and HasParent(Mary, John), then HasRelative(Mary, John).

Given Ontology :
AssistantProfessor(AssistantProfessor0)
SportsFan(AssistantProfessor0)
(··· more context here ···)

Logical Implications:
Man(AssistantProfessor0)
SportsLover(AssistantProfessor0)
(··· more context here ···)

Answer:

Sample answer for instance checking:

(··· more context here ···)
1. Man(AssistantProfessor0) - True, since AssistantProfessor0 is explicitly stated to be a Man.
(··· more context here ···)

Prompt for property characteristic probing:

Task Description:
There are a DL-Lite ontology and some logical implications, and your task is to determine whether each of these logical implications can be deduced from the given ontology. Give reasons or inferring process.

Given Ontology :
(··· more context here ···)
$\text{WorksIn}^- \sqsubseteq$ Employs
Employs $\sqsubseteq \text{WorksIn}^-$
(funct WorksIn)
WorksIn(John, Google)
WorksIn(John, x)
(··· more context here ···)

Logical Implications:
Employs(Google, John)
x ≡ John
(··· more context here ···)

Answer:

Sample answer for property characteristic probing:

(··· more context here ···)

1. Employs(Google, John) - True, Since WorksIn(John, Google) is given and Employs is the inverse of WorksIn, hence Employs(Google, John) can be deduced.

2. x ≡ John - True, Given: WorksIn(John, Google) and WorksIn(John, x). Since WorksIn is a functional property, John can only work at one company. Thus, x must be Google to satisfy the functional constraint.

(··· more context here ···)

Prompt for query answering:

Task Description:
There are a DL-Lite ontology and a query, and your task is to answer the query over the given DL-Lite ontology. Because ontology is relatively large, it will be entered in several times.

Given Ontology :

(⋯ more context here ⋯)
PhDStudent $\sqsubseteq$ Student
Student $\sqsubseteq \neg\exists$hasStaffID
$\exists$hasStaffID^{-} $\sqsubseteq$ ID
Student $\sqsubseteq \neg$ID
PhdStudent(John)
(⋯ more context here ⋯)

Queries:
Q1(x) $\leftarrow$ Student(x)
(⋯ more context here ⋯)

Answer:

Sample answer for query answering:

(⋯ more context here ⋯)

1. From axiom 5, we know PhDStudent(John). From axiom 1, we have PhDStudent $\sqsubseteq$ Student. This means every PhDStudent is a Student. Therefore, PhDStudent(John) implies Student(John). The answer is q(John).

(⋯ more context here ⋯)

Prompt for ontology satisfiability checking:

Task Description:
There is a DL-Lite ontology, and your task is to determine whether the ontology is satisfiable, i.e., iff it has at least one model.

Given Ontology :
(⋯ more context here ⋯)
MasterStudent $\sqsubseteq$ Student
MasterStudent $\sqsubseteq$ Employee
Student $\sqsubseteq \neg$ Employee
MasterStudent(John)
(⋯ more context here ⋯)

Answer:

Sample answer for ontology satisfiability checking:

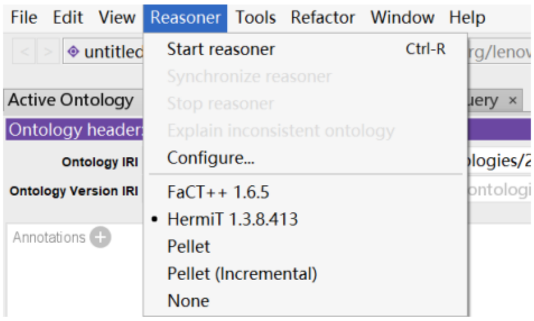
(⋯ more context here ⋯)

The axioms lead to a logical inconsistency regarding the concept of MasterStudent MasterStudent. Therefore, the given DL-Lite ontology is not satisfiable.

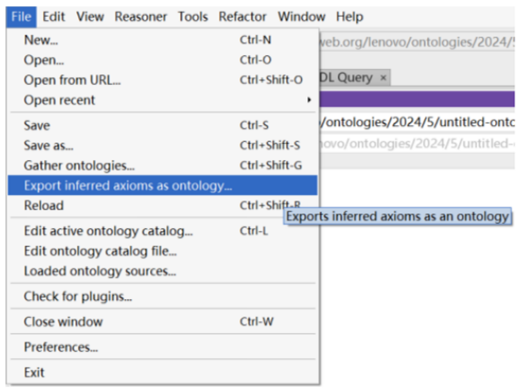
(⋯ more context here ⋯)

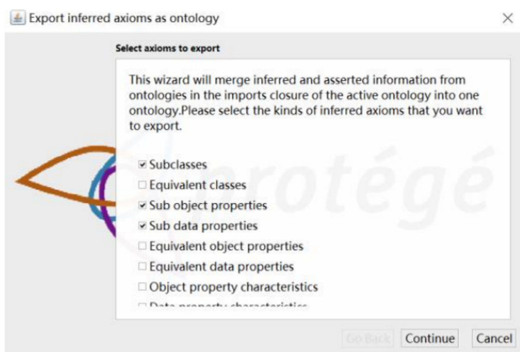
D Instructions about Protégé

We import the selected ontological datasets into Protégé and utilize the reasoning engine HermiT 1.3.8.413 to infer logical implications.



Then we export the inferred axioms. For subsumption of concepts or roles, the chosen categories of inferred axioms exported are subclasses, sub object properties, and sub data properties. For instance checking, the chosen categories of inferred axioms exported are class assertions and property assertions





DL-Lite Ontology	**Logical Implications**
Case 1: TBox = { $C_1 \sqsubseteq C_2$, $C_2 \sqsubseteq \neg C_3$, $C_4 \sqsubseteq \neg C_2$, $R_1 \sqsubseteq R_2$, $\exists R_2 \sqsubseteq \neg C_5$, $C_6 \sqsubseteq \neg \exists R_2$, $R_3 \sqsubseteq R_4$, $\exists R_4^- \sqsubseteq \neg C_7$, $C_8 \sqsubseteq \neg \exists R_4^-$, $R_5 \sqsubseteq R_6$, $R_6 \sqsubseteq \neg R_7$, $R_8 \sqsubseteq \neg R_6$ }	$C_1 \sqsubseteq \neg C_3$, $C_1 \sqsubseteq \neg C_4$, $\exists R_1 \sqsubseteq \neg C_5$, $\exists R_1 \sqsubseteq \neg C_6$, $\exists R_3^- \sqsubseteq \neg C_7$, $\exists R_3^- \sqsubseteq \neg C_8$, $R_5 \sqsubseteq \neg R_7$, $R_5 \sqsubseteq \neg R_8$
Case 2: TBox = { $C_1 \sqsubseteq C_2$, $C_1 \sqsubseteq C_3$, $C_2 \sqsubseteq C_4$, $R_1 \sqsubseteq R_2$, $R_3 \sqsubseteq R_4$, $C_5 \sqsubseteq \exists R_5$, $\exists R_6 \sqsubseteq C_6$, $\exists R_7 \sqsubseteq \exists R_8$}; ABox = { $C_1(a)$, $C_1(b)$, $R_1(c,d)$, $R_3(e,f)$, $C_5(a)$, $R_6(a,k)$, $R_7(g,h)$}	$C_2(a)$, $C_3(a)$, $C_2(b)$, $C_3(b)$, $R_2(c,d)$, $R_4(e,f)$, $C_4(a)$, $C_4(b)$, $R_5(a,\_)$, $C_6(a)$, $R_8(h,\_)$
Case 3: TBox = {$C_1 \sqsubseteq C_2$, $C_1 \sqsubseteq C_4$, $C_1 \sqsubseteq C_6$, $C_2 \sqsubseteq C_3$, $C_4 \sqsubseteq \neg C_5$, $C_7 \sqsubseteq \neg C_6$, $R_1 \sqsubseteq R_2$, $R_4 \sqsubseteq R_5$, $R_6 \sqsubseteq R_7$, $R_2 \sqsubseteq R_3$, $\exists R_2 \sqsubseteq \neg C_8$, $C_9 \sqsubseteq \neg \exists R_2$, $C_{10} \sqsubseteq \neg \exists R_5^-$, $\exists R_5^- \sqsubseteq \neg C_{11}$, $R_7 \sqsubseteq \neg R_8$, $R_9 \sqsubseteq \neg R_7$, $R_{10} \sqsubseteq \neg R_{10}$, $\exists R_{11} \sqsubseteq \neg \exists R_{11}$, $\exists R_{12}^- \sqsubseteq \neg \exists R_{12}^-$}	$C_1 \sqsubseteq C_3$, $C_1 \sqsubseteq \neg C_5$, $C_1 \sqsubseteq \neg C_7$, $R_1 \sqsubseteq R_3$, $\exists R_1 \sqsubseteq \neg C_8$, $\exists R_1 \sqsubseteq \neg C_9$, $\exists R_4^- \sqsubseteq \neg C_{10}$, $\exists R_4^- \sqsubseteq \neg C_{11}$, $R_6 \sqsubseteq \neg R_8$, $R_6 \sqsubseteq \neg R_9$, $\exists R_{10} \sqsubseteq \neg \exists R_{10}$, $\exists R_{10}^- \sqsubseteq \neg \exists R_{10}^-$, $R_{11} \sqsubseteq \neg R_{11}$, $\exists R_{11}^- \sqsubseteq \neg \exists R_{11}^-$, $R_{12} \sqsubseteq \neg R_{12}$, $\exists R_{12} \sqsubseteq \neg \exists R_{12}$.
Case 4: TBox = { $C_1 \sqsubseteq C_2$, $C_1 \sqsubseteq \exists R_1$, $\exists R_2 \sqsubseteq C_3$, $\exists R_3 \sqsubseteq \exists R_4$, $R_5 \sqsubseteq R_5$}; ABox = {$C_1(a)$, $C_1(b)$, $R_2(c,d)$, $R_3(e,f)$, $R_5(g,h)$}	$C_2(a)$, $R_1(b,\_)$, $C_3(c)$, $R_4(e,\_)$, $R_6(g,h)$.
Case 5: TBox = {$C_1 \sqsubseteq C_2$, $C_2 \sqsubseteq C_3$, $C_3 \sqsubseteq C_4$, $C_4 \sqsubseteq C_5$, $C_3 \sqsubseteq C_6$, $C_6 \sqsubseteq C_7$, $R_1 \sqsubseteq R_2$, $R_2 \sqsubseteq R_3$, $R_3 \sqsubseteq R_4$}	$C_1 \sqsubseteq C_3$, $C_1 \sqsubseteq C_4$, $C_1 \sqsubseteq C_5$, $C_1 \sqsubseteq C_6$, $C_1 \sqsubseteq C_7$, $C_2 \sqsubseteq C_3$, $C_2 \sqsubseteq C_4$, $C_2 \sqsubseteq C_5$, $C_2 \sqsubseteq C_6$, $C_2 \sqsubseteq C_7$, $C_3 \sqsubseteq C_5$, $C_3 \sqsubseteq C_6$, $C_3 \sqsubseteq C_7$, $R_1 \sqsubseteq R_3$, $R_1 \sqsubseteq R_4$, $R_2 \sqsubseteq R_4$.

Table 6: Handcrafted ontologies in case study of transitivity rules.

Data Sources	#ax.	#as.	#inv.	#fun.	#inv. fun.	#impli.inv.	#impli.fun.
Academic Hierarchy	36	120	6	3	1	12	12
E-Commerce System	32	51	4	2	1	8	6
Library System	22	70	3	0	3	7	6
Social Network Relations	29	102	3	1	4	5	3
Medical Medical Relationships	16	21	3	1	0	13	4

Table 7: Statistics about data sources for property characteristics probing. # denotes "the number of ", and ax., as., inv., fun., inv. fun., impli., impli. fun. denote class axioms, class assertions, inverse roles, functional roles, inverse functional roles, logical implications for inverse roles, logical implication for functional roles.

E Ontologies in Case Study of Transitivity Rules

Table 6 demonstrate the handcrafted ontologies in case study of transitivity rules.

F Data Stastistics for Property Characteristic Probing

Table 7 demonstrate statistics about data sources for property characteristics probing.