TU WIEN Informatics

Leveraging Ontologies for Flexible Access to Graph-structured Data

DISSERTATION

zur Erlangung des akademischen Grades

Doktorin der Technischen Wissenschaften

eingereicht von

Medina Andreşel, MSc
Matrikelnummer 1428616

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Assistant Prof. Dr.techn. Magdalena Ortiz
Zweitbetreuung: Privatdoz. Dr.techn. Mantas Šimkus

Diese Dissertation haben begutachtet:

Assoc. Prof. Antonella Poggi	Assoc. Prof. Martin Homola

Wien, 30. Jänner 2024

Medina Andreşel

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Leveraging Ontologies for Flexible Access to Graph-structured Data

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktorin der Technischen Wissenschaften

by

Medina Andreşel, MSc
Registration Number 1428616

to the Faculty of Informatics

at the TU Wien

Advisor: Assistant Prof. Dr.techn. Magdalena Ortiz
Second advisor: Privatdoz. Dr.techn. Mantas Šimkus

The dissertation has been reviewed by:

Assoc. Prof. Antonella Poggi

Assoc. Prof. Martin Homola

Vienna, 30th January, 2024

Medina Andreşel

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Medina Andreşel, MSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 30. Jänner 2024

Medina Andreşel

To my family

Acknowledgements

This thesis is the result of many years of hard work and perseverance. Reflecting on this journey, I have learned so many new things that have shaped the person I am today, the most important lesson is to keep following my path and my dreams no matter the challenge. I would like to take the opportunity to thank the people that inspired and helped me in reaching the finish line.

First and foremost, I would like to thank my PhD adviser Magdalena Ortiz for her guidance, patience and support. Her feedback and advice have been instrumental in shaping my scientific growth and I could not have asked for a better mentor and role model. My gratitude also goes to my co-adviser Mantas Šimkus for his support and always having that insightful idea or comment which turned out to be most of the times crucial. I would also like to thank Yazmín Ibáñez-García for her invaluable support and guidance. I am also very grateful to Daria Stepanova and Trung-Kien Tran for the opportunity to do a PhD Sabattical at Bosch Research Center for AI which has been a transformative experience for my scientific career. This thesis would have not been possible without any of them and I am profoundly grateful for the immeasurable contributions they have made to my development.

My sincere gratitude also goes to the PhD thesis committee members - Antonella Poggi and Martin Homola for their dedication and insightful feedback that has positively impacted the comprehensibility of this thesis. I would also like to express my sincere gratitude to Juan L. Reutter for facilitating my research stay at PUC Chile where I had the opportunity to learn about novel topics, attend workshops and seminars and have so many interesting discussions. Special thanks also go to my other co-authors Pasquale Minervini, Csaba Domokos, Julien Corman and Ognjen Savković for the very fruitful collaborations. I would also like to thank my undergraduate mentor Laura Dioşan who has firstly nurtured my interest in artificial intelligence.

During my PhD studies I had the pleasure to work with so many incredible people who have inspired me every day. My gratitude also goes to my colleagues from the Institute of Logic and Computation from TU Wien for providing the necessary environment for my PhD study. Special thanks to the administrative staff - Beatrix Buhl, Juliane Auerböck and Eva Nedoma for their assistance in navigating all the bureaucracy.

It was a pleasure to share this experience with my PhD colleagues from the Doctoral Program Logic in Computer Science (LogiCS) to which I thank for all the interesting discussions, coffee breaks, culinary experiences and hiking trips. In particular, many thanks to Tobias Kaminski, Ilina Stoilkovska, Anna Lukina, Emir Demirović, Adrian Rebola-Pardo, Matthias Schleipfer,

Mihaela Rozman, Adrian Haret, Jan Maly, Ana Costa, and Jure Kukovec. This journey would have been far less enjoyable without all these nice moments together.

I could not thank enough my "Unicorns" friends, Ilina Stoikovska, Tobias Kaminski, Alina Alexandrova and Serge Stratan for their friendship, kindness and constant support. Our friendship helped me in navigating the most challenging moments and for that and many more I am forever grateful. I would also like to thank my dear flatmates Juli and Lena Nemestothy, Alexandra Langer and Anna Rigoni for all the amazing WG experiences, their kind support and encouragements. I am also grateful to Camelia Georgiu for her positivity, encouragements and interesting discussions. To my "Cluj family" - Alina Moldovan, Maria Andrei and Ligia Bildea for always being close even when far.

To Dominic Staschitz for sailing into my life when unexpected and offering me so many happy moments together. His encouragements and support have been crucial in finalizing this thesis. I am also grateful to my AIT colleagues Pietro Saggese, Mina Schütz and Ana Jalali for allowing our collegiality to turn into a friendship.

Last but not least, to my family without whom any of this would have not been possible. To my sister Violeta Andreşel for always being my rock, my best friend and my partner in crime for my whole life, I could not express in words how much that meant for me. To my parents Maria and Petru Andreşel for their unconditional love, support and all the sacrifices they had to make for me to be where I am today. To my uncles Marian and Gheorghe Jula, my aunt Viorica Andreşel as well as to the rest of my family, I am grateful to have them always by my side.

Finally, I would also like to acknowledge the Austrian Science Fund (FWF) for funding the LogiCS doctoral program I was part of and for directly supporting the research presented in this thesis through the projects P30360, P30873 and W1255. I do not take this chance for granted.

Kurzfassung

Wissensgraphen (engl. Knowledge Graphs, KGs) sind Datensätze, die aus miteinander verbundenen, gekennzeichneten Entitäten in einem teilbaren Format bestehen und sowohl für Menschen als auch Maschinen verwendbar gemacht werden sollen. Anwendungen von KGs umfassen unter anderem Natural Language Question Answering, Web-Suche, Datenanalyse und Expertensysteme. Eine wichtige Aufgabe bei vielen KG-Anwendungen ist die Beantwortung von Abfragen, d.h. das Problem, alle möglichen Antworten auf eine bestimmte Abfrage in einer konkreten Abfragesprache wie den üblichen konjunktiven Abfragen (engl. conjunctive queries, CQs) zu finden. Obwohl viel Aufmerksamkeit der effizienten Abrufung von Antworten für verschiedene Abfragesprachen gewidmet wurde, gehen die meisten Arbeiten davon aus, dass eine ideale Abfrage existiert, die die Antworten, die der Benutzer im Sinn hat, genau erfasst, und dass der Abfrageformulierungsschritt, der entscheidend und für Endbenutzer potenziell herausfordernd ist, als selbstverständlich angesehen wird. Darüber hinaus wird auch implizit angenommen, dass die Informationsbedürfnisse des Benutzers klar definiert und statisch sind, was oft nicht der Fall ist, insbesondere wenn auf KGs zugegriffen wird, die Daten aus umfangreichen und unbekannten Domänen enthalten.

Ontologien sind logische Formalismen, die verwendet werden, um KGs mit menschlicher Expertise und Allgemeinwissen anzureichern. Sie wurden erfolgreich eingesetzt, um die Abfrageformulierung zu unterstützen, mehrere und heterogene Datenquellen zu integrieren und die Abrufung von vollständigeren und informativeren Antworten auf Anfragen zu ermöglichen. Bestehende ontologiebasierte Abfrage-Technologien sind jedoch unzureichend für Szenarien, die einen inhärent interaktiven Abfrageprozess erfordern, der die Exploration von Daten unterstützt: Ausgehend von einer anfänglichen Abfrage werden Benutzer dabei unterstützt, diese schrittweise zu verfeinern, um ihre Informationsbedürfnisse genau zu erfassen. Darüber hinaus muss jede Reformulierung für die interaktive Abfrage effizient und dynamisch ausgewertet werden. Aktuelle Abfrageantwortmaschinen unterstützen nur die Auswertung jeweils einer Abfrage gleichzeitig, und die Abrufung der vollständigen Menge von Antworten auf jede Abfrage ist mit einem hohen Rechenaufwand verbunden. Um diesen Herausforderungen zu begegnen, konzentriert sich diese Arbeit auf die Ontologiesprache $DL\text{-}Lite_{\mathcal{A}}$, die ein gutes Verhältnis zwischen der Komplexität der Schlussfolgerungen und der Ausdrucksstärke bietet.

Der erste Hauptbeitrag dieser Arbeit besteht darin, einen explorativen Rahmen vorzuschlagen, der Ontologien nutzt, um die folgenden Fähigkeiten zu unterstützen: a) die Formulierung von Abfragen, die es dem Benutzer ermöglicht, die Informationsbedürfnisse durch Verfassen einer Abfragenvorlage zu approximieren, die eine große Menge semantisch verwandter Abfragen

prägnant beschreibt, b) die effiziente Abrufung vollständiger Antworten für die große Menge verwandter Abfragen, c) die dynamische Abfrageverfeinerung, die die interaktive Exploration von Daten ermöglicht.

Unsere Techniken für die Ontologie-getriebene Abfrage-Reformulierung erfordern eine Erweiterung von *DL-Lite*$_{\mathcal{A}}$ um komplexe Rolleneinschlüsse. Als zweiter Hauptbeitrag präsentieren wir ein vollständiges Komplexitätsbild mehrerer *DL-Lite*$_{\mathcal{A}}$-Erweiterungen, die wir während unserer Suche identifiziert haben, um dessen positive Berechnungseigenschaften zu erhalten. Wir betrachten auch die sichere Integration von Aggregation in sowohl Ontologie als auch Abfragesprache, um begrenzte Datenanalyse zu unterstützen.

Um die inhärente Unvollständigkeit von KGs anzugehen, schlagen wir als drittes Hauptergebnis zwei Techniken vor, um flexible Abfragen von unvollständigen KGs angereichert mit Ontologien zu unterstützen. Ein erster solcher Ansatz ist die Annahme-basierte Abfragebeantwortung, bei der Abfragen mit Annahmemustern ausgestattet werden, die dazu dienen, mehrere hypothetische Erweiterungen des KGs zu beschreiben und informativere Antworten über alle solchen Erweiterungen zu konstruieren. Eine Antwort ist in diesem Fall nicht nur ein Kandidaten-Entitäten-Tupel, sondern stattdessen eine bedingte Antwort, die ein solches Tupel mit den Annahmen paart, die das Tupel zu einer wahren bestimmten Antwort machen. Wir zeigen, dass die Annahme-basierte Abfragebeantwortung in der Datenkomplexität berechenbar ist und schlagen ontologiebasierte Umschreibungstechniken vor, um die bedingten Antworten zu konstruieren, auch in Gegenwart von geschlossenen Prädikaten, einer Form von Vollständigkeitserklärungen zu bestimmten Beziehungen.

Als letzter Ansatz für flexibles Abfragebeantwortung betrachten wir auch das Embedding-basierte Ontologie-vermittelte Abfragebeantwortung über unvollständige KGs. Dafür bauen wir auf modernsten Embedding-Modellen auf, die darauf ausgelegt sind, plausible Antworten auf Abfragen vorherzusagen, und untersuchen einige Möglichkeiten, Ontologien entweder in den Trainingsdaten oder in der Trainingsziel-Funktion zu integrieren, um eine hohe Genauigkeit bei der Vorhersage fehlender Antworten zu erzielen, die im Allgemeinen ontologisches Schlussfolgern erfordern, um auf vorhergesagten Fakten durchgeführt zu werden.

Abstract

Knowledge Graphs (KGs), understood broadly as datasets consisting of interconnected labeled entities in a sharable format, have emerged as means to make different kinds of knowledge available to both humans and machines. Applications of KGs include, but are not limited to, natural question answering, web search, data analytics and expert systems. A crucial task in many KG applications is that of query answering, that is, the problem of finding all possible answers to a given query in a concrete query language, such as the common conjunctive queries (CQs). While much attention has been devoted to the efficient retrieval of answers for various query languages, most of the works assume an ideal query which accurately captures the answers the user has in mind, and take for granted the query formulation step, which is critical and potentially challenging for end-users. Furthermore, it is also implicitly assumed that the user's information needs are well-defined and static, which is often not the case, especially when accessing KGs that include data from vast and unfamiliar domains.

Ontologies are logical formalisms used to enrich KGs with human expertise and common-sense knowledge. They have been successfully deployed to support query formulation, for integrating multiple and heterogeneous data sources, and to enable the retrieval of more complete and informative answers to queries. However, existing ontology-based querying technologies are insufficient for scenarios that call for an inherently interactive querying process that supports the exploration of data: starting from an initial query, users are guided in gradually refining it to accurately capture their information needs. Moreover, for interactive querying, each reformulation must be efficiently evaluated *on-the-fly*. Current query answering engines only support the evaluation of one query at a time, and retrieving the complete set of answers to each query is computationally costly. In order to cope with these challenges, in this thesis we focus on the ontology language *DL-Lite*$_{\mathcal{A}}$ which offers a good trade-off between complexity of reasoning and expressiveness.

The first main contribution of this thesis is to propose an exploratory framework which leverages ontologies to support: a) query formulation, allowing the user to approximate the information needs by writing a query template that succinctly describes a large set of semantically related queries, b) efficient retrieval of complete answers for the large set of related queries, c) on-the-fly query refinement, which enables the interactive exploration of data.

Our techniques for ontology-driven query reformulation call for extending *DL-Lite*$_{\mathcal{A}}$ with *complex role inclusions*. As a second main contribution, we present a complete complexity picture of several *DL-Lite*$_{\mathcal{A}}$ extensions which we identified along our quest to preserve its nice computational

properties. We also consider the safe integration of aggregation into both the ontology and query language to support limited data analytics.

To address the intrinsic incompleteness of KGs, as a third main contribution we propose two techniques to support *flexible querying* of incomplete KGs enriched with ontologies. A first such proposal is *assumption-based query answering*, in which queries are equipped with assumption patterns meant for describing multiple hypothetical extensions of the KG, and construct more informative answers over all such extensions. An answer in this case is not only a candidate entity tuple, but instead a conditional answer that pairs such a tuple with the assumptions that make the tuple a true certain answer. We show that assumption-based query answering is tractable in data complexity and propose ontology-based rewriting techniques for constructing the conditional answers, also in the presence of *closed predicates*, a form of completeness statements about particular relations.

As a final proposal for flexible query answering, we also consider *embedding-based ontology-mediated query answering* over incomplete KGs. For that we build on state-of-the-art embedding models, tailored for predicting plausible answers queries, and explore some means to incorporate ontologies, either in the training data or in the training objective function, in order to obtain high accuracy for predicting missing answers that in general require ontological reasoning to be performed on-top of predicted facts.

Contents

CHAPTER 1

Introduction

Knowledge and information derived from the rapidly increasing amounts of available data are important to create innovative solutions that contribute to the advancement of several diverse domains such as science, technology and business. Graph-structured models have emerged as a leading data representation model to make (expert) knowledge and information available to data-driven applications. By means of graphs we can encode various kinds of knowledge: from relatively simple knowledge such as how computers interact in a network to more complex ones such as protein interaction and social networks. Whenever the graph has several types of relations between the existing nodes, it is often referred to as a knowledge graph.

Knowledge Graphs (KGs), understood broadly as datasets consisting of interconnected labeled entities in a sharable format, have emerged as means to make different kinds of knowledge available to both humans and machines. Applications of KGs include, but are not limited to, natural question answering, web search, data analytics and expert systems. A KG typically encompasses a representation of the underlying domain, called *ontology*, which is used to encode human expertise or common-sense knowledge. An ontology encodes information about key domain concepts and their inter-dependencies, by means of multiple relations.

The idea of enriching data with expert knowledge is not new and it has been extensively investigated in the *Knowledge Representation and Reasoning* area, where first-order logic is used as a suitable formalism for representing the domain knowledge given its well-defined semantics. *Description Logics* have emerged as suitable languages to encode ontologies since they represent decidable fragments of first-order logic, given that full first-order logic is undecidable. DLs have become the most popular ontology formalism insofar-as the W3C ontology standards are based on existing and well-studied description logics. In DLs, a KG is called a *knowledge base* and there is a clear separation between *intentional knowledge*, often denoted as TBox, which represents the knowledge encoded in an ontology, and the *extensional knowledge*, denoted as ABox, which simply encodes all existing facts about concrete individuals or data values.

Ontologies are not only used to structure the existing information in a KG, but also they are being used for integrating multiple heterogeneous data sources by linking the underlying data to the ontology, via *mappings*, thus creating a large *virtual KG*. In this paradigm, called *Ontology-based Data Access* (OBDA), the ontology becomes the common entry point to all disparate data sources. OBDA is also relevant to the Semantic Web, where the main goal is to interconnect information on the web for effective use by both humans and machines. To facilitate this purpose the particular Semantic Web standards such as RDFS and OWL are based on well-studied DLs.

A crucial task in many KG applications is that of *query answering*, that is, the problem of finding all possible answers to a given query in a concrete query language, such as the commonly used conjunctive queries (CQs). A *conjunctive query* is a formula in first-order logic composed using only conjunction and existential quantification. In the context of DLs, the corresponding problem of querying ontology-mediated data has received a lot of attention.

In this thesis we focus on the problem of querying graph-structured data enriched with ontologies, thus we will employ the DL formalization and make a clear distinction between the intensional and extensional parts of a KG, and in particular consider that a KG consists only of extensional knowledge. This formalization allows more generality as the same data can be queried using different ontologies since the ontology is more dependent on the application domain. Moreover, most of the available KGs do not include ontological knowledge and have almost the same expressiveness as an ABox. Therefore we will refer to a KG as an ABox and sometimes use such notions interchangeably.

In general the expressiveness of the ontology language influences the complexity of evaluating queries, with more expressive languages having higher complexity. There are however also tractable languages that offer a good trade-off between expressive power and efficient query evaluation. One such family of DLs is the so-called *DL-Lite* family among which the most popular fragments are *DL-Lite*$_{\mathcal{R}}$ and its extension *DL-Lite*$_{\mathcal{A}}$ with allows in addition data values and functionality restrictions. Compared to other DLs, such as the super-languages of the well-known $\mathcal{ALC}$, the *DL-Lite* family of languages seem rather simple, however they are sufficient to encode knowledge usually represented in class and entity-relationship diagrams, and due to their nice computational properties, *DL-Lite*$_{\mathcal{A}}$ is employed as the main ontology language for integrating and accessing relational data (i.e., via OBDA) while *DL-Lite*$_{\mathcal{R}}$ is used as basis for the OWL QL profile recommended by W3C.

Another important aspect of querying data in presence of ontologies is the assumption that we only have a partial view over the data, thus the existing information is by default incomplete. The existing information in the data may not represent the ground truth, thus there could be in principle additional facts which may not be directly accessible. Among such facts are the implicit facts which can be derived by applying the ontology axioms over existing facts, a process called *deductive reasoning*. However, they fall short in identifying missing facts beyond those that can be inferred from existing information.

Since most of the publicly available KGs are semi-automatically constructed from textual data, they are notoriously incomplete. Predicting missing edges in a KG has received a lot of attention and recently statistical methods based on embedding nodes and edges into low-dimensional

vector space are becoming popular tools for KGs completion. The embedding-based methods aim at learning a representation of the graph by means of *inductive reasoning*. This means that if enough examples are given, the embedding model is able to learn certain inference patterns that are then being used to predict missing links between entities in the graph. More recent, embedding-based techniques for answering logical queries have been proposed as means for predicting missing answers to queries, however they have not been tailored for answering queries mediated by ontologies.

1.1 Motivation

While much attention has been devoted to the efficient retrieval of answers for various query and ontology languages, most of the works assume an ideal query which accurately captures the answers the user has in mind. They take for granted the query formulation step, which is critical and potentially challenging for end-users. Furthermore, it is also implicitly assumed that the user's information needs are well-defined and static, which is often not the case, especially when accessing vast data sources that include data from different and unfamiliar domains. This problem is referred in the literature as the *information mismatch*: users that do not have technical background or that are not domain experts cannot express their information needs by means of a formal query. Ontologies can support the query formulation step, however no generic solutions for exploration and analysis of ontology-mediated data have been proposed so far.

In order to explore the underlying data for different purposes, the querying process must be inherently interactive. An interactive query-answering system would then facilitate the following functionality: starting from an initial query, users are guided in gradually refining it, by taking into account the available facts, to accurately capture their information needs. This means that the query answering system should suggest relevant query reformulations and be able to answer them efficiently. All the queries created in this process are similar with respect to their syntax and meaning, thus a large set of related queries emerges with each querying scenario. From the technical perspective, each query that might be relevant to the query scenario should be efficiently answered and meaningful related queries should be easily constructed. Such services are currently not supported by the existing ontology-mediated query answering systems since they evaluate queries *one-at-a-time*, thus computing the answers from scratch when some of the answers can be preserved if the query is slightly modified.

One of the most successful paradigms for data analysis is *Online Analytical Processing* (OLAP) in which the data (containing mostly historical information) is analyzed from different perspectives (such as time and location) and on multiple granularity levels (such as month, trimester, year for time dimension, and city, region, country for the location dimension). Another important functionality in OLAP is the use of aggregation to define various measures which can then be automatically computed for each level in the dimensional hierarchy. Thus, OLAP relies on a *multi-dimensional data model* for querying similarly as the OBDA paradigm relies on an ontology. Drawing a comparison between OLAP and OBDA, the ontology naturally captures hierarchy of concepts and roles which can also be used for exploratory purposes. Moreover, OBDA offers some advantages over OLAP since the expensive pre-processing steps needed to create the multi-

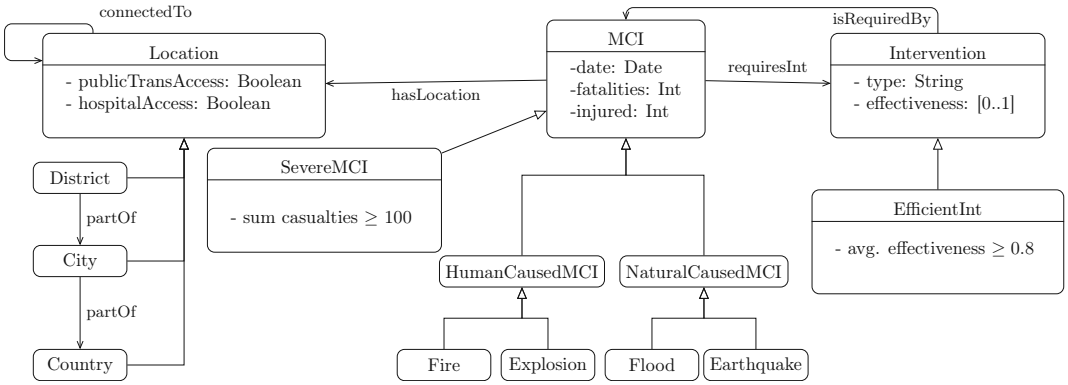


Figure 1.1: Ontology for risk assessment in the disaster management domain.

dimensional data structure are not needed. Therefore, creating for OBDA the type of functionalities that OLAP supports would enable on-the-fly analysis of disperse and heterogeneous data.

In order to facilitate OLAP functionalities for OBDA, it is important to model in the ontology additionally to concept and role hierarchies also *part-of* hierarchies. Encoding multi-dimensional models in DLs requires the use of an ontology language that is more expressive than *DL-Lite* [FS99]. Therefore, we aim at investigating tractable extensions of *DL-Lite* which allow the encoding of part-of hierarchies.

In the following example, which focuses on the risk management domain, we present some concrete use-cases to demonstrate the need for interactive query answering as well as for data exploration and analysis tools.

Example 1.1 (MCI management)**.** *A* mass casualty incident *(MCI) is an event which requires an immediate intervention and in which emergency resources, such as personnel and equipment, are overwhelmed by the number or severity of casualties. The main concerns regarding MCIs are mitigation and prevention, therefore a risk assessment analyst is in general concerned with:*

a) early identification of severe MCIs *(events that have a higher potential to cause harm),*
b) analysis and evaluation of the risk associated with such hazards (e.g., areas affected, population at risk etc.), and ultimately
c) evaluation of the response measures in order to assess the effectiveness of various mitigating interventions.

Consider the graphical representation of an ontology in Figure 1.1 which models the information needed for reporting an MCI. The actual definition will be given in Chapter 2 once the necessary DL notions are introduced. Intuitively, the graphical representation captures the following knowledge:

- *An* MCI *has a starting date, and a possible number of (estimated) casualties which are partitioned into* fatalities *and* injured *cases. If the total number of casualties is greater than 100, then the MCI is considered to be severe. The particular types of events considered are classified based on their origin: they are either caused by humans or caused by nature. The*

most specific categories of the considered MCIs are fires, explosions, floods and earthquakes. When reporting an MCI, requests for various types of interventions can be done.

- *The concept* Intervention *denotes an intervention request which is typically associated to a reported MCI. The type of the intervention, such as request for an emergency service or an evacuation, must also be included in the request, while the effectiveness of the intervention is assessed at a later stage depending on the type of the intervention. For example, an effectiveness measure for a medical intervention can be the percentage of cases that were covered in a particular time interval.*
- *When reporting an MCI, its location must be specified. The more general concept* Location *captures any type of location, with focus on districts, cities and countries. Additional useful information regarding the access to the location include information such as if is reachable via public transport and if a hospital is located in the proximity.*

A first observation is that in order to capture such domain, the chosen ontology language should allow *analytic-aware concept definitions* such is the case for concepts SevereMCI and EfficientInt, and to model *part-of* hierarchies for analyzing the data at various granularity levels. Moreover, to support decision making, the system should allow interactive querying and the use of aggregation at query time.

In general the ontology is useful for: (i) offering a conceptual overview over the data as illustrated in Figure 1.1, (ii) obtaining more complete answers to queries by applying the domain knowledge, and (iii) supporting the query formulation process. For task b in Example 1.1 having complete answers is crucial. For example, when querying all areas affected by severe MCIs the expert system should automatically return all locations points of all MCIs, such as floods, fires etc., for which the total number of casualties exceeds 100. If the number of retrieved locations is too large, meaningful ways to specialize the query can be suggested. For instance one way to specialize the query is by zooming in only on the MCIs that have either a human or a natural cause. Similarly, ways to generalize the query are needed when not enough information is retrieved.

A third observation is that, while adding a lot of value to data, ontologies cannot directly help with missing facts beyond those that can be inferred from existing information and, in practice, the data might be only partially available at query time. For example if the number of casualties is not provided, one cannot infer that the reported MCI is severe or not. This is also the case with KGs that are automatically constructed, queries might fail due to lack of available information. Therefore techniques tailored towards coping with incomplete data are needed for evaluating queries mediated by ontologies.

Towards that, querying the data based on relevant assumptions that model various data completion scenarios can be useful. For example in task c of Example 1.1, to make sure an evacuation is efficient, one has to verify various possible scenarios regarding the location accessibility. For instance, assuming that a person is located in the proximity of some metro station, can that person reach a hospital? Answering such question is relevant for identifying possible vulnerable locations and improving the efficacy of interventions.

Another, interesting direction in coping with incompleteness is to rely on embedding-based models for answering ontology-mediated queries. For example, task a of Example 1.1 requires

the prediction of additional answers, such as predicting potential human-caused MCIs. Current KG embedding models are able to predict plausible answers to queries, however when taking into account the ontology, in order to obtain accurate predictions, the embedding model would have to learn to apply also the ontology knowledge. Thus, for such task the model should be *ontology-aware* thus be capable of performing both inductive and deductive reasoning.

1.2 Problem Formulation and State-of-the-art

The main goal of the thesis is to formalize and implement techniques that allow flexible access to graph-structured data enriched with ontologies. Concretely, we want to provide the basis for the following services:

1. *Query formulation*: support the user's attempts to formulate their information needs, by developing a suitable specification language for sets of queries to enable exploration of the targeted data;
2. *Interactive ontology-mediated query answering for data exploration*: based on the specified information need, efficiently compute answers to a large set of related queries, and describe them in an understandable and informative way. In addition, suggest meaningful query reformulations (i.e., specializations or generalizations) for gradual exploration of answers;
3. *Support for data analysis:* automatize operations that facilitate analysis of data enriched with domain knowledge. The focus is on incorporating aggregation and *part-of* hierarchies that resemble OLAP capabilities;
4. *Incompleteness tolerance*: alternative techniques for answering ontology-mediated queries even when the data is partially unattainable. In particular we focus on querying hypothetical data completions, and on neuro-symbolic approaches to predict missing answers.

1.2.1 State-of-the-art

In this section we present the state-of-the-art regarding the problems which the thesis is addressing. To the best of our knowledge, there is no approach that can support all enumerated services, therefore we categorize the related work as follows.

Query Reformulation. There are several interface systems that help the user in formulating queries which differ based on the underlying supported query language and on the use of reasoning to suggest possible query reformulation. The main motivation is the need for end-users to access disparate data sources and the limitations imposed by the machine-oriented query languages used to access data. To cope with such issues, *visual query systems* (*VQSs*) such as SemFacet [ACGK+14], Quelo [FGTT11], Optique VQS [SKZ+14], focus on bridging the gap between user's information needs and query formulation. They support limited exploration functionalities since, in general, they support mainly ways to specialize queries. SemFacet takes also into account the underlying data, by discarding options that would not change the set of answers. The expressivity of the supported query language is also limited as they do not capture arbitrary conjunctive queries. More recently, Vargas et al. [VAHL19] supports the formulation of more complex queries, namely SPARQL queries, however it does not consider ontological reasoning

to answer such queries. In the work by Nutakki et al. [Nut11] the authors present solutions for specializing queries over $\mathcal{EL}$ ontologies [BBL05], to avoid information overload. The presented approach computes minimal specializations which are then clustered for reducing information load by a significant amount. The approach is evaluated over synthetic data, which is uniformly generated, and contains small number of individuals, so it is not clear how such solution behaves over real data sources with millions of instances. In addition, the converse problem of identifying minimal generalizations, useful for exploration purposes, was not considered.

Explorative and Analytical Querying. Existing solutions enhance *query navigation*, by computing ways to specialize or generalize the query based on terminological support for a given application domain. The idea of navigating from one query to another as means to explore the data has been investigated mostly in the context of web search, to improve the quality of the search engines. The authors of [HYY05] use a particular concept terminology to generate a query space that creates relations among the terms, i.e., words in natural language, and provide visual tools that help the user in refining the query to correspond the intended information need. Another approach is to translate queries, expressed in natural language, into suitable conceptual graphs, which represent concept specializations or generalizations of the initial query, which are then matched over annotated documents [CHH10]. This work has been extended in [PHH11] to include a pivot language in which the user can express relations among query terms. However, all such approaches are designed for hiding the complexity of formulating queries and focus mostly on improving natural language search over domain specific documents on the Web. The actual query matching process is almost syntactical (e.g., terms can be directly matched or via synonyms), since the the terminological support represents simple taxonomies, therefore it is not clear how such solutions can be used if DL ontologies are used as knowledge support, in which case more complex reasoning services would be required.

There are several engines that support ontological reasoning to compute complete answers to queries, most noteworthy Pellet [SPG+07], HermiT [GHM+14] and PAGOdA [ZGN+15], which work for evaluating queries over graph-structured data, and Ontop [RKZ13] which implements an OBDA service over databases. However, such systems do not support interactive querying, meaning that answering a sequence of related queries can be done only by evaluating queries *one-at-a-time*, answers are built from scratch in each evaluation, and this is rather inefficient considering that some answers might be preserved from one evaluation to the other. Computing meaningful specializations or generalizations for a given query can be done using such engines only in a naive manner, by means of brute-force evaluation of queries and comparison on answers. Moreover, traditional solutions do not support aggregation in the ontology, and this lead to the extension of *DL-Lite* to incorporate a safe form of aggregation [KKM+16].

The need of aggregation in the query has already been acknowledged in the case of OBDA, however, due to the open-world semantics, such queries have to be carefully handled to avoid undecidability [KR13] and in [CKNT08] an epistemic semantics for evaluating queries with aggregation was introduced. The combination of $\textit{DL-Lite}_{\mathcal{A}}^{\text{agg}}$ and aggregating queries is very important towards having analytics-aware OBDA services however all such formalism are not enough for supporting OLAP-like operations. In the case of RDF data, there have been approaches

in building data structures suitable for data analytics [CGMR14], and to develop efficient OLAP operations over such analytical schemas [AGMR15]. From this perspective, ontologies have been leveraged only for designing *extract-transform-load* (ETL) processes to clean and integrate the data [SSS09], or for defining new granularity levels for analytical queries ([NAS12], [ANS12], [GGR+18]). More recently, the work by Schütz et al. [SBN+21] studies the adaptation of OLAP paradigm to KGs.

Querying Incomplete Data Sources. This problem has been addressed in the setting of relational data in which techniques for querying hypothetical extended versions of the database have been proposed in [GGMO95, CA98]. Among the proposed formalism over disjunctive databases is *conditional query answering* [Dem92] in which tuples are coupled with the assumptions needed in order to become answers to queries. In the work by Griffin and Hull [GH97] hypothetical queries are rewritten into equivalent traditional queries which are evaluated using standard techniques.

In the case of ontology-mediated data, ten Cate et al. [tCCST15] define so-called *why-not queries*, where an ontology is leveraged to obtain explanations of why tuples are not an answer, and study the complexity of obtaining most general explanations. Another similar formalism based on ABox abduction is proposed by Calvanese et al. [COvS13] where *negative answers* are employed for describing a tuple of individuals and an associated explanation, which is represented by a small ABox, to why it is not an answer to the given query.

Knowledge graph completion has received also a lot of attention. Recently, statistical methods to compute plausible facts in a KG have been proposed [NMTG16, WMWG17]. The general idea is that the statistical model is able to learn some form graph similarity which is then used to predict missing edges between existing nodes. All such models are based on the idea of embedding nodes and relations into a low-dimensional vector space. Building on their success, methods that can perform so-called multi-hop querying in the embedding space have been successfully applied to predict answers to queries of limited expresivity. The existing proposals can be divided into *query-based* [RHL20, RL20, LDJ+21, CRK+21, KLN21, SAB+20] and *atom-based* [ADMC21]. Friedman et al. [FdB20] and Bogwardt et al. [BCL19] study the relation between the problem of conjunctive query answering in the embedding space and over probabilistic databases. None of the embedding-based models are designed for querying in presence of ontologies, thus the question if they can be used or adapted for such case is natural and not answered by the existing body of work.

1.3 Research Challenges and Methodology

The first step towards achieving the goal of the thesis is to formalize a framework that takes as input an ontology and a dataset and firstly allows the user to succinctly define a set of related queries. Then, based on the query specification, it constructs a compilation to support interactive query answering and data analysis operations. Furthermore, in case the queries of interest do not produce the desired output, we can employ the functionalities of answering queries considering relevant completions of the data or predict missing answers with the associated score. In order to achieve all the envisioned services, we have to provide a theoretical framework. For that we

identify the following research challenges associated to each particular problem that the thesis is addressing.

RC1 Find tractable extensions of ontology language *DL-Lite* to facilitate operations similar to OLAP *roll-up*, *drill-down* and query rewriting operations to specialize and generalize queries that help the user in gradually explore the data.

RC2 Provide means to the user to write an expression that succinctly describes sets of related (i.e., semantically similar) conjunctive queries. In particular, the expression should define a core conjunctive query, which represents the user's basic information need, and create techniques to automatically derive various modifications of the query that the user deems interesting.

RC3 Given the user's query specification, provide solutions to compile the relevant fragments of the ontology and the data to answer any query that matches the specification. Constructing such compilation is important in order to efficiently navigate from one query to another. Such task requires reasoning to identifying explicit and implicit objects matching some query of the family.

RC4 Define reasoning services over the compilation, that are geared towards supporting interactive exploration of answers for queries that target user's information needs. This challenge is relevant for identifying meaningful specializations and generalizations, considering that there can be such modifications which are redundant for navigation, i.e., specializations that produce the same set of answers.

RC5 Formalize and provide effective solutions for answering queries over incomplete data mediated by ontologies. In particular provide reasoning services to support *assumption-based query answering*, and adapt existing KG embedding techniques for predicting answers to ontology-mediated queries.

For addressing **RC1**, we firstly point out that the concept taxonomy which is usually present in an ontology, which are related via *is-a* relation, is a natural way to modify queries for data exploration. However, this is different than the dimension hierarchy typically used in OLAP which contains concepts that are related via *part-of* relation (such as city is part of a country and country part of a political or geographical group). This type of relation can be encoded in an DL ontology by means of *complex role inclusions*, however this can significantly increase the complexity of reasoning and we need to identify in which conditions they can be incorporated into *DL-Lite* such that the tractability of the ontology language is preserved.

For addressing **RC2**, we aim at having a query template language whose syntax allows flexible query constructs while the semantics uses the derivation-based order to create a large collection of semantically related queries, which we denote as a *query space*. This leads us to address **RC3** for which we first define rules to derive concrete queries from the query template and given ontology we can apply the ontology axioms on the template atoms in a specializing or generalizing fashion. This idea is inspired by the query rewriting approach for *DL-Lite*$_{\mathcal{R}}$ [CDL$^+$07], however

we want to identify other rewriting rules to obtain more meaningful reformulations and to support navigation along both is-a and part-of hierarchies. Furthermore, we aim at having a Datalog translation of the query generation process which is then evaluated over the data alone to create the compilation.

In order to address **RC4**, we want to support query navigation by means of identifying, for any given query, all specializations and generalizations that minimally change the set of answers. Next, we rely once more on Datalog rules to identify and answer such reformulations. Thus by combining both the query generation and navigation Datalog programs we obtain the implementation of a fully-fledged exploratory framework. Moreover, we also aim at evaluating the framework in practice using existing Datalog engines.

Regarding **RC5** to address the problem of evaluating queries in possible extensions of the data enriched with ontologies, we formalize the problem of *assumption-based ontology-mediated query answering* and provide efficient techniques to compute informative answers. We also are interested in possible extensions of this formalism to partially incorporate closed-world semantics. We also want to combine ontological reasoning and existing KG embedding models that support query answering, to compute plausible answers to queries over incomplete KGs.

1.4 Contributions

The first main contribution of this thesis is to propose an *exploratory framework* which leverages ontologies to support:

- Query formulation, allowing the user to approximate the information needs by writing a *query template* that succinctly describes a large set of related queries.
- Efficient retrieval of answers for the large set of related queries.
- On-the-fly query refinement, which enables the interactive exploration of data.

Our techniques for ontology-driven query reformulation call for extending *DL-Lite*$_{\mathcal{A}}$ with complex role inclusions. As a second main contribution, we present a detailed complexity picture of several *DL-Lite*$_{\mathcal{A}}$ extensions which we identified along our quest to preserve its nice computational properties. We also consider the safe integration of aggregation into both the ontology and query language to support limited data analytics.

To address the intrinsic incompleteness of data, as a third main contribution we propose two techniques to support *flexible querying* of incomplete data enriched with ontologies. A first such proposal is *assumption-based ontology-mediated query answering*, in which queries are equipped with assumption patterns meant for creating multiple hypothetical extensions of the data, and construct more informative answers. An answer in this case is not only a candidate answer tuple, but instead a *conditional answer* that pairs such a tuple with the assumptions that make the tuple a true answer. We show that assumption-based query answering is tractable and propose ontology-based rewriting techniques for constructing the conditional answers, also in the presence of *closed predicates*, which enables completeness statements about particular concepts and relations.

As a final proposal for flexible query answering, we also define the task of *embedding-based ontology-mediated query answering*. As concrete approaches to such problem, we build on two state-of-the-art embedding models as representative of query-based and atom-based models. Such models are tailored for predicting plausible answers to a restricted form of positive existential queries, however they do not take into account any ontological knowledge. Therefore, we explore some means to incorporate ontologies, either in the training data or into the training objective function, in order to obtain high accuracy in predicting answers that require the application of both inductive and deductive reasoning.

The enumerated contributions regarding the extension of *DL-Lite* with complex role inclusions and the proposed exploratory framework have been published in the following peer-reviewed venues:

- Medina Andresel, Yazmín Angélica Ibáñez-García, Magdalena Ortiz, and Mantas Simkus. Taming complex role inclusions for DL-Lite. In *Proceedings of the 31st International Workshop on Description Logics co-located with 16th International Conference on Principles of Knowledge Representation and Reasoning (KR 2018), Tempe, Arizona, US, October 27th - to - 29th, 2018*, volume 2211 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018.

- Medina Andresel, Yazmín Ibáñez-García, Magdalena Ortiz, and Mantas Simkus. Relaxing and restraining queries for OBDA. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 2654–2661. AAAI Press, 2019.

- Medina Andresel, Yazmín Ibáñez-García, and Magdalena Ortiz. A framework for exploratory query answering with ontologies. In *Proceedings of the 33rd International Workshop on Description Logics (DL 2020) co-located with the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020), Online Event [Rhodes, Greece], September 12th to 14th, 2020*, volume 2663 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.

The contributions regarding assumption-based querying have been presented in the following peer-reviewed publication:

- Medina Andresel, Magdalena Ortiz, and Mantas Simkus. Query rewriting for ontology-mediated conditional answers. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 2734–2741. AAAI Press, 2020.

Regarding the latest work of embedding-based ontology-mediated query answering, the results presented in this thesis are collected into a publication published in a top-tier peer-reviewed conference:

- Medina Andresel, Trung-Kien Tran, Csaba Domokos, Pasquale Minervini, and Daria Stepanova. Combining inductive and deductive reasoning for query answering over incomplete knowledge graphs. In Ingo Frommholz, Frank Hopfgartner, Mark Lee, Michael Oakes, Mounia Lalmas, Min Zhang, and Rodrygo L. T. Santos, editors, *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023*, pages 15–24. ACM, 2023.

1.5 Thesis Structure

This thesis is structured into seven chapters. In Chapter 1 (this chapter) we present the main motivation of this work, goals, research challenges and contributions. The next chapter Chapter 2 presents a more in-depth introduction of the problem of ontology-mediated query answering with particular focus on *DL-Lite*$_{\mathcal{A}}$ and its analytic-aware extension *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$, and on the evaluation of conjunctive queries which may contain aggregating functions. We also present the main techniques for answering ontology-mediated queries in *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ and the notions required to study the complexity of relevant reasoning tasks. The following Chapters 3–6 present the main contributions of the thesis:

- In Chapter 3 we address **RC1** and we extend *DL-Lite*$_{\mathcal{A}}$ with complex role inclusions, denoted as *DL-Lite*$_{\mathcal{A}}^{++}$. We show that such language is undecidable and then investigate sub-languages that preserve decidability and study the complexity of reasoning tasks such as consistency testing and computing certain answer. We show that restricting the language so that only *regular* complex role inclusions are allowed is enough for ensuring decidability, however the complexity remains high, namely exponential. In order to obtain tractable sub-languages we propose to either disallow recursion involving relations appearing in any complex role inclusion axiom, or to limit the recursion to have only paths of fixed length over which complex role inclusions are triggered.
- In Chapter 4 we address **RC2**, **RC3** and **RC4**, and propose a query template language that allows the marking of query atoms with the purpose of specialization or generalization. Further, we present rules to derive a query space which is then used for query navigation by leveraging a derivation-based order between queries. We also define means to identify meaningful reformulations to queries and finally present a Datalog-based translation to realize such framework which we evaluate using existing Datalog engines.
- In Chapter 5 we address **RC5** and define the problem of computing conditional answers to queries in the presence of ontologies. A conditional answer includes facts which are not implied and if added to the data would produce additional answers. To obtain interesting conditional answers, we propose to pair a given ontology-mediated query with assumption patterns in the form of $\mathcal{ELI}$ atoms which can be applied to concrete atoms in the query. We show that such novel formalism is *first-order rewritable*, meaning that we can construct the conditional answers in a data-independent fashion by rewriting the query w.r.t. the

ontology and the set of assumption patterns. We also consider a more involved case in which closed predicates are allowed in the assumption patterns and show that our formalism is both useful to model interesting domains that require the use of closed predicates, while not increasing the complexity. This is remarkable since the presence of closed predicates is known to significantly increase the complexity of query answering [LSW15].

- In Chapter 6 we also address **RC5** and investigate a different approach to tackle data incompleteness. In this chapter we focus on embedding-based approaches for answering queries and investigate their capabilities in performing both deductive and inductive reasoning. We investigate how to use and adapt such models for predicting answers to ontology-mediated queries and propose ontology-driven training strategies to incorporate the ontology axiom into the training data, and novel ways to enforce the rules in the vector space. We note that such technique is different in coping with missing facts than the assumption-based approach.

In the last chapter, namely Chapter 7 we present an overview of the thesis together with final remarks and a discussion regarding future research directions.

CHAPTER 2

Ontology-mediated Query Answering

In this chapter we provide the basic knowledge regarding the problems studied in this thesis. We introduce the syntax and semantics of lightweight ontology languages and also define the query languages that we consider. Based on these two notions we define ontology-mediated queries, their semantics and existing techniques for evaluating them over given datasets. We define the decision problems associated to the reasoning tasks of interest and describe the standard complexity measurements together with the relevant complexity classes.

2.1 Description Logics Ontologies

Description logics (DLs) [BCM+03, BHLS17] are a family of *knowledge representation and reasoning* languages used to formalize domain knowledge for various knowledge-driven applications. The domain is usually described by means of an *ontology*, therefore it is common to refer to DLs as *ontology languages*. DLs in fact are nothing but decidable fragments of first-order logic, thus their main advantages include the well-understood semantics and automated reasoning capabilities, useful to infer new knowledge.

The general modeling principle in DLs is to structure the domain of interest into *concepts* (i.e., classes), which describe similar objects, and *roles* (i.e., properties), which denote relations between objects. The usual DL constructs are suitable to express constrains for concepts and roles. Among the basic constraints that each DL allows are *concept and role inclusions* and, depending on the DL expressivity, more involved constraints are supported. The constrains are then collected into what we call a *DL ontology*, typically denoted as a *TBox* since it encodes the relevant terminological knowledge about the domain.

In this thesis our main focus is on the lightweight families of DLs called *DL-Lite* [CGL+05, CGL+07, ACKZ09] and $\mathcal{EL}$ [BBL05, BLB08] which offer limited expressive power in exchange for good computational properties, important for scalability of data-driven applications.

In the remaining of this section, we first present the syntax and semantics of the considered languages, illustrate using examples what type of knowledge they are able to capture and present the classical reasoning tasks.

2.1.1 Terminology, Data-value Domains, Interpretations

In DLs, concepts and roles are encoded using first-order logic predicates. We start with defining the notion of a *DL vocabulary* consisting of atomic predicates distinguished into: a set of unary predicates called *concept names*, two sets of binary predicates denoting *role names* and *feature names*, and lastly a set of predicates of arity 0 denoting *constant symbols* that is partitioned into two disjoint sets denoting *object constants (or individuals)* and *value constants*.

Definition 2.1 (Vocabulary)**.** *A* DL vocabulary *consists of quadruple* $(\mathbf{C}, \mathbf{R}, \mathbf{F}, \mathbf{K})$ *of countable infinite, pairwise disjoint sets such that* $\mathbf{C}$ *denotes the set of* concept names*,* $\mathbf{R}$ *denotes the set of* role names*,* $\mathbf{F}$ *denotes the set of* feature names *and* $\mathbf{K}$ *denotes the set of* constants*. We further consider that* $\mathbf{K}$ *is partitioned in two infinite disjoint sets, namely* $\mathbf{K_O}$ *denoting constant symbols for* objects *and* $\mathbf{K_V}$ *denoting constant symbols for* data values*.*

Example 2.1. *Recall the mass casualty incident (MCI) ontology presented in the previous chapter. For such domain, the vocabulary consists of all the terms that appear in the ontology. In particular, we have the following:*

- *concept names:* MCI, Location, Intervention, City ...
- *role names:* hasLocation, connectedTo, partOf, ...
- *feature names:* date, injured, interventionType, effectiveness, ...
- *object constants:* Vienna, Austria, distr_1, ...
- *value constants:* "02/10/2015", "50", "medical intervention", "7.8", ...

By convention, concept names start with uppercase, role and feature names start with lowercase and are written in Roman *font. Constants start with either lower or uppercase and are written in* Sans Serif *font.*

In this thesis we assume an arbitrary but fixed DL vocabulary $(\mathbf{C}, \mathbf{R}, \mathbf{F}, \mathbf{K})$ that is not referring to one particular application domain. We use the following naming notation: (possibly sub-indexed) A, A' for concept names, r, s for role names, f for feature names, p for either a role or feature name, a, b for objects, v for values and c for any constant symbol.

Data domains. The constant symbols denoting data-values correspond to data types such as *String*, *Integer*, *Real* and others. In order for such sets of elements to construct an abstract *data-value domain*, they are typically equipped with binary predicates for comparisons and potentially with aggregating functions mapping multi-sets of elements to elements in the domain. We recall that a multi-set is a collection of elements, which unlike a set, allows for multiple occurrences of the same element.

Definition 2.2 (Data domain)**.** *A* data-value domain $\mathcal{D}$ *is a triple* $(\Delta_D, \Phi_D, \Lambda_D)$*, where* Δ_D *is a set of elements called* the domain of $\mathcal{D}$*,* Φ_D *is a set of* comparison predicates $<_D, \leq_D, =_D, \geq_D, >_D, \neq_D$,

and Λ_D *contains some of the following aggregating functions* min_D, max_D, avg_D, sum_D, cnt_D, cntd_D *such that each* $\mathsf{agg} \in \Lambda_D$ *is a function that maps any multi-set of elements to an element in* Δ_D.

Given a data-value domain $\mathcal{D}$ and a multi-set $M \subseteq \Delta_D$, functions min_D, max_D, are always defined for M since they output the minimal, respectively maximal, element in M with respect to the ordered imposed by the comparison predicates. Similarly, the functions cnt_D, cntd_D are used to count the number of (distinct) elements in M. However for sum_D and avg_D, which output the sum, respectively the arithmetic mean of the elements in M, we consider them to be defined only for numerical domains like $\mathbb{N}$, $\mathbb{Z}$ or $\mathbb{R}$.

To exemplify, the *Real* data-value domain can be represented as follows: $\mathbb{R}$ is the set of all elements, the comparison predicates are defined as usual and each aggregating function computes the expected output over any multi-set of real numbers. Similarly, the *String* data-value domain contains as elements set any possible word over some given alphabet, and the comparison operators defined according to the lexicographical ordering of strings.

Let $\mathcal{D}_1, \ldots, \mathcal{D}_n$ denote all the considered data-value domains. Based on all the considered domains, we assume that $\mathbf{K_V}$ is further partitioned into n sets $\mathbf{K}_{\mathbf{V}_1}, \ldots, \mathbf{K}_{\mathbf{V}_n}$ such that each $\mathbf{K}_{\mathbf{V}_i} = \Delta_{\mathcal{D}_i}$ is the set of constant symbols for $\mathcal{D}_i$.

The semantic of DLs languages is given in terms of *first-order logic interpretations* which, intuitively, give meaning to each element in the vocabulary.

Definition 2.3 (Interpretation)**.** *An* interpretation $\mathcal{I}$ *consists of a pair* $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$*, where* $\Delta^{\mathcal{I}}$ *is a non-empty (possibly) infinite set of elements called* the domain of $\mathcal{I}$*. We assume that* $\Delta^{\mathcal{I}}$ *consists of two pairwise disjoint sets* $\Delta^{\mathcal{I}}_{\mathbf{O}}$ *and* $\Delta^{\mathcal{I}}_{\mathbf{V}} = \Delta^{\mathcal{I}}_{\mathcal{D}_1} \cup \cdots \cup \Delta^{\mathcal{I}}_{\mathcal{D}_n}$*. Function* $\cdot^{\mathcal{I}}$ *is called* the valuation of $\mathcal{I}$ *which maps vocabulary terms as follows:*

- *for each* $a \in \mathbf{K_O}$ *we have* $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}_{\mathbf{O}}$,
- *for each* $v \in \mathbf{K_V}$*, we have* $v^{\mathcal{I}} \in \Delta^{\mathcal{I}}_{\mathbf{V}}$,
- *for each* $A \in \mathbf{C}$*,* $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$,
- *for each data-value domain* $\mathcal{D}$*, we have* $\mathcal{D}^{\mathcal{I}} = \Delta^{\mathcal{I}}_{\mathcal{D}}$,
- *for each* $r \in \mathbf{R}$*,* $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}_{\mathbf{O}} \times \Delta^{\mathcal{I}}_{\mathbf{O}}$,
- *for each* $f \in \mathbf{F}$*,* $f^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}_{\mathbf{O}} \times \Delta^{\mathcal{I}}_{\mathbf{V}}$.

We say that $\mathcal{I}$ adopts *the* unique name assumption *(UNA) if for each* $c_1, c_2 \in \mathbf{K}$ *if* $c_1 \neq c_2$ *then* $c_1^{\mathcal{I}} \neq c_2^{\mathcal{I}}$*. Similarly,* $\mathcal{I}$ adopts *the* standard name assumption *(SNA) if* $\mathbf{K} \subseteq \Delta^{\mathcal{I}}$ *and for each* $c \in \mathbf{K}$ *we have that* $c^{\mathcal{I}} = c$.

Under the UNA, different individuals must be mapped to different elements, while under the stronger SNA (which implies the UNA) individuals must be mapped to themselves. The UNA is useful whenever different objects must refer to different things in the world. The SNA, adopted by default in relational databases, is typically used when the information is assumed to be complete (i.e., closed), or when data-value domains are being considered.

A *homomorphism* between interpretations is a structure-preserving mapping.

Definition 2.4 (Homomorphism)**.** *Let $\mathcal{I}$ and $\mathcal{J}$ be two interpretations. A* homomorphism *from $\mathcal{I}$ to $\mathcal{J}$ is a function $h : \Delta^{\mathcal{I}} \mapsto \Delta^{\mathcal{J}}$ such that:*

a) for each $c \in A^{\mathcal{I}}$ there exists $c' \in A^{\mathcal{J}}$ such that $h(c) = c'$, and
b) for each $(c_1, c_2) \in p^{\mathcal{I}}$ there exists $(c'_1, c'_2) \in p^{\mathcal{J}}$ such that $h(c_1) = c'_1$ and $h(c_2) = c'_2$,

where $A \in \mathbf{C}$, $p \in \mathbf{R} \cup \mathbf{F}$ and $c, c', c_1, c_2, c'_1, c'_2 \in \mathbf{K}$. If such homomorphism exists, we say that $\mathcal{I}$ is homomorphically embedded into *$\mathcal{J}$ and write $\mathcal{I} \rhd \mathcal{J}$.*

If there exists a homomorphism h that is bijective (i.e., its inverse is also a homomorphism), then we say that $\mathcal{I}$ and $\mathcal{J}$ are isomorphic.

2.1.2 Lightweight DLs

In this section, we present the main DL languages which are being studied in this thesis and their semantics in terms of interpretations.

***DL-Lite* Family.** Some of the most popular ontology languages are part of the *DL-Lite* family of DLs, since on the one hand they can represent data models such as entity-relation and UML class diagrams, and on the other they enjoy low computational complexity of inference. In particular we focus on *DL-Lite*$_{\mathcal{A}}$ [ACKZ09], a language especially tailored for the *ontology-based data access* (OBDA) paradigm [PLC+08]. In *DL-Lite*$_{\mathcal{A}}$, data-value domains play an important role and allow the declaration of ranges for feature roles. The interpretations in this case adopt the SNA for all constants[1].

Definition 2.5 (*DL-Lite*$_{\mathcal{A}}$ syntax, semantics)**.** *DL-Lite*$_{\mathcal{A}}$ *expressions are constructed according to the following grammar:*

$$B := \top \mid A \mid \exists s.\top \mid \exists f \qquad s := r \mid r^-,$$
$$F := \exists f^- \qquad E := \mathcal{D}_1 \mid \cdots \mid \mathcal{D}_n$$

where $\top$ is called top concept *$A \in \mathbf{C}$, $r \in \mathbf{R}$ and $f \in \mathbf{F}$. Concepts of the form B, F are called* general concepts *and E denotes any of the considered value-domains $\mathcal{D}_1, \ldots, \mathcal{D}_n$. Roles the form r^- are called* inverse roles*, features of form f^- are called* inverse features.

A DL-Lite$_{\mathcal{A}}$ axiom *has any of the following shapes:*

$$B_1 \sqsubseteq B_2 \quad s_1 \sqsubseteq s_2 \quad B \sqsubseteq \exists s.A \quad \mathsf{disj}(B_1, B_2) \quad \mathsf{disj}(s_1, s_2) \quad \mathsf{disj}(f_1, f_2)$$
$$F \sqsubseteq E \quad f_1 \sqsubseteq f_2 \quad B \sqsubseteq \exists s.\top \quad (\mathsf{funct}\ s) \quad (\mathsf{funct}\ f).$$

Example 2.2. *Some of the constraints ilustrated in the MCI reporting ontology can be written into DL-Lite*$_{\mathcal{A}}$ *as follows. For instance, we encode the inheritance relations using the following*

[1] For *DL-Lite*$_{\mathcal{A}}$, UNA is sufficient, however, we adopt the SNA as aggregation in the DL expressions, which is considered later on, requires it.

axioms:

$$\begin{aligned}
\text{SevereMCI} &\sqsubseteq \text{MCI} & \text{EfficientInt} &\sqsubseteq \text{Intervention}\\
\text{Fire} &\sqsubseteq \text{HumanCausedMCI} & \text{Explosion} &\sqsubseteq \text{HumanCausedMCI}\\
\text{Flood} &\sqsubseteq \text{NaturalCausedMCI} & \text{Earthquake} &\sqsubseteq \text{NaturalCausedMCI}\\
\text{HumanCausedMCI} &\sqsubseteq \text{MCI} & \text{NaturalCausedMCI} &\sqsubseteq \text{MCI}\\
\text{District} &\sqsubseteq \text{Location} & \text{City} &\sqsubseteq \text{Location}\\
\text{Country} &\sqsubseteq \text{Location}.
\end{aligned}$$

Such axioms enforce for example that each instance of Fire *must be an instance of* HumanCausedMCI *and implicitly also of* MCI. *We can also encode other constraints, such as each* MCI *must have a location and a date, and the domain and range constraints can be encoded as follows.*

$$\begin{aligned}
\text{MCI} &\sqsubseteq \exists \text{hasLocation} & \exists \text{hasLocation}^- &\sqsubseteq \text{Location}\\
\text{MCI} &\sqsubseteq \exists \text{date} & \text{Intervention} &\sqsubseteq \exists \text{type}\\
\exists \text{fatalities} &\sqsubseteq \text{MCI} & \exists \text{injured} &\sqsubseteq \text{MCI}\\
\exists \text{requiresInt} &\sqsubseteq \text{MCI} & \exists \text{requiresInt}^- &\sqsubseteq \text{Intervention}\\
\exists \text{connectedTo} &\sqsubseteq \text{Location} & \exists \text{connectedTo}^- &\sqsubseteq \text{Location}\\
\exists \text{date}^- &\sqsubseteq \textit{String} & \exists \text{fatalities}^- &\sqsubseteq \textit{Integer}\\
\exists \text{injured}^- &\sqsubseteq \textit{Integer} & \exists \text{casualties}^- &\sqsubseteq \textit{Integer}\\
\exists \text{type}^- &\sqsubseteq \textit{String} & \exists \text{effectiveness}^- &\sqsubseteq \textit{Real}.
\end{aligned}$$

The next block of axioms involve roles and features. For example we can encode that fatalities *and* injured *are sub-relations of* casualties *and that role* isRequiredBy *is the inverse of* requiresInt.

$$\begin{aligned}
\text{fatalities} &\sqsubseteq \text{casualties} & \text{injured} &\sqsubseteq \text{casualties}\\
\text{requiresInt}^- &\sqsubseteq \text{isRequiredBy}.
\end{aligned}$$

Note that domain and range constraints for requiresInt *become applicable also for* isRequiredBy *(the domain of* requiresInt *becomes the range of* isRequiredBy, *and the range becomes the domain). Lastly, to encode the mandatory participation constraints, such as each district is part of a city or that whenever there exists an incident with fatalities it automatically requires some intervention, we resort to the following set of axioms:*

$$\begin{aligned}
\text{District} &\sqsubseteq \exists \text{partOf}.\text{City} & \text{City} &\sqsubseteq \exists \text{partOf}.\text{Country}\\
\exists \text{fatalities} &\sqsubseteq \exists \text{requiresInt}.
\end{aligned}$$

We collect all the axioms above into a DL ontology which we denote as $\mathcal{O}_{MCI}$ *and use as a running example in the following chapters as well.*

A *DL-Lite*$_{\mathcal{A}}$ ontology is a finite set of axioms in which the interaction between role or feature inclusions and functionality axioms is not allowed. The restriction on functional roles and features is to ensure the same computational properties that the languages *DL-Lite* $_{core}$, *DL-Lite*$_{\mathcal{R}}$ and *DL-Lite*$_{\mathcal{F}}$ enjoy [CGL+07]. For simplicity, let $\mathbf{R}^{\pm}$ be the extension of $\mathbf{R}$ containing additionally r^{-} for each $r \in \mathbf{R}$ and similarly let $\mathbf{F}^{\pm}$ be the extension of $\mathbf{F}$ that adds f^{-} for each $f \in \mathbf{F}$.

Definition 2.6 (*DL-Lite*$_{\mathcal{A}}$ ontology)**.** *A DL-Lite*$_{\mathcal{A}}$ ontology (or TBox) $\mathcal{O}$ *is a finite set of DL-Lite*$_{\mathcal{A}}$ *axioms which respects the following condition: for each* $p \in \mathbf{R}^{\pm} \cup \mathbf{F}^{\pm}$ *such that* (funct p)*,* $\mathcal{O}$ *does not contain any axiom of the form* $p' \sqsubseteq p$*.*

Incorporating aggregating concepts. In order to enable reasoning about data values we consider *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ [KKM+16] which is an extension of *DL-Lite*$_{\mathcal{A}}$ that allows *aggregating functions* (such as sum, min, avg) and value comparisons in concept expressions. Note that the concept SevereMCI is not expressible in *DL-Lite*$_{\mathcal{A}}$ since it requires any of its instances to have a total number of casualties greater or equal than 100, and similarly for the concept EfficientInt. Therefore, to model our running example we must resort to additional concept constructs that allow aggregation over features. The concrete syntax of *DL-Lite*$_{\mathcal{A}}$ with aggregating concepts is defined as follows:

Definition 2.7 (*DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ syntax, semantics)**.** *Let* $\mathcal{D}$ *be a data-value domain. An* aggregating concept w.r.t. $\mathcal{D}$ *is of form*

$$C := \odot_d(\mathsf{agg}_D\ f),$$

where $\odot$ *is one of* $<_D, >_D, \leq_D, \geq_D, =_D, \neq_D$*,* $d \in \Delta_D$*,* $f \in \mathbf{F}$ *and* agg *is one of the defined aggregating functions in* $\mathcal{D}$*. Semantics of such expression are presented in Table 2.1, where* $\{\!|\cdot|\!\}$ *denotes a multi-set.*

A DL-Lite$_{\mathcal{A}}^{\mathsf{agg}}$ axiom *is either a DL-Lite*$_{\mathcal{A}}$ *axiom or has the following form* $C \sqsubseteq B$ *where* B *is a DL-Lite*$_{\mathcal{A}}$ *concept (as in Definition 2.5).*

A *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ ontology is a finite set of such axioms, but with an additional syntactic restriction.

Definition 2.8 (*DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ ontology)**.** *A DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ ontology (or TBox) $\mathcal{O}$ *is a finite set of DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ *axioms such that (i) for each* $p \in \mathbf{R}^{\pm} \cup \mathbf{F}$ *such that* (funct p)*,* $\mathcal{O}$ *does not contain any axiom* $p' \sqsubseteq p$*, and (ii) for each* $f \in \mathbf{F}^{\pm}$*,* $\mathcal{O}$ *does not contain any axiom having* $\exists f$ *on the right-hand side.*

The first condition is inherited from *DL-Lite*$_{\mathcal{A}}$, while the second is required due to the interpretation of features as *closed predicates* (condition imposed for evaluating queries in the presence of aggregates to avoid empty answers [CKNT08]) and the fact that unrestricted use of closed predicates leads to intractability [LSW19].

Whenever the data-value domain is clear from the context we omit the subscript $\mathcal{D}$. Using concept inclusion axioms expressible in *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$, we can encode what is an instance of SevereMCI or an instance of EfficientIntervention, according to the MCI reporting ontology. If such concept

inclusion axioms are added to $\mathcal{O}_{MCI}$, then we have to discard concept inclusions axioms in which features are existentially implied. For example, MCI $\sqsubseteq \exists$date is no longer allowed, since all possible dates must be known in order to apply aggregating functions. However, this restriction could be eased for features which do not participate in aggregating concepts.

Example 2.3. *In our running example, a severe incident is defined as having the total number of casualties greater or equal to 100. We can express such property using aggregating concept* $\geq_{100}$ (sum casualties). *Likewise, the concept* $\geq_{0.8}$ (avg effectiveness) *captures each instance for which the average effectiveness measurements are greater or equal to 0.8. To encode the fact that each such instance must be labeled with concept name* SevereMCI, *respectively with* EfficientInt, *we consider* $\mathcal{O}^{\text{agg}}_{MCI}$ *which*

- *extends* $\mathcal{O}_{MCI}$ *with the following axioms:*

$$\geq_{100} (\text{sum casualties}) \sqsubseteq \text{SevereMCI} \qquad \geq_{0.8} (\text{avg effectiveness}) \sqsubseteq \text{EfficientInt},$$

- *and also drops the following axioms:*

$$\text{MCI} \sqsubseteq \exists\text{date} \qquad \text{Intervention} \sqsubseteq \exists\text{type}.$$

$\mathcal{EL}$ ontologies. A different family of DLs is the $\mathcal{EL}$ family [BBL05] tailored for capturing life science domains, among others. In addition to *DL-Lite*, they allow conjunction and existential restrictions on the left-hand-side of concept inclusions. We recall the syntax and semantics of language $\mathcal{ELI}$ and its extension $\mathcal{ELHI}_\bot$.

Definition 2.9 ($\mathcal{ELI}$, $\mathcal{ELHI}_\bot$ syntax, semantics)**.** *A* $\mathcal{ELI}$ concept expression *is defined as follows:*

$$B := \top \mid A \mid \exists r.B \mid B_1 \sqcap B_2 \qquad r := p \mid p^-$$

where $A \in \mathbf{C}$, $p \in \mathbf{R}$. *Concepts of shape* B *are called* basic concepts, *with* B_1, B_2 *denoting as well basic concepts. The semantics of such expressions is also captured in Table 2.1.*

An $\mathcal{ELI}$ axiom *is any axiom of the form* $B_1 \sqsubseteq B_2$, *where* B_1 *and* B_2 *are basic concepts. An* $\mathcal{ELHI}_\bot$ axiom *is any* $\mathcal{ELI}$ *axiom or an axiom in one of the following forms:*

$$r_1 \sqsubseteq r_2, \quad \text{disj}(B_1, B_2), \quad \text{disj}(r_1, r_2),$$

where $r_1, r_2 \in \mathbf{R}^\pm$, *and* B_1, B_2 *are basic concepts.*

Concept expressions in $\mathcal{ELI}$ are strictly more expressive than in *DL-Lite*$_\mathcal{A}$. For example, we can encode that in the case of a severe fire, an intervention must be requested: SevereMCI $\sqcap$ Fire $\sqsubseteq$ $\exists$requiresInt.

Semantics of $\mathit{DL\text{-}Lite}_{\mathcal{A}}^{(\mathsf{agg})}$ and $\mathcal{ELI}$ concepts under $\mathcal{I}$			
Top concept	$\top^{\mathcal{I}}$	$=$	$\Delta_{\mathbf{O}}^{\mathcal{I}}$
Data-value domain	$\mathcal{D}^{\mathcal{I}}$	$=$	$\Delta_{\mathcal{D}}^{\mathcal{I}}$
Concept name	$A^{\mathcal{I}}$	$\subseteq$	$\Delta_{\mathbf{O}}^{\mathcal{I}}$
Role	$r^{\mathcal{I}}$	$\subseteq$	$\Delta_{\mathbf{O}}^{\mathcal{I}} \times \Delta_{\mathbf{O}}^{\mathcal{I}}$
Feature	$f^{\mathcal{I}}$	$\subseteq$	$\Delta_{\mathbf{O}}^{\mathcal{I}} \times \Delta_{\mathbf{V}}^{\mathcal{I}}$
Inverse relation	$(p^{-})^{\mathcal{I}} = \{(c,c') \mid (c',c) \in p^{\mathcal{I}}\}$		
Role existential restriction	$(\exists s.\top)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \exists d'.(d,d') \in s^{\mathcal{I}} \text{ and } d \in \Delta_{\mathbf{O}}^{\mathcal{I}}\}$		
Full role existential restriction	$(\exists r.A)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \exists d'.(d,d') \in r^{\mathcal{I}} \text{ and } d \in A^{\mathcal{I}}\}$		
Feature existential restriction	$(\exists f)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \exists v.(d,v) \in f^{\mathcal{I}} \text{ and } v \in \Delta_{\mathbf{V}}^{\mathcal{I}}\}$		
Aggregate concept	$(\odot_d \mathsf{agg}_D\, f)^{\mathcal{I}} = \{o \in \Delta^{\mathcal{I}} \mid \mathsf{agg}_D(\{\!\{v \in \Delta_{\mathcal{D}}^{\mathcal{I}} \mid (o,v) \in f^{\mathcal{I}}\}\!\}) \odot_D d\}$		
Conjunction	$(B_1 \sqcap B_2)^{\mathcal{I}} = B_1^{\mathcal{I}} \cap B_2^{\mathcal{I}}$		

Table 2.1: Semantics of the considered DLs concept and role expressions. We use p to denote either a role or a feature, s denotes a possibly inverse role or feature name.

2.1.3 Semantics of DL Ontologies

The semantics of $\mathit{DL\text{-}Lite}_{\mathcal{A}}$, $\mathit{DL\text{-}Lite}_{\mathcal{A}}^{\mathsf{agg}}$ and $\mathcal{ELI}$ expressions are defined in Table 2.1. We consider that interpretations adopt the SNA. A model of a DL ontology is an interpretation that satisfies each axiom.

Definition 2.10 (Modelhood)**.** *Let $\mathcal{I}$ be an interpretation. For a DL language $\mathcal{L}$, let α be an axiom in $\mathcal{L}$. We say that $\mathcal{I}$* satisfies *α, written $\mathcal{I} \vDash \alpha$, when one of the following cases apply:*

- *If α is of form $C_1 \sqsubseteq C_2$, then $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$,*
- *If α is of form $p_1 \sqsubseteq p_2$, then $p_1^{\mathcal{I}} \subseteq p_2^{\mathcal{I}}$,*
- *If α is of form $(\mathsf{funct}\ \ p)$ then whenever $(c,c_1) \in p^{\mathcal{I}}$ and $(c,c_2) \in p^{\mathcal{I}}$, then $c_1 = c_2$,*
- *If α is of form $\mathsf{disj}(C_1, C_2)$, then $C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} = \emptyset$,*
- *If α is of form $\mathsf{disj}(p_1, p_2)$, then $p_1^{\mathcal{I}} \cap p_2^{\mathcal{I}} = \emptyset$,*

where C_1, C_2 are concepts expressions in $\mathcal{L}$ and p, p_1, p_2 are role or feature expressions in $\mathcal{L}$.

For a given DL ontology $\mathcal{O}$ in $\mathcal{L}$, we say that an interpretation $\mathcal{I}$ models *$\mathcal{O}$, written $\mathcal{I} \vDash \mathcal{O}$, if $\mathcal{I}$ satisfies each $\alpha \in \mathcal{O}$.*

The classical reasoning problems with respect to a DL ontology are as follows. Let $\mathcal{O}$ be a given DL ontology in language $\mathcal{L}$. For $\mathcal{O}$, the following decision problems are of interest:

1. *Ontology satisfiability:* Test whether $\mathcal{O}$ has a model. In this case we say that $\mathcal{O}$ is *satisfiable*, otherwise we say that $\mathcal{O}$ is *unsatisfiable*.
2. *Concept satisfiability w.r.t. $\mathcal{O}$:* For a given DL concept C, test if $C^{\mathcal{I}} \neq \emptyset$, for some $\mathcal{I} \vDash \mathcal{O}$.
3. *Subsumption testing w.r.t $\mathcal{O}$:* For a given axiom $C_1 \sqsubseteq C_2$, test if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$, for each $\mathcal{I} \vDash \mathcal{O}$. In this case we write $\mathcal{O} \vDash C_1 \sqsubseteq C_2$, otherwise we write $\mathcal{O} \nvDash C_1 \sqsubseteq C_2$.

In this thesis we do not focus on concept satisfiabiliy or subsumption testing as they can be reduced to ontology unsatisfiability.

2.1.4 Normal Form

We use notation *DL-Lite*$_{\mathcal{A}}^{(\mathsf{agg})}$ for an ontology that is in either *DL-Lite*$_{\mathcal{A}}$ or *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$. We will further consider that *DL-Lite*$_{\mathcal{A}}^{(\mathsf{agg})}$ and $\mathcal{ELHI}_{\bot}$ ontologies are given in the following restricted syntactical form, called *normal form* which ensures that in each axiom must always be a primitive symbol (i.e., concept, role or feature name).

Normal Form *DL-Lite*$_{\mathcal{A}}^{(\mathsf{agg})}$. Formally, we say that a *DL-Lite*$_{\mathcal{A}}^{(\mathsf{agg})}$ ontology is in *normal form* whenever each axiom is in one of the forms:

$$\begin{array}{llll} A \sqsubseteq \exists p & \exists p \sqsubseteq A & r^- \sqsubseteq r' & \mathsf{disj}(A, A') \quad (\mathsf{funct}\ \ p) \\ \odot_d \mathsf{agg}_D \sqsubseteq A & \exists f^- \sqsubseteq D & p_1 \sqsubseteq p_2 & \mathsf{disj}(p_1, p_2) \end{array}$$

where $A, A' \in \mathbf{C}$, $r, r' \in \mathbf{R}$, $f \in \mathbf{F}$, $p, p_1, p_2 \in \mathbf{R} \cup \mathbf{F}$ and D is a data-value domain.

We can transform every *DL-Lite*$_{\mathcal{A}}^{(\mathsf{agg})}$ ontology into an equivalent one that respects the normal form by doing the following transformations: (i) for each full existential restriction axiom of the form $B \sqsubseteq \exists s.A$ we replace it with the following axioms $B \sqsubseteq \exists s_{aux}$, $\exists s_{aux}^- \sqsubseteq A$ and $s_{aux} \sqsubseteq s$, where s_{aux} is a fresh role name, and (ii) for each general or aggregating concept C we use a fresh concept name A_C adding an axiom $C \sqsubseteq A_C$ if C appears on the left-hand-side of an axiom or $A_C \sqsubseteq C$ otherwise, and then replacing C by A_C in all other axioms.

Such transformation uses polynomially many fresh symbols.

Normal Form $\mathcal{ELHI}_{\bot}$. Similarly, an $\mathcal{ELHI}_{\bot}$ ontology is in *normal form* if it only contains inclusions of the following form:

$$A \sqsubseteq A', \quad A_1 \sqcap A_2 \sqsubseteq A, \quad A \sqsubseteq \exists r.A', \quad \exists r.A \sqsubseteq A', \quad r^- \sqsubseteq r', \quad \mathsf{disj}(A, A'), \text{ or } \mathsf{disj}(r, r'),$$

where $A, A', A_1, A_2 \in \mathbf{C}$, $r, r' \in \mathbf{R}$. The normalization rules use the same "naming principle" as before which is now applied also to conjunction of complex concepts and nested existential restrictions [BHLS17].

Let $\mathsf{sig}(\mathcal{O})$ denote the *signature* of any given ontology $\mathcal{O}$, consisting of all concept, role and feature names appearing in $\mathcal{O}$.

Proposition 2.1. *For any arbitrary ontology $\mathcal{O}$ in DL-Lite*$_{\mathcal{A}}^{(\mathsf{agg})}$ *or $\mathcal{ELHI}_{\bot}$ there exists an ontology $\mathcal{O}'$ in normal form of the same DL language such that:*

a) $\mathsf{sig}(\mathcal{O}) \subseteq \mathsf{sig}(\mathcal{O}')$,
b) each $\mathcal{I}'$ model of $\mathcal{O}'$ restricted to $\mathsf{sig}(\mathcal{O})$ *is also a model of $\mathcal{O}$, and*
c) each $\mathcal{I}$ model of $\mathcal{O}$ can be extended over $\mathsf{sig}(\mathcal{O}')$ *into $\mathcal{I}'$ such that $\mathcal{I}'$ is a model of $\mathcal{O}'$.*

2.1.5 Reasoning with ABoxes

A *DL dataset*, or *ABox* in DL jargon, consists of a set of facts encoding concept instance declarations and explicit relations between constants, which are part of the assertional knowledge in the domain of interest. The semantics of ABoxs are also given in terms of interpretations.

In this thesis, from now on, we assume that interpretations use the SNA, which is important in the presence of aggregation and functional constraints as argued in [KKM+16]: for *DL-Lite*$_{\mathcal{A}}$ the complexity of reasoning increases when the UNA is dropped [CGL+09b], while for *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ the dropping of the SNA can lead to empty answer when evaluating queries [CKNT08].

Definition 2.11 (ABox)**.** *An* ABox (or dataset) $\mathcal{A}$ *is a finite set of* concept assertions *of the form* $A(a)$, role assertions *of the form* $r(a, b)$, *or* feature assertions *of the form* $f(a, v)$, *where* $A \in \mathbf{C}$, $r \in \mathbf{R}$, $f \in \mathbf{F}$, $a, b \in \mathbf{K_O}$ *and* $v \in \mathbf{K_V}$. *The set of constants occurring in* $\mathcal{A}$ *is denoted by* $cst(\mathcal{A})$, *among which we distinguish* $ind(\mathcal{A}) = cst(\mathcal{A}) \cap \mathbf{K_O}$ *as* the set of objects (or individuals) *and* $val(\mathcal{A}) = cst(\mathcal{A}) \cap \mathbf{K_V}$ *as* the set of data values.

An interpretation $\mathcal{I}$ satisfies an assertion α, *written* $\mathcal{I} \models \alpha$ *if (i)* $\alpha = A(a)$ *and* $a \in A^{\mathcal{I}}$, *or (ii)* $\alpha = p(c_1, c_2)$ *and* $(c_1, c_2) \in p^{\mathcal{I}}$. *We say that* $\mathcal{I}$ satisfies a dataset $\mathcal{A}$, *written* $\mathcal{I} \models \mathcal{A}$, *if* $\mathcal{I}$ *satisfies each assertion in* $\mathcal{A}$.

Note that a dataset is always satisfiable. In fact, any dataset can be viewed as an interpretation, which is sometimes convenient to do. In the same way, an interpretation can be viewed as a (possibly infinite) dataset. Therefore, given a dataset $\mathcal{A}$, the interpretation $\mathcal{I}_{\mathcal{A}}$ *corresponding to* $\mathcal{A}$ is defined as follows:

$$\Delta_{\mathbf{O}}^{\mathcal{I}_{\mathcal{A}}} = ind(\mathcal{A}) \qquad \Delta_{\mathbf{V}}^{\mathcal{I}_{\mathcal{A}}} = val(\mathcal{A})$$
$$A^{\mathcal{I}_{\mathcal{A}}} = \{a \mid A(a) \in \mathcal{A}\} \qquad p^{\mathcal{I}_{\mathcal{A}}} = \{(c_1, c_2) \mid p(c_1, c_2) \in \mathcal{A}\},$$

for each $A \in \mathbf{C}$ and $p \in \mathbf{R} \cup \mathbf{F}$. Conversely, for an interpretation $\mathcal{I}$, the (possible infinite) dataset $\mathcal{A}_{\mathcal{I}}$ *corresponding to* $\mathcal{I}$ contains for each $a \in A^{\mathcal{I}}$ an assertion $A(a)$, and for each $(c_1, c_2) \in p^{\mathcal{I}}$ an assertion $p(c_1, c_2)$.

A standard reasoning problem in DLs is to test if a given ABox is consistent with respect to a DL ontology. In general, this means that one has to check if there exists an interpretation that satisfies each assertion in the data and which models the ontology. However, in some cases, the notion of modelhood for a dataset and an ontology can be strictly more restrictive, as it is the case for *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$. We define below the dataset semantics with respect to each of the previously introduced ontology languages.

The semantics of a KB $(\mathcal{O}, \mathcal{A})$ is defined in terms of interpretations that model both $\mathcal{O}$ and $\mathcal{A}$. When $\mathcal{O}$ contains aggregating concepts, it is convenient to restrict the models to those that do not allow other feature assertions than those that follow directly from the data and the ontology.

Definition 2.12 (Knowledge Base)**.** *Let* $\mathcal{A}$ *be an ABox and* $\mathcal{O}$ *a DL ontology. The pair* $(\mathcal{O}, \mathcal{A})$ *is called a* knowledge base (KB).

An interpretation $\mathcal{I}$ *is* feature closed for $(\mathcal{O}, \mathcal{A})$ *if for each* $f \in \mathbf{F}$, $f^{\mathcal{I}} = \{(a, v) \mid f'(a, v) \in \mathcal{A}$ *and* $f' \sqsubseteq_{\mathcal{O}}^{*} f\}$, *where* $\sqsubseteq_{\mathcal{O}}^{*}$ *denotes the reflexive and transitive closure of* $\sqsubseteq$ *in* $\mathcal{O}$.

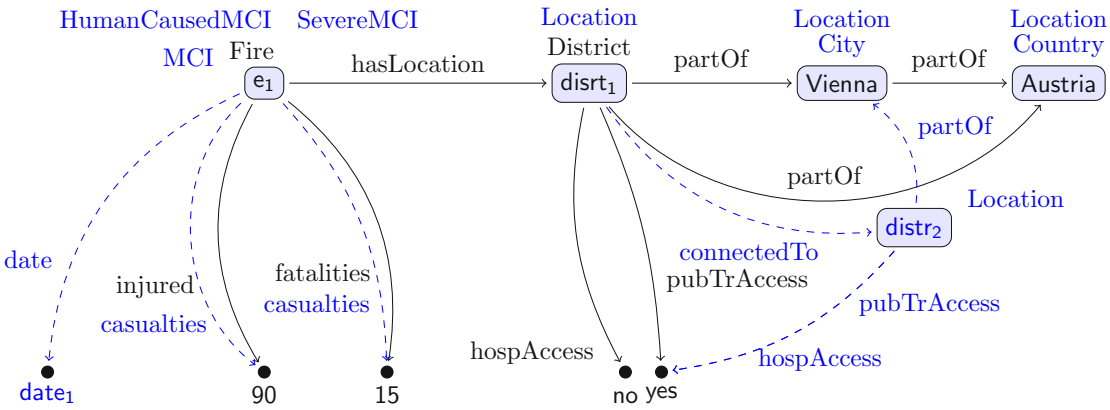


Figure 2.1: Example of the semantics of a dataset $\mathcal{A}$ w.r.t. $\mathcal{O}_{MCI}$ in terms of a model $\mathcal{I}$.

An interpretation $\mathcal{I}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}$, *written $\mathcal{I} \models (\mathcal{O}, \mathcal{A})$, if $\mathcal{I}$ satisfies $\mathcal{A}$ and models $\mathcal{O}$. If $\mathcal{O}$ is in DL-Lite$_{\mathcal{A}}^{\mathsf{agg}}$ then $\mathcal{I}$ must additionally be feature closed for $(\mathcal{O}, \mathcal{A})$.*

As argued by Kharlamov et al. [KKM+16], the restriction for *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ to models that are feature closed does not increase the complexity of reasoning. This is due to the fact that existential restrictions concerning features do not occur on the right-hand-side of inclusion axioms.

The following example illustrates an interpretation is a model of a dataset with respect to the $\mathcal{O}_{MCI}$ ontology.

Example 2.4. *Consider the following ABox $\mathcal{A}$ which encodes facts such as* e1 *is a fire incident, with multiple reported casualties and it occurs in the first district of Vienna. In addition to the previously introduced vocabulary, the ABox contains as features:* hosAccess *to denote whether a hospital is accessible from some location, and* pubTrAccess *to denote if some public transport is available on locations.*

$$\begin{aligned}\mathcal{A} = \{&\mathsf{Fire}(\mathsf{e1}), \mathsf{hasLocation}(\mathsf{e1}, \mathsf{distr}_1), \mathsf{District}(\mathsf{distr}_1), \mathsf{partOf}(\mathsf{distr}_1, \mathsf{Vienna}),\\ &\mathsf{partOf}(\mathsf{Vienna}, \mathsf{Austria}), \mathsf{partOf}(\mathsf{distr}_1, \mathsf{Austria}), \mathsf{injured}(\mathsf{e1}, 90), \mathsf{fatalities}(\mathsf{e1}, 15),\\ &\mathsf{hospAccess}(\mathsf{distr}_1, \mathsf{no}), \mathsf{pubTrAccess}(\mathsf{distr}_1, \mathsf{yes})\}.\end{aligned}$$

We consider interpretation $\mathcal{I}$ with domain:

$$\Delta^{\mathcal{I}} = \{\mathsf{e1}, \mathsf{distr}_1, \mathsf{Vienna}, \mathsf{Austria}, 90, 15, \mathsf{no}, \mathsf{yes}, \mathsf{distr}_2, \mathsf{date}_1\},$$

and which interprets each symbol in the vocabulary of $\mathcal{O}_{MCI}$ (from Example 2.2) as follows:

$$\begin{aligned}
\text{Fire}^{\mathcal{I}} = \text{HumanCausedMCI}^{\mathcal{I}} = \text{SevereMCI}^{\mathcal{I}} = \text{MCI}^{\mathcal{I}} &= \{\mathsf{e1}\},\\
\text{District}^{\mathcal{I}} &= \{\mathsf{distr}_1\}, & \text{City}^{\mathcal{I}} &= \{\mathsf{Vienna}\},\\
\text{Country}^{\mathcal{I}} &= \{\mathsf{Austria}\}, & \text{Location}^{\mathcal{I}} &= \{\mathsf{distr}_1, \mathsf{distr}_2, \mathsf{Vienna}, \mathsf{AT}\},\\
\text{date}^{\mathcal{I}} &= \{(\mathsf{e1}, \mathsf{date}_1)\}, & \text{hasLocation}^{\mathcal{I}} &= \{(\mathsf{e1}, \mathsf{distr}_1)\},\\
\text{injured}^{\mathcal{I}} &= \{(\mathsf{e1}, 90)\}, & \text{fatalities}^{\mathcal{I}} &= \{(\mathsf{e1}, 15)\},\\
\text{hospAccess}^{\mathcal{I}} &= \{(\mathsf{distr}_1, \mathsf{no}), (\mathsf{distr}_2, \mathsf{yes})\} & \text{pubTrAccess}^{\mathcal{I}} &= \{(\mathsf{distr}_1, \mathsf{yes}), (\mathsf{distr}_2, \mathsf{yes})\},\\
\text{partOf}^{\mathcal{I}} &= \{(\mathsf{distr}_1, \mathsf{Vienna}), (\mathsf{distr}_2, \mathsf{Vienna}),\\
&\quad (\mathsf{Vienna}, \mathsf{Austria})\}, & \text{connectedTo}^{\mathcal{I}} &= \{(\mathsf{distr}_1, \mathsf{distr}_2)\}.
\end{aligned}$$

All other symbols are interpreted as $\emptyset$. Note that each fact in $\mathcal{A}$ is satisfied by $\mathcal{I}$, and in addition $\mathcal{I}$ contains some constants not in $\mathcal{A}$: date_1 which denotes the date of the incident e_1, and an additional location distr_2 which is also part of Vienna. Given that $\mathcal{I}$ satisfies each axiom in $\mathcal{O}_{MCI}$, it is also a model of $\mathcal{O}_{MCI}$.

Figure 2.1 illustrates why $\mathcal{I}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}_{MCI}$. In the figure, boxed nodes denote object constants, while dot nodes denote value constants. Node labels denote concept assertions while edge labels denote role or feature assertions. In order to model $\mathcal{A}$ and $\mathcal{O}_{MCI}$ there are implicit facts, inferred from existing ABox assertions and ontology axioms, which must be satisfied. Moreover, $\mathcal{I}$ can also include other additional facts which are not necessarily derived from existing knowledge. All implicit and additional facts in $\mathcal{I}$ are colored in blue.

Let us now focus on $\mathcal{O}^{\mathsf{agg}}_{MCI}$ previously defined in Example 2.3. In this case, $\mathcal{I}$ is no longer a model given that for feature date*, there exists pair $(\mathsf{e}_1, \mathsf{date}_1)$ which is not present nor derivable from the data (using the reflexive and transitive closure of $\sqsubseteq$ in $\mathcal{O}^{\mathsf{agg}}_{MCI}$). However, by dropping such pair from $\text{date}^{\mathcal{I}}$, all axioms in $\mathcal{O}^{\mathsf{agg}}_{MCI}$ are still satisfied. Hence, for $\mathcal{I}'$ which agrees with $\mathcal{I}$ except for $\text{date}^{\mathcal{I}'} = \emptyset$ and for $\text{hospAccess}^{\mathcal{I}} = \{(\mathsf{distr}_1, \mathsf{no})\}$ and $\text{pubTrAccess}^{\mathcal{I}} = \{(\mathsf{distr}_1, \mathsf{yes})\}$, we have that $\mathcal{I}'$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}^{\mathsf{agg}}_{MCI}$.*

Whenever for a given KB $(\mathcal{O}, \mathcal{A})$ a model exists, we say that $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$. ABox consistency is an important prerequisite for various reasoning tasks. A given ABox is trivially consistent whenever the ontology does not contain *negative constraints* (i.e., disjointness or functionality axioms). In the presence of negative axioms, the existence of a model is no longer guaranteed since there can be that no interpretation that models the ABox and that satisfies all inclusion axioms without violating a negative axiom.

Definition 2.13 (ABox consistency)**.** *Let $\mathcal{A}$ be an ABox and $\mathcal{O}$ an ontology. We say that $\mathcal{A}$ is* consistent w.r.t. *$\mathcal{O}$ if and only if there exists an interpretation $\mathcal{I}$ that is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}$. Whenever $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$ we say that the KB $(\mathcal{O}, \mathcal{A})$ is* satisfiable*.*

Ensuring consistency is important since otherwise *ex falso quodlibet*—the principle in classical logic meaning "from falsehood everything follows"— becomes applicable when testing logical entailment. Therefore, a central decision problem for any given KB is:

$\mathcal{L}$-ABox Consistency	
Input:	An ontology $\mathcal{O}$ in $\mathcal{L}$ and an ABox $\mathcal{A}$.
Question:	Is $\mathcal{A}$ consistent w.r.t. $\mathcal{O}$?

2.2 Ontology-mediated Query Answering

Query answering is one of the most fundamental tasks in data management. However, in presence of ontologies evaluating queries is more challenging compared to querying plain data, since the main goal is to leverage the domain knowledge for complete answers, a task that boils down to logical entailment. In spite of the fact that testing logical entailment in first-order logic is in general undecidable, the expressivity of the ontology and query languages does play a role in ensuring decidability and in designing efficient answering techniques.

2.2.1 Query Languages

In this thesis, we consider queries that can be formulated in first-order logic (FOL). Our focus is mainly on so-called *conjunctive queries* (CQs), which are FOL expressions composed using conjunction and existential quantification only. CQs are of interest considering that they capture the majority of queries that occur in practical settings. Moreover, it is also a well-studied query language in relational databases since it has the same expressive power as the *select-project-join* fragment of the SQL query language.

Query atoms. Let $\mathbf{V}$ be a countable infinite set of *variables*. A *term* t is either a variable or an element in $\mathbf{K}$. A *query atom* has one of the following forms: $A(x)$, $p(x, y)$ or $t_1 = t_2$, where $x, y \in \mathbf{V}$, $A \in \mathbf{C}$, $p \in \mathbf{R} \cup \mathbf{F}$ and t_1, t_2 are terms. An *aggregating query atom* is an atom of form $C(x)$, where C is a *DL-Lite*$_{\mathcal{A}}^{\text{agg}}$ aggregating concept.

First-order query language. We use query atoms as primitive FOL formulas and we recall that FOL formulas are constructed inductively as follows: a) a query atom is a formula, b) if ϕ is a formula then $\neg\phi$, $\exists x.\phi$, $\forall x.\phi$ are formulas, and c) if ϕ_1, ϕ_2 are formulas, then $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$ and $\phi_1 \rightarrow \phi_2$ are also formulas.

For a FOL formula ϕ, we denote by $term(\phi)$ the set of *terms in* ϕ among which we distinguish : $vars(\phi)$— the set of *variables in* ϕ, and $free(\phi)$— the set of variables that are not in the scope of any of the quantifiers $\exists$ or $\forall$ (i.e., they occur freely). For a FOL formula ϕ, we sometimes write $\phi(x_1, \ldots, x_n)$ to denote that tuple $(x_1, \ldots, x_n)$ are the *free variables* in ϕ. If a variable occurs in the scope of a quantifier (i.e, is not free), then it is called *bound*. A *first-order query* (FO-query) is a FOL formula $\phi(\vec{x})$ and, in this context, $\vec{x}$ are called *answer variables*.

(Union of) conjunctive queries. FO queries that are formulated using only conjunction and existential quantification are called *conjunctive queries* (CQ). A generalization of such language is to take the disjunction of a set of CQs.

Definition 2.14 ((Union of) CQs)**.** *A* conjunctive query $q(\vec{x})$ *is a FOL query of the form* $\exists\vec{y}.\phi(\vec{x},\vec{y})$*, such that* ϕ *is a FOL formula constructed using only* $\wedge$*, and* $\vec{y}$ *denote all the bound variables in* q*.*

A union of conjunctive queries *(UCQ)* $Q(\vec{x})$ *is a FOL query of the form* $\exists\vec{y}.(\phi_1(\vec{x},\vec{y}_1)\vee\cdots\vee\phi_n(\vec{x},\vec{y}_n))$*, such that, for* $1\leq i\leq n$*, each* $\phi_i(\vec{x},\vec{y}_i)$ *is a FOL query constructed using only* $\wedge$*, and* $\vec{y}=\bigcup_{i=1}^{n}\vec{y}_i$ *denotes all the bound variables in* Q*.*

Example 2.5. *Given the MCI ontology, we can express using a CQ, the following information request: find all severe human-caused incidents that have occurred in Austria:*

$$q(x)\leftarrow\exists yz\,\mathsf{HumanCausedMCI}(x)\wedge\mathsf{SevereMCI}(x)\wedge\mathsf{hasLocation}(x,y)\wedge\mathsf{partOf}(y,z)\wedge z=\mathsf{Austria}.$$

Query languages with aggregation. We consider a first extension of (U)CQs that allow for aggregating query atoms in the query expressions. Such query language is useful to exploit *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ ontologies when querying.

Definition 2.15 ((U)CQs with aggregating atoms)**.** *A* CQ with aggregating query atoms (CQA) $q(\vec{x})$ *is an expression of the form* $\exists\vec{y}.\phi_{\mathsf{agg}}(\vec{x},\vec{y})$ *such that* ϕ_{agg} *denotes a conjunction of (possible aggregating) query atoms, and* $\vec{y}$ *denote all the bound variables in* q*.*

Similarly, a UCQ with aggregating query atoms (UCQA) $Q(\vec{x})$ *is an expression of the form* $\exists\vec{y}.(q_1(\vec{x},\vec{y}_1)\vee\cdots\vee q_n(\vec{x},\vec{y}_n))$ *where for* $1\leq i\leq n$*, each* $q_i(\vec{x},\vec{y})$ *is a CQA and* $\vec{y}=\bigcup_{i=1}^{n}\vec{y}_i$ *denotes all the bound variables in* Q*.*

Each (U)CQ is implicitly a (U)CQA. In addition, using a (U)CQA we can capture objects having a certain aggregated value on some particular feature. A concrete example is to consider a query that captures non-severe fires located in Austria. For that, the aggregating concept $<_{100}$ (sum casualties) is used to identify such incidents.

$$q(x)\leftarrow\exists y\,\mathsf{Fire}(x)\wedge<_{100}(\mathsf{sum\ casualties})(x)\wedge\mathsf{hasLocation}(x,y),y=\mathsf{Austria}.$$

Having aggregating concepts in the query does not support some of the basic analytical needs, such as retrieving the output of the aggregating function. Consider for instance, the task to analyze vulnerable locations that are prone to human-caused MCIs. For that, one has to identify locations and the number of MCIs that have occurred in each location. Hence, we expand (U)CQAs such that the aggregating functions can be applied to query terms in order to retrieve the aggregating outcome as part of the answer. Such query language is similar to CQs with aggregating functions, previously introduced in [CKNT08]. We call an *aggregating term* any expression of the form $\mathsf{agg}(z)$, where agg is an aggregating function over some data-value domain $\mathcal{D}$, and z is called an *aggregating variable*.

Definition 2.16 (Analytical queries)**.** *An* analytical CQA $q(\vec{x}, \mathsf{agg}(z))$ *consists of a CQA* $q(\vec{x})$ *and an aggregating term* $\mathsf{agg}(z)$ *such that* z *is a variable in* q *not from* $\vec{x}$*.*

Similarly, an analytical UCQA $Q(\vec{x}, \mathsf{agg}(z))$ *consists of UCQA* $Q(\vec{x})$ *and an aggregating term* $\mathsf{agg}(z)$ *such that* z *is a variable in* Q *not from* $\vec{x}$*.*

For an analytical query, $\vec{x}$ *is called the* grouping variables*, the expression denoted by (U)CQA is called the* condition for $\vec{x}$ *and* $\mathsf{agg}(z)$ *is called the* the measurement for $\vec{x}$*.*

Analytical queries and queries with aggregating concepts complement each other in the sense that each query language can express some information needs that the other cannot.

Example 2.6. *In order to obtain the total number of casualties of MCIs per location where an intervention was required, we can use the following analytical CQ:*

$$q_1(z, \mathsf{sum}(y)) \leftarrow \exists xu\ \mathrm{MCI}(x) \wedge \mathrm{casualties}(x, y) \wedge \mathrm{requiresInt}(x, u) \wedge \mathrm{hasLocation}(x, z).$$

Using an aggregating concept atom in the query we can focus only on incidents with more than 100 casualties:

$$\begin{aligned} q_2(z, \mathsf{sum}(y)) \leftarrow & \exists xu\ \mathrm{MCI}(x) \wedge \mathrm{casualties}(x, y) \wedge \mathrm{requiresInt}(x, u) \wedge \mathrm{hasLocation}(x, z) \\ & \wedge \geq_{100} (\mathsf{sum}\ \mathrm{casualties})(x). \end{aligned}$$

The answers to q_2 *will be tuples of the form* (ℓ, k) *where* ℓ *is a location where an MCI which required some intervention and had more than 100 casualties, and* k *is the total number of casualties at location* ℓ*.*

For each introduced query language, whenever a given query has no answer variables it is called a *Boolean query*.

2.2.2 Ontology-mediated Queries

Ontology-mediated queries (OMQs) are pairs consisting of a query, of some concrete query language, and an ontology in a particular DL dialect. In this thesis, we focus on OMQs where the ontology is in *DL-Lite*$_{\mathcal{A}}^{(\mathsf{agg})}$ and queries are CQs which may contain some form of aggregation. For an overview of other existing OMQ classes, we refer the reader to the in-dept overview [BO15].

Definition 2.17 (Ontology-mediated Queries)**.** *Let* $\mathcal{O}$ *be a DL ontology. An* ontology-mediated query *(OMQ)* $\mathcal{Q}$ *is a pair* $(\mathcal{O}, q(\vec{x}))$ *where* $q(\vec{x})$ *is a CQA. An* analytical OMQ *(OMQ*$^{\mathsf{agg}}$*)* $\mathcal{Q}$ *is a pair* $(\mathcal{O}, (q(\vec{x}, \mathsf{agg}(z))))$ *where* $q(\vec{x}, \mathsf{agg}(z)$ *is an analytical CQA.*

Answering OMQs. Finding answers to a query implies finding all variables assignments that make the query true in a given interpretation. For an OMQ, we need to find all such assignments that hold in *every model* of the ABox w.r.t. the ontology. Each answer to an (ontology-mediated) query is witnessed by a variable assignment, called *query match*.

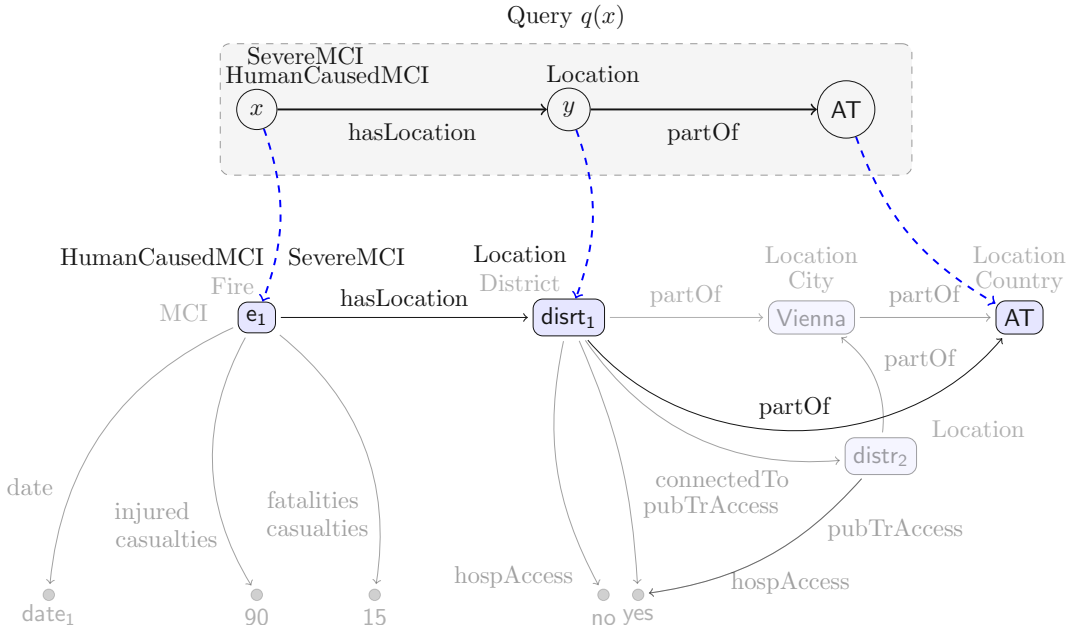


Figure 2.2: Example of matching a query over an interpretation.

Definition 2.18 (Query match)**.** *Let $\varphi(\vec{x})$ be a either a FO-query or a UCQA such that $\vec{x} = (x_1, \ldots, x_n)$. For an interpretation $\mathcal{I}$, let $\vec{a} = (a_1, \ldots, a_n)$ be a tuple of elements in $\Delta^{\mathcal{I}}$ of same arity as $\vec{x}$. We say that $\vec{a}$ is* an answer of $\varphi(\vec{x})$ in $\mathcal{I}$ *if there exists a mapping π : term$(\varphi) \mapsto \Delta^{\mathcal{I}}$ such that:*

(i) $\pi(x_i) = a_i$,
(ii) $\pi(c) = c^{\mathcal{I}}$, *for each* $c \in cst(\varphi)$, *and*
(iii) $\mathcal{I}$ *satisfies* $\varphi\pi$.

In this case we call π a match for φ in $\mathcal{I}$. *We use $ans(\varphi(\vec{x}), \mathcal{I})$ to denote the set of all answers to $\varphi(\vec{x})$ in $\mathcal{I}$.*

Example 2.7. *Recall the query in Example 2.5 and the interpretation in Example 2.4. Figure 2.2 illustrates the fact that for $q(x)$ there exists a query match over $\mathcal{I}$ such that e_1 is an answer of $q(x)$ in $\mathcal{I}$. Faded nodes, edges and labels denote facts in $\mathcal{I}$ that are not relevant for mapping q.*

Without an ontology, the answers to a query over an ABox $\mathcal{A}$ are computed by finding all query mappings over $\mathcal{I}_{\mathcal{A}}$. The semantics of answering OMQs is to find all sets of answers in every model and then taking their intersection.

Definition 2.19 (OMQ certain answers)**.** *Let $\mathcal{A}$ be an ABox and $\mathcal{Q} = (\mathcal{O}, q(\vec{x}))$ an OMQ, with $\vec{x} = (x_1, \ldots, x_n)$. A tuple $\vec{a} = (a_1, \ldots, a_n)$ of constants from $\mathcal{A}$, of same arity as $\vec{x}$ is called a* certain answer of $\mathcal{Q}$ over $\mathcal{A}$ *if $\vec{a}$ is an answer in every model $\mathcal{I}$ of $\mathcal{A}$ w.r.t. $\mathcal{O}$.*

We denote by $cert(q(\vec{x}), \mathcal{O}, \mathcal{A})$ the set of certain answers of $(\mathcal{O}, q(\vec{x}))$ over $\mathcal{A}$.

Example 2.8 (Continued)**.** *Resuming the Example 2.7, when evaluating* $(\mathcal{O}_{MCI}, q(x))$ *over the ABox* $\mathcal{A}$ *from Example 2.4, we have that* $cert(q(x), \mathcal{O}_{MCI}, \mathcal{A}) = \emptyset$*, since no instances of* SevereMCIs *can be deduced. However, in the case of the OMQ* $(\mathcal{O}^{\mathsf{agg}}_{MCI}, q(x))$*,* ev_1 *is a certain answer since from the axiom* $\geq_{100}$ (sum casualties) $\sqsubseteq$ SevereMCI*, we obtain that in each feature closed model of* $\mathcal{A}$ *w.r.t.* $\mathcal{O}^{\mathsf{agg}}_{MCI}$*, constant* ev_1 *is an answer to* $q(x)$*.*

Answering OMQ$^{\mathsf{agg}}$. Intuitively an answer to an analytical query $q(\vec{x}, \mathsf{agg}(\vec{z}))$ over an interpretation $\mathcal{I}$ consists of a tuple $\vec{a}$ such that $\vec{a}$ is an answer to the CQ $q(\vec{x})$ in $\mathcal{I}$, and a numeric value n which is the output value of agg over the multi-set $V = \{\!\{\vec{c} \mid$ there is π a match of q in $\mathcal{I}$ such that $\pi(\vec{z}) = \vec{c}$ and $\pi(\vec{x}) = \vec{a}\}\!\}$. Since the output of the aggregating function is not the same in each model $\mathcal{I}$, the set of certain answers is in general empty for OMQ$^{\mathsf{agg}}$.

For this reason, a new semantics for evaluating OMQ$^{\mathsf{agg}}$, proposed in [CKNT08], is to apply the aggregating function only to sets V that consist of "known values", i.e., constants in the data. Such semantics is denoted as *epistemic certain answer semantics*. We can construct such answers in a natural way by computing the certain answers to the OMQ in which the aggregating variables become answer variables.

Definition 2.20 (OMQ$^{\mathsf{agg}}$ certain answer)**.** *Let* $(\mathcal{O}, q(\vec{x}, \mathsf{agg}(z)))$ *be an OMQ*$^{\mathsf{agg}}$ *and* $\mathcal{A}$ *an ABox. A tuple* $(\vec{a}, n)$*, such that* $\vec{a}$ *is a tuple of constants from* $\mathcal{A}$ *of the same arity as* $\vec{x}$ *and* n *is an element in some data-value domain* $\mathcal{D}$*, is called a* certain answer of $(\mathcal{O}, (q(\vec{x}, \mathsf{agg}(z))))$ over $\mathcal{A}$ *if for*

$$G_{\vec{a}} = \{\!\{d \mid (\vec{a}, d) \in cert(q(\vec{x}, z), \mathcal{O}, \mathcal{A})\}\!\}$$

we have that $\mathsf{agg}_{\mathcal{D}}(G_{\vec{a}}) =_{\mathcal{D}} n$*.*

Example 2.9. *Recall query* $q_1(z, \mathsf{sum}(y))$ *in Example 2.6. Evaluating* $(\mathcal{O}_{MCI}, q_1(z, \mathsf{sum}(y)))$ *over* $\mathcal{A}$ *we obtain that tuple* $(\mathsf{distr}_1, 105)$ *is a certain answer since* e_1 *is a* HumanCausedMCI*, therefore implicitly an* MCI*, that occurs at that particular location, and since* fatalities$(e_1, 15)$*, by axiom* $\exists$fatalities $\sqsubseteq \exists$requiresInt *it implicitly required some intervention. The total number of casualties* 105 *is obtained by reasoning over existing facts using axioms* injured $\sqsubseteq$ casualties *and* fatalities $\sqsubseteq$ casualties *and taking the sum of both injured and fatalities.*

Since evaluating OMQs$^{\mathsf{agg}}$ in an ontology language $\mathcal{L}$ relies on evaluating an OMQs in $\mathcal{L}$, we focus on defining the relevant decision problems, which are central in this thesis, only for OMQs.

$\mathcal{L}$-Instance Checking	
Input:	A DL ontology $\mathcal{O}$ in $\mathcal{L}$, a dataset $\mathcal{A}$ and an assertion α.
Question:	For each $\mathcal{I}$ model of $\mathcal{A}$ w.r.t. $\mathcal{O}$, does $\mathcal{I}$ satisfy α?

$\mathcal{L}$-Certain Answers	
Input:	An OMQ $\mathcal{Q}$ in $\mathcal{L}$, an ABox $\mathcal{A}$ and a tuple $\vec{a}$.
Question:	Is $\vec{a}$ a certain answer of $\mathcal{Q}$ over $\mathcal{A}$?

It is often the case that the ontology and the query is comparably smaller than the data, therefore it is reasonable to assume that the only parameter that is most influential in having efficient query answering techniques is the size of the data. This is a standard assumption in the database setting [Var82], and it has also been considered in the case of DLs [CGL+07]. Formally, let $\mathcal{L}$ be a DL ontology language, $\mathcal{O}$ an ontology in $\mathcal{L}$ and $q(\vec{x})$ a CQ. We are then also interested in the following decision problem.

$\mathcal{L}$-CERTAIN ANSWERS($\mathcal{O}$, q)	
Input:	An ABox $\mathcal{A}$ and a tuple $\vec{a}$.
Question:	Is $\vec{a}$ a certain answer of $(\mathcal{O}, q)$ over $\mathcal{A}$?

Containment of OMQs. In relational databases, query containment is the problem of deciding whether the answers of a query are contained in the answers of another query for any dataset and sometimes the dataset may be required to satisfy particular contraints, such as the case of integrity constraints, problem known as *query containment under constraints*. In our setting, the constraints are encoded in the ontology. The definition can be easily lifted to the OMQ case.

Definition 2.21 (OMQ containment)**.** *Let $(\mathcal{O}, q_1(\vec{x}))$ and $(\mathcal{O}, q_2(\vec{x}))$ be two OMQs defined over the same ontology $\mathcal{O}$. We say that query q_1* is contained in *q_2* w.r.t. *$\mathcal{O}$, written $q_1 \subseteq_{\mathcal{O}} q_2$, if for each ABox $\mathcal{A}$ consistent w.r.t. $\mathcal{O}$ we have that $cert((\mathcal{O}, q_1(\vec{x})), \mathcal{A}) \subseteq cert((\mathcal{O}, q_2(\vec{x})), \mathcal{A})$.*

It is convenient to identify pairs of queries for which containment holds only for a particular dataset. This enables query answering optimizations and designing ontology-mediated techniques to support data exploration, which we discuss in this thesis.

Definition 2.22 (OMQ containment w.r.t ABoxes)**.** *Let $(\mathcal{O}, q_1(\vec{x}))$ and $(\mathcal{O}, q_2(\vec{x}))$ be two OMQs defined over the same ontology $\mathcal{O}$ and let $\mathcal{A}$ be an ABox consistent w.r.t. $\mathcal{O}$. We say that q_1* is contained in *q_2* w.r.t. *$(\mathcal{O}, \mathcal{A})$, written $q_1 \subseteq_{(\mathcal{O},\mathcal{A})} q_2$, if $cert((\mathcal{O}, q_1), \mathcal{A}) \subseteq cert((\mathcal{O}, q_2), \mathcal{A})$.*

Note that containment w.r.t. ABoxes can also be reduced to OMQ answering as by evaluating each OMQ over the given dataset, we can establish the containment relation. However, to decide containment w.r.t. the ontology alone would require more sophisticated techniques, as the answer containment needs to hold for all possible ABoxes.

2.3 Techniques for Answering DL OMQs

For *DL-Lite* family of languages, the main techniques for answering OMQs are based on *query rewriting* and *saturation*. The general idea of the rewriting-based approach is to embed the ontology into the query thus creating a more complex FO-query which is then evaluated over the data alone. Such approach is desirable considering that the certain answers can be computed in a *data-independent* fashion. Saturation-based techniques enrich the data with additional information, implied by existing facts and the ontology, which is then made available for query answering. We

extend the saturation and rewriting approaches for *DL-Lite*$_{\mathcal{A}}$ to capture also *DL-Lite*$_{\mathcal{A}}^{\text{agg}}$ ontologies. Afterwards, we show that we can focus solely on *DL-Lite*$_{\mathcal{A}}$ OMQs since answering *DL-Lite*$_{\mathcal{A}}^{\text{agg}}$ OMQs can be polynomially reduced to answering *DL-Lite*$_{\mathcal{A}}$ OMQs.

2.3.1 Chase Procedure and Construction of the Canonical Model

A KB can have infinitely many models, however to compute certain answers to CQs it is sufficient to evaluate a query over some *canonical model*, which is a model that is universal in the sense that it can be homomorphically mapped into any other model. This also means that a canonical model is in a sense minimal, since it satisfies only facts that are required by the ontology.

Definition 2.23 (Canonical model)**.** *Given an ABox $\mathcal{A}$ and an ontology $\mathcal{O}$, a* canonical model of $\mathcal{A}$ w.r.t. $\mathcal{O}$ *is a model $\mathcal{I}$ of $\mathcal{A}$ w.r.t. $\mathcal{O}$ such that $\mathcal{I} \rhd \mathcal{J}$, for each $\mathcal{J}$ model of $\mathcal{A}$ w.r.t. $\mathcal{O}$.*

Intuitively, given an answer $\vec{a}$ to a CQ $q(\vec{x})$ in some canonical model $\mathcal{I}$, there exists a match π of q in $\mathcal{I}$ such that $\pi(\vec{x}) = \vec{a}$. Since there exists a homomorphism h from $\mathcal{I}$ into each model $\mathcal{J}$ of $\mathcal{A}$ w.r.t. $\mathcal{O}$, we obtain that the composition of π and h yields a match of $q(\vec{x})$ in $\mathcal{J}$, hence $\vec{a}$ is a certain answer.

For a given ontology we say that $\mathcal{O}$ satisfies the *canonical model property* if for each ABox $\mathcal{A}$ a canonical model of $\mathcal{A}$ w.r.t. $\mathcal{O}$ exists.

Proposition 2.2. *Let $\mathcal{O}$ be a DL ontology that satisfies the canonical model property. For any OMQ $(\mathcal{O}, q(\vec{x}))$ and any ABox $\mathcal{A}$ consistent w.r.t. $\mathcal{O}$, we have that*

$$cert(q(\vec{x}), \mathcal{O}, \mathcal{A}) = ans(q(\vec{x}), \mathcal{I}_{\mathcal{O},\mathcal{A}})$$

where $\mathcal{I}_{\mathcal{O},\mathcal{A}}$ is a canonical model of $\mathcal{A}$ w.r.t. $\mathcal{O}$.

It is known that Horn-DLs, such as *DL-Lite* and $\mathcal{EL}$, enjoy the canonical model property. In the case of *DL-Lite*$_{\mathcal{A}}^{(\text{agg})}$, one of the core techniques to construct the canonical model is to extend the ABox with facts that are entailed by the data together with the ontology, procedure which is sometimes referred as *the chase*. We extend the standard definition of the chase procedure of [CGL+07] to capture also axioms specific to *DL-Lite*$_{\mathcal{A}}^{\text{agg}}$.

Definition 2.24 (Chase procedure)**.** *Let $\mathcal{O}$ be a DL-Lite*$_{\mathcal{A}}^{(\text{agg})}$ *ontology and $\mathcal{A}$ an ABox. We consider a function g that takes as input a multi-set of assertions $\mathcal{A}$ and an axiom α, and outputs the effect of applying α to $\mathcal{A}$. We assume that $\mathcal{O}$ is normalized. We define*

$$chase_{\mathcal{O}}(\mathcal{A}) = \bigcup_{j \in \mathbb{N}} \mathcal{A}_j,$$

with $\mathcal{A}_0 = \mathcal{A}$, and $\mathcal{A}_{j+1} = \mathcal{A}_j \cup g_\alpha(\mathcal{A}')$, where $\mathcal{A}' \subseteq \mathcal{A}_j$, $\alpha \in \mathcal{O}$ and $g_\alpha(\mathcal{A}')$ are in one of the cases below:

- *If $\mathcal{A}' = \{A_1(a)\}$, and $A_2(a) \notin \mathcal{A}_j$, then $g_{A_1 \sqsubseteq A_2}(\mathcal{A}') = \{A_2(a)\}$.*

- *If* $\mathcal{A}' = \{A(a)\}$, *and there is no constant* c *such that* $p(a,c) \in \mathcal{A}_j$, *then* $g_{A \sqsubseteq \exists p}(\mathcal{A}') = \{p(a, c_{new})\}$, *where* c_{new} *is a fresh constant not appearing in* $\mathcal{A}_j$.
- *If* $\mathcal{A}' = p(a,c)$, *and* $A(a) \notin \mathcal{A}_j$ *then* $g_{\exists p \sqsubseteq A}(\mathcal{A}') = \{A(a)\}$.
- *If* $\mathcal{A}' = \{p(c_1, c_2)\}$, *and* $p'(c_1, c_2) \notin \mathcal{A}_j$, *then* $g_{p \sqsubseteq p'}(\mathcal{A}') = \{p'(c_1, c_2)\}$.
- *If* $\mathcal{A}' = \{p(c_1, c_2)\}$, *and* $p'(c_2, c_1) \notin \mathcal{A}_j$, *then* $g_{p^- \sqsubseteq p'}(\mathcal{A}') = \{p'(c_2, c_1)\}$.
- *If* $\mathcal{A}' = \{\!\{ f(c, v_1), \ldots, f(c, v_n) \}\!\}$, $\mathsf{agg}(\{\!\{ v_1, \ldots, v_n \}\!\}) \odot n$, *and* $A(c) \notin \mathcal{A}_j$, *then* $g_{\odot_n(\mathsf{agg}\ f) \sqsubseteq A} = \{A(c)\}$

where $A, A_1, A_2 \in \mathbf{C}$, $p, p' \in \mathbf{R} \cup \mathbf{F}$, $f \in \mathbf{F}$. *We denote* $chase^i_{\mathcal{O}}(\mathcal{A}) = \bigcup_{0 \le j \le i} \mathcal{A}_j$ *to be the chase obtained after* i *applications of the rules above. We use* $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ *to denote the interpretation corresponding to* $chase_{\mathcal{O}}(\mathcal{A})$.

If $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$, then $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}$. This holds from the construction of $chase_{\mathcal{O}}(\mathcal{A})$: $\mathcal{A}$ is trivially satisfied, and since axioms in $\mathcal{O}$ are exhaustively applied in such a way that functionality and disjointness constraints are not violated, we obtain that the interpretation corresponding to $chase_{\mathcal{O}}(\mathcal{A})$ models $\mathcal{O}$. Moreover, for *DL-Lite*$_{\mathcal{A}}$, it is known that $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ is a canonical model [CGL$^+$09b]. It is not difficult to see that this holds also for *DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$.

Lemma 2.1. *Let* $\mathcal{O}$ *be a DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ *ontology. For any ABox* $\mathcal{A}$ *consistent w.r.t.* $\mathcal{O}$, $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ *is a model of* $\mathcal{A}$ *w.r.t.* $\mathcal{O}$.

Proof. Let $\mathcal{A}$ be an arbitrary ABox that is consistent w.r.t. $\mathcal{O}$ and we want to show that $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}$. We proceed with showing the following:

Claim 1. *For* $0 \le i$, *if* $\mathcal{I}$ *is a model of* $chase_i$ *w.r.t.* $\mathcal{O}$, *then* $\mathcal{I}$ *is also a model of* $chase_{i+1}$ *w.r.t.* $\mathcal{O}$.

This claim follows almost immediately given that the only additional facts that $chase_{i+1}$ contains follow immediately from some multi-set of facts in $chase_i$ and some axiom in $\mathcal{O}$.

We assume that $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ is not a model of $\mathcal{A}$ w.r.t. $\mathcal{O}$, hence there exists the smallest step k such that $chase_k$ is inconsistent w.r.t. $\mathcal{O}$. Using the above claim we obtain that for each $0 < i < k$, we have that $chase_i$ is inconsistent w.r.t. $\mathcal{O}$ and since $chase_0 = \mathcal{A}$ we obtain a contradiction with the fact that $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$. Thus our assumption is false and we conclude that $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}$. This concludes the lemma proof.

□

From the fact that the chase procedure introduces only the necessary new facts and since it uses fresh constants, it is not difficult to deduce that it is also a canonical model of $\mathcal{A}$ w.r.t. $\mathcal{O}$.

Proposition 2.3. *Let* $\mathcal{O}$ *be a DL-Lite*$^{(\mathsf{agg})}_{\mathcal{A}}$ *ontology. For any ABox* $\mathcal{A}$ *consistent w.r.t.* $\mathcal{O}$, *we have that* $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ *is a canonical model of* $\mathcal{A}$ *w.r.t.* $\mathcal{O}$.

Proof. Let $\mathcal{A}$ be an arbitrary ABox and $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an arbitrary model of $\mathcal{A}$ w.r.t. $\mathcal{O}$. We want to show that $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}} \rhd \mathcal{I}$. Since this is known to hold for any *DL-Lite*$_{\mathcal{A}}$ ontology, it suffices to prove that a homomorphism exists whenever axioms of the form $\odot_d \mathsf{agg}_D \sqsubseteq A$ are applied in the chase procedure. We show this inductively on the chase procedure.

Basic Step. Since $chase_0 = \mathcal{A}$ and $\mathcal{I}$ models $\mathcal{A}$ w.r.t $\mathcal{O}$ it implies that $\mathcal{I}$ it is feature closed. Then by mapping each constant to itself we obtain that a homomorphism exists.

Inductive Step. We assume that $chase^{i+1}_{\mathcal{O}}(\mathcal{A})$ is obtained from $chase^{i}_{\mathcal{O}}(\mathcal{A})$ by applying an axiom of the form $\odot_d \mathsf{agg}_D \sqsubseteq A$. This means that there is some $\mathcal{A}' = \{\!\{f(c, v_1), \ldots, f(c, v_n)\}\!\}$, such that $\mathsf{agg}(\{\!\{v_1, \ldots, v_n\}\!\}) \odot n$, and $A(c) \notin chase^{i}_{\mathcal{O}}(\mathcal{A})$, and $chase^{i+1}_{\mathcal{O}}(\mathcal{A}) = chase^{i}_{\mathcal{O}}(\mathcal{A}) \cup \{A(c)\}$. By induction hypothesis and the fact that $\mathcal{I}$ is feature closed, we obtain that there must be some $c' \in \Delta^{\mathcal{I}}$ such that $\mathcal{I}$ satisfies all $f(c', v_1), \ldots, f(c', v_n)$. Since $v_1, \ldots, v_n$ are constants from $\mathcal{A}$ and $\mathcal{I}$ maps each such constant to itself, we get that $\mathsf{agg}(\{\!\{v_1, \ldots, v_n\}\!\}) \odot n$ holds also under $\mathcal{I}$. Lastly, since $\mathcal{I}$ must satisfy all axioms in $\mathcal{O}$ it must be that $c' \in A^{\mathcal{I}}$.

Hence a homomorphism exists and we have that $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}} \rhd \mathcal{I}$, for each model $\mathcal{I}$ of $\mathcal{A}$ w.r.t. $\mathcal{O}$. □

Therefore, from the above proposition and Proposition 2.2 follows that we can rely on the chase procedure to obtain certain answers.

Corollary 1. *Let $\mathcal{O}$ be a DL-Lite$^{(\mathsf{agg})}_{\mathcal{A}}$ ontology. For any ABox $\mathcal{A}$ consistent w.r.t. $\mathcal{O}$ and any OMQ $(\mathcal{O}, q(\vec{x}))$ we have that $cert(q(\vec{x}), \mathcal{O}, \mathcal{A}) = ans(q(\vec{x}), \mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}})$.*

The chase procedure does not always terminate since the canonical model can be in general infinite. Techniques that either limit the number of the chase steps needed to evaluate the query [BOSX13], or that embed the ontology axioms into the query expression [CGL$^+$07], namely query rewriting, or a combination of both [KLT$^+$10] represent the existing solutions for the evaluation of *DL-Lite* OMQs. We focus next on the query rewriting approach, which we extend to *DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ in a natural way.

2.3.2 Rewriting *DL-Lite*$^{(\mathsf{agg})}_{\mathcal{A}}$ OMQs

Rewriting an OMQ boils down to finding a query, in a particular target query language, that produces the same answers as the original OMQ when evaluated over any dataset.

Definition 2.25 (Rewriting of a OMQ)**.** *Let $(\mathcal{O}, q(\vec{x}))$ be an OMQ in $\mathcal{L}_o$, and $\mathcal{L}_q$ be the target query language.*

A query $\varphi_{\mathcal{O}}$ is called an $\mathcal{L}_q$-rewriting of q w.r.t. *$\mathcal{O}$ if $\varphi_{\mathcal{O}}$ is in $\mathcal{L}_q$, and for any ABox $\mathcal{A}$ consistent w.r.t. $\mathcal{O}$*

$$cert(q(\vec{x}), \mathcal{O}, \mathcal{A}) = cert(\varphi_{\mathcal{O}}, \emptyset, \mathcal{A}).$$

We say that $\mathcal{L}_o$-OMQs are $\mathcal{L}_q$-rewritable *if for each OMQ in $\mathcal{L}_o$ there exists an $\mathcal{L}_q$-rewriting.*

In general, the preferred target rewriting language is FOL, since FO-queries can be evaluated efficiently using off-the-shelves database query answering engines. For *DL-Lite*$_\mathcal{A}$ OMQs, FO-rewritability holds.

Proposition 2.4 ([ACKZ09]). *DL-Lite*$_\mathcal{A}$*-OMQs are UCQ-rewritable.*

The existing rewriting algorithm for *DL-Lite*$_\mathcal{A}$, denoted as *PerfectRef* and presented in [CGL$^+$07], proposes to reformulate query atoms by applying ontology axioms to consider all possible specializations. The union of all different specializations is then evaluated over the data alone. For example, recall query $q(x)$ in Example 2.5. The rewriting of q w.r.t. $\mathcal{O}_{MCI}$ is the UCQ:

$$\begin{aligned} q_{\mathcal{O}_{MCI}}(x) \leftarrow \exists yz\ & (\text{HumanCausedMCI}(x) \wedge \text{SevereMCI}(x) \wedge \text{hasLocation}(x, y) \wedge \text{partOf}(y, z) \\ & \wedge z = \text{Austria}) \vee \\ & (\text{Fire}(x) \wedge \text{SevereMCI}(x) \wedge \text{hasLocation}(x, y) \wedge \text{partOf}(y, z) \wedge z = \text{Austria}) \vee \\ & (\text{Explosion}(x) \wedge \text{SevereMCI}(x) \wedge \text{hasLocation}(x, y) \wedge \text{partOf}(y, z) \wedge z = \text{Austria}). \end{aligned}$$

We present next the rewriting procedure for *DL-Lite*$_\mathcal{A}^{(\text{agg})}$ OMQs. In the following, it is convenient to consider a CQA as a set of query atoms. We recall that two query atoms α_1 and α_2 *unify* if and only if there exists a substitution $\sigma : \mathit{vars}(\{\alpha_1, \alpha_2\}) \mapsto \mathit{term}(\{\alpha_1, \alpha_2\})$ such that $\alpha_1\sigma = \alpha_2\sigma$. The *most general unifier* of α_1 and α_2 is a unifier σ such that for each unifier θ there exists a substitution θ' such that $\theta = \sigma \circ \theta'$. We extend the rewriting rules of *DL-Lite*$_\mathcal{A}$ from [CGL$^+$09b] to capture also *DL-Lite*$_\mathcal{A}^{\text{agg}}$ OMQs in a natural way.

Definition 2.26 (Rewriting of *DL-Lite*$_\mathcal{A}^{(\text{agg})}$ OMQs). *Let $\mathcal{O}$ be a DL-Lite*$_\mathcal{A}^{(\text{agg})}$ *ontology in normal form and $(\mathcal{O}, q(\vec{x}))$ an OMQ, with q a CQA. A* rewriting of $q(\vec{x})$ w.r.t. $\mathcal{O}$ *is a CQA $q'(\vec{x})$, for which we write $q \rightsquigarrow_\mathcal{O} q'$, and which is obtained either by means of unification*

S1 *If atoms α_1 and α_2 unify, apply their most general unifier to q,*

or by applying an atom substitution *to q, as follows:*

S2 *If $A_1 \sqsubseteq A_2 \in \mathcal{O}$ and $A_2(x) \in q$, then replace $A_2(x)$ by $A_1(x)$ in q;*

S3 *If $A \sqsubseteq \exists p \in \mathcal{O}$ and $p(x, y) \in q$ such that y is a non-answer variable occurring only once in q, then replace $p(x, y)$ with $A(x)$ in q;*

S4 *If $\exists p \sqsubseteq A \in \mathcal{O}$ and $A(x) \in q$ then replace $A(x)$ by $p(x, z)$ in q, where z is a fresh variable;*

S5 *If $p \sqsubseteq p' \in \mathcal{O}$ and $p'(x, y) \in q$ then replace $p'(x, y)$ by $p(x, y)$ in q;*

S6 *If $p^- \sqsubseteq p' \in \mathcal{O}$ and $p'(x, y) \in q$ then replace $p'(x, y)$ by $p(y, x)$ in q;*

S7 *If $\odot_n(\text{agg}\ \ f) \sqsubseteq A \in \mathcal{O}$ and $A(x) \in q$ then replace $A(x)$ by $\odot_n(\text{agg}\ \ f)(x)$ in q.*

Let $\rightsquigarrow^*_{\mathcal{O}}$ be the reflexive and transitive closure of $\rightsquigarrow_{\mathcal{O}}$. The rewriting of $(\mathcal{O}, q)$, *denoted as $rew_{\mathcal{O}}(q)$, is the UCQA obtained by taking as a disjunct each q' such that $q \rightsquigarrow^*_{\mathcal{O}} q'$ that is unique up to isomorphism.*

The rewriting can be exponential in the size of the ontology, however the size of each CQ is polynomial. In order to show that the rewriting is correct also for *DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ OMQs, an adaptation of the proof for *DL-Lite*$_{\mathcal{A}}$ is needed. We proceed with the following theorem:

Theorem 2.1. *Let $\mathcal{O}$ be a DL-Lite$^{\mathsf{agg}}_{\mathcal{A}}$ OMQ. For each OMQ $(\mathcal{O}, q)$ and each ABox $\mathcal{A}$ consistent w.r.t. $\mathcal{O}$,*

$$cert(q, \mathcal{O}, \mathcal{A}) = cert(rew_{\mathcal{O}}(q), \emptyset, \mathcal{A}_f),$$

where $\mathcal{A}_f$ is the closure of $\mathcal{A}$ w.r.t. feature inclusion axioms in $\mathcal{O}$.

Proof. Let $\mathcal{A}$ be an ABox consistent w.r.t. $\mathcal{O}$ and $\mathcal{A}_f$ be the closure of $\mathcal{A}$ w.r.t. the feature inclusion axioms in $\mathcal{O}$.

Direction "$\supseteq$". We have to show that for any two arbitrary CQAs q_1, q_2 such that q_1 is a one-step rewriting of q_2 w.r.t. $\mathcal{O}$ obtained by applying axiom $\alpha \in \mathcal{O}$, we have that each match of q_1 in $chase^k_{\mathcal{O}}(\mathcal{A}_f)$ is a match of q_2 in $chase^{k+1}_{\mathcal{O}}(\mathcal{A}_f)$, where $chase^{k+1}_{\mathcal{O}}(\mathcal{A}_f)$ is obtained by applying g_α on $chase^k_{\mathcal{O}}(\mathcal{A}_f)$, for each $k \geq 0$. If q_1 is obtained by applying any of the rules **S**1, **S**2- **S**6, the claim follows from the fact that the rewriting is sound for *DL-Lite*$_{\mathcal{A}}$. Then, suppose q_1 is obtained from q_2 using rule **S**7. This means that for some $\odot_n(\mathsf{agg}\ \ f) \sqsubseteq A \in \mathcal{O}$, we have that $\odot_n(\mathsf{agg}\ \ f)(x) \in q_1$ and $A(x) \in q_2$. Let π be an arbitrary match for q_1 in $chase^k_{\mathcal{O}}(\mathcal{A})$ such that $\pi(x) = c$. Then, there exist facts $f(c, v_1), \ldots f(c, v_n)$ in $chase^k_{\mathcal{O}}$ such that $\mathsf{agg}(\{\!\{v_1, \ldots, v_n\}\!\}) \odot n$ which means that $A(c) \in chase^{k+1}_{\mathcal{O}}$. Since all other atoms of q_1 and q_2 coincide, π is also a match of q_2 in $chase^{k+1}_{\mathcal{O}}(\mathcal{A})$. This means that, for any $k \geq 0$, we have that $ans(q_1, chase^k_{\mathcal{O}}(\mathcal{A}_f)) \subseteq ans(q_2, chase^{k+1}_{\mathcal{O}}(\mathcal{A}_f)))$. This implies that:

$$cert(q_1, \mathcal{O} \setminus \{\alpha\}, \mathcal{A}_f) \subseteq cert(q_2, \mathcal{O}, \mathcal{A}_f). \tag{2.1}$$

Take any arbitrary query q and any arbitrary rewriting q' of q w.r.t. $\mathcal{O}$. From (2.1) it follows that $cert(q', \emptyset, \mathcal{A}_f) \subseteq cert(q, \mathcal{O}, \mathcal{A}_f)$ and due to the fact that $\mathcal{A}_f$ is part of $chase_{\mathcal{O}}(\mathcal{A})$ which suffices for obtaining certain answers (according to Corollary 1) we obtain that $cert(q', \emptyset, \mathcal{A}_f) \subseteq ans(q, chase_{\mathcal{O}}(\mathcal{A})) = cert(q, \mathcal{O}, \mathcal{A})$.

Direction "$\subseteq$". Note that $chase_{\mathcal{O}}(\mathcal{A}) = chase_{\mathcal{O}}(\mathcal{A}_f)$. We have to show that each certain answer is covered by some rewriting when evaluated over $\mathcal{A}_f$. For that we first show the following claim: Let q_1 be an arbitrary CQA. For each answer $\vec{a}$ of q_1 in $chase^k_{\mathcal{O}}(\mathcal{A}_f)$, there exists a rewriting q_2 of q_1 such that $\vec{a}$ is an answer of q_2 in $chase^{k-1}_{\mathcal{O}}(\mathcal{A}_f)$, for each $k \geq 1$. Let α be the axiom such that for some $\mathcal{A}' \subseteq chase^{k-1}_{\mathcal{O}}(\mathcal{A}_f)$, $chase^k_{\mathcal{O}}(\mathcal{A}_f) = chase^{k-1}_{\mathcal{O}}(\mathcal{A}_f) \cup g_\alpha(\mathcal{A}')$. For α in *DL-Lite*$_{\mathcal{A}}$, then this holds from completeness of the rewriting for *DL-Lite*$_{\mathcal{A}}$. Hence, let α be of the following form: $\odot_n(\mathsf{agg}\ \ f) \sqsubseteq A$. W.l.o.g., assume that $\vec{a}$ is not an answer to q_1 in

$chase^{k-1}_{\mathcal{O}}(\mathcal{A}_f)$. Let π be the match of $\vec{a}$ in $chase^{k}_{\mathcal{O}}(\mathcal{A}_f)$. Then, there is some $A(x) \in q_1$ such that $A(c) \in chase^{k}_{\mathcal{O}}$ and $\pi(x) = c$, and $A(c) \notin chase^{k-1}_{\mathcal{O}}(\mathcal{A}_f)$. From α applicable to $\mathcal{A}'$, there must be that $\mathcal{A}' = \{\!\{f(c, v_1), \ldots, f(c, v_n)\}\!\}$ such that $\mathsf{agg}(\{\!\{v_1, \ldots, v_n\}\!\}) \odot n$. Clearly, α is applicable to q_1 obtaining some q_2 in which $A(x)$ is replaced by $\odot_n(\mathsf{agg}\ \ f)(x)$, and all other atoms remain unchanged. Hence, π is also a match of q_2 in $chase^{k-1}_{\mathcal{O}}(\mathcal{A}_f)$.

Let $q(\vec{x})$ be an arbitrary CQ and $\vec{a}$ an arbitrary certain aswer of $(\mathcal{O}, q(\vec{x}))$ over $\mathcal{A}$. Let π be the match of q in the $chase^{k}_{\mathcal{O}}(\mathcal{A}_f)$ such that $\pi(\vec{x}) = \vec{a}$. From claim above, it follows that for each i such that $k \leq i \leq 0$ a rewriting q_i of q exists such that $\vec{a}$ is an answer of q_i in $chase^{i}_{\mathcal{O}}(\mathcal{A}_f)$. Therefore, for q_0 we have that $\vec{a}$ is an answer to q_0 over $\mathcal{A}_f$.

□

Provided that the data is complete w.r.t. the feature inclusion axioms, we have that *DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ OMQs are UCQA-rewritable.

Proposition 2.5. *DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ *OMQs are UCQA-rewritable when the data is complete w.r.t. the feature inclusion axioms in the ontology.*

In this approach in the case of *DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ OMQs the evaluation of the aggregating function is done at query time. Another possible approach is to complete the data also with facts that are implied by aggregating concepts in the ontology. Then, such axioms can be dropped and the resulting OMQ is in *DL-Lite*$_{\mathcal{A}}$. This translation is presented next.

2.3.3 Applying *DL-Lite*$_{\mathcal{A}}$ techniques for *DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ OMQs

Answering *DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ OMQs can be done using the same techniques as for *DL-Lite*$_{\mathcal{A}}$ OMQs, by means of the following translation. Let $\mathcal{O}$ be a *DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ OMQ and $\mathcal{A}$ an ABox. We denote by $\mathcal{O}'$ and $\mathcal{A}'$ the *DL-Lite*$_{\mathcal{A}}$ *translations* of $\mathcal{O}$ and $\mathcal{A}$, obtained as follows: For each aggregating concept C in $\mathcal{O}$, take a fresh concept name U_C and then:

1. $\mathcal{O}'$ is obtained by replacing each C by U_C in $\mathcal{O}$.
2. $\mathcal{A}'$ is obtained by extending $\mathcal{A}$ with an assertion $U_C(c)$, for each c such that $\mathsf{agg}\{\!\{ f'(c, v) \mid f' \sqsubseteq^*_{\mathcal{O}} f \}\!\} \odot n$, and each C of form $\odot_n(\mathsf{agg}\ \ f)$ in $\mathcal{O}$.

The translation preserves the structure of the canonical model. In particular, the following claim states that the canonical models are *isomorphic* modulo renaming of the aggregating concepts.

Lemma 2.2. *For arbitrary DL-Lite*$^{\mathsf{agg}}_{\mathcal{A}}$ *ontology $\mathcal{O}$ and ABox $\mathcal{A}$, let $\mathcal{O}'$ and $\mathcal{A}'$ denote their DL-Lite*$_{\mathcal{A}}$ *translations. We assume that $chase_{\mathcal{O}}(\mathcal{A}_f)$ and $chase_{\mathcal{O}'}(\mathcal{A}')$ are constructed by applying the axioms in $\mathcal{O}$ and their translations in $\mathcal{O}'$ in the same order.*

Let Δ_1 be the domain of $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ and Δ_2 be the domain of $\mathcal{I}^{chase}_{\mathcal{O}',\mathcal{A}'}$. Then, there exists a bijection $h : \Delta_1 \mapsto \Delta_2$ such that

a) h is the identity on cst($\mathcal{A}$),
b) for each $A \in \mathbf{C}$, $A(c) \in chase_{\mathcal{O}}(\mathcal{A})$ if and only if $A(h(c)) \in chase_{\mathcal{O}'}(\mathcal{A}')$,

c) for each $p \in \mathbf{R} \cup \mathbf{F}$, $p(c_1, c_2) \in chase_{\mathcal{O}}(\mathcal{A})$ *if and only if* $p(h(c_1), h(c_2)) \in chase_{\mathcal{O}'}(\mathcal{A}')$, *and*
d) for each aggregating concept C *in* $\mathcal{O}$, $C(c)$ *is satisfied in* $chase_{\mathcal{O}}(\mathcal{A})$ *if and only if* $U_C(h(c)) \in chase_{\mathcal{O}'}(\mathcal{A}')$.

Proof. Let $\mathcal{A}_f$ be the closure of $\mathcal{A}$ w.r.t. the set of feature inclusions in $\mathcal{O}$. The proof is done by induction on $i \geq 0$ for $chase^i_{\mathcal{O}}(\mathcal{A}_f)$ and $chase^i_{\mathcal{O}'}(\mathcal{A}')$. Let Δ^i_1 denote the domain of $chase^i_{\mathcal{O}}(\mathcal{A}_f)$ and Δ^i_2 the domain of $chase^i_{\mathcal{O}'}(\mathcal{A}')$.

For $i = 0$, we have that $chase^0_{\mathcal{O}}(\mathcal{A}_f) = \mathcal{A}_f$ and $chase^0_{\mathcal{O}}(\mathcal{A}') = \mathcal{A}'$. Since their domains coincide, let h be the identity on Δ^0_1. The assertions in $\mathcal{A}$ are simply copied into $\mathcal{A}'$, therefore conditions a)-c) are trivially satisfied. Let C be an arbitrary aggregating concept in $\mathcal{O}$. Suppose that there exists some o such that $C(o)$ is satisfied in $\mathcal{A}_f$. By definitions of satisfiability for aggregating concepts and the translation of $\mathcal{A}$, we obtain that $U_C(o) \in \mathcal{A}'$.

Suppose that the claim holds for $i = k$. This means that there exists a bijection $h : \Delta^k_1 \mapsto \Delta^k_2$ such that conditions a)-d) hold for $chase^k_{\mathcal{O}}(\mathcal{A}_f)$ and $chase^k_{\mathcal{O}'}(\mathcal{A}')$. Let $\alpha \in \mathcal{O}$ be an arbitrary axiom applicable to $chase^k_{\mathcal{O}}(\mathcal{A}_f)$ and α' be its translation in $\mathcal{O}'$. If $\alpha = \alpha'$, the claim is straightforward. Then, suppose that α is of form $\odot_n(\mathsf{agg}\ \ f) \sqsubseteq A$ and hence its translation α' is $U_{\odot_n(\mathsf{agg}\ f)} \sqsubseteq A$.

We show that α is applicable iff α' is applicable. If α is applicable to $chase^k_{\mathcal{O}}(\mathcal{A}_f)$, then there exists some $c \in \Delta^k_1$ such that $f(c, v_1), \ldots, f(c, v_n) \in chase^k_{\mathcal{O}}(\mathcal{A}_f)$ and $\mathsf{agg}\{\!\{v_1, \ldots, v_n\}\!\} \odot n$. By induction hypothesis and the fact that $v_1, \ldots, v_n \in val(\mathcal{A})$ (since feature invention is not allowed in $DL\text{-}Lite^{\mathsf{agg}}_{\mathcal{A}}$) we obtain that there exists $h(c) \in \Delta^k_2$ and for each $f(c, v_i)$ we have that $f(h(c), v_i) \in chase^k_{\mathcal{O}'}(\mathcal{A}')$. From the construction of $\mathcal{A}'$, we obtain that $U_{\odot_n(\mathsf{agg}\ f)}(h(c)) \in chase^k_{\mathcal{O}'}(\mathcal{A}')$. Therefore, α' is applicable to $chase^k_{\mathcal{O}'}(\mathcal{A}')$. We obtain that if $A(c) \in chase^{k+1}_{\mathcal{O}}(\mathcal{A}_f)$ then $A(h(c)) \in chase^{k+1}_{\mathcal{O}'}(\mathcal{A}')$. Analogously for the other direction.

By induction base and induction step, we can conclude that there exists a bijection h such that the conditions a)-d) hold for $chase_{\mathcal{O}}(\mathcal{A}_f)$ and $chase_{\mathcal{O}'}(\mathcal{A}')$. Since $chase_{\mathcal{O}}(\mathcal{A}_f) = chase_{\mathcal{O}}(\mathcal{A})$, we obtain that the claim holds. □

The following result follows immediately from the result above and the semantics of $DL\text{-}Lite^{\mathsf{agg}}_{\mathcal{A}}$.

Proposition 2.6. *Let* $\mathcal{O}$ *be a* $DL\text{-}Lite^{\mathsf{agg}}_{\mathcal{A}}$ *ontology,* $\mathcal{A}$ *be an ABox, and let* $\mathcal{O}'$, $\mathcal{A}'$ *be their* $DL\text{-}Lite_{\mathcal{A}}$ *translations. Then, for each CQ* q *defined over* $\mathsf{sig}(\mathcal{O})$*:*

$$cert(q, \mathcal{O}, \mathcal{A}) = cert(q, \mathcal{O}', \mathcal{A}').$$

Proof. We consider first the case when $\mathcal{A}$ is inconsistent w.r.t. $\mathcal{O}$. Since the translation does not affect concepts, roles and features that appear in functionality and disjointness axioms in $\mathcal{O}$, which represent the only cause of inconsistency, it easily follows that $\mathcal{A}'$ is also inconsistent w.r.t. $\mathcal{O}'$, hence each possible tuple of constants in $\mathcal{A}$ is a certain answer of $(\mathcal{O}, q)$ over $\mathcal{A}$ and as well of $(\mathcal{O}', q)$ over $\mathcal{A}'$.

Suppose that $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$. We have to show that each answer is preserved under the translation and that each answer over the translation is an answer in the original setting. Since we can rely on the canonical model to evaluate $\textit{DL-Lite}_{\mathcal{A}}^{(\mathsf{agg})}$ OMQs, and since FOL cannot distinguish between isomorphic structures, it follows from Lemma 2.2 that for any CQ q we have that $ans(q, \mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}) = ans(q, \mathcal{I}_{\mathcal{O}',\mathcal{A}'}^{chase})$ and since we can rely on canonical models to evaluate OMQs we obtain the desired result.

□

From the theoretical perspective, this result allows us to focus in the next chapters mainly on $\textit{DL-Lite}_{\mathcal{A}}$ OMQs, however via practical examples we advocate for the use of aggregating concepts and sometimes analytical queries since their addition, as presented in this chapter, is harmless. Moreover, in practice, allowing aggregation in the query may often be better than pre-processing.

2.4 Complexity of Reasoning

In order to understand for a given decision problem how difficult it is to be solved in practice, the characterization of the *computational complexity* in terms of *time* or *space* required by some computational model such as *Turing machines* is usually studied. This allows us to classify decision problems into *complexity classes*. We present only the complexity classes which are relevant for the decision problems we study and refer the reader to the book by Papadimitriou [Pap94] for an in-depth introduction to *complexity theory*.

2.4.1 Complexity Classes

The complexity classes of interest are organized into the following hierarchy:

$$\mathrm{AC}^0 \subsetneq \mathrm{LogSpace} \subseteq \mathrm{NLogSpace} \subseteq \mathrm{PTime} \subseteq \mathrm{NP} \subseteq \mathrm{PSpace} \subseteq \mathrm{ExpTime}.$$

The inclusion relationship is not known to be strict except for $\mathrm{AC}^0 \subsetneq \mathrm{LogSpace}$ and $\mathrm{PTime} \subsetneq \mathrm{ExpTime}$.

A problem is said to be *hard* for a complexity class C if any problem belonging to C can be reduced to it. If additionally it is also known to *belong to* C, then it is said to be *complete* for class C. This means that C-complete problems are the most difficult problems in C. Proving that a problem is hard for a class C is usually done by means of a *reduction*. Formally, a reduction from a problem P to a problem P' is a mapping f from each input word w in the language of P, $\mathcal{L}_P$, to some word in the language of P', $\mathcal{L}_{P'}$, such that $w \in \mathcal{L}_P$ if and only if $f(w) \in \mathcal{L}_{P'}$. A reduction is said to be polynomial if it can be computed in polynomial time in the size of each input word. All the reductions presented in this thesis are polynomial.

We now introduce the complexity classes that are relevant to this thesis. A problem belongs to AC^0 if it can be decided using a boolean circuit of polynomial size and constant depth, that uses unlimited fan-in gates. Intuitively, it can be decided in constant time using a polynomial number

of processors, with respect to the input size. An important problem that is relevant to this thesis and which belongs to this class, is the evaluation of FO-queries (i.e., SQL queries) over relational databases when the query is considered to be fixed and only the database is considered to be part of the input [AHV95]. This result in a sense justifies the ability of relational database engines to handle large amounts of data. Moreover, whenever a problem is known to be at least as hard as another problem belonging to a class that strictly contains AC^0, it cannot be reduced to FO-query evaluation.

For the remaining complexity classes, the computational model relies on Turing machines (TM). A (non)deterministic TM consists of two infinite tapes: read and write, a set of states which includes an input and an accepting state, and a transition function which determines the behavior of the TM on particular inputs. If for a pair consisting of a state and an input symbol the transition function uniquely determines the next state, then the TM is said to be *deterministic* (DTM), otherwise it is called *nondeterministic* (NTM). A problem is in LogSpace if it can be decided by a DTM which uses at most logarithmic number of tape cells, in the size of the input. An example of a notorious problem that belongs to LogSpace (but not to AC^0) is *reachability in undirected graphs* [LP82]. Similarly, NLogSpace is the class of problems that can be decided using a NTM which uses at most logarithmic number of tape cells (also measured in the size of the input). *Reachability in directed graphs* is a well-known problem that is in NLogSpace [Jon75].

A problem belongs to PTime, respectively NP, if it is solvable by a DTM, respectively a NTM, using polynomially many steps in the size of the input. We are also interested in the class CoNP, which contains all the problems for which their complement belongs to NP. Similarly, problems that are in PSpace are solvable using a DTM in space that is polynomial in the size of the input. Lastly, a problem belongs to ExpTime if it can be solved using a DTM in time that is exponential in the size of the input. A decision problem is said to be *undecidable* if it is provable that an algorithm that always can output yes or no for any given input does not exists. A famous undecidable problem is the *halting problem*: given a TM T and an arbitrary input word w, decide if T halts on w, by either accepting or rejecting, or if it runs forever. Therefore, a common technique for proving undecidability is by encoding the halting problem of an arbitrary TM.

Data and Combined Complexity of Answering OMQs. Following the complexity measures for query evaluation in the database setting, the *data* and *combined* complexity are also of interest in the case of OMQ evaluation over datasets. In the database setting, whenever we want to find the complexity of evaluating any query over any database, then both have to be considered as input of the decision problem. The complexity of such problem represents the combined complexity. However, based on the observation that the size of the query is generally much smaller than the size of the database, Vardi also identified another variation of the query answering problem in which the query is considered as fixed (i.e., not part of the input) and only the data is considered as input [Var82].

In the case of OMQ answering, the data and the combined complexity measurements are defined similarly:

- the *data complexity* is measured when the ontology and the query are fixed and only the

ABox is part of the input. The complexity of $\mathcal{L}$-CERTAIN ANSWERS$(\mathcal{O}, q)$ problem for each ontology language $\mathcal{L}$ denotes the data complexity of answering $\mathcal{L}$-OMQs.

- the *combined complexity* is measured in the size of all, i.e., the ontology, query and the data. Similarly as before, the complexity of $\mathcal{L}$-CERTAIN ANSWERS problem for each ontology language $\mathcal{L}$ denotes the combined complexity of answering $\mathcal{L}$-OMQs.

Part I

Interactive Ontology-mediated Query Answering

CHAPTER 3

Taming Complex Role Inclusions for *DL-Lite*

Among the limits of *DL-Lite*$_{\mathcal{A}}$ is that it cannot capture the propagation of properties along paths, such as *is part of* or *is located in* relations, which are fundamental in modeling so-called *aggregating objects* [Sat00]. A typical example regarding the usefulness of modeling part-whole relationships is the representation of geographical locations, since in general a location can be composed of multiple sub-locations and every property from the lower granularity level is preserved to the upper levels. For instance, it is natural to infer that whenever an event occurs in a city, it implicitly occurs also in the country the city is part of. In DLs such knowledge can be modeled by allowing role composition, which are usually referred as *complex role inclusions* [HS04]. While much desired, complex role inclusions are computationally hard. The addition of *transitive roles* already means losing FO-rewritability, given that reachability can be encoded by means of a single transitive role [ACKZ09]. Moreover, if no restrictions are imposed on their usage, they can easily lead to undecidability [HS04, SS89].

In this chapter, we study in which conditions we can add such axioms to *DL-Lite*$_{\mathcal{A}}$ in such a way that FO-rewritability is preserved. This allows us to capture our motivating example while keeping the computational cost under control. Our findings show that, like for very expressive DLs, also for *DL-Lite* the addition of unrestricted complex role inclusions results in an undecidable logic. For highly expressive DLs, decidability is restored by restricting complex role inclusions to generate a regular language when viewed as production rules of context-free grammars. Such complex role inclusions are therefore called *regular*. For *DL-Lite*$_{\mathcal{A}}$ with regular complex role inclusions we obtain that ABox consistency is ExpTime-complete, however an additional condition is imposed to avoid a 2-ExpTime lower-bound inherited from the more expressive $\mathcal{RIQ}$ [Kaz08]. As it turns out, such condition is also useful to avoid an exponential blowup when performing query rewriting. To remain FO-rewritable, we identified two possibilities: either to disallow recursion involving complex role inclusions or to ensure that the recursion is bounded. The picture on combined complexity is then completed with the following results: the non-recursive fragment is

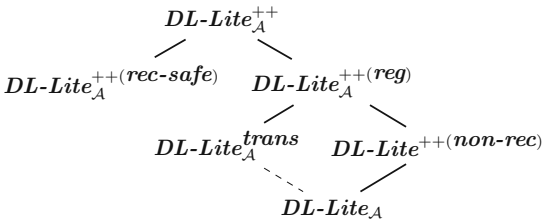


Figure 3.1: Inclusion relations for $DL\text{-}Lite_{\mathcal{A}}^{++}$ family of languages.

$\mathcal{L}$	Complexity		
	Combined		Data
	$\mathcal{L}$-ABOX CONSIST	$\mathcal{L}$-CERT ANSW	$\mathcal{L}$-ABOX CONSIST($\mathcal{O}$) and $\mathcal{L}$-CERT ANSW($\mathcal{O}$,q)
$DL\text{-}Lite_{\mathcal{A}}$	in PTime [a]	NP-c[b]	AC^0 [a,b]
$DL\text{-}Lite_{\mathcal{A}}^{++(rec\text{-}safe)}$	in PTime (Thm. 3.6)	NP-c (Thm. 3.7)	AC^0 (Thm. 3.7)
$DL\text{-}Lite_{\mathcal{A}}^{trans}$	NLogSpace-c[c]	in PSpace [e]	NLogSpace-c[d]
$DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$	CONP-c (Thm. 3.5)	NP-c (Thm. 3.4)	AC^0 (Thm. 3.4)
$DL\text{-}Lite_{\mathcal{A}}^{++(reg)}$	ExpTime-c (Thm. 3.2)	ExpTime-c (Thm. 3.3)	PTime-c (Thm. 3.3)[f]
$DL\text{-}Lite_{\mathcal{A}}^{++}$	undecidable (Thm. 3.1)	-	-

[a] Theorem 4.22 in [CGL+09b]
[b] Theorem 5.17 and Theorem 5.18 in [CGL+09b]
[c] Corollary 5.20 in [ACKZ09]
[d] Corollary 6.4 in [ACKZ09] (for instance checking)
[e] Reduction to C2RPQ answering in *DL-Lite* and Theorem 6.8 in [BOS15]
[f] Theorem 4 in [ORS10] (for satisfiability)

Table 3.1: Summary of complexity results.

CONP-complete, and the recursion-safe fragment is in PTime. The expresivity relation between such languages is presented in Figure 3.1, where $DL\text{-}Lite_{\mathcal{A}}^{trans}$ denotes the extension of $DL\text{-}Lite_{\mathcal{A}}$ with transitive roles, previously introduced in [ACKZ09], while all other extensions are new. The data and combined complexity of testing ABox consistency and OMQ answering are presented in Table 3.1 where each previously known or straightforward result points to the reference. In particular we note that for the extension with transitive roles a PSpace upper-bound is directly obtained from answering conjunctive two-way regular path queries in $DL\text{-}Lite_{\mathcal{R}}$ [BOS15]. All the other results in Table 3.1, as the references in parenthesis denote, are new and represent the main contribution of this chapter.

3.1 Extending $DL\text{-}Lite_{\mathcal{A}}$ with Complex Relation Inclusions

In this section we present the syntax and semantics of the extensions of $DL\text{-}Lite_{\mathcal{A}}$ with complex role inclusions, for which we adopt the following formalization. In order to capture also features, we use the term *relation* to denote either a role or a feature. Then, the notion of a *complex relation incusion* (CRI) extends the notion of a complex role inclusion to allow composition between roles and features.

Let us assume that $\mathbf{R}^{\pm}$ contains two disjoint subsests that are closed under inverses: $\mathbf{R_s}$ - denotes the set of *simple roles* , while $\overline{\mathbf{R}}_\mathbf{s}$ - denotes the set of *non-simple roles*. Similarly for features, $\mathbf{F}^{\pm}$ contains two partitions: $\mathbf{F_s}$ denotes *simple features*, and $\overline{\mathbf{F}}_\mathbf{s}$ denotes *non-simple features*.

Definition 3.1 (Complex relation inclusions)**.** *A* complex relation inclusion *(CRI) is an expression of one of the following forms:*

- $r_1 \circ r_2 \sqsubseteq r$, *where* $r_1, r_2 \in \mathbf{R}^{\pm}$ *and* $r \in \overline{\mathbf{R}}_\mathbf{s}$, *or*
- $r_1 \circ f_1 \sqsubseteq f$, *where* $r_1 \in \mathbf{R}^{\pm}$, $f_1 \in \mathbf{F}^{\pm}$ *and* $f \in \overline{\mathbf{F}}_\mathbf{s}$, *or*
- $r_1 \sqsubseteq r_2$, *where* $r_1, r_2 \in \mathbf{R}^{\pm}$ *and if* $r_1 \in \overline{\mathbf{R}}_\mathbf{s}$ *then* $r_2 \in \overline{\mathbf{R}}_\mathbf{s}$, *or*
- $f_1 \sqsubseteq f_2$, *where* $f_1, f_2 \in \mathbf{F}^{\pm}$ *and if* $f_1 \in \overline{\mathbf{F}}_\mathbf{s}$ *then* $f_2 \in \overline{\mathbf{F}}_\mathbf{s}$.

An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ satisfies *a CRI* $p_1 \circ p_2 \sqsubseteq p$ *if for all* $d_1, d_2, d_3 \in \Delta^{\mathcal{I}}$, $(d_1, d_2) \in p_1^{\mathcal{I}}$ *and* $(d_2, d_3) \in p_2^{\mathcal{I}}$ *imply* $(d_1, d_3) \in p^{\mathcal{I}}$, *where* p, p_1, p_2 *are relations.*

Intuitively, a relation is non-simple if it occurs on the right-hand-side of a CRI or if it is implied by such a relation, otherwise it is a simple relation.

Note that the composition between two features is not possible due to the fact that a feature always connects an object and a data value.

In our running example, we are interested in propagating the occurrence of an event along the location dimension, composed of district, city and country. For that, we can add the following CRI to our ontology:

$$\text{hasLocation} \circ \text{partOf} \sqsubseteq \text{hasLocation} \tag{3.1}$$

Given that the dataset contains the facts: $\text{Fire}(\mathsf{e}_1)$, $\text{hasLocation}(\mathsf{e}_1, \mathsf{distr}_1)$ and the following *part-of* path: $\text{partOf}(\mathsf{disrt}_1, \mathsf{Vienna}), \text{partOf}(\mathsf{Vienna}, \mathsf{Austria})$, by applying axiom 3.1 on these facts we can automatically infer $\text{hasLocation}(\mathsf{e}_1, \mathsf{Vienna})$ and $\text{hasLocation}(\mathsf{e}_1, \mathsf{Austria})$. Moreover, by allowing features to occur in CRIs, we are able to capture knowledge such as "locations have hospital access if they are connected to another location which has access to a hospital". Then, using the following CRI:

$$\text{connectedTo} \circ \text{hospAccess} \sqsubseteq \text{hospAccess}, \tag{3.2}$$

where hospAccess is a feature with Boolean range, we can model such knowledge.

3.1.1 Regular and Linear CRIs

There is a tight connection between CRIs and the study of formal languages, since the language that a role or feature determines given a set of CRIs can be generated using *context-free grammars* [SN07]. Recall that in formal language theory, given an alphabet, a grammar defines the rules used to form words belonging to that language. Such rules are called *production rules* and in context-free grammars each production rule is of the form $S \to \alpha$, where S is a non-terminal (i.e., syntactical variable) and α is a string of alphabet symbols and/or non-terminals. Naturally

for each relation p an associated grammar can be defined based on the CRIs involving p, thus for simplicity we view the CRIs as production rules.

Let R denote an arbitrary composition chain of (possibly inverse) relations, i.e. R is of form $p_1 \circ p_2 \circ \ldots \circ p_n$, where $p_i \in (\mathbf{R} \cup \mathbf{F})^{\pm}$. For a given set of CRIs $\mathcal{R}$ and arbitrary relation chains R_1, R_2, we write $R_1 \sqsubseteq_{\mathcal{R}} R_2$ whenever there exists $R \sqsubseteq p \in \mathcal{R}$ and either (i) $R_1 = S \circ R \circ S'$ and $R_2 = S \circ p \circ S'$, or (ii) $R_1 = S \circ R^- \circ S'$ and $R_2 = S \circ p^- \circ S'$.

We denote by $\sqsubseteq^*_{\mathcal{R}}$ the reflexive and transitive closure of $\sqsubseteq_{\mathcal{R}}$. Then, for any relation p, the *language of p w.r.t. $\mathcal{R}$* is defined as a set of words over $sign(\mathcal{R})$ such that:

$$\mathcal{L}_{\mathcal{R}}(p) = \{p_1 \ldots p_n \mid p_1 \circ \ldots \circ p_n \sqsubseteq^*_{\mathcal{R}} p\}.$$

A formal language is said to be *linear* if it can be generated using a linear grammar i.e., a context-free grammar that has at most one non-terminal symbol on the right-hand side in each production rule. A context-free grammar is said to be *left-linear* if each non-terminal symbol that occurs on the right-hand side of some production rule exists at the leftmost place, i.e., left end. Similarly, a context-free grammar is said to be *right-linear* if each non-terminal symbol that occurs on the right-hand side of some production rule exists at the rightmost place, i.e., right end.

A formal language is said to be *regular* if it is produced by a context-free grammar that is either *left-linear* or *right-linear*. Note that, in general a linear language is not regular, a typical example is $\{s^i r s^i \mid i \geq 0\}$ which can be generated using the following CRIs: $r \circ s \sqsubseteq r'$ and $s \circ r' \sqsubseteq r$.

The connection to context-free grammars allows us to identify whenever a set of CRIs induces a language that is either *linear*, *regular* or both. Such classification on CRIs is not novel, for instance, regularity restrictions have been imposed in more expressive DLs. For example to $\mathcal{RIQ}$, the language that extends $\mathcal{ALC}$ [SS91] with role inverses, qualifying number restrictions and regular CRIs, and $\mathcal{SROIQ}$ which further extends $\mathcal{RIQ}$ with nominals, (a)symmetric and (ir)reflexive role axioms.

However, for such DLs testing ABox consistency is 2-ExpTime-complete even when CRIs are only of the form $p_1 \circ p_2 \sqsubseteq p_1$ or $p_2 \circ p_1 \sqsubseteq p_1$ [Kaz08]. In order to avoid such exponential blow-up we opt for a different (more restrictive) syntactic characterization: for a set of CRIs, the production rule denoted by each CRI has at most one non-terminal symbol, meaning that for each $p_1 \circ p_2 \sqsubseteq p$, at most one of p_1 and p_2 can occur on the right-hand-side of another CRI. This restriction ensures that each word in the language of a set of CRIs is polynomially bounded.

We proceed with the syntactic restrictions on CRIs to ensure such properties.

Definition 3.2. *For a given set of CRIs $\mathcal{R}$, we say that $\mathcal{R}$ is*

a) linear *if for each $p_1 \circ s \sqsubseteq p_2 \in \mathcal{R}$ we have that $s \in \mathbf{R_s} \cup \mathbf{F_s}$ is a simple relation.*

b) regular *if there exists a strict partial order $\prec$ on $\mathbf{R} \cup \mathbf{F}$ such that for each CRI $p_1 \circ p_2 \sqsubseteq p \in \mathcal{R}$ and for each $i \in \{1, 2\}$ we have either (i) $p_i = p$, (ii) p_i is the inverse of p, or (iii) $p_i \prec p$.*

Linearity is then imposed by allowing at most one non-simple relation on the left-hand-side of a CRI. Note that such restriction does not capture regularity since using inverses, we can still generate $s^i r s^i$ using $r \circ s \sqsubseteq r'$ and $r'^- \circ s^- \sqsubseteq r^-$, where s is simple.

For the regular case, we use an adaptation of the standard syntactical restriction from [HS04, HKS06]. Such characterization does not capture all regular languages, however it is easy to grasp and captures most of the examples that motivate the need for CRIs. For capturing all regular languages, we refer to [Kaz10].

The following proposition bridges the gap between the restrictions on CRIs and the languages they generate.

Proposition 3.1. *Let $\mathcal{R}$ be a set of CRIs. We have that*

1. *If $\mathcal{R}$ is linear, then $\mathcal{L}_\mathcal{R}(p)$ is a linear language for each p in the signature of $\mathcal{R}$.*
2. *If $\mathcal{R}$ is regular, then $\mathcal{L}_\mathcal{R}(p)$ is a regular language for each p in the signature of $\mathcal{R}$.*

It is known that any regular language can be characterized by means of a regular expression. We show next that for regular CRIs that are also linear, the size of the regular expression is polynomial in the size of $\mathcal{R}$.

Lemma 3.1. *Let $\mathcal{R}$ be a set of regular CRIs. Then, for every relation $p \in sign(\mathcal{R})$ there exists a regular expression that generates $\mathcal{L}_\mathcal{R}(p)$. If in addition $\mathcal{R}$ is linear, then each regular expression is of size that is polynomial in $|\mathcal{R}|$.*

Proof. Let $\mathcal{R}$ be a set of regular CRIs. W.l.o.g., we assume that $\mathcal{R}$ contains $p_1 \circ p_2 \sqsubseteq p$ iff $p_2^- \circ p_1^- \sqsubseteq p^-$ and $r_1 \sqsubseteq r_2$ iff $r_1^- \sqsubseteq r_2^-$, with $(p^-)^- = p$. Then, for each relation p we define:

$$\rho_p := \Big(\bigcup_{\substack{s \circ p \sqsubseteq p \in \mathcal{R},\\ s \neq p}} s\Big)^* \Big(p \cup \bigcup_{\substack{r \circ s \sqsubseteq p \in \mathcal{R},\\ r,s \neq p}} rs\Big)\Big(\bigcup_{\substack{p \circ s \sqsubseteq p \in \mathcal{R},\\ s \neq p}} s\Big)^*$$

$$\tau_p := p \cup \bigcup_{\substack{p' \sqsubseteq^*_\mathcal{R} p\\ p' \neq p}} p'.$$

Then, for each relation p, the resulting regular expression is obtained as follows: if p is simple then $rex_p := \tau_p$, otherwise:

1. $rex_p := (\rho_p$ with p replaced by $\tau_p)$ and $V_p = \emptyset$
2. for each $p' \neq p$ occurring in rex_p such that $p' \notin V_p$ do $rex_p := (rex_p$ with p' replaced by $rex_{p'})$ and $V_p := V_p \cup \{p'\}$.

This procedure to construct a regular expression of each $p \in sign(\mathcal{R})$ terminates since cycles in $\mathcal{R}$ can occur only between simple relations (given the imposed $\prec$), for which we keep track of the replaced relations (using V_p).

For each $p \in sign(\mathcal{R})$, let $L(rex_p)$ denote the language described by the regular expression rex_p. We first show the following claim:

Claim 2. *(i) For each $p \in sign(\mathcal{R})$ we have that $p \in L(rex_p)$.*
(ii) If $p_1 \circ p_2 \sqsubseteq p \in \mathcal{R}$ then $p_1 p_2 \in L(rex_{p'})$ for all $p \sqsubseteq^*_{\mathcal{R}} p'$.
(iii) If $\mathcal{R}$ is linear, then for each $p \in sign(\mathcal{R})$ the size of rex_p is polinomially bounded in the size of $\mathcal{R}$.

Proof. Let $p \in sign(\mathcal{R})$ be arbitrarily chosen. Claim (i) is obvious since rex_p is of the form: $(...)^*(... \cup p \cup ...)(...)^*$, thus clearly $p \in L(rex_p)$. For (ii), let $p_1 \circ p_2 \sqsubseteq p \in \mathcal{R}$ and let $p \sqsubseteq^*_{\mathcal{R}} p'$. In the first step, we get that $\tau_{p'} = p' \cup \cdots \cup p \cup \dots$ and

ρ_p is of the form:

- $(\dots)^*(p \cup \cdots \cup p_1 p_2 \cup \dots)(\dots)^*$, if $p_1, p_2 \neq p$;
- $(\dots)^*(p \cup \dots)^*(\cdots \cup p_2 \cup \dots)^*$, if $p_1 = p$;
- $(\cdots \cup p_1 \cup \dots)^*(p \cup \dots)(\dots)^*$, if $p_2 = p$.

From the replacement steps, we get that $rex_{p'}$ is of the form $(p' \cup \cdots \cup rex_p \cup \dots)$. From (i) we get that for $i \in \{1, 2\}$ we have $p_i \in L(rex_{p_i})$, hence $p_1 p_2 \in L(rex_p)$. Since $rex_{p'}$ results from replacing p with rex_p we also get that $p_1 p_2 \in L(rex_{p'})$.

We now argue that (iii) holds. Suppose that $\mathcal{R}$ is linear, then when replacing a simple role the size of the regular expression grows linearly in the size of $\mathcal{R}$ and all freshly introduced symbols are simple. Since in each CRI at least one of the involved relations is simple, we obtain that the size of any regular expression is at most polynomial in the size of $\mathcal{R}$. □

Lastly, from the above claim and the definition of $\mathcal{L}_{\mathcal{R}}(p)$ we obtain that $L(rex_p)$ contains $\mathcal{L}_{\mathcal{R}}(p)$, for each $p \in sign(\mathcal{R})$.

□

3.1.2 *DL-Lite*$_{\mathcal{A}}^{++}$ and *DL-Lite*$_{\mathcal{A}}^{++(reg)}$

We consider the following extensions of *DL-Lite*$_{\mathcal{A}}$: *DL-Lite*$_{\mathcal{A}}^{++}$ is the extension with linear CRIs, and *DL-Lite*$_{\mathcal{A}}^{++(reg)}$ is the extension with CRIs that are both linear and regular.

Definition 3.3 (*DL-Lite*$_{\mathcal{A}}^{++}$, *DL-Lite*$_{\mathcal{A}}^{++(reg)}$)**.** *A DL-Lite*$_{\mathcal{A}}^{++}$ *ontology $\mathcal{O}$ is a DL-Lite*$_{\mathcal{A}}$ *ontology that may also contain linear CRIs such that for each $p_1 \circ p_2 \sqsubseteq p$, there is no axiom* (funct p) $\in \mathcal{O}$.

A DL-Lite$_{\mathcal{A}}^{++(reg)}$ *ontology $\mathcal{O}$ is a DL-Lite*$_{\mathcal{A}}^{++}$ *ontology such that the set of CRIs is in addition regular.*

For *DL-Lite*$_{\mathcal{A}}^{++}$, as for *DL-Lite*$_{\mathcal{A}}$, we disallow functional roles or features to be specialized. In the remaining of this chapter we focus only on linear CRIs, therefore we may omit calling them linear.

The chase procedure in Definition 2.24 can be extended for *DL-Lite*$_{\mathcal{A}}^{++}$ by considering the following additional case:

- If $\mathcal{A}' = \{p_1(c_1, c_2), p_2(c_2, c_3)\}$ and $p(c_1, c_3) \notin \mathcal{A}_j$, then $g_{p_1 \circ p_2 \sqsubseteq p}(\mathcal{A}') = \{p(c_1, c_3)\}$.

Lemma 2.1 can be lifted for *DL-Lite*$_{\mathcal{A}}^{++}$ given that also the additional case preserves satisfiability and, similarly to *DL-Lite*$_{\mathcal{A}}$, the chase procedure constructs a model that is canonical.

Proposition 3.2. *Let $\mathcal{O}$ be a DL-Lite$_{\mathcal{A}}^{++}$ ontology. For any ABox $\mathcal{A}$ consistent w.r.t. $\mathcal{O}$ we have that $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$ is a canonical model of $\mathcal{A}$ w.r.t. $\mathcal{O}$.*

3.1.3 Complexity of Reasoning in *DL-Lite*$_{\mathcal{A}}^{++}$ and *DL-Lite*$_{\mathcal{A}}^{++(reg)}$

As it is the case for more expressive DLs, without regularity, the addition of linear CRIs to *DL-Lite*$_{\mathcal{A}}$ leads to undecidability. The proof is done by encoding the behavior of a deterministic Turing Machine. Towards that, we show first that using regular CRIs, we can capture the axioms of $\mathcal{ELI}$. This preliminary result serves us in two ways: we obtain that testing ABox consistency in *DL-Lite*$_{\mathcal{A}}^{++(reg)}$ is ExpTime-hard and also we can rely on $\mathcal{ELI}$ axioms in the reduction.

The translation of normalized $\mathcal{ELI}$ axioms into *DL-Lite*$_{\mathcal{A}}^{++(reg)}$ is presented in Table 3.2. Intuitively, the combination of regular CRIs and inverse roles can be used to capture conjunction and qualified existential axioms.

$\mathcal{ELI}$ **axiom**	*DL-Lite*$_{\mathcal{A}}^{++(reg)}$ **translation**
$A \sqcap B \sqsubseteq C$	$A \sqsubseteq \exists r_A$ $\quad r_A^- \circ r_B \sqsubseteq p$
	$B \sqsubseteq \exists r_B$ $\quad p \circ r_B^- \sqsubseteq s_{A\sqcap B}$
	$\exists s_{A\sqcap B}^- \sqsubseteq C$
$A \sqsubseteq \exists r.B$	$A \sqsubseteq \exists p_{r_B}$ $\quad p_{r_B} \sqsubseteq r$
	$\exists p_{r_B}^- \sqsubseteq B$
$\exists r.A \sqsubseteq B$	$A \sqsubseteq \exists s_A$ $\quad r \circ s_A \sqsubseteq p_{r_A}$
	$\exists p_{r_A} \sqsubseteq B$

Table 3.2: $A, B, C \in \mathbf{C}$, $r \in \mathbf{R}^{\pm}$, and r_A, r_B, p, $s_{A\sqcap B}$, p_{r_A}, p_{r_B} s_A, are fresh role names.

We show next that the translation semantically emulates the original $\mathcal{ELHI}_{\bot}$ ontology.

Lemma 3.2. *For each $\mathcal{ELHI}_{\bot}$ ontology $\mathcal{O}$, there exists a DL-Lite$_{\mathcal{A}}^{++(reg)}$ $\mathcal{O}'$ such that $sign(\mathcal{O}) \subseteq sign(\mathcal{O}')$, and*

a) for each $\mathcal{I}'$ such that $\mathcal{I}' \models \mathcal{O}'$, $\mathcal{I}'$ restricted to $sign(\mathcal{O})$ is a model of $\mathcal{O}$, and
b) for each $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$, there exists $\mathcal{I}'$ which coincides with $\mathcal{I}$ over $sign(\mathcal{O})$ such that $\mathcal{I}' \models \mathcal{O}$.

Proof (sketch). Let $\mathcal{O}$ be an arbitrary $\mathcal{ELHI}_{\bot}$ ontology. W.l.o.g. we assume $\mathcal{O}$ to be in normal form. We obtain $\mathcal{O}'$ by copying all axioms which are of *DL-Lite*$_{\mathcal{A}}$ form, and using the transformations in Table 3.2 for the remaining axioms. By construction, it is clear that $sign(\mathcal{O}) \subseteq sign(\mathcal{O}')$.

Claim a) also holds from the construction of $\mathcal{O}'$. For b), since in the translation we are using fresh role names, we can easily extend any model of $\mathcal{O}$ over all predicates in $sign(\mathcal{O}') \setminus sign(\mathcal{O})$ in such a way that each axiom of $\mathcal{O}'$ is satisfied. □

This result allows us to make use of $\mathcal{ELHI}_\bot$ axioms in proving the following.

Theorem 3.1. *DL-Lite*$_\mathcal{A}^{++}$-ABox CONSISTENCY *is undecidable.*

Proof. Given a deterministic Turing machine (TM) M and an input word w, we can construct a *DL-Lite*$_\mathcal{A}^{++}$ ontology $\mathcal{O}$ and ABox $\mathcal{A}$ such that $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$ if and only if M does not accept input w. Since we can translate axioms expressible in $\mathcal{ELHI}_\bot$ into *DL-Lite*$_\mathcal{A}^{++}$, we make use of such axioms in our reduction. The reduction is inspired by the undecidability proof for answering CQs with safe negation in $\mathcal{ELI}_\bot$ [GIKK15].

Let $M = (\Sigma, Q, q_0, q_f, \sigma)$ be a deterministic TM, where Σ is the alphabet (including blank symbol '␣'), Q is the set of states, $q_0, q_f \in Q$ represent the initial, respectively accepting state, and $\sigma : (Q \times \Sigma) \mapsto (Q \times \Sigma \times \{L, R\})$ is the transition function. We encode the sequence of configurations of M using the following concept and role names:

- concept H_q marks the cell that is under the head of the cursor and the current state is q;
- concept $H_\emptyset$ marks all other cells on the tape which are not under the head of the cursor;
- concept C_a, where $a \in \Sigma$ encodes the information that symbol a is the content of that cell;
- role nconf connects content of a cell in configuration i to the content of the same tape cell in configuration $i + 1$;
- role ncell connects content of two adjacent tape cells in the same configuration;
- concepts B_τ^q, B_τ for $q \in Q$ and $\tau \in \{L, R\}$ propagate left and right along the tape the head and no-head markers, respectively;
- concept D encodes the fact that the content of the cell is blank, since in the initial configuration the tape content is empty after the input word.

Let $\mathcal{O}_M$ be the following $\mathcal{ELHI}_\bot$ ontology:

$$H_q \sqcap C_a \sqsubseteq \exists \text{nconf}.(C_b \sqcap B_\tau^{q'}), \qquad \sigma(q,a) = (q', b, \tau), \tau \in \{L, R\} \tag{3.3}$$

$$H_\emptyset \sqcap C_a \sqsubseteq \exists \text{nconf}.C_a, \qquad a \in \Sigma. \tag{3.4}$$

For $q \in Q$:

$$H_q \sqsubseteq B_L \sqcap B_R, \tag{3.5}$$

$$\exists \text{ncell}.B_L^q \sqsubseteq H_q \qquad \exists \text{ncell}^-.B_R^q \sqsubseteq H_q. \tag{3.6}$$

$$\exists \text{ncell}.B_L \sqsubseteq H_\emptyset \sqcap B_L \qquad \exists \text{ncell}^-.B_R \sqsubseteq H_\emptyset \sqcap B_R \tag{3.7}$$

$$\text{ncell} \circ \text{nconf} \sqsubseteq d \qquad \text{nconf}^- \circ d \sqsubseteq \text{ncell} \tag{3.8}$$

$$D \sqsubseteq \exists \text{ncell}.(D \sqcap C_␣) \tag{3.9}$$

$$H_{q_f} \sqsubseteq \bot. \tag{3.10}$$

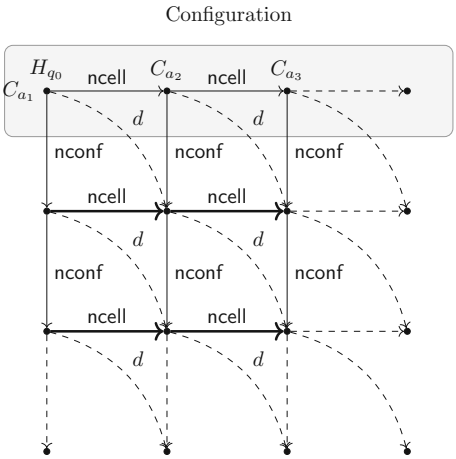


Figure 3.2: Encoding the computations of a Turing Machine.

For every input word $w = a_1 \dots a_n$ over Σ, we take the following ABox $\mathcal{A}_w$ with constants $c_1, \dots, c_n$, and assertions:

$$H_{q_0}(c_1), \quad C_{a_i}(c_i),\ \mathrm{ncell}(c_i, c_{i+1}),\ \text{for } 1 \leq i < n,\ \text{and } C_{a_n}(c_n), D(c_n).$$

We claim that:

Claim 3. *$\mathcal{A}_w$ is consistent w.r.t. $\mathcal{O}_M$ if and only if M does not accept w.*

Consider a model $\mathcal{I}$ of $(\mathcal{O}_M, \mathcal{A}_w)$. Since $\mathcal{I} \vDash \mathcal{A}_w$, by (3.9) we obtain that there exists an infinite sequence of (not necessarily distinct) elements $d_1, d_2, \dots$ such that $d_i \in C_{a_i}^{\mathcal{I}}$, for $1 \leq i \leq n$ and $d_i \in C_{\_}{}^{\mathcal{I}}$ for $i > n$. By (3.5) - (3.7), we obtain that $d_i \in H_{\emptyset}^{\mathcal{I}}$, for $i > 1$. Using axiom (3.3) and (3.4), we obtain that there exists elements $d_1', d_2', \dots$ such that $(d_i, d_i') \in \mathrm{nconf}^{\mathcal{I}}$, which denote the cells in the next configuration. The content and state are updated according to σ while for unaffected cells, the content remains unchanged. Again by (3.5) - (3.7), the head of the cursor is updated. Using CRIs in (3.8), we obtain that $(d_i', d_{i+1}') \in \mathrm{ncell}^{\mathcal{I}}$. By applying the same reasoning, we deduce that for each configuration in the computation such sequence of elements exists. Finally, (3.10) ensures that the accepting state is not reached during computation, that is, M does not accept w.

Conversely, if M does not accept w, then the computation of M can be encoded into an infinite two-dimensional grid interpretation $\mathcal{I}$ that is a model of $\mathcal{O}_M$ and satisfies $\mathcal{A}_w$. The interpretation $\mathcal{I}$ is sketched in Figure 3.2 and it clearly satisfies $\mathcal{A}_w$ and axioms (3.3)-(3.9). Since M does not accept w, then the accepting state q_f is not reached, therefore $H_{q_f}^{\mathcal{I}} = \emptyset$, hence $\mathcal{I} \vDash (\mathcal{O}_M, \mathcal{A}_w)$.

Lastly, since the problem of deciding whether a given deterministic Turing machine accepts or rejects a given input is undecidable, it follows that the theorem holds. □

If we impose regularity on the CRIs, decidability is preserved. Since testing if a given ABox is consistent w.r.t. a $\mathcal{ELHI}_\bot$ ontology is ExpTime-hard, from Lemma 3.2 we obtain the following result.

Lemma 3.3. *DL-Lite*$_\mathcal{A}^{++(reg)}$-ABOX CONSISTENCY *is* ExpTime-*hard.*

Unlike *DL-Lite*$_\mathcal{A}^{++}$, the regular fragment is decidable [HS04]. The ExpTime upper-bound can be obtained following the same procedure as in Ortiz et al. [Ort10] which is used to show 2-ExpTime upper-bound for $\mathcal{SRIQ}, \mathcal{SROQ}$ and $\mathcal{SROI}$. In our case, we can translate any given *DL-Lite*$_\mathcal{A}^{++(reg)}$ ontology into an equisatisfiable $\mathcal{ZIQ}$ ontology ($\mathcal{ZIQ}$ is also known as $\mathcal{ALCHIQ}b_{reg}^{\mathsf{Self}}$ which extends the better known $\mathcal{ACHIQ}$ with safe booleans over roles, regular role expressions and concepts of the form $\exists p.\mathsf{Self}$) by replacing each relation p with its regular expression. Unlike for the mentioned sub-languages of $\mathcal{SROIQ}$, such reduction is polynomial in the size of the original ontology, given that each regular expression is of polynomial size (Lemma 3.1). Lastly, since testing ABox consistency in $\mathcal{ZIQ}$ is in ExpTime (Theorem 3.4.2 in [Ort10]), from Lemma 3.3 we obtain:

Theorem 3.2. *DL-Lite*$_\mathcal{A}^{++(reg)}$-ABOX CONSISTENCY *is* ExpTime-*complete.*

Query Answering in *DL-Lite*$_\mathcal{A}^{++(reg)}$. The lower-bounds for query answering decision problems also follow directly from Lemma 3.2 and the fact that the CQ answering in $\mathcal{ELHI}_\bot$ is hard for ExpTime in combined complexity and PTime-hard in data complexity [BLB08].

A 2-ExpTime upper-bound is inherited from answering conjunctive two-way regular path queries in Horn-$\mathcal{SRIQ}$ [ORS11], however given that our CRIs are more succinct, a tighter upper-bound can be obtained as follows.

Using the standard automata encoding from [Kaz08] we can encode the set of CRIs into a set of Horn-$\mathcal{ALCHI}$ axioms. Based on the regular expression for each relation p, we can construct a non-deterministic finite automata (NFA) that can recognize each word in the language of p. Let $\mathcal{B}_\mathcal{R}(p)$ denote an NFA for $\mathcal{L}_\mathcal{R}(p)$ with the set of states Q_p, starting state $q_0 \in Q_p$, accepting states $F \subseteq Q_p$, and transition relation $\delta \subseteq Q_p \times \mathbf{R} \cup \mathbf{F} \times Q_p$. Given $\mathcal{B}_\mathcal{R}(p)$ we define the following axioms:

$$\begin{aligned} &\exists r \sqsubseteq A^p_{q_0}, && \text{for each } (q_0, r, q') \in \delta \\ &A^p_q \sqsubseteq \forall r.A^p_{q'}, && \text{for each } (q, r, q') \in \delta, \\ &A^p_q \sqsubseteq C_p, && \text{for each } q \in F. \end{aligned}$$

Then, given a *DL-Lite*$_\mathcal{A}^{++(reg)}$ OMQ $\mathcal{Q}$ we can construct a Horn-$\mathcal{ALCHI}$ OMQ $\mathcal{Q}'$ such that for any ABox the certain answers of $\mathcal{Q}$ and $\mathcal{Q}'$ coincide. We obtain such transformation from $\mathcal{Q}$ by adding the above axioms for each non-simple role p and then replace each occurrence of $\exists p$ by $A^p_{q_0}$, and similarly, each occurrence of $\exists p^-$ by C_p. Then, each $p(x, y)$ in the query is replaced with $A^p_{q_0}(x) \wedge C_p(y)$. The correctness of such transformation follows from Proposition 1 in [ORS11],

however, it follows from Lemma 3.1 that our transformation is polynomial, given that automata are in general more succinct than regular expressions. Lastly, since Horn-$\mathcal{ALCHIQ}^{\mathsf{Disj}}$ is strictly contained in Horn-$\mathcal{ALCHOIQ}^{\mathsf{Disj}}_{\mathsf{Self}}$, from Lemma 4 in [ORS11] we obtain the desired upper bounds.

Theorem 3.3. *DL-Lite*$_{\mathcal{A}}^{++(reg)}$-CERTAIN ANSWERS *is* ExpTime-*complete and DL-Lite*$_{\mathcal{A}}^{++(reg)}$-CERTAIN ANSWERS($\mathcal{O}$,q) *is P-complete.*

3.2 FO-rewritable Fragments of *DL-Lite*$_{\mathcal{A}}^{++}$

In this section we present two fragments of *DL-Lite*$_{\mathcal{A}}^{++}$ for which OMQs can be answered via rewriting into a union of conjunctive queries such that standard database evaluation techniques can be used to obtain complete answers. This means that the data complexity for evaluating CQs in these logics is the same as for core *DL-Lite*. We also establish complexity results for testing ABox consistency for each fragment.

3.2.1 Non-recursive *DL-Lite*$_{\mathcal{A}}^{++}$

We start with the observation that given an ontology $\mathcal{O}$ consisting of a single CRI $r \circ s \sqsubseteq r$ is sufficient for reducing reachability in a directed graph, problem known to be NLogSpace-hard, to *instance query answering* (i.e., query is atomic and with no existential variables) [ACKZ09]. For regaining rewritability, we need to ensure that the paths of simple roles on which CRIs are applicable have bounded size in some canonical model of the KB. To achieve this, a first restriction is to disallow cyclic dependencies between roles occuring in CRIs. We then extend the rewriting rules for *DL-Lite*$_{\mathcal{A}}$ in a natural way and prove correctness of the rewriting for this fragment.

In order to define the notion of a *recursive CRI* in an ontology, we employ the notion of an *ontology recursion graph*. For a *DL-Lite*$_{\mathcal{A}}^{++}$ ontology $\mathcal{O}$, the *recursion graph of* $\mathcal{O}$ is the directed graph containing a node v_A for each concept name A, and a node v_p for each relation p occurring in $\mathcal{O}$, and for each:

- $A_1 \sqsubseteq A_2 \in \mathcal{O}$, there exists an edge from v_{A_2} to v_{A_1};
- $p_1 \sqsubseteq p_2 \in \mathcal{O}$, there exists an edge from v_{p_2} to v_{p_1};
- $p_1 \sqsubseteq p_2^- \in \mathcal{O}$, there exists an edge from v_{p_2} to v_{p_1};
- $A \sqsubseteq \exists p \in \mathcal{O}$, there exists an edge from v_p to v_A;
- $\exists p \sqsubseteq A \in \mathcal{O}$, there exists an edge from v_A to v_p;
- $p_1 \circ p_2 \sqsubseteq p \in \mathcal{O}$, there exists an edge from v_p to v_{p_1} and one to v_{p_2}.

A relation p is *recursive in* $\mathcal{O}$ if v_p participates in a cycle in the recursion graph of $\mathcal{O}$ and a CRI $p_1 \circ p_2 \sqsubseteq p$ is *recursive in* $\mathcal{O}$ if p is.

Definition 3.4. *A DL-Lite*$_{\mathcal{A}}^{++(non\text{-}rec)}$ *ontology is a DL-Lite*$_{\mathcal{A}}^{++}$ *ontology* $\mathcal{O}$ *with no recursive CRIs.*

Note that for a *DL-Lite*$_{\mathcal{A}}^{++(non\text{-}rec)}$ ontology, the set of CRIs is regular since non-simple relations cannot occur in a cycle in the ontology graph, therefore based on the ontology graph an order on

relations can be derived such that regularity restrictions are satisfied. Moreover, we show next that restricting CRIs to be non-recursive indeed guarantees FO-rewritability.

The rewriting is the same as for $DL\text{-}Lite_{\mathcal{A}}$, extended with an additional case for handling CRIs.

Definition 3.5 ($DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$ rewriting)**.** *Let $\mathcal{O}$ be a $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$ ontology. For CQs q, q' we say that q' is* a rewriting *of q w.r.t. $\mathcal{O}$, written $q \rightsquigarrow_{\mathcal{O}} q'$ whenever q' is obtained from q by applying a $DL\text{-}Lite_{\mathcal{A}}$ rewriting rule* ***S1–S6*** *from in Definition 2.26, or*

S8 *by replacing $p(x, y) \in q$ with $p_1(x, z), p_2(z, y)$, if $p_1 \circ p_2 \sqsubseteq p \in \mathcal{O}$, where z is a fresh variable.*

The rewriting of q w.r.t. $\mathcal{O}$ *is $rew(q, \mathcal{O}) = \{q\} \cup \{q' \mid q \rightsquigarrow_{\mathcal{O}}^{*} q'\}$, where $\rightsquigarrow_{\mathcal{O}}^{*}$ denotes the reflexive and transitive closure of $\rightsquigarrow_{\mathcal{O}}$.*

For any CQ q, $rew(q, \mathcal{O})$ is a finite set of CQs such that each $q' \in rew(q, \mathcal{O})$ the derivation path of q' is polynomially bounded in the size of $\mathcal{O}$ and q.

Lemma 3.4. *Let $\mathcal{O}$ be a $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$ ontology, let q a CQ and let n denote the size of $(\mathcal{O}, q)$. Each $q' \in rew(q, \mathcal{O})$ is polynomial in n and can be obtained in polynomially many rewriting steps.*

Proof. Due to the non-recursiveness of the dependency graph and the restriction on simple roles, we show that we can assign to queries a (suitably bounded) degree that roughly corresponds to the number of rewriting steps that can be further applied. We prove that for each q' such that $q \rightsquigarrow_{\mathcal{O}}^{*} q'$, the degree does not increase, and after polynomially many steps we will reach $q \rightsquigarrow_{\mathcal{O}}^{*} q''$ such that the degree strictly decreases.

We first define $\mathcal{G}_{acycl}$ as the acyclic version of the recursion graph of $\mathcal{O}$ in which nodes n are labeled with a bag of predicates symbols, $bag(n)$, and each maximal cycle in the recursion graph of $\mathcal{O}$ denotes a single node with a bag containing all symbols participating in the cycle. All other nodes are labeled with a bag consisting of single predicate symbol. The edges in $\mathcal{G}_{acycl}$ are obtained from the recursion graph, namely there is an edge between node n and n' if there exists an edge in the recursion graph between some $P \in bag(n)$ and $P' \in bag(n')$, where P, P' are concept or relation symbols.

The function $mpath$ assigns a level to each node n in $\mathcal{G}_{acycl}$ as follows:

- if n has no outgoing edges, then $mpath(n) = 0$;
- otherwise, $mpath(n) = max\{mpath(n') + 1 \mid n \to n' \in \mathcal{G}_{acycl}\}$.

For a given query q, we define a function $dgr_{\mathcal{O}}(q)$ that, roughly, bounds the number of rewriting steps that may be iteratively applied to it. It is defined as follows:

$$dgr_{\mathcal{O}}(q) = \sum_{P(\vec{x}) \in q, P \in bag(n)} mpath(n).$$

We will show that the application of the rules decreases the degree, except for some cases where the degree stays the same, but can only do so for polynomially many rewriting steps (in the size of largest bag of $\mathcal{G}_{acycl}$). We show this bound before proving the main claim:

(‡) For each query of the form $q_1 = q \cup \{P(\vec{x})\}$ such that P participates in a cycle, then there are at most k^2 different queries of the form $q_2 = q \cup \{P'(\vec{x'})\}$ that can be obtained by the rewriting rules and such that $dgr_{\mathcal{O}}(q_2) = dgr_{\mathcal{O}}(q_1)$, where k is the size of the bag in $\mathcal{G}_{acycl}$ containing P (there is a unique bag containing P, since if P participates in two cycles, all symbols in the cycles belong to the same bag).

Let query q_2 be obtained by replacing $P(\vec{x})$ with $P'(\vec{x'})$ in q_1. From the rewriting rules of $DL\text{-}Lite_{\mathcal{A}}$, $\vec{x'}$ differs from $\vec{x}$ by at most one fresh variable. Since $dgr_{\mathcal{O}}(q_2) = dgr_{\mathcal{O}}(q_1)$ it must be that P, P' belong to the same bag in $\mathcal{G}_{acycl}$. If P occurs in a cycle then, by the restriction of CRIs in $DL\text{-}Lite_{\mathcal{A}}^{++}$, P cannot be a non-simple role, hence q_2 is not obtained by applying **S**8. Then, applying the axioms in $\mathcal{O}$ that participate in this cycle in $\mathcal{G}_{\mathcal{O}}$, it must be that q_2 is obtained again after at most k^2 rewriting steps (number of distinct pairs of symbols in the bag) therefore there are at most k^2 rewritings of q_1 that have the same degree.

Now that we have a bound on the number of times that the degree can stay the same for rewritings of a specific form, we can proceed with proving the lemma. We will distinguish between the types of queries produced by the rules in Definition 3.5:

1. for rules **S**2– **S**6: $q \cup \{P(\vec{x}\}) \rightsquigarrow_{\mathcal{O}} q \cup \{P'(\vec{x'})\}$, and there is an arc between node labeled with P, and node labeled with P', or they occur in the same bag in $\mathcal{G}_{acycl}$;
2. for rule **S**8 $q \cup \{p(x, y)\} \rightsquigarrow_{\mathcal{O}} q \cup \{p_1(x, z), p_2(z, y)\}$, where z is a variable not occurring in q and there exists arcs between the node labeled with p and nodes labeled with p_1 and p_2;
3. for rule **S**1 $q(\vec{x}) \rightsquigarrow_{\mathcal{O}} \sigma(q(\vec{x}))$, where a variable replaces another variable in q.

We now show that if $q_1 \rightsquigarrow_{\mathcal{O}} q_2$, then either *(i)* $dgr_{\mathcal{O}}(q_2) < dgr_{\mathcal{O}}(q_1)$, or *(ii)* $dgr_{\mathcal{O}}(q_2) = dgr_{\mathcal{O}}(q_1)$ if q_1, q_2 are as in (‡), and thus can only preserve the same degree for at most $k^2 \cdot |q_1|$ rewriting steps, or if q_2 is obtained by applying a substitution on q_1. This will imply that, after at most $dgr_{\mathcal{O}}(q_1) \cdot (|q_1| \cdot k)^2$ steps, the degree will be zero and no more steps will be applicable.

In what follows we show a proof by cases that matches cases 1–3 above. Firstly, for case 1 above, if P and P' do not occur in same cycle, then $dgr_{\mathcal{O}}(q \cup \{P(\vec{x}\})) < dgr_{\mathcal{O}}(P'(\vec{x'}))$ since there is an arc between P and P', hence $mpath(n) > mpath(n')$, where $P \in bag(n)$ and $P' \in bag(n')$. The other sub-case follows from (‡), therefore the degree of the queries obtained using rules **S**3– **S**6 decreases after at most $k^2 \cdot |q|$ rewriting steps.

Next, we show for case 2 above that: for each pair of queries $q_1 = q \cup \{p(x, y)\}$, $q_2 = q \cup \{p_1(x, z), p_2(z, y)\}$ such that $q_1 \rightsquigarrow_{\mathcal{O}} q_2$, we have that $dgr_{\mathcal{O}}(q_2) < dgr_{\mathcal{O}}(q_1)$. Since q_2 is obtained by applying $p_1 \circ p_2 \sqsubseteq p \in \mathcal{O}$, the degree strictly decreases in this case since p cannot occur in a cycle and there must be an arc between node labeled p and nodes labeled p_1 and p_2 in $\mathcal{G}_{acycl}$.

Lastly, for case 3 above: for each pair of queries $q_1 = q(\vec{x}) \rightsquigarrow_{\mathcal{O}} q_2 = \sigma(q(\vec{x}))$, we have that $dgr_{\mathcal{O}}(q_2) = dgr_{\mathcal{O}}(q_1)$, however in this case $vars(q_1) \subsetneq vars(q_2)$, hence such rule will eventually either reduce the size of q_1 and potentially make applicable cases 1 or 2 hence $dgr_{\mathcal{O}}(q_2) < dgr_{\mathcal{O}}(q_1)$ after at most $(k \cdot |q_1|)^2$ applications.

Therefore, we can conclude that each query $q' \in rew(q, \mathcal{O})$ can be obtained after applying at most $dgr_{\mathcal{O}}(q) \cdot (|q| \cdot k)^2$ rewriting steps, where k is the size of the largest cycle in $\mathcal{O}$.

We now argue the other part of the lemma, namely that each query in the rewriting has polynomial size. In case 1 at most one new variable is introduced but the size of the query remains the same, and in case 2 the size of the query increases by one, however only one of the newly introduced atoms (the non-simple role atom) may further trigger application of rule **S**8, but only a polynomially bounded number of times, since the degree decreases. Therefore the size of each query in the rewriting is polynomially bounded in the size of $\mathcal{O}$ and q. □

The next result is shown analogously as for *DL-Lite*$_{\mathcal{A}}$ [CGL+09b], considering the additional rewriting case.

Lemma 3.5. *Let $\mathcal{O}$ be a DL-Lite$_{\mathcal{A}}^{++(non\text{-}rec)}$ ontology and q a CQ. For every ABox $\mathcal{A}$ consistent with $\mathcal{O}$:*

$$cert(q, \mathcal{O}, \mathcal{A}) = \bigcup_{q' \in rew(q,\mathcal{O})} cert(q', \emptyset, \mathcal{A}).$$

Proof. Direction "$\supseteq$". We show that for each $q' \in rew(q, \mathcal{O})$ we have that $cert(q', \mathcal{O}, \mathcal{A}) \subseteq cert(q, \mathcal{O}, \mathcal{A})$.

- Base step: trivial since $q \in rew(q, \mathcal{O})$.
- Inductive step: Let q_i denote the rewriting obtained from q after i application of rewriting rules **S**1– **S**8 and suppose that $cert(q_i, \mathcal{O}, \mathcal{A}) \subseteq cert(q, \mathcal{O}, \mathcal{A})$. For q_{i+1} obtained from q_i by means of one-step application of some rule **S**1– **S**8, we want to show that $cert(q_{i+1}, \mathcal{O}, \mathcal{A}) \subseteq cert(q_i, \mathcal{O}, \mathcal{A})$. If q_{i+1} is obtained by applying any of the rewriting cases **S**1– **S**6, then this follows from the soundness of the rewriting for *DL-Lite*$_{\mathcal{A}}$. It remains to argue for **S**8. Suppose that $p_1 \circ p_2 \sqsubseteq p \in \mathcal{O}$ is applicable onto q_i, then q_{i+1} is obtained by replacing $p(x, y)$ in q with $p_1(x, z) \wedge p_2(z, y)$. From Proposition 3.2 we obtain that we can rely on the chase procedure to answer queries, and it is easy to check that each answer of q_{i+1} over $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$ is an answer of q_i over $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$.

From the base and induction steps, we conclude that for each $q' \in rew(q, \mathcal{O})$, we have that $cert(q', \emptyset, \mathcal{A}) \subseteq cert(q', \mathcal{O}, \mathcal{A}) \subseteq cert(q, \mathcal{O}, \mathcal{A})$.

Direction "$\subseteq$". We need to show that for each $\vec{a} \in cert(q, \mathcal{O}, \mathcal{A})$ there exists $q' \in rew(q, \mathcal{O})$ such that $\vec{a} \in cert(q', \emptyset, \mathcal{A})$. Let $\vec{a}$ be an arbitrary certain answer of $(q, \mathcal{O})$ over $\mathcal{A}$. This means that there exists some match π of q over $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$ such that $\pi(\vec{x}) = \vec{a}$, where $\vec{x}$ denote the free variables in q. Since we can rely on the chase to construct the canonical model it means that for some $k \geq 0$ we have that $\pi(q) \in chase_{\mathcal{O}}^{k}(\mathcal{A})$. We show that for $k - i$, with $0 \leq i \leq k$, there exist a rewriting

$q_i(\vec{x}_i) \in rew(q, \mathcal{O})$ and a match π_i such that $\pi_i(\vec{x}_i) = \vec{a}$ and $\pi_i(q_i) \in chase_{\mathcal{O}}^{k-i}(\mathcal{A})$. The claim then follows for $i = k$ since $chase_{\mathcal{O}}^{0} = \mathcal{A}$.

- Base step: This is an immediate consequence of the fact that $q \in rew(q, \mathcal{O})$.
- Inductive step: Let $k' = k - i + 1$ and suppose that there exist $q_{k'}(\vec{x'}) \in rew(q, \mathcal{O})$ and $\pi_{k'}$ such that $\pi_{k'}(\vec{x'}) = \vec{a}$ and $\pi_{k'}(q_{k'}) \in chase_{\mathcal{O}}^{k'}(\mathcal{A})$. We show that the claim also holds for $k' - 1$. For the $DL\text{-}Lite_{\mathcal{A}}$ rewriting rules (covered by cases **S**1– **S**6) it follows directly from the fact that the rewriting procedure is complete. We then show for the remaining case **S**8. We assume that a CRI $p_1 \circ p_2 \sqsubseteq p \in \mathcal{O}$ is applied in $chase^{k'-1}$, therefore there are $p_1(a_1, a_2), p_2(a_2, a_3) \in chase^{k'-1}$ and there is no $p(a_1, a_3) \in chase^{k'-1}$, and $chase^{k'} = chase^{k'-1} \cup \{p(a_1, a_3)\}$. If $p_1 \circ p_2 \sqsubseteq p$ is not applicable on $q_{k'}$ then the claim follows directly since $\pi_{k'}$ is a match of $q_{k'}$ in $chase^{k'-1}$. If $p_1 \circ p_2 \sqsubseteq p$ is applicable on $q_{k'}$ then we obtain a rewriting $q_{k'-1}$ which replaces $p(x, y) \in q_{k'}$ with $p_1(x, z) \wedge p_2(z, y)$ where z is a fresh variable. Then match $\pi_{k'-1}$ is obtained by extending $\pi_{k'}$ to map z to a_2. Therefore the claim holds.

From the base and inductive step we conclude that for each $\vec{a} \in cert(q, \mathcal{O}, \mathcal{A})$ there exists $q' \in rew(q, \mathcal{O})$ such that $\vec{a} \in cert(q, \emptyset, \mathcal{A})$. □

From Lemmas 3.4 and 3.5, we obtain that non-recursive CRIs preserve FO-rewritability of $DL\text{-}Lite_{\mathcal{A}}$. Moreover, the combined complexity remains NP-complete as for $DL\text{-}Lite_{\mathcal{A}}$: NP-hardness is inherited from answering CQs over plain relational databases, while the NP membership follows from the fact that guessing a rewriting and a mapping it is possible to verify in polynomial time if it is a match over the ABox.

Theorem 3.4. *For consistent ABoxes,* $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$-CERTAIN ANSWERS *is* NP*-complete and* $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$-CERTAIN ANSWERS($\mathcal{O}$,$q$) *is in* AC^0.

The addition of non-recursive CRI is far from harmless for consistency testing. Indeed, unlike the extension with transitive roles, even non-recursive CRIs increase the complexity of testing ABox consistency w.r.t. $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$ ontologies.

Theorem 3.5. $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$-ABOX CONSISTENCY *is* CONP*-complete.*

Proof. Upper-bound: Similarly as for $DL\text{-}Lite_{\mathcal{A}}$, inconsistency checking can be reduced to UCQ answering, using a CQ q_α for testing whether each disjointness or functionality axiom α is violated in the canonical model.

We consider first the case when α is a functionality axiom. From the restrictions on functional roles it follows that they can be violated only within the ABox since analogous to $DL\text{-}Lite_{\mathcal{A}}$, for each $(\mathsf{funct}\ p)$ such that $\mathcal{O} \models (\mathsf{funct}\ p)$, either $(\mathsf{funct}\ p) \in \mathcal{O}$ or $p^{\mathcal{I}} = \emptyset$ for each $\mathcal{I} \models \mathcal{O}$. Therefore, the violation of each functionality axiom $\alpha = (\mathsf{funct}\ p)$ can be tested by evaluating the following CQ with inequalities over the ABox alone: $q_\alpha() \leftarrow \exists x, y, z\ p(x, y) \wedge p(x, z) \wedge y \neq z$.

When α is a disjointness axiom, by Lemmas 3.4 and 3.5, an NP procedure can guess a query q_α, guess a q'_α in its rewriting, and evaluate q'_α over $\mathcal{A}$. Since the rewriting is of polynomial size, the evaluation of such boolean CQs over the ABox can be done in polynomial time.

We can conclude then that the ABox is inconsistent iff $() \in cert(q_\alpha, \mathcal{O}, \mathcal{A})$ for some disjointness or functionality axiom α. As discussed above, this test is in NP, therefore we conclude that $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$-ABOX CONSISTENCY is in CONP.

Lower-bound: We reduce the complement of 3SAT to testing ABox consistency in $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$. Suppose we are given a conjunction $\varphi = c_1 \wedge \cdots \wedge c_n$ of clauses of the form $\ell_{j_1} \vee \ell_{j_2} \vee \ell_{j_3}$, where for $1 \leq j \leq n$ and $1 \leq k \leq 3$, each ℓ_{j_k} is a literal, i.e., propositional variables or their negation. Let $x_0, \ldots, x_m$ be all the propositional variables occurring in φ.

In order to encode the possible truth assignments of each variable x_i, we take two fresh roles r_{x_i} and $\bar{r}_{x_i}$, intended to be disjoint. We construct a $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$ ontology $\mathcal{O}_\varphi$ containing, for every $0 \leq i \leq m$, the following axioms:

$$\mathsf{disj}(r_{x_i}, \bar{r}_{x_i}), \qquad A_i \sqsubseteq \exists r_{x_i} \sqcap \exists \bar{r}_{x_i}, \qquad \exists r_{x_i}^- \sqsubseteq A_{i+1},$$
$$\exists (\bar{r}_{x_i})^- \sqsubseteq A_{i+1}, \qquad r_{x_i} \sqsubseteq p, \qquad \bar{r}_{x_i} \sqsubseteq p$$

These axioms have a model that is a full binary tree, rooted at A_0 and whose edges are labeled with the role p, and with different combinations of the roles r_i and $\bar{r}_{x_i}$. Intuitively, each path represents a possible variable truth assignment. Further, $\mathcal{O}_\varphi$ contains axioms relating each variable assignment with the clauses it satisfies, using roles $s_{c_1}, \ldots, s_{c_n}$. More precisely, we have the following role inclusions for $0 \leq i \leq m$, and $1 \leq j \leq n$:

$$r_{x_i} \sqsubseteq s_{c_j}, \quad \text{if } x_i \in c_j \qquad\qquad \bar{r}_{x_i} \sqsubseteq s_{c_j}, \quad \text{if } \neg x_i \in c_j \tag{3.11}$$

To encode the evaluation of all clauses, we have axioms propagating down the tree all clauses satisfied by some assignment. Note that we could do this easily using a CRI such as $s_{c_j} \circ p \sqsubseteq s_{c_j}$. However, this would need a recursive role s_{c_j}. Since the depth of the assignment tree is bounded by m, we can encode this (bounded) propagation using at most m roles $s^i_{c_j}$ for each clause c_j, which will be declared as subroles of another role $s^*_{c_j}$. For $1 \leq j \leq n$ and $1 \leq i < m$, we have the following CRIs:

$$s_{c_j} \circ p \sqsubseteq s^1_{c_j} \qquad s^i_{c_j} \circ p \sqsubseteq s^{i+1}_{c_j} \qquad s^i_{c_j} \sqsubseteq s^*_{c_j}$$

Thus, if c_j is satisfied in a p-branch of the assignment tree, its leaf will have an incoming $s^*_{c_j}$ edge. Now, in order to encode that there is at least one clause that is not satisfied, we need to forbid the existence of a leaf satisfying the concept $\exists (s^*_{c_1})^- \sqcap \cdots \sqcap \exists (s^*_{c_n})^-$. This cannot be straightforwardly written in $DL\text{-}Lite_{\mathcal{A}}^{++(non\text{-}rec)}$, but we resort again to CRIs to propagate information:

$$\exists (s^*_{c_1})^- \sqsubseteq \exists t_1 \qquad\qquad s^*_{c_k} \circ t_1 \sqsubseteq p^1_k,\ 2 \leq k \leq n \tag{3.12}$$

Next, for $2 \leq i \leq n$, $i < k \leq n$ we have the following:

$$\exists (p_i^{i-1})^- \sqsubseteq \exists t_i \qquad\qquad p_k^{i-1} \circ t_i \sqsubseteq p_k^i \tag{3.13}$$

By adding the axiom $\exists t_n \sqsubseteq \bot$, we obtain the required restriction. Then, φ is unsatisfiable if and only if $\{A_0(a)\}$ is consistent w.r.t. $\mathcal{O}_\varphi$. □

3.2.2 Recursion-safe fragment of *DL-Lite*$_{\mathcal{A}}^{++}$

Additionally to the increased complexity, *DL-Lite*$_{\mathcal{A}}^{++(non\text{-}rec)}$ has another relevant limitation: it cannot express CRIs like $r \circ s \sqsubseteq r$ which means that the propagation of properties along paths is very limited. As s is simple, it cannot be implied by non-simple roles, thus only ABox constants are instances of these guarding roles. However, as motivated by our running example, CRIs such as 3.1 are useful to propagate information along the location hierarchy denoted by concepts District, City, Country. Indeed, in this case, the application of the CRI is very local, meaning that each path that triggers the application of a CRI involves known objects from the data. Following this observation, we define yet another fragment of *DL-Lite*$_{\mathcal{A}}^{++}$ that allows recursive CRIs, however their application "fires" only very close to the ABox. We call such fragment *recursion-safe*, for which regularity conditions are not imposed.

Definition 3.6. *A DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$ *ontology $\mathcal{O}$ is any DL-Lite*$_{\mathcal{A}}^{++}$ *ontology such that for each CRI $p_1 \circ s \sqsubseteq p_2$, there is no axiom of the form $B \sqsubseteq \exists p \in \mathcal{O}$ with $p \sqsubseteq_{\mathcal{O}}^{s} s$ or $p \sqsubseteq_{\mathcal{O}}^{s} s^-$, where $\sqsubseteq_{\mathcal{O}}^{s}$ denotes the reflexive and transitive closure of $s_1 \sqsubseteq s_2 \in \mathcal{O}$ with $s_2 \in \mathbf{R_s} \cup \mathbf{F_s}$.*

The key idea behind recursion safety is that every recursive CRI is "guarded" by a simple role that is not existentially implied. For query answering, we can assume that only ABox constants are connected by these guarding roles, and thus recursive roles can connect only pairs of objects in which at least one must be an ABox constant.

Standard reasoning problems like consistency checking and answering instance queries are tractable for recursion safe *DL-Lite*$_{\mathcal{A}}^{++}$ ontologies. To see this, we first note that for a given ABox $\mathcal{A}$ testing consistency can be done in a two step procedure: a) first verify if $\mathcal{A}$ satisfies all functionality axioms, and then b) test if there exists a model of $\mathcal{A}$ that satisfies each remaining axiom.

Claim 4. *Let $\mathcal{O}$ be a DL-Lite*$_{\mathcal{A}}^{++}$ *ontology and let $\mathcal{O}_f$ denote the functionality axioms in $\mathcal{O}$. For any ABox $\mathcal{A}$, $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$ if and only if $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}_f$ and $\mathcal{A}$ is consistent w.r.t. $\mathcal{O} \setminus \mathcal{O}_f$.*

Proof. **Direction** $\Rightarrow$ is trivial. For the other direction, suppose that (i) $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}_f$ and (ii) $\mathcal{A}$ is consistent w.r.t. $\mathcal{O} \setminus \mathcal{O}_f$. From (i), it follows that $\mathcal{I}_\mathcal{A}$ satisfies each functionality axiom in $\mathcal{O}_f$. Due to the fact that each functional relation cannot be specialized nor can occur on the right-hand-side of a CRI, the following property holds: for each $(\mathsf{funct}\ p)$ such that $\mathcal{O} \models (\mathsf{funct}\ p)$ we have that either $(\mathsf{funct}\ p) \in \mathcal{O}_f$ or $p^\mathcal{I} = \emptyset$, for each $\mathcal{I} \models \mathcal{O}$. Using this property and the facts: $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$ extends $\mathcal{I}_\mathcal{A}$ and it creates a fresh constant when applying axioms of the form $A \sqsubseteq \exists p$, we

obtain that $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ satisfies $\mathcal{O}_f$. Lastly, from (ii) we have that $\mathcal{I}^{chase}_{\mathcal{O},\mathcal{A}}$ satisfies also each axiom in $\mathcal{O} \setminus \mathcal{O}_f$. Therefore, $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$. □

For the second part of the procedure to test consistency we can build a polynomial-sized interpretation that is a model whenever $\mathcal{A}$ is consistent w.r.t. $\mathcal{O} \setminus \mathcal{O}_f$. This model can also be used for answering instance queries. In the following we use the notation $\alpha \sqsubseteq_{\mathcal{O}} \beta$ whenever $\mathcal{O} \models \alpha \sqsubseteq \beta$.

Definition 3.7. *Let $\mathcal{O}$ be a recursion safe DL-Lite$^{++}_{\mathcal{A}}$ ontology and $\mathcal{A}$ an ABox. We define an interpretation $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ as follows. As domain we use the constants in $\mathcal{A}$, fresh constants a_p that serve as p-fillers for individual a, and fresh constants c^i_p for $i \in \{1,2,3\}$ that serve as shared p-fillers for constants not from $\mathcal{A}$, where p denotes either a role or a feature.*

That is, $\Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}} = D_0 \cup D_1 \cup D_2$, where:

$$D_0 = cst(\mathcal{A}), \qquad D_1 = \{a_p \mid a \in D_0, A \sqsubseteq \exists p \in \mathcal{O}\}, \quad D_2 = \{c^1_p, c^2_p, c^3_p \mid A \sqsubseteq \exists p \in \mathcal{O}\}.$$

We again distinguish between constant symbols denoting objects $\Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}_{\mathbf{O}}$ and those denoting data values $\Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}_{\mathbf{V}}$. The interpretation function has $c^{\mathcal{E}_{\mathcal{O},\mathcal{A}}} = a$ for each $c \in \Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$, and assigns to each concept name A, each role name r and each feature name f in $\mathsf{sig}(\mathcal{O})$ the minimal set of the form $A^{\mathcal{E}_{\mathcal{O},\mathcal{A}}} \subseteq \Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}_{\mathbf{O}}$, $r^{\mathcal{E}_{\mathcal{O},\mathcal{A}}} \subseteq \Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}_{\mathbf{O}} \times \Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}_{\mathbf{O}}$, $f^{\mathcal{E}_{\mathcal{O},\mathcal{A}}} \subseteq \Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}_{\mathbf{O}} \times \Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}_{\mathbf{V}}$ such that the following conditions hold, for all $A \in \mathbf{C}$, B a basic concept, and $f \in \mathbf{F}$ and $r, r_1, r_2 \in \mathbf{R}$. $p, p_1, p_2 \in \mathbf{R} \cup \mathbf{F}$:

1. *If $A(a) \in \mathcal{A}$ then $a \in A^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$, if $r(a,b) \in \mathcal{A}$ then $(a,b) \in r^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ and if $f(a,v) \in \mathcal{A}$ then $(a,v) \in f^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$.*
2. *If $B \sqsubseteq \exists p \in \mathcal{O}$, $a \in B^{\mathcal{E}_{\mathcal{O},\mathcal{A}}} \cap D_0$ then $(a, a_p) \in p^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$.*
3. *If $B \sqsubseteq \exists p \in \mathcal{O}$, $d \in B^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ and*
   a) *if $d \in D_1$ then $(d, c^1_p) \in p^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$, or*
   b) *if $d \in D_2$ then (i) if $d = c^i_r$, for some $i \in \{1,2\}$, then $(d, c^{i+1}_p) \in p^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$, (ii) if $d = c^3_r$ then $(d, c^1_p) \in p^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$.*
4. *If $B \sqsubseteq A \in \mathcal{O}$, $d \in B^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ then $d \in A^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$.*
5. *If $p_1 \sqsubseteq p_2 \in \mathcal{O}$, $(c_1,c_2) \in p_1^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ then $(c_1,c_2) \in p_2^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$.*
6. *If $p_1 \sqsubseteq p_2^- \in \mathcal{O}$, $(c_1,c_2) \in p_1^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ then $(c_2,c_1) \in p_2^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$.*
7. *If $p_1 \circ p_2 \sqsubseteq p \in \mathcal{O}$, $(c_1,c_2) \in p_1^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ and $(c_2,c_3) \in p_2^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ then $(c_1,c_3) \in p^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$.*

For $\mathcal{E}_{\mathcal{O},\mathcal{A}}$, we can show the following useful properties:

Proposition 3.3. *Let $\mathcal{O} = \mathcal{O}_p \cup \mathcal{O}_n \cup \mathcal{O}_f$ be a recursion safe DL-Lite$^{++}_{\mathcal{A}}$ TBox, where $\mathcal{O}_p$ contains only positive inclusion axioms, $\mathcal{O}_n$ contains only disjointness axioms and $\mathcal{O}_f$ contains only functionality axioms. Then, for every ABox $\mathcal{A}$:*

- **P1** *If $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$ then $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}_p \cup \mathcal{O}_n$.*
- **P2** *$\mathcal{A}$ is inconsistent w.r.t $\mathcal{O}_p \cup \mathcal{O}_n$ if and only if $\mathcal{E}_{\mathcal{O},\mathcal{A}} \not\models \alpha$ for some $\alpha \in \mathcal{O}_n$.*
- **P3** *If $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$ and q is an instance query (i.e., an atomic query with no existential variables) then $cert(q,\mathcal{O},\mathcal{A}) = ans(q, \mathcal{E}_{\mathcal{O},\mathcal{A}})$.*

Proof. To prove **P1**, we assume that $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$. Verifying that $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ satisfies all but the disjointness axioms is easy from the definition of $\mathcal{E}_{\mathcal{O},\mathcal{A}}$. Let $\mathcal{I}$ be an arbitrary model of $\mathcal{A}$ w.r.t. $\mathcal{O}$. For $d, d' \in \Delta^{\mathcal{I}}$, let $\mathsf{tp}_{\mathcal{I}}(d) = \{B \mid d \in B^{\mathcal{I}}\}$ the set of basic concepts satisfied at d in $\mathcal{I}$, and $\mathsf{tp}_{\mathcal{I}}(d, d') = \{p \mid (d, d') \in p^{\mathcal{I}}\}$, the set of relations connecting d and d' in $\mathcal{I}$. The following claim shows a key property of $\mathcal{E}_{\mathcal{O},\mathcal{A}}$.

Claim 5. *For any given* $d \in \Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ (i) *there exists* $e \in \Delta^{\mathcal{I}}$ *such that* $\mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(d) \subseteq \mathsf{tp}_{\mathcal{I}}(e)$ *and* (ii) *for each* $d' \in \Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ *such that* $\mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(d, d') \neq \emptyset$ *we have that there exists* $e' \in \Delta^{\mathcal{I}}$ *such that* $\mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(d, d') \subseteq \mathsf{tp}_{\mathcal{I}}(e, e')$.

Proof (Sketch). Given that $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$ then $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$ is a canonical model of $\mathcal{A}$ w.r.t. $\mathcal{O}$. Since any canonical model can be homomorphically mapped into any other model, it suffices to show the claim for $\mathcal{I} = \mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$. By construction of the chase, we obtain that for the case when $d \in D_0$ the claim follows easily.

Suppose that $d = a_p \in D_1$, then from the construction of $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ we have that $a \in D_0$, $B \in \mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(a)$ and $B \sqsubseteq \exists p \in \mathcal{O}$. In this case we have that $\mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(d) = \{B \mid \exists p \sqsubseteq_{\mathcal{O}} B\}$. Using the induction hypothesis we have that $B \in \mathsf{tp}_{\mathcal{I}}(a)$ hence from the construction of the chase we obtain that there exists some fresh constant c_p such that $(a, c_p) \in p^{\mathcal{I}}$. Since $\mathcal{I}$ is a model, we easily conclude that $\mathsf{tp}_{\mathcal{I}}(c_p) \supseteq \mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(d)$. Let d' be such that there is some $p' \in \mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(d, d')$. Since $\mathcal{O}$ is recursion-safe, we cannot have that $d' \in D_1$, hence if $d' \in D_0$ we have that $p \in \mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(d', d)$ and $p \sqsubseteq \exists (p')^- \in \mathcal{O}$, case which is already considered. If $d' \in D_1$, then there is some $B \in \mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(d)$ and $B \sqsubseteq \exists s$ and $s \sqsubseteq_{\mathcal{O}} p'$ case which is also covered by the induction hypothesis and the chase construction.

For the remaining case when $d \in D_2$, given that the construction of $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ avoids creating cycles of length 1 or 2 thus the re-use of constants does not create extra types for edges, thus condition (ii) in Claim is satisfied also in this case. Thus the proof in this case is done analogously to the previous one. □

We resume the proof of **P1**. Towards a contradiction, assume there is $\alpha = \mathsf{disj}(B_1, B_2) \in \mathcal{O}$, where B_1, B_2 are general concepts, such that $\mathcal{E}_{\mathcal{O},\mathcal{A}} \nvDash \alpha$; the case of other disjointness axioms is analogous. Then there is $d \in \Delta^{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$ with $B_1, B_2 \in \mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}$, and by the claim above, for each model $\mathcal{I}$ there exists some $d' \in \Delta^{\mathcal{I}}$ such that $\mathsf{tp}_{\mathcal{E}_{\mathcal{O},\mathcal{A}}}(d) \subseteq \mathsf{tp}_{\mathcal{I}}(d')$, therefore $B_1, B_2 \in \mathsf{tp}_{\mathcal{I}}(d')$. Since this holds for each model $\mathcal{I}$, we obtain a contradiction with the fact that $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$. Therefore, it must be that $\mathcal{E}_{\mathcal{O},\mathcal{A}} \vDash \alpha$ for each $\alpha \in \mathcal{O}_n$. This concludes proof of **P1**.

For property **P2**, **direction** $\Leftarrow$ follows directly from Claim 5. For the other direction, suppose that $\mathcal{A}$ is inconsistent w.r.t. $\mathcal{O}_p \cup \mathcal{O}_n$. This means that there exists some $\alpha \in \mathcal{O}_n$ that is not satisfied in $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$. Since $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$ is a canonical model of $\mathcal{A}$ w.r.t. $\mathcal{O}_p$ and since $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ is also a model of $\mathcal{A}$ w.r.t. $\mathcal{O}_p$, there exists a homomorphism from $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$ to $\mathcal{E}_{\mathcal{O},\mathcal{A}}$. However, since $\mathcal{A}$ is inconsistent w.r.t. $\mathcal{O}_p \cup \mathcal{O}_n$, there is some axiom $\alpha \in \mathcal{O}$ such that $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase} \nvDash \alpha$, which means that there exists some tuple of constants $\vec{c}_1$ in $\mathcal{I}_{\mathcal{O},\mathcal{A}}^{chase}$ whose types contain two disjoint concepts or roles. Since $\vec{c}$ can be homomorphically mapped into some tuple of constants $\vec{c}_2$ in $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ that satisfies the same types as $\vec{c}_2$, we directly obtain that $\mathcal{E}_{\mathcal{O},\mathcal{A}} \nvDash \alpha$, which concludes the proof of **P2**.

For **P3**, since $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}$, using Claim 4 we have that $\mathcal{A}$ satisfies each functionality axiom and $\mathcal{A}$ is consistent w.r.t. $\mathcal{O}_p \cup \mathcal{O}_n$. From **P1** we have that $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}_p \cup \mathcal{O}_n$ and using Claim 5 we obtain that each OMQ with query the form $q(x) \leftarrow A(x)$ or $q(x, y) \leftarrow r(x, y)$ can be evaluated over $\mathcal{E}_{\mathcal{O},\mathcal{A}}$. □

By definition, the procedure to construct $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ runs in polynomial time in the size of the ontology and the ABox. Then, from Claim 4 and Proposition 3.3 we obtain the following:

Theorem 3.6. *DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$-ABOX CONSISTENCY *and DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$-CERTAIN ANSWERS *for instance queries are in* PTime.

The recursion safe fragment of *DL-Lite*$_{\mathcal{A}}^{++}$ is not FO-rewritable, since reachability can be encoded using one CRI of the form $r \circ s \sqsubseteq r$. However, we can get rid of recursive CRIs and regain rewritability if we have guarantees that they will only be relevant on paths of bounded length. We formalize this rough intuition next.

Definition 3.8 (k-bounded ABox)**.** *Let $\mathcal{O}$ be a DL-Lite*$_{\mathcal{A}}^{++}$ *ontology and $\mathcal{A}$ an ABox. Let S be a set of simple relations. Given $a, b \in cst(\mathcal{A})$, we say that there exists an S-*path of length n between a and b (in $\mathcal{A}$ w.r.t. $\mathcal{O}$) *if there exist pairwise distinct $d_1, \ldots, d_{n-1} \in cst(\mathcal{A})$ with $d_i \notin \{a, b\}$, and $s_1(a, d_1), \ldots, s_i(d_{i-1}, d_i), \ldots, s_n(d_{n-1}, b) \in \mathcal{A}$ such that $s_i \sqsubseteq^{s}_{\mathcal{O}} s$ and $s \in S$, $1 \leq i < n$. Let*

$$S_p = \{p_i \mid p_1 \circ \ldots \circ p_n \sqsubseteq_{\mathcal{O}} p\} \cap (\mathbf{R}^{\pm}_{\mathbf{s}} \cup \mathbf{F}^{\pm}_{\mathbf{s}}).$$

*We say that $\mathcal{A}$ is k-*bounded for $\mathcal{O}$ *if for each recursive $p \in \mathcal{O}$ there is no S_p-path of size larger than k.*

Since the path for each each recursive CRI is bounded, we simulate recursive CRIs by unfolding them into non-recursive ones. The general idea is to have a new predicate denoting the size of each path that triggers the application of a recursive CRI.

Definition 3.9 (k-unfolding, k-rewriting)**.** *Let $\mathcal{O}$ be an arbitrary DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$ *ontology and $k \geq 0$ fixed. For each non-simple relation $p \in sign(\mathcal{O})$, let p^i where $0 \leq i \leq k$ and $\hat{p}$ be fresh relation names not occurring in $\mathcal{O}$.*

*A k-*unfolding of $\mathcal{O}$ *is a DL-Lite*$_{\mathcal{A}}^{++(non\text{-}rec)}$ *ontology $\mathcal{O}_k$ obtained as follows:*

a) For each non-simple relation p, add new axioms $p \sqsubseteq p^0$ and $p^i \sqsubseteq \hat{p}$, for $0 \leq i \leq k$.
b) Replace each CRI $p_1 \circ s \sqsubseteq p_2$ with $p_1^i \circ s \sqsubseteq p_2^{i+1}$, for $0 \leq i \leq k - 1$.
c) Replace each non-simple p relation in all other axioms with $\hat{p}$ and copy the remaining axioms.

*For a query q, the k-*rewriting of $(\mathcal{O}, q)$ *is the OMQ $(\mathcal{O}_k, \hat{q})$, where $\hat{q}$ is obtained from q by replacing each $p(x, y) \in q$ by $\hat{p}(x, y)$, for each non-simple relation p.*

Applying this transformation yields a non-recursive *DL-Lite*$_{\mathcal{A}}^{++}$ ontology since each CRI in the new ontology is non-recursive. By disallowing $A \sqsubseteq \exists s$, simple roles cannot occur in cycles involving non-simple relations. Therefore, we conclude that $\mathcal{O}_k$ is in *DL-Lite*$_{\mathcal{A}}^{++(non\text{-}rec)}$.

We proceed next with showing that the translation preserves *query non-entailment*, which is the problem of deciding whether there exists a model of an ABox w.r.t. the given ontology that admits no match for a given query, written $\mathcal{I} \not\models q$.

Proposition 3.4. *Let $\mathcal{O}$ be a DL-Lite$_{\mathcal{A}}^{++(rec\text{-}safe)}$ ontology and $k \geq 1$ fixed. For any k-bounded ABox $\mathcal{A}$ and CQ q we have that:*

- ***P1*** *For each $\mathcal{I}$ model of $\mathcal{A}$ w.r.t. $\mathcal{O}$ such that $\mathcal{I} \not\models q$ there exists $\widehat{\mathcal{I}}$ model of $\mathcal{A}$ w.r.t. $\mathcal{O}_k$ such that $\widehat{\mathcal{I}} \not\models \hat{q}$.*
- ***P2*** *For each $\widehat{\mathcal{I}}$ model of $\mathcal{A}$ w.r.t. $\mathcal{O}_k$ such that $\widehat{\mathcal{I}} \not\models \hat{q}$ there exists $\mathcal{I}$ model of $\mathcal{A}$ w.r.t. $\mathcal{O}$ such that $\mathcal{I} \not\models q$.*

Proof. **Proof of P1:** Let $\mathcal{I}$ be an arbitrary model of $\mathcal{A}$ w.r.t. $\mathcal{O}$. We construct interpretation $\widehat{\mathcal{I}}$, with $\Delta^{\widehat{\mathcal{I}}} = \Delta^{\mathcal{I}}$ as follows:

- $A^{\widehat{\mathcal{I}}} = A^{\mathcal{I}}$, for each $A \in \mathbf{C}$;
- $s^{\widehat{\mathcal{I}}} = s^{\mathcal{I}}$, for each simple relation s;
- for each non-simple relation p_2 and each $p_1 \circ s \sqsubseteq p_2 \in \mathcal{O}$:
  - (i) $(p_2^0)^{\widehat{\mathcal{I}}} = \{(d, d') \mid (d, d') \in p_2^{\mathcal{I}} \text{ such that } \{d, d'\} \cap cst(\mathcal{A}) \neq \emptyset\}$
  - (ii) $(p_2^i)^{\widehat{\mathcal{I}}} = \{(d, d') \mid (d, d'') \in {p_1^{i-1}}^{\widehat{\mathcal{I}}} \text{ and } (d'', d') \in s^{\widehat{\mathcal{I}}}\}$, for $1 \leq i \leq k$;
  - (iii) $\hat{p}_2^{\widehat{\mathcal{I}}} = \bigcup_{0 \leq i \leq k} {p_2^i}^{\widehat{\mathcal{I}}}$.

We start with showing that:

$$\text{For each non-simple relation } p \text{ and for } 1 \leq i \leq k \text{ if } (d_1, d_2) \in (p^i)^{\widehat{\mathcal{I}}}, \text{ then } (d_1, d_2) \in p^{\mathcal{I}}. \quad (3.14)$$

If $i = 0$, since $(p^0)^{\widehat{\mathcal{I}}} \subseteq p^{\mathcal{I}}$ the proposition holds. For each $t^{i-1} \circ s \sqsubseteq p^i \in \mathcal{O}_k$ suppose that for each $(d_1, d_2) \in (t^{i-1})^{\widehat{\mathcal{I}}}$ we have that $(d_1, d_2) \in t^{\mathcal{I}}$. Let $(d_1, d_2) \in (t^{i-1} \circ s)^{\widehat{\mathcal{I}}}$ arbitrarily chosen; then there exists d' such that $(d_1, d') \in (t^{i-1})^{\widehat{\mathcal{I}}}$ and $(d', d_2) \in s^{\widehat{\mathcal{I}}} \subseteq s^{\mathcal{I}}$ (since s must be simple). Using our assumption we get that $(d_1, d') \in t^{\mathcal{I}}$ and since $t \circ s \sqsubseteq p \in \mathcal{O}$ we obtain that $(d_1, d_2) \in p^{\mathcal{I}}$. Therefore, from the induction base and induction step, we can conclude that (3.14) holds.

We proceed now with the proof that $\widehat{\mathcal{I}}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}_k$. Clearly, $\widehat{\mathcal{I}} \models \mathcal{A}$ and for each $\alpha \in \mathcal{O} \cap \mathcal{O}_k$ it is easy to check that $\widehat{\mathcal{I}} \models \alpha$. If $\alpha \in \mathcal{O}_k \setminus \mathcal{O}$ we have the following cases:

(i) if α if of one of the forms $p \sqsubseteq p^0$, $p^i \sqsubseteq \hat{p}$ or $t^{i-1} \circ s \sqsubseteq p^i$ then from the construction of $\widehat{\mathcal{I}}$ we have that $\widehat{\mathcal{I}} \models \alpha$;

(ii) if α is of the form $\hat{p_1} \sqsubseteq \hat{p_2}$, then there is $p_1 \sqsubseteq p_2 \in \mathcal{O}$. Let $(d_1, d_2) \in {\hat{p_1}}^{\widehat{\mathcal{I}}}$, then using the construction of $\widehat{\mathcal{I}}$ and (3.14), we get that $(d_1, d_2) \in p_1^{\mathcal{I}}$. Then it must be that $(d_1, d_2) \in p_2^{\mathcal{I}}$ and by construction of $\widehat{\mathcal{I}}$, we get that $(d_1, d_2) \in {\hat{p_2}}^{\widehat{\mathcal{I}}}$, hence $\widehat{\mathcal{I}} \models \hat{p_1} \sqsubseteq \hat{p_2}$. The proof is analogous for $\mathsf{disj}(\hat{p_1}, \hat{p_2})$;

(iii) if α is of the form $\exists \hat{p} \sqsubseteq B$, $B \sqsubseteq \exists \hat{p}$ or $\mathsf{disj}(\hat{p}, p')$, this holds from the construction of $\widehat{\mathcal{I}}$, property (3.14) and the fact that $\mathcal{I}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}$.

We show now that $\mathcal{I} \not\models q$ implies $\widehat{\mathcal{I}} \not\models \hat{q}$. From the construction of $\widehat{\mathcal{I}}$ and the fact that $\widehat{\mathcal{I}}$ is a model, we have that:

- for each $A \in \mathbf{C}$ if $d \in A^{\widehat{\mathcal{I}}}$ then $d \in A^{\mathcal{I}}$;
- for each simple relation s, if $(d, d') \in s^{\widehat{\mathcal{I}}}$ then $(d, d') \in s^{\mathcal{I}}$;
- for each non-simple relation p, if $(d, d') \in \hat{p}^{\widehat{\mathcal{I}}}$ then $(d, d') \in p^{\mathcal{I}}$;

Let q be a query over the signature of $\mathcal{O}$ such that $\mathcal{I} \not\models q$, and let us assume there exists a match π of $\hat{q}$ over $\widehat{\mathcal{I}}$. From the above arguments we have π is also a match of q over $\mathcal{I}$ which contradicts the fact that $\mathcal{I} \not\models q$.

Proof of P2: W.l.o.g., let $\widehat{\mathcal{I}}$ be a minimal model of $\mathcal{A}$ w.r.t. $\mathcal{O}_k$. We define interpretation I, with $\Delta^{\mathcal{I}} \subseteq \Delta^{\widehat{\mathcal{I}}}$, as follows:

- $A^{\mathcal{I}} = A^{\widehat{\mathcal{I}}}$, for all $A \in \mathbf{C}$;
- $p^{\mathcal{I}} = \bigcup_{0 \le i \le k} (p^i)^{\widehat{\mathcal{I}}}$, for each non-simple relation p;
- $s^{\mathcal{I}} = \{(a, b) \in s^{\widehat{\mathcal{I}}} \mid a, b \in cst(\mathcal{A})\}$, for each simple relation s.

It is clear from the construction of $\mathcal{I}$ and the fact that $\widehat{\mathcal{I}}$ satisfies $\mathcal{A}$ and each axiom that occurs in $\mathcal{O} \cap \mathcal{O}_k$ that the same holds for $\mathcal{I}$. It remains to argue that $\mathcal{I}$ satisfies each axiom α in $\mathcal{O} \setminus \mathcal{O}_k$. For that we distinguish the following cases:

(i) if α is of the form $p_1 \circ s \sqsubseteq p_2$, suppose that there is some $(d_1, d_2) \in (p_1 \circ s)^{\mathcal{I}}$ such that $(d_1, d_2) \notin p_2^{\mathcal{I}}$. Then there is some d' such that $(d_1, d') \in p_1^{\mathcal{I}}$ therefore $(d_1, d') \in (p_1^i)^{\widehat{\mathcal{I}}}$, for some i such that $0 \le i \le k$. The only way this can happen is if there exists an S_p-path larger than k connecting d_1 and d_2. By construction of $\mathcal{I}$ this path involves only constants from the ABox, hence we obtain that in $\mathcal{A}$ there is an S_p-path larger than k, which contradicts the fact that $\mathcal{A}$ is k-bounded. Therefore our assumption was wrong and we conclude that $(d_1, d_2) \in p_2^{\mathcal{I}}$, hence $\mathcal{I}$ satisfies α.

(ii) if α is of the form $\exists p \sqsubseteq B$, it follows from the fact that $p^{\mathcal{I}} \subseteq \hat{p}^{\widehat{\mathcal{I}}}$ and $\widehat{\mathcal{I}}$ satisfies $\exists \hat{p} \sqsubseteq B$. Similarly for axioms of the form $\mathsf{disj}(p, p')$.

(iii) if α is of the form $B \sqsubseteq \exists p$, it follows from the facts that $B \sqsubseteq \exists \hat{p} \in \mathcal{O}_k$, $\hat{p}$ does not occur in $\mathcal{A}$ and $\widehat{\mathcal{I}}$ is minimal, therefore $p^{\mathcal{I}} = \hat{p}^{\widehat{\mathcal{I}}}$.

Therefore we obtain that $\mathcal{I} \models \alpha$, for all $\alpha \in \mathcal{O}$, therefore $\mathcal{I}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{O}$. It remains to argue that for $\hat{q}$ if $\widehat{\mathcal{I}} \not\models \hat{q}$ then $\mathcal{I} \not\models q$. This follows from the construction of $\mathcal{I}$. □

From proposition above and Lemma 3.5 we obtain that for k-bounded ABoxes, the k-rewriting ensures FO-rewritability of recursion safe *DL-Lite*$_{\mathcal{A}}^{++}$ OMQs.

Lemma 3.6. *Let $\mathcal{O}$ be a DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$ *ontology, $\mathcal{O}_k$ a k-unfolding of $\mathcal{O}$, for some $k \ge 0$, and q a CQ. Then, for every k-bounded ABox $\mathcal{A}$:*

$$cert(q, \mathcal{O}, \mathcal{A}) = \bigcup_{q' \in rew(\hat{q}, \mathcal{O}_k)} cert(q', \emptyset, \mathcal{A}).$$

The combined complexity of answering CQs in the recursion safe fragment of *DL-Lite*$^{++}_{\mathcal{A}}$ is NP-complete, since the lower-bound is inherited from answering CQs over relational databases while the upper-bound follows from Theorem 3.6.

Theorem 3.7. *For k-bounded and consistent ABoxes, DL-Lite*$^{++(rec\text{-}safe)}_{\mathcal{A}}$-CERTAIN ANSWERS *is* NP*-complete and DL-Lite*$^{++(rec\text{-}safe)}_{\mathcal{A}}$-CERTAIN ANSWERS$(\mathcal{O},q)$ *is in* AC^0.

Ensuring k-boundedness. As we defined before, an ABox is k-bounded whenever each S_p-path has length at most k. We can guarantee this by enforcing each simple role to connect constants in an orderly manner: ordering the concepts which are connected via simple roles from "smaller" to "larger" and enforcing that each S_p-path is respecting this order.

Definition 3.10 (Order constraints)**.** *For a simple role s, an* order constraint (along s) *takes the form $ord(s, \mathbf{A}, \prec)$, with $\mathbf{A} \subseteq \mathbf{C}$ finite, and $\prec$ a strict partial order over $\mathbf{A}$. An interpretation $\mathcal{I}$ satisfies $ord(s, \mathbf{A}, \prec)$ if*

$$s^{\mathcal{I}} \subseteq \bigcup_{A_1, A_2 \in \mathbf{A}} (A_1^{\mathcal{I}} \times A_2^{\mathcal{I}}), \quad s^{\mathcal{I}} \cap \bigcup_{A_1 \not\prec A_2} (A_1^{\mathcal{I}} \times A_2^{\mathcal{I}}) = \emptyset.$$

Let $\mathbf{C}$ be a set of order constraints. We say that $\mathbf{C}$ covers a relation p in a given ontology $\mathcal{O}$ *if there exists a partial order $(\mathbf{A}, \prec)$ such that for every simple relation s in the set $\{s \mid p' \circ s \sqsubseteq p \in \mathcal{O}\}$, $ord(s, \mathbf{A}', \prec) \in \mathbf{C}$ for some $\mathbf{A} \subseteq \mathbf{A}'$. We say that $\mathbf{C}$ covers $\mathcal{O}$, if it covers every relation p in $\mathcal{O}$.*

Further, for a recursion safe DL-Lite$^{++}_{\mathcal{A}}$ *ontology $\mathcal{O}$ and an ABox $\mathcal{A}$, we say that $\mathcal{A}$ is* $\mathbf{C}$-admissible w.r.t. *$\mathcal{O}$ if (i) $\mathbf{C}$ covers $\mathcal{O}$, and (ii) $\mathcal{E}_{\mathcal{O},\mathcal{A}}$ satisfies each $c \in \mathbf{C}$.*

Concepts capturing different types of geographical locations, can be in general ordered according to a granularity level. In our running example, it is natural to consider the order District $\prec$ City $\prec$ Country. By enforcing such order using order constraints, the instances of such concepts are implicitly connected via partOf-paths complying with such order. One result of such enforcement is that each partOf-path is bound by the size of the order, which in this case is 3. Thus, for a $\mathbf{C}$-admissible ABox $\mathcal{A}$, the parameter k corresponds to the maximal cardinality of $\mathbf{A}$ in an order constraint in $\mathbf{C}$.

Lemma 3.7. *Let $\mathcal{O}$ be a recursion-safe DL-Lite*$^{++}_{\mathcal{A}}$ *KB, and let $\mathbf{C}$ be a set of order constraints covering $\mathcal{O}$. For any given ABox $\mathcal{A}$, if $\mathcal{A}$ is $\mathbf{C}$-admissible w.r.t. $\mathcal{O}$, then then $\mathcal{A}$ is $\ell(\mathbf{C})$-bounded for $\mathcal{O}$, where $\ell(\mathbf{C}) = max\{|\mathbf{A}| \mid ord(s, \mathbf{A}, \prec) \in \mathbf{C}\}$.*

Finally, we note that $\mathbf{C}$-admissibility amounts to evaluating simple queries on $\mathcal{E}_{\mathcal{O},\mathcal{A}}$, which can be done in time that is polynomial in $\mathbf{C}$, $\mathcal{O}$, and $\mathcal{A}$. Moreover, although testing $\mathbf{C}$-admissibility is data dependent, once it is established, FO-rewritability is guaranteed for any CQ.

Proposition 3.5. *Checking $\mathbf{C}$-admissibility for recursion safe DL-Lite*$^{++}_{\mathcal{A}}$ *KBs is feasible in polynomial time in combined complexity.*

This concludes the theoretical part of this chapter. In the next section we discuss how to incorporate aggregation into *DL-Lite*$^{++(rec\text{-}safe)}_{\mathcal{A}}$ to capture functionalities reminiscent of OLAP.

3.2.3 The Case for Recursion-safe *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$

Similarly as for *DL-Lite*$_{\mathcal{A}}$, adding aggregating concepts to the recursion safe *DL-Lite*$_{\mathcal{A}}^{++}$ is not computationally problematic. We can lift the results from the previous chapter to show that each aggregating concept can be replaced by a fresh concept name such that the canonical models, of the original ontology and that of the translated ontology are isomorphic. This is due to the fact that CRIs can be applied only close to the ABox, therefore the application of axioms with aggregating concepts is also bounded by the largest path of simple relations in the ABox.

Definition 3.11. *A* recursion safe *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}++}$ *ontology is a DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ *ontology which may contain in addition a set of recursion safe CRIs.*

In order to define the notion of modelhood also for this case, we extend the notion of a feature closed interpretation to take into account CRIs: an interpretation $\mathcal{I}$ is called *feature closed* for $(\mathcal{O}, \mathcal{A})$, where $\mathcal{O}$ is a *DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$ ontology and $\mathcal{A}$ is an ABox, if for each $f \in \mathbf{F}$, $f^{\mathcal{I}} = \{(a, v) \mid \mathcal{I} \vDash p_1 \circ \ldots \circ p_n(a, v), a, v \in cst(\mathcal{A})$ and $p_1 \circ \ldots \circ p_n \sqsubseteq_{\mathcal{O}} f\}$. Then Lemma 2.2 is lifted by extending the closure of the ABox $\mathcal{A}_f$ to contain also objects of the form a_r, where $a \in cst(\mathcal{A})$ and r is a non-simple role. Since features and simple roles cannot be existentially implied, due to the restrictions inherited from *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}}$ and those imposed by recursion safeness, we can apply each CRI that involves features in an apriori procedure since such CRIs are not triggered at a later stage in the chase procedure. Threfore, the remaining of the proof of Lemma 2.2 stays the same. Furthermore, the translation into the aggregation-free counterparts of $\mathcal{O}, \mathcal{A}$ and q remain the same. Therefore the following result is straightforward:

Proposition 3.6. *Let $\mathcal{O}$ be a recursion safe DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}++}$ *ontology. For each ABox $\mathcal{A}$ and CQA q, we have that $cert(q, \mathcal{O}, \mathcal{A}) = cert(q', \mathcal{O}', \mathcal{A}')$, where $\mathcal{O}'$, $\mathcal{A}'$ and q' are their DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$ *translations.*

In our running example, extending $\mathcal{O}_{MCI}^{\mathsf{agg}}$ from Example 2.3 with the CRI 3.1 falls into the recursion safe fragment of *DL-Lite*$_{\mathcal{A}}^{\mathsf{agg}++}$. Such extension, allows us to propagate incidents along locations connected by a partOf-path. Hence, for the query

$$q_1(x) \leftarrow \text{HumanCausedMCI}(x) \wedge \text{SevereMCI}(x) \wedge \text{hasLocation}(x, y) \wedge y = \mathsf{Austria}.$$

when evaluated over ABox $\mathcal{A}$ from Example 2.4, $\mathsf{e1}$ is a certain answer, given that its location is within the geographical borders of Austria. Moreover, since partOf is a simple role, infinite sequences of objects connected by it are disallowed. Moreover, using CRIs, we can also define integrity constrains that must be satisfied by the sub-component in order to be part of the whole. In our example, suppose that constants belonging to the *String* data-value domain are lexicographically ordered, that is we can compare them: i.e., $\mathsf{no} < \mathsf{yes}$. Then, using the following axioms we can define the concept of a location with hospital access by enforcing that some sub-location or a connected location must have access to a hospital.

$$=_{\mathsf{yes}} \mathsf{min}(\text{hospAccess}) \sqsubseteq \text{HospAccessible}$$
$$\exists \text{partOf}.\text{HospAccessible} \sqsubseteq \text{HospAccessible}$$
$$\exists \text{connectedTo}.\text{HospAccessible} \sqsubseteq \text{HospAccessible}.$$

By the first axiom, we obtain that distr_2 is an instance of HospAccessible, while by the latter two we infer that also distr_1, Vienna and Austria are as well instances of HospAccessible. Note that the latter axioms can be simulated given that $\mathcal{ELI}$ concepts can be translated using recursion safe CRIs and it does not require value invention using simple relations.

We show next the usefulness of considering queries with aggregation and recursion-safe *DL-Lite*$^{++}_{\mathcal{A}}$ ontologies. Suppose that a new MCI is reported using the following facts: Earthquake(e_2), hasLocation($\mathsf{e}_2, \mathsf{loc}_1$), partOf($\mathsf{loc}_1$, Austria), date($\mathsf{e}_2$, 10/02/2020). We again consider that constant symbols of data-value domain Date can be chronologically ordered: e.g., 10/02/2020 $<$ 10/04/2020. Using the following CQA, we can then identify all incidents that have occurred in Austria since the beginning of 2020:

$$q_2(x) \leftarrow \text{MCI}(x) \wedge \leq_{01/01/2020} (\mathsf{min}\ \text{date})(x) \wedge \text{hasLocation}(x, y) \wedge y = \mathsf{Austria}.$$

Since for e_1 an exact date is not given, only e_2 is a certain answer to such query. As discussed before, computing certain answers to such query can be done using the rewriting procedure, provided that for the ABox $\mathcal{A}$, we have that $\mathcal{I}_{\mathcal{A}}$ is feature closed. Moreover, using the following analytical query we can also calculate the average number of casualties in any location in Austria that occurred since the beginning of 2020.

$$\begin{aligned} q_3(\mathsf{avg}(u), z) \leftarrow & \text{MCI}(x) \wedge \leq_{01/01/2020} (\mathsf{min}\ \text{date})(x) \wedge \text{casualties}(x, u) \wedge \text{hasLocation}(x, z) \\ & \wedge \text{partOf}(z, y) \wedge y = \mathsf{Austria}. \end{aligned}$$

Since we rely on the epistemic semantics to compute answers to analytical queries, answering q_3 is achieved by computing certain answers to the CQA obtained by replacing $\mathsf{avg}(u)$ with u in q_3, grouping by sub-location and then averaging over the number of casualties.

These examples show the potential of enriching standard OMQA to allow CRIs, aggregating concepts in the ontology and queries with aggregation. The good news is that this novel formalism does not come with a high computational cost, thus existing OMQA evaluation techniques can be easily adapted to this new setting.

3.3 Related Work and Discussion

The importance of CRIs was acknowledged since the earliest works on description logics, when *role value maps* were considered very desirable and included in the first DL system KL-ONE [BS85]. Such construct is useful to define path-complying concepts such as (friendOf $\circ$ isChildOf $\sqsubseteq$ knowsParent) which denotes that each person knows the parents of their friends. CRIs are equivalent to the global version of role value maps, i.e., the CRI friendOf $\circ$ isChildOf $\sqsubseteq$ knowsParent corresponds to $\top \sqsubseteq$ (friendOf $\circ$ isChildOf $\sqsubseteq$ knowsParent). As shown in [SS89], the extension of $\mathcal{ALC}$ with role value maps leads to undecidability of subsumption and consistency testing, however the proof uses a strictly more expressive language thus it cannot be inherited for the *DL-Lite* case. More recently, some positive results are shown in [BT20] for which subsumption in the presence of role value maps is decidable for $\mathcal{FL}_0$.

As described in [HS04], there is a thight connection between CRIs and *grammar logics* [dCP88] which represent a class of modal logics in which the accessibility relations are given by means of a grammar and their semantics is defined in terms of relational structures that satisfy that grammar. Similarly as for CRIs, the expressiveness of the grammar influences the complexity of reasoning for such languages: for regular grammars, consistency testing is EXPTIME-complete [Dem01] whereas this problem is undecidable for context-free grammars [Bal03, BGM98]. The undecidability result is obtained from reducing the emptiness problem for the intersection of context-free grammars, however due to our restrictions on CRIs, not all context-free languages are captured in our case. Similarly for the regular fragment of $\textit{DL-Lite}_{\mathcal{A}}^{++}$, the ExpTime-hardness cannot be directly obtained from the regular grammar logics, given that our CRIs cannot capture all regular languages, as argued in [Kaz10].

Besides the already discussed extensions of $\mathcal{ALC}$ that consider CRIs, also in the $\mathcal{EL}$ family there exists an extension which captures them, namely $\mathcal{EL}^{++}$ [BBL05]. In distinction to all the other DLs, consistency testing in $\mathcal{EL}^{++}$ is tractable even for unrestricted CRIs. However, $\mathcal{EL}^{++}$ does not allow for inverses, which means that the underlying cause of undecidability for $\textit{DL-Lite}_{\mathcal{A}}^{++}$ is the combination of CRIs and inverse roles. Indeed, for the extension of $\mathcal{EL}^{++}$ with *range restrictions* it has been shown that the subsumption problem becomes undecidable unless the ranges of the path and the closure of the path coincide [BLB08]. Regarding query answering, results are not as optimistic given that entailment of CQs in $\mathcal{EL}$ ++ is undecidable [Ros07].

Capturing recursion in DLs is among the desired effects of having CRIs. This is a central property of Datalog and its exended version Datalog$^{\pm}$ [CGL12, BLMS11] which allows for the central property of DLs, namely *value invention*. It is then not a surprise that Datalog$^{\pm}$ is undecidable and quests for finding decidable and even tractable languages have been lunched while correspondences to DLs languages have been established. Among the tractable fragments is the *linear* Datalog$^{\pm}$ [CGL09a] which captures $\textit{DL-Lite}_{\mathcal{R}}$, however each rule can have at most one atom in the body, therefore CRIs are not captured. Similarly, all the FO-rewritable fragments: *sticky* and its extension *sticky-join* [CGP10] and *binary guarded* [CR15] do not capture the expresivity of *DL-Lite* with CRIs.

Our work is also related to *regular path queries* (RPQs) and their extensions. In fact, the kind of query answering we advocate is naturally supported in any ontology-mediated setting where the DL has CRIs, or the query language contains conjunctive RPQs. Many such settings have been considered in the literature and their complexity is well understood, see [Ort13, OS12] for references. However, any such combination is necessarily NLogSpace-hard in data complexity, and the combined complexity is usually PSpace-hard even for lightweight DLs [BOS15].

CHAPTER 4

Ontology-enhanced Exploratory Framework

For graph-structured data, ontologies act as an interface to data sources, facilitating query formulation by providing a vocabulary closer to the users' understanding. However, accessing data in large and unfamiliar sources can still be challenging for non-domain experts. In practice, users may find that the formulated queries have too many answers to be informative, or at the other extreme, they have very few or no answers at all. Refining queries can be difficult for users who may not know how to formulate their information needs according to the underlying data since some changes in the query may have no effect on the answers, or dramatically affect them in unexpected ways.

Manual exploration via iterative query evaluation can be computationally very costly if each minor query variation is evaluated independently and from scratch. For example, querying all mass casualty incidents (MCIs) over a huge historical dataset can produce a large and rather uninformative set of answers. In this case a natural way to filter out some of the answers is to select among suitable *specializations* of the query that partition the answers into subsets which are more informative and easier to handle by the user. Such partitions could be generated by choosing more fine-grained categories of MCIs such as floods, fires, earthquaques but also by selecting a smaller geographical area. Such partitions can be conversely aggregated into larger partitions such as MCIs that are caused either by humans or nature, which intuitively represent *generalizations* of the more selective queries.

The ontology-enhanced specializing and generalizing operations on queries resemble the *rolling-up* and *drilling-down* functionalities supported by the *online analytical processing (OLAP)* paradigm which allows the users to analyze multidimensional data in an interactive fashion. In a nutshell, a multidimensional dataset encodes relations between several independent attributes, called *dimensions*, such as location, time, type of MCI, etc., and a certain number of numerical attributes, called *measures*, such as number of casualties or number of severe MCIs, which depend on the

particular chosen values in each dimension. Typically, a record in a multidimensional dataset encodes an observation of a value (e.g., number 160) and the attributes describing that value (e.g. date of MCI: “05/10/2014", location: “Vienna", type of MCI: “fire"). A dimension usually consists of several categories denoting different granularity levels, for instance, the location dimension can be composed of district, city, country and region. Querying in this case offers a multidimensional perspective over the statistical facts encoded in the dataset: e.g., “number of casualties caused by fires that occurred in Vienna in 2014”. A rolling-up operation changes the perspective to an upper level on some particular dimension (e.g., from “Vienna” to “Austria”), while drilling-down represents the converse operation (e.g., from “Vienna” to “Innerestadt”–a specific district of Vienna). Intuitively, the number of answers to queries can potentially increase with each rolling-up and decrease with each drilling-down operation similarly to each specialization and generalization operation.

In this chapter we propose an exploratory framework for graph-structured data that supports some analytical functionalities in the style of OLAP. The general idea is to generate and interactively navigate a large set of queries which are related via query containment relation. The main ingredients of our abstract framework are: a) a query template language that extends the expressivity of CQs by allowing concept and relations in the query body to be relaxed or restrained, b) a set of rules based on the chosen knowledge-base to generate valid CQs from the given template, c) a Datalog encoding to automatically compile the data, the ontology and the set of generated queries such that answering each specialization and generalization can be done very efficiently, thus enabling online exploration of the targeted data.

4.1 Abstract Exploratory Framework

In this section we propose a general approach for data exploration by means of identifying meaningful query reformulations.

4.1.1 Motivating Example

It is often the case that the answer to a given query rises many follow-up questions. Suppose that for the following query mediated by the MCI ontology we obtain following tabular representation of the answers:

$$q(x, y) \leftarrow \text{MCI}(x) \wedge \text{hasLocation}(x, y) \wedge \text{Location}(y)$$

MCI	Location(s)
ev_1	Vienna, Austria
ev_2	Austria
ev_3	Austria
ev_4	Austria, Croatia

Such answer is not very informative, for example it does not present the specific type of each MCI. In addition, for incident ev_4 we obtain it has occurred in multiple locations and would require further inquires to understand why we obtained such answer tuples. In principle, it could be

that ev_4 is either some severe MCI that has indeed affected all those areas, or it could be a data anomaly, therefore this may require further investigation and potential fixes.

In order to explore the topic of interest, namely types of existing MCIs and their locations, there are several approaches. Once could potentially create a more involved SPARQL query that incorporates such information request. For example, the following query:

```
1  SELECT *
2  WHERE {
3          ?x a ?y .
4          ?y rdfs:subClassOf* :MCI .
5          ?x :hasLocation ?z.
6          ?z a ?u .
7          ?u rdfs:subClassOf* :Location .
8  }
```

retrieves all existing MCIs, their specific kind (lines 3-4), and their specific type of location (lines 6-7).

However, in general, there are several issues with such approach: (i) it requires knowledge of SPARQL 1.1. query language and of the reasoning steps needed to ensure completeness of answers, (ii) for ontology languages beyond *DL-Lite*, SPARQL is not expressive enough to capture full reasoning [BKPR14], (iii) the returned answer may be too large and uninformative to the user, thus requiring further manipulation and analysis. Indeed the above SPARQL query is not complete w.r.t. $\mathcal{O}_{MCI}$ and the answer would require some careful filtering due to repetitive and irrelevant information, since it is not possible to identify only the most specific type of MCIs and their most specific location.

Alternatively, the user can start with the most specific query and then manually change the type of MCI and the type of location based on the retrieved answers. This process can be very tedious and time consuming. What we propose instead is to automatically construct all the possible follow-up queries using the relevant concept and relation hierarchies in the ontology. Moreover, we also want to take into account individuals and data values that the user deems relevant. In our running example, the relevant follow-up queries are captured by the following querying template:

$$q_{ref}(x, y) \leftarrow Atom_1(x) \wedge \text{hasLocation}(x, y) \wedge Atom_3(y),$$

where:

$Atom_1(x) \in \{\text{MCI}(x), \text{NaturalCausedMCI}(x), \text{HumanCausedMCI}(x), \text{Earthquake}(x), \text{Flood}(x),$
$\text{SevereMCI}(x)\}$, and
$Atom_3(y) \in \{\text{District}(y), \text{City}(y), \text{Country}(y), \text{Location}(y), y = \mathsf{Vienna}, y = \mathsf{Austria}\}.$

This means that even for simple queries, consisting of only few atoms, the number of possible follow-up queries can be rather large. The question is then how to identify the most relevant reformulations for a given query? We consider that ideally the best follow-up queries are selected

based on the following criterias: *semantic relatedness* and *answer quality*. What we mean by a query being *semantically related* is that it tries to preserves as much as possible of the meaning captured by the original query. For example,

$q_1(x, y) \leftarrow \text{NaturalCausedMCI}(x) \wedge \text{hasLocation}(x, y) \wedge \text{Location}(y)$

is semantically more related to q than

$q_2(x, y) \leftarrow \text{Fire}(x) \wedge \text{hasLocation}(x, y) \wedge \text{Location}(y)$

given that NaturalCausedMCI is an immediate subconcept of MCI while Fire is a more specific subconcept.

The second criteria used to identify the relevant reformulations is how well the reformulation can discriminate the underlying data. If we look at Figure 4.1 we can see the distribution of the answers among the relevant concepts such as type of MCI and the concrete location it is based in. Using this diagram, there are several observations to be made regarding the answer quality of the relevant reformulations:

1. There are certain specializations and generalizations which do not affect the answers, that is they are *neutral* reformulations. For instance, the query

   $$q_1(x, y) \leftarrow \text{MCI}(x) \wedge \text{hasLocation}(x, y) \wedge \text{Country}(y)$$

   has the same answers though it is more specific. Similarly, since Vienna is the only existing city in which MCIs have been reported, queries

   $$q_2(x, y) \leftarrow \text{MCI}(x) \wedge \text{hasLocation}(x, y) \wedge \text{City}(y), \text{ and}$$
   $$q_3(x, y) \leftarrow \text{MCI}(x) \wedge \text{hasLocation}(x, y) \wedge y = \text{Vienna}$$

   produce the same answers, namely $\{(\text{ev}_1, \text{Vienna})\}$, with q_3 more specific than q_2.
2. Among the neutral reformulations, those that bound the answers from above and below are of interest since they respectively encode the most general and the most specific properties that the answers have in common. If we consider for instance the tuples $(\text{ev}_4, \text{Austria})$, $(\text{ev}_4, \text{Croatia})$, the most specific query which captures and implicitly describes these answers is:

   $q_s(x, y) \leftarrow \text{Earthquake}(x) \wedge \text{SevereMCI}(x) \wedge \text{NaturalCausedMCI}(x) \wedge \text{hasLocation}(x, y)$
   $\wedge \text{Country}(y)$,
   while the most general is:
   $q_g(x, y) \leftarrow \text{NaturalCausedMCI}(x) \wedge \text{SevereMCI}(x) \wedge \text{hasLocation}(x, y) \wedge \text{Location}(y)$.

3. There are multiple ways to specialize or generalize queries such that answers are either dropped or added. For example, when specializing q, one possibility is to focus on one of NaturalCausedMCIs or HumanCausedMCIs, but another is to filter out the answers by choosing a more specific location. However, in order to allow the user to understand and create a complete picture of the data, much like the illustrated diagram, gradual exploration of answers is important, therefore the queries which minimally modify answers should

be chosen among all the possible follow-up queries. For example to specialize q_1 while preserving most of the answers, query

$$q_4(x, y) \leftarrow \text{MCI}(x) \wedge \text{hasLocation}(x, y) \wedge y = \text{Austria}$$

is preferred among the other specializations that filter out answer tuples such as q_2 and q_3.

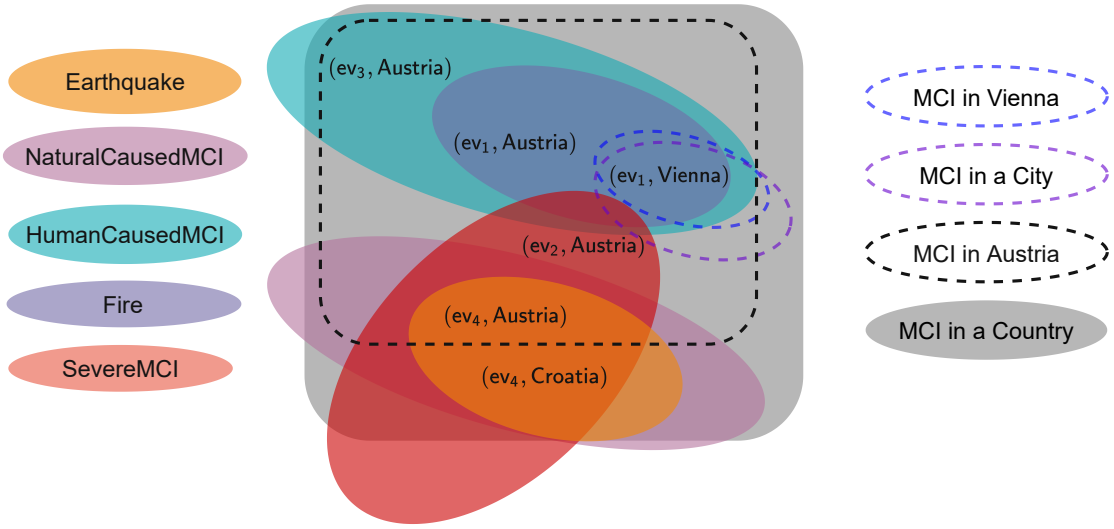


Figure 4.1: Answers representation based on relevant semantic properties.

In the next section, we formalize the problem of *exploring datasets by means of query navigation*. The formalization of the exploratory framework relies on an abstract preorder between queries used to denote the semantic relation between queries, and we rely on containment of certain answers to identify which are the most interesting reformulations according to the answer quality criteria.

4.1.2 Problem Formulation

We start by formalizing the notion of *query space* which represents a set of ontology-mediated queries that are related by means of a preorder $\preceq$ and whenever such relation ensures containment we can use it for exploratory purposes like the ones previously exemplified.

Definition 4.1. *(Query space) Let $\mathcal{O}$ be an arbitrary ontology. For any given ABox $\mathcal{A}$, a preordered set $\mathcal{Q}_\mathcal{O} = \langle Q, \preceq \rangle$ of queries mediated by $\mathcal{O}$ is called an* exploratory query space for $\mathcal{A}$ *if $(q_1, \mathcal{O}) \preceq (q_2, \mathcal{O})$ implies $q_1 \subseteq_{(\mathcal{O},\mathcal{A})} q_2$. We will write $q_1 \preceq q_2$, whenever $\mathcal{O}$ is clear from the context and similarly write $\mathcal{Q}$ instead of $\mathcal{Q}_\mathcal{O}$.*

In the following we focus on the formalization of query space navigation to ensure exploration of answers. For that we consider an arbitrary query q within the space and identify two kinds of reformulations for q by exploring the preorder $\preceq$: We are interested in identifying other queries in the space that are either more specific or more general than q which can be categorized based on containment of answers into (i) *strict* refomulations, which strictly change the set of answers, and (ii) *neutral* reformulations, which do not change the set of answers.

In the remaining of this chapter it is convenient to consider an arbitrary but fixed setting. For that, let $\mathcal{A}$ be an ABox, $\mathcal{O}$ an ontology and $\mathcal{Q}_\mathcal{O} = (Q, \preceq)$ an exploratory query space for $\mathcal{A}$.

Definition 4.2 (Neutral and strict reformulations)**.** *For queries $q_1 \neq q_2$ in $\mathcal{Q}_\mathcal{O}$ such that $q_1 \preceq q_2$, we say that q_1 is a* neutral *specialization of q_2, written $q_1 \simeq q_2$, if $cert(q_1, \mathcal{O}, \mathcal{A}) = cert(q_2, \mathcal{O}, \mathcal{A})$, and it is a* strict *specialization of q_2, written $q_1 \prec q_2$, if $cert(q_1, \mathcal{O}, \mathcal{A}) \subsetneq cert(q_2, \mathcal{O}, \mathcal{A})$. Conversely, q_2 is a* neutral*, respectively* strict*, generalization of q_1.*

Note that $\simeq$ is not symmetric since $\preceq$ is not symmetric either and we want to avoid cases such as $q_1 \preceq q_2$ and $q_1 \simeq q_2$ hold but $q_2 \not\preceq q_1$. This restriction prevents that a neutral reformulation is both a specialization and a generalization.

Using the neutral reformulations, we can identify the common semantic properties of the targeted answers. For that, we want to identify the most specific and the most general properties that the answers share. For example, considering Figure 4.1, for query $q(x, y) \leftarrow \text{Earthquake}(x) \wedge \text{hasLocation}(x, y) \wedge y = \text{Austria}$ which captures only tuple $(\text{ev}_4, \text{Austria})$, the most specific neutral specializations would be: $q_s(x, y) \leftarrow \text{Earthquake}(x) \wedge \text{NaturalCausedMCI}(x) \wedge \text{SevereMCI}(x) \wedge \text{hasLocation}(x, y) \wedge y = \text{Austria}$, while the most general neutral generalizations would be $q_g^1(x, y) \leftarrow \text{NaturalCausedMCI}(x) \wedge \text{hasLocation}(x, y) \wedge y = \text{Austria}$, and $q_g^2(x, y) \leftarrow \text{SevereMCI}(x) \wedge \text{hasLocation}(x, y) \wedge y = \text{Austria}$.

In general there are multiple such reformulations which may be incomparable w.r.t. $\preceq$.

Definition 4.3 (MSN specializations and MGN generalizations)**.** *Let $q_1 \preceq q_2$. If $q_1 \simeq q_2$ in $\mathcal{Q}_\mathcal{O}$, we say that q_1 is a* most specific neutral *(MSN) specialization of q_2 if for each $q' \in Q_\mathcal{O}$ such that $q' \preceq q_1$ and $q_1 \not\preceq q'$, it holds that $q' \prec q_2$. Conversely, q_2 is a* most general neutral *(MGN) generalization of q_1 if for each $q' \in Q$ such that $q_2 \preceq q'$ and $q' \not\preceq q_2$, we have $q_1 \prec q'$.*

In order to ensure a gradual exploration, we need to identify the reformulations that modify the answers in a minimal way. For example if our query is $q(x, y) \leftarrow \text{MCI}(x) \wedge \text{hasLocation}(x, y) \wedge \text{Location}(y)$, the noteworthy changes on q which have a minimal impact on the set of answers are either to choose the HumanCausedMCI or NaturalCausedMCI instead of MCI. We call such queries the most general strict specializations of q, and for the generalizing case such reformulations are called the most specific strict generalizations.

Definition 4.4 (MGS specializations and MSS generalizations)**.** *If $q_1 \prec q_2$, we say that q_1 is a* most general strict *(MGS) specialization of q_2, if for each $q' \in Q$ such that $q_1 \preceq q' \preceq q_2$ we have $q' \simeq q_2$. Conversely, q_2 is a* most specific strict *(MSS) generalization of q_1 if for each $q' \in Q$ such that $q_1 \preceq q' \preceq q_2$ we have $q_1 \simeq q'$.*

For a given query $q \in \mathcal{Q}_\mathcal{O}$, we denote *the set of all most specific neutral specializations* and of *all most general strict specializations* of q given $\mathcal{A}$ by $msnSpe_\mathcal{A}(q, \mathcal{Q}_\mathcal{O})$ and $mgsSpe_\mathcal{A}(q, \mathcal{Q}_\mathcal{O})$, respectively. Similarly, we denote *the set of all most general neutral generalizations* and *all most specific strict generalizations* of q given $\mathcal{A}$ by $mgnGen_\mathcal{A}(q, \mathcal{Q}_\mathcal{O})$ and $mssGen_\mathcal{A}(q, \mathcal{Q}_\mathcal{O})$, respectively.

The decision problems associated with query space navigation are the following:

$\mathcal{Q}_\mathcal{O}$-MGS-SPE	
Input:	A query space $\mathcal{Q}_\mathcal{O}$ exploratory for ABox $\mathcal{A}$, q_1, q_2 in $\mathcal{Q}_\mathcal{O}$.
Question:	Is q_1 a most general strict specialization of q_2?

$\mathcal{Q}_\mathcal{O}$-MSS-GEN	
Input:	A query space $\mathcal{Q}_\mathcal{O}$ exploratory for ABox $\mathcal{A}$, q_1, q_2 in $\mathcal{Q}_\mathcal{O}$.
Question:	Is q_1 a most specific strict generalization of q_2?

To solve such problems in general we need to test for containment of answers. In the worst case containment has to be tested at most $(n \cdot (n-1))/2$, where $n = |Q|$ (i.e., the number of unique query pairs).

If we know already for each query $q \in \mathcal{Q}$ the set of $msnSpe_\mathcal{A}(q, \mathcal{Q})$ and the set of $mgnGen_\mathcal{A}(q, \mathcal{Q})$ then we can identify all pairs $q_1, q_2 \in \mathcal{Q}$ such that $q_1 \simeq q_2$. Moreover, for pairs q_1, q_2 such that $q_1 \preceq q_2$ and $q_1 \not\simeq q_2$ we can infer that $q_1 \prec q_2$. Then if we consider $\prec^1$ to be the one-step $\prec$ relation, then each q' such that there is some $q_n \simeq q$ and $q' \prec^1 q_n$ is a most general strict specialization of query q. Similarly, each q' such that there is some $q \simeq q_n$ and $q_n \prec^1 q'$ is a most specific strict generalization of q. In another words, if we know which are the most specific neutral specializations and most general neutral generalizations, we are able to identify also the most general strict specializations and most specific strict generalizations without the need to test for containment. We formally state this observation.

Observation 4.1. *Let $\mathcal{A}$ be an ABox and $\mathcal{Q}_\mathcal{O} = \langle Q, \preceq \rangle$ be an exploratory query space for $\mathcal{A}$ and let $q_1 \preceq q_2$ in $\mathcal{Q}_\mathcal{O}$. Then, we have that $q_1 \simeq q_2$ iff either (i) there is some $q_n \in msnSpe_\mathcal{A}(q_2, \mathcal{Q}_\mathcal{O})$ such that $q_n \preceq q_1$, or (ii) there is some $q_n \in mgnGen_\mathcal{A}(q_1, \mathcal{Q}_\mathcal{O})$ such that $q_2 \preceq q_n$.*

Similarly, we have that $q_1 \prec q_2$ iff either (i) for all $q_n \in msnSpe_\mathcal{A}(q_2, \mathcal{Q}_\mathcal{O})$ we have that $q_n \not\simeq q_1$, or (ii) for all $q_n \in mgnGen_\mathcal{A}(q_1, \mathcal{Q}_\mathcal{O})$ we have that $q_2 \not\simeq q_n$. Let $q_1 \prec^1 q_2$ if $q_1 \prec q_2$ and there is no other q' such that $q_1 \preceq q' \preceq q_2$.

Then, for any $q \in \mathcal{Q}_\mathcal{O}$, we have that:

- *$q_1 \in mgsSpe_\mathcal{A}(q, \mathcal{Q}_\mathcal{O})$ iff $q_1 \prec^1 q_n$, for some $q_n \simeq q$.*
- *$q_1 \in mssGen_\mathcal{A}(q, \mathcal{Q}_\mathcal{O})$ iff $q_n \prec^1 q_2$, for some $q \simeq q_n$.*

Therefore, the utility of most specific neutral specializations and most general neutral generalization is two-fold: (i) they allow us to identify for a set of target individuals which are the most

specific and most general common properties, and (ii) they enable us to identify the relevant strict reformulations. To show that, we first define the following decision problems:

$\mathcal{Q}_\mathcal{O}$-MSN-SPE	
Input:	A query space $\mathcal{Q}_\mathcal{O}$ exploratory for ABox $\mathcal{A}$, q_1, q_2 in $\mathcal{Q}_\mathcal{O}$
Question:	Is q_1 a most specific neutral specialization of q_2?

$\mathcal{Q}_\mathcal{O}$-MGN-GEN	
Input:	A query space $\mathcal{Q}_\mathcal{O}$ exploratory for ABox $\mathcal{A}$, q_1, q_2 in $\mathcal{Q}_\mathcal{O}$.
Question:	Is q_1 a most general neutral generalization of q_2?

Lemma 4.1. *$\mathcal{Q}_\mathcal{O}$-MSN-SPE is CONP-hard, even for query spaces consisting of only tree-shaped CQs with one answer variable.*

Proof. We reduce the problem of verifying if D is the $\mathcal{EL}$ least common subsummer (LCS) of concepts $C_1, \ldots, C_n$. This amounts to verifying if for each $i \in \{1, \ldots, n\}$ we have that $\models C_i \sqsubseteq D$ and for any $\mathcal{EL}$ concept D' such that $\models C_i \sqsubseteq D'$ we have that $\models D \sqsubseteq D'$. This problem is known to be CONP-complete [JLW20].

Given an $\mathcal{EL}$ concept C we can construct a tree-shaped ABox $\mathcal{A}_C$ denoting concept C with root denoted by constant a_C. Similarly, we can construct a tree-shaped CQ $q_C(x)$ denoting C.

Let $C_1, \ldots, C_n$ and D be $\mathcal{EL}$ concepts. We construct an ABox $\mathcal{A}$ and an exploratory query space $\langle Q, \preceq \rangle$ for $\mathcal{A}$ as follows:

- $\mathcal{A} = \mathcal{A}_{C_1} \cup \cdots \cup \mathcal{A}_{C_n}$ and in addition we add $A(a_{C_i})$ for $i = 1, \ldots, n$, where A is a fresh concept name;
- Q contains each tree-shaped CQ $q_B(x)$ where B is a $\mathcal{EL}$ concept such that $\mathcal{A} \models B(a_{C_i})$;
- The preorder $\preceq$ is defined as follows: $q_1 \preceq q_2$ if all query atoms in q_2 are contained in q_1, modulo renaming variables.

Clearly, $\langle Q, \preceq \rangle$ is an exploratory space for $\mathcal{A}$.

We show next that D is the LCS of $C_1, \ldots, C_n$ iff $q_{A \sqcap D}(x)$ is a most specific specialization of $q_A(x)$.

Direction $\Rightarrow$: Assume that D is the LCS of $C_1, \ldots, C_n$. It follows from Lemma 1 in [JLW20] that $\mathcal{A}_{C_i} \models D(a_{C_i})$ for all $i = 1, \ldots, n$. This means that each a_{C_i} is an answer to both q_A and $q_{A \sqcap D}$ over $\mathcal{A}$. Moreover, since A is a fresh concept name we can conclude that $q_{A \sqcap D}$ is a neutral specialization of q_A. Lastly, from the fact that D is the LCS it cannot be that there is another $q' \preceq q_{A \sqcap D}$ such that each a_{C_i} is an answer to q' over $\mathcal{A}$. Therefore we conclude that $q_{A \sqcap D}$ is a most specific neutral specialization of q_A in $\langle Q, \preceq \rangle$.

Direction $\Leftarrow$: Assume that $q_{A \sqcap D}$ is a most specific neutral specialization of q_A. Clearly $q_{A \sqcap C_i} \preceq q_{A \sqcap D}$ and using again Lemma 1 in [JLW20] we obtain that $\models C_i \sqsubseteq D$, for all $i = 1, \ldots, n$. From

the construction of $\mathcal{A}$ and $\langle Q, \preceq \rangle$, and the fact that for each q_B such that $q_B \preceq q_{A \sqcap D}$ we have that $q_B \prec q_A$ (which means that at least one of a_{C_i} is not an answer to any further specialization q_B) we conclude that D must be the LCS of $C_1, \ldots, C_n$.

□

For $\mathcal{Q}_\mathcal{O}$-MGN-GEN the complexity remains open even for tree-shaped query spaces. Nevertheless, based on the previous result, we can conclude that navigating within a query space is not feasible in polynomial time in general. Moreover, there are several challenges to be addressed for realizing such approach in practice: (i) automatic generation of meaningful query spaces, and (ii) techniques for efficient answer retrieval and navigation within the query space. We propose concrete solutions to such challenges in the following sections.

4.2 Generating Meaningful Query Spaces

We now introduce the notion of *query template*, which is the starting point to build exploratory query spaces. Syntactically, they are CQs whose atoms may be marked if we want to consider more specialized ($\cdot^{\mathsf{s}}$) or generalized ($\cdot^{\mathsf{g}}$) versions of them.

Definition 4.5 (Query Templates)**.** *A* query template $\Psi[\vec{x}]$ *is an expression* $\exists \vec{y}.\tau_1 \wedge \cdots \wedge \tau_n$*, where each* τ_i *is an atom of the form:*

$$A(x) \mid r(x,y) \mid x = a \mid A^{\mathsf{s}}(x) \mid A^{\mathsf{g}}(x) \mid r^{\mathsf{s}}(x,y) \mid r^{\mathsf{g}}(x,y) \mid x = a^{\mathsf{s}} \mid x = a^{\mathsf{g}},$$

where $x, y \in \vec{x} \cup \vec{y}$*,* $\vec{x}$ *are* the answer variables*,* $A \in \mathbf{C}$ *or* $\top$*,* $r \in \mathbf{R}$ *and* $a \in \mathbf{I}$*.*

Example 4.1. *We can succintly describe CQs that retrieve (possibly special types of) MCI in* Vienna*, or more general locations that contain* Vienna*, using template:*

$$\Psi[x] = \exists z\ \mathrm{MCI}^{\mathsf{s}}(x) \wedge \mathrm{hasLocation}^{\mathsf{g}}(x,z) \wedge z = \mathsf{Vienna}^{\mathsf{g}}.$$

Given some query template Ψ as starting point, we will use *reformulation rules* to build a set of related CQs. To guarantee that these queries can be ordered according to their specificity (or generality), these rules will be guided by a set of *reformulation axioms*. In general, $\mathcal{O}$ can be used to guide the rules application. However, to take also the data into account, we allow other sets $\mathcal{K}$ of reformulation axioms as well. For example, $\mathcal{K}$ can be a subset of $\mathcal{O}$ that the user considers *relevant*. It may contain assertions from $\mathcal{A}$, or other axioms implied by $\mathcal{O}$ and $\mathcal{A}$, enabling *data-driven* query reformulations in the style of [AIOS19].

The *reformulation rules* are presented in Table 4.1. We use '_' as a placeholder for a fresh variable, symbol $\tilde{y}$ denotes the condition $(y \notin \mathit{vars}(\Psi') \cup \vec{x}) \vee (y = \_)$ and $\hat{y}$ denotes that y is not an answer variable in the template, and r^{x} stands for any of r^{s}, r^{g}, or r.

Definition 4.6 (*DL-Lite*$_\mathcal{A}^{++(\textit{rec-safe})}$ query space)**.** *A* reformulation axiom *is either an ABox assertion or an ontology axiom of DL-Lite*$_\mathcal{A}^{++(\textit{rec-safe})}$ *form.*

(R1)	If $B \sqsubseteq A \in \mathcal{K}$	then: $\Psi' \wedge A^{\mathsf{s}}(x) \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge B^{\mathsf{s}}(x)$	$\Psi' \wedge B^{\mathsf{g}}(x) \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge A^{\mathsf{g}}(x)$
(R2)	If $A \sqsubseteq \exists r \in \mathcal{K}$	then: $\Psi' \wedge r^{\mathsf{s}}(x, \tilde{y}) \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge A^{\mathsf{s}}(x)$	$\Psi' \wedge A^{\mathsf{g}}(x) \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge r^{\mathsf{g}}(x, \_)$
(R3)	If $\exists r \sqsubseteq A \in \mathcal{K}$	then: $\Psi' \wedge A^{\mathsf{s}}(x) \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge r^{\mathsf{s}}(x, \_)$	$\Psi' \wedge r^{\mathsf{g}}(x, \tilde{y}) \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge A^{\mathsf{g}}(x)$
(R4)	If $r \sqsubseteq p \in \mathcal{K}$	then: $\Psi' \wedge p^{\mathsf{s}}(x, y) \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge r^{\mathsf{s}}(x, y)$	$\Psi' \wedge r^{\mathsf{g}}(x, y) \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge p^{\mathsf{g}}(x, y)$
(R5)	If $s^- \sqsubseteq p \in \mathcal{K}$	then: $\Psi' \wedge p^{\mathsf{s}}(x, y) \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge s^{\mathsf{s}}(y, x)$	$\Psi' \wedge s^{\mathsf{g}}(x, y) \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge p^{\mathsf{g}}(y, x)$
(R6)	If $\{B \sqsubseteq \exists s.A,$ $r \circ s \sqsubseteq r\} \subseteq \mathcal{K}$	then: $\Psi' \wedge r^{\mathsf{x}}(x, \hat{y}) \wedge A^{\mathsf{s}}(\hat{y}) \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge r^{\mathsf{x}}(x, y) \wedge B^{\mathsf{s}}(y)$ $\Psi' \wedge r^{\mathsf{x}}(x, \hat{y}) \wedge B^{\mathsf{g}}(\hat{y}) \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge r^{\mathsf{x}}(x, y) \wedge A^{\mathsf{g}}(y)$	
(R7)	If $A(a) \in \mathcal{K}$	then: $\Psi' \wedge A^{\mathsf{s}}(x) \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge x = a^{\mathsf{s}}$	$\Psi' \wedge x = a^{\mathsf{g}} \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge A^{\mathsf{g}}(x)$
(R8)	If $r(a, b) \in \mathcal{K}$	then: $\Psi' \wedge r^{\mathsf{s}}(x, \tilde{y}) \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge x = a^{\mathsf{s}}$	$\Psi' \wedge x = a^{\mathsf{g}} \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge r^{\mathsf{g}}(x, \_)$
		$\Psi' \wedge r^{\mathsf{s}}(\tilde{x}, y) \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge y = b^{\mathsf{s}}$	$\Psi' \wedge x = b^{\mathsf{g}} \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge r^{\mathsf{g}}(\_, x)$
(R9)	If $\{s(a, b),$ $r \circ s \sqsubseteq r\} \subseteq \mathcal{K}$	then: $\Psi' \wedge r^{\mathsf{s}}(x, \hat{y}) \wedge \hat{y} = b^{\mathsf{s}} \overset{\mathsf{s}}{\rightsquigarrow} \Psi' \wedge r^{\mathsf{s}}(x, y) \wedge y = a^{\mathsf{s}}$ $\Psi' \wedge r^{\mathsf{g}}(x, \hat{y}) \wedge \hat{y} = a^{\mathsf{g}} \overset{\mathsf{g}}{\rightsquigarrow} \Psi' \wedge r^{\mathsf{g}}(x, y) \wedge y = b^{\mathsf{g}}$	

Table 4.1: Rules to derive CQs from query template $\Psi[\vec{x}]$.

Given a template Ψ *and a set of reformulation axioms* $\mathcal{K}$*, we say that* Ψ' is derivable from Ψ using $\mathcal{K}$ *if there is a sequence of reformulations* $\Psi \overset{\mathsf{x}}{\rightsquigarrow} \Psi_1 \ldots \overset{\mathsf{x}}{\rightsquigarrow} \Psi_n = \Psi'$*, with* $n \geq 0$ *and* $\mathsf{x} \in \{\mathsf{s}, \mathsf{g}\}$*, using the rules in Table 4.1. We will use* $\Psi \overset{\mathsf{x}}{\rightsquigarrow}_{\mathcal{K}} \Psi'$*, to indicate that* Ψ' *is derivable from* Ψ *w.r.t.* $\mathcal{K}$*, and we will write* $q_\Psi(\vec{x})$ *to denote the CQ obtained from* $\Psi[\vec{x}]$ *by removing all the* s *and* g *labels. Then we define:*

- $Q^{\mathcal{K}}_\Psi$ *is the set of all CQs* $\{q_{\Psi'}(\vec{x}) \mid \Psi[\vec{x}] \rightsquigarrow_{\mathcal{K}} \Psi'[\vec{x}]\}$.
- *We write* $q_{\Psi_1} \preceq_{\mathcal{K}} q_{\Psi_2}$ *whenever* $\Psi_2 \overset{\mathsf{s}}{\rightsquigarrow}_{\mathcal{K}} \Psi_1$ *or* $\Psi_1 \overset{\mathsf{g}}{\rightsquigarrow}_{\mathcal{K}} \Psi_2$.

Note that, following [AIOS19], we allow CRIs in our reformulation axioms to capture query navigation similar to roll-up and drill-down operations in OLAP. For example, the *location dimension* has different granularity levels connected by a *part-of* rather than by a *subclass-of* relation. Using axioms like City $\sqsubseteq \exists$partOf.Country and the CRI hasLocation$\circ$partOf $\sqsubseteq$ hasLocation, we can generalize a query asking for MCIs in a city to one that asks for MCIs in a country instead. The roll-up and its specializing counterpart drill-down are captured by rules **R6** and **R9**, respectively.

To understand the process of creating the query space, let us discuss the rules in Table 4.1 in more detail. Atom $A^{\mathsf{s}}(x)$ can be specialized as: $B(x)$ using (**R1**) or $r(x, \_)$ using (**R3**), or as $A(a)$, with a a known instance of concept A using (**R7**). Atom $p^{\mathsf{s}}(x, y)$ is specialized as $r(x, y)$ with r a role more specific than r using (**R4**) or (**R5**), or $B(x)$ (or $B(y)$), with B a concept that is more specific than $\exists r$ (or $\exists r^-$) with rule (**R2**). The rules work similarly for the generalizing case. The process can be stopped by simply dropping the special markers and obtaining a CQ. The derivation process also establishes the exploratory order in the query space.

Example 4.2. *Let us consider again the template* $\Psi[x]$ *from Example 4.1, and the set of reformulation axioms:* $\mathcal{K}_1 = \{\text{hasLocation}\circ\text{partOf} \sqsubseteq \text{hasLocation}, \text{partOf}(\text{Vienna}, \text{Austria})\}$.

*Rule (**R9**) is applicable for* Ψ *and* $\mathcal{K}_1$ *(since* z *is not an answer variable):*

$$\text{MCI}^{\text{s}}(x) \wedge \text{hasLocation}^{\text{g}}(x, z) \wedge z = \text{Vienna}^{\text{g}} \overset{\text{g}}{\rightsquigarrow} \text{MCI}^{\text{s}}(x) \wedge \text{hasLocation}^{\text{g}}(x, z) \wedge z = \text{Austria}^{\text{g}}.$$

The query space generated by template Ψ *and* $\mathcal{K}_1$ *is denoted by* $Q_\Psi^{\mathcal{K}_1}$*; by removing the special markers, we get that the following query is part of it:*

$$q_1(x) \leftarrow \text{MCI}(x) \wedge \text{hasLocation}(x, z) \wedge z = \text{Austria}.$$

Thus, $q_\Psi \preceq_{\mathcal{K}_1} q_1$ *in* $Q_\Psi^{\mathcal{K}_1}$*. The semantics of* Ψ *change if we consider, for example, the set of reformulation axioms* $\mathcal{K}_2 = \{\text{Fire} \sqsubseteq \text{MCI}\}$*, based on which by applying rule (**R1**) we can obtain the query* $q_2(x) \leftarrow \text{Fire}(x) \wedge \text{hasLoccation}(x, z) \wedge z = \text{Vienna}$ *thus allowing flexibility and control in choosing what it means to specialize or generalize template atoms.*

The reformulation axioms might be derived from the existing knowledge base, but also they might come from an external source and in that case we must ensure that they are compatible with the existing knowledge.

Let $\mathcal{O}$ be written in a language that enjoys the canonical model property. Then we call a set $\mathcal{K}$ of reformulation axioms *compatible with* $(\mathcal{O}, \mathcal{A})$ if $\mathcal{I}_{\mathcal{O},\mathcal{A}} \vDash \mathcal{K}$, where $\mathcal{I}_{\mathcal{O},\mathcal{A}}$ denotes a canonical model. Compatibility of $\mathcal{K}$ ensures that we obtain an exploratory query space.

Lemma 4.2. *Let* Ψ *be a template,* $\mathcal{K}$ *a set of reformulation axioms, and* $\mathcal{O}$ *an ontology in a language that enjoys the canonical model property. If* $\mathcal{A}$ *is such that* $\mathcal{K}$ *is compatible with* $(\mathcal{O}, \mathcal{A})$*, then each query space* $\langle Q_\Psi^{\mathcal{K}}, \preceq_{\mathcal{K}}\rangle$ *is exploratory for* $\mathcal{A}$*.*

Proof (Sketch). Since the symbol '_' stands for a variable which does not occur free nor in a join, and since only such variables can be freshly introduced during derivation, we easily obtain that $Q_\Psi^{\mathcal{K}}$ is finite. Since $\mathcal{K}$ is compatible with $(\mathcal{O}, \mathcal{A})$ and since each query generation rule applies only axioms and assertions from $\mathcal{K}$ in such a way that containment of answers is preserved in both generalizing and specializing case, we directly obtain that for any given queries $q_1, q_2 \in Q_\Psi^{\mathcal{K}}$ such that $q_1 \preceq_{\mathcal{K}} q_2$ we have that $q_1 \subseteq_{(\mathcal{O},\mathcal{A})} q_2$. □

Example 4.3. *Let* $\mathcal{O} = \{\text{Fire} \sqsubseteq \text{NatureCausedMCI}, \text{NatureCausedMCI} \sqsubseteq \text{MCI}\}$ *and*

$$\mathcal{A} = \{\text{Fire}(\text{e}_1),\ \text{hasLocation}(\text{e}_1, \text{Vienna}), \text{partOf}(\text{Vienna},\ \text{Austria}),\\ \text{hasLocation}(\text{e}_1, \text{Austria})\}.$$

$\mathcal{K} = \{\text{hasLocation}\circ\text{partOf} \sqsubseteq \text{hasLocation}\}$ *is compatible with* $(\mathcal{O}, \mathcal{A})$ *since in* $\mathcal{A}$ *the CRI is already satisfied. Similarly, it is also easy to check that* $\mathcal{K}_1$ *and* $\mathcal{K}_2$ *from the previous example, are compatible with* $(\mathcal{O}, \mathcal{A})$*, so both* $(Q_\Psi^{\mathcal{K}_1}, \preceq_{\mathcal{K}_1})$ *and* $(Q_\Psi^{\mathcal{K}_2}, \preceq_{\mathcal{K}_2})$ *are exploratory for* $\mathcal{A}$*.*

If $\mathcal{K}$ is not a subset of $\mathcal{O} \cup \mathcal{A}$, in order to test compatibility, property that is important for exploration purposes, we need to check if each axiom holds in the canonical model. When $\mathcal{O}$ is in $\mathcal{DL\text{-}Lite}_{\mathcal{A}}^{++(\textit{rec-safe})}$ such test can be done in PTime (cf. Theorem 3.6).

4.3 Generating Query Spaces with Datalog

In this section, we describe a way to realize our exploratory framework using Datalog. The general idea is to create a Datalog program that encodes the query generation and answering process, based on the query template and the derivation rules (in Section 4.3.2), and the exploration process by identifying the most relevant generalizations and specializations for each query in the space (in Section 4.3.4).

Before proceeding with the Datalog encoding, we recall the syntax and semantics of *Datalog programs with stratified negation* [ABW88].

4.3.1 Datalog with Equality and Stratified Negation

The vocabulary of a Datalog program is a collection of *object contants* and *relation contants*. Object constants include, but are not limited to, constant symbols in $\mathbf{K}$, therefore we use the same notation to denote Datalog constants (e.g., Vienna, Austria, etc.). Relation constants behave similarly to symbols in $\mathbf{C} \cup \mathbf{R} \cup \mathbf{F}$ but they consists of arbitrary aritities. In order to distinguish between DL predicate names and Datalog relations, we denote a relation constant as an alphanumeric string in TypewriterFont with possible subscript e.g., $\texttt{capitalOf}$, $\texttt{hasConnection}_{\texttt{metro}}$, and so on.

Similarly to any other query language, a Datalog program allows the use of *variables* to point to objects without specifically naming them. To distinguish Datalog variables among the variables present in CQs, we write Datalog variables as alphanumeric strings with possible subscript that begin with capital letter, e.g., X, Y, U, U_1 and so on. A Datalog *term* is either a Datalog constant or a variable.

An atom β is either (i) $\mathrm{p}(\vec{T})$ with $\vec{T}$ a tuple of terms of the same arity as p, or (ii) $T = T'$ for terms T, T'. A Datalog rule ρ has the form:

$$\mathrm{p}(\vec{V}) \leftarrow \beta_1, \ldots, \beta_k, \neg\beta_{k+1}, \ldots, \neg\beta_m$$

where $m \geq k$, $\mathrm{p}(\vec{V})$ is called the *head of* ρ and denoted by $head(\rho)$, while the set of atoms $\{\beta_1, \ldots, \beta_m\}$ is the *body of* ρ and $body^+(\rho)$ denotes the positive atoms while $body^-(\rho)$ the negative ones. All variables in $head(\rho)$ and in $body^-(\rho)$ must also occur in $body^+(\rho)$. A Datalog *fact* is a variable-free rule with empty body which may be written as $\mathrm{p}(\vec{c})$.

A *Datalog program* Π is a set of Datalog rules. Π is *stratified* if it can be partitioned as $\Pi_1, \ldots, \Pi_n$ such that for each p, all rules with p in the head are in the same partition Π_i, and for all atoms in the bodies of those rules, the definitions of such predicates are in some Π_j, with $j \leq i$ if the atom is positive, or $j < i$ for negative atoms.

A set of facts I *satisfies a rule* ρ if there exists a mapping $h : vars(\rho) \mapsto cst(I)$ such that whenever $h(body^+(\rho)) \subseteq I$ and $h(body^-(\rho)) \nsubseteq I$, then $h\,(head\,(\rho)) \subseteq I$. A *model of* Π is any set of facts that satisfies each $\rho \in \Pi$. For a stratified program Π and finite set of facts D, $\Pi(D)$ denotes the minimal model of Π (which is unique and always exists) consisting of $\bigcup_{i=0}^{n} I_i$, where $I_0 = D$ and for $1 \leq i \leq n$, each I_i minimally extends I_{i-1} such that I_i is a model of Π_i.

4.3.2 Datalog Encoding of Query Spaces

We now build a Datalog program that derives and evaluates all the queries in a space triggered by a template and a set of reformulation axioms. Thus, in this section we fix a template $\Psi[x_1, \ldots, x_k] = \exists \vec{y}.\tau_1 \wedge \cdots \wedge \tau_n$ as well as a set of reformulation axioms $\mathcal{K}$. We also assume that each non-answer variable that occurs exactly once in Ψ is substituted by the symbol _. Further, for each variable $z \in \mathit{vars}(\Psi)$, we consider an associated Datalog variable V_z.

In our encoding, we reason over concepts, relations and constants that appear in Ψ and $\mathcal{K}$, thus for simplicity we use A as the Datalog constant denoting concept $A \in \mathbf{C}$ and similarly p as the Datalog constant denoting relation $p \in \mathbf{R} \cup \mathbf{F}$. Moreover, we also consider a Datalog constant v_z and c_τ for each variable z and each atom τ in Ψ, respectively. A special constant null acts as placeholder for _ and as a "filler" for irrelevant positions in our designated Datalog predicates.

The initial step of our Datalog encoding consists in translating the given Ψ and $\mathcal{K}$ into a set of Datalog facts. Such translation is presented in the first part of Table 4.2. Firstly, we encode all relevant information in $\mathcal{K}$ into a set of facts denoted by $\mathbf{D}_\mathcal{K}$. For that, we map ($\mapsto$) each concept A in the signature of Ψ and $\mathcal{K}$ to a fact $\mathtt{conc}(\mathsf{A})$. Similarly, for each relation p occuring in $\mathcal{K}$ or Ψ we consider a fact $\mathtt{role}(\mathsf{p})$. Then, each inclusion axiom in $\mathcal{K}$ is encoded using Datalog relations $\mathtt{cISA}$ and $\mathtt{rISA}$, and each inverse axiom is encoded using $\mathtt{rISAinv}$. For each constant symbol c we consider a fact $\mathtt{const}(\mathsf{c})$. To encode unary $A(c)$, respectively binary $r(a, b)$, assertions in $\mathcal{K}$ we use $\mathtt{uAssrt}(\mathsf{A}, \mathsf{c})$, respectively $\mathtt{bAssrt}(\mathsf{r}, \mathsf{a}, \mathsf{b})$. For each $r \circ s \sqsubseteq r$ and $s(a, b) \in \mathcal{K}$, respectively $A \sqsubseteq \exists s.A' \in \mathcal{K}$, we have facts $\mathtt{rup}(\mathsf{a}, \mathsf{b}, \mathsf{r})$ and $\mathtt{ddn}(\mathsf{b}, \mathsf{a}, \mathsf{r})$, respectively $\mathtt{rup}(\mathsf{A}, \mathsf{A}', \mathsf{r})$ and $\mathtt{ddn}(\mathsf{A}', \mathsf{A}, \mathsf{r})$. Next, we encode all the relevant information regarding the structure of Ψ into a set of facts $\mathbf{D}_\Psi$. For each non-answer variable z in Ψ we consider a fact $\mathtt{nAns}(\mathsf{c}_z)$ and proceed as before if new concepts, relations or constants appear. Then, for each atom τ if it is a specializing atom $A^{\mathsf{s}}(x)$, $r^{\mathsf{s}}(x, y)$, we consider a fact $\mathtt{atm}_{\mathsf{s}}(\mathsf{c}_\tau, \mathsf{A}, \mathsf{c}_x, \mathsf{null})$, $\mathtt{atm}_{\mathsf{s}}(\mathsf{c}_\tau, \mathsf{r}, \mathsf{c}_x, \mathsf{c}_y)$, respectively. If it is a generalizing atom the encoding is similar but using the predicate $\mathtt{atm}_{\mathsf{g}}$, while for τ a query atom we use predicate $\mathtt{refAt}$.

The second step in our encoding is to simulate the query generation process captured by the rules in Table 4.1. Using the relevant information captured by $\mathbf{D}_\Psi \cup \mathbf{D}_\mathcal{K}$, the Datalog set of rules Π_{rules} are in one-to-one correspondance to the query generation rules and are explicitly defined in the second part of the Table 4.2. These rules derive an $\mathtt{atm}_{\mathsf{s}}$ atom for each reformulation of a specializing atom, and an $\mathtt{atm}_{\mathsf{g}}$ atom for each reformulation of a generalizing atom in Ψ.

The final group of rules Π_Q in Table 4.2 build up the set of OMQs in the space. In this group of Datalog rules, we first collect all reformulated atoms using $\mathtt{refAt}$. Next, all possible combinations of valid query formulations in the space are created using relation $\mathtt{refAtms}$ which has the same arity as the number of atoms in the template. The last part of the encoding is to define the views needed to evaluate each query in the space. For that we use for each template atom τ a predicate $\mathtt{qAtm}_\tau$ which intuitively captures all possible matches for any query atom that is derivable from τ. We collect all the matches of all the queries in the space using view $\mathtt{query}_\Psi(\vec{U}, V_{x_1}, \ldots, V_{x_k})$ where tuple $\vec{U}$ has arrity n–the number of atoms in Ψ, and each variable in it is mapped to Datalog constants denoting symbols from the signature of $\mathcal{K}$ and Ψ. The tuple $(V_{x_1}, \ldots, V_{x_n})$

corresponds to the tuple of answer variables $(x_1, \ldots, x_n)$ in Ψ and each tuple $\vec{V}_i$ in the body of the view corresponds to the answer variables in each template atom τ_i, for $1 \leq i \leq n$.

We can now describe how the query space is captured. We denote by $tr(q)$ the translation of a given CQ $q(\vec{x})$ into the signature of our Datalog program, obtained by applying to each atom in q the function tr such as:

- $tr(x=a) = V_x = a$,
- $tr(A(x)) = \texttt{uAtm}(\mathsf{A}, V_x)$,
- $tr(r(x,y)) = \texttt{bAtm}(\mathsf{r}, V_x, V_y)$,
- replace each non-join and non-answer variable in $tr(q)$ by null.

Definition 4.7 (Datalog encoding)**.** *We let* $\Pi_\Psi = \Pi_{rules} \cup \Pi_Q$ *and* $D_{\Psi,\mathcal{K}} = D_\Psi \cup D_\mathcal{K}$*, and call the pair* $\langle \Pi_\Psi, D_{\Psi,\mathcal{K}} \rangle$ *the* Datalog encoding of $(\Psi, \mathcal{K})$.

Let $\vec{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_n)$ *be a tuple of Datalog constants from* $\mathbf{C} \cup \mathbf{R} \cup \mathbf{F} \cup \mathbf{K}$*. The* unfolding of Π_Ψ for $\vec{c}$ *is the rule obtained from*

$$\texttt{query}_\Psi(\vec{c}, V_{x_1}, \ldots, V_{x_k}) \leftarrow \texttt{refAtms}(\vec{c}), \texttt{qAtm}_{\tau_1}(\mathsf{c}_1, \vec{V}_1), \ldots, \texttt{qAtm}_{\tau_n}(\mathsf{c}_n, \vec{V}_n)$$

by choosing some rule $\rho \in \Pi_Q$ *for each* $\texttt{qAtm}_{\tau_i}(\mathsf{c}_i, \vec{V}_i)$*, and a substitution* σ *such that*

- $head(\rho)\sigma = \texttt{qAtm}_{\tau_i}(\mathsf{c}_i, \vec{V}_i)$*, and*
- *exactly one of* $\texttt{conc}(\mathsf{c}_i)$, $\texttt{role}(\mathsf{c}_i)$ *or* $\texttt{const}(\mathsf{c}_i)$ *is contained in* $D_{\Psi,\mathcal{K}}$

and then: (i) replacing $\texttt{qAtm}_{\tau_i}(\mathsf{c}_i, \vec{V}_i)$ *with* $body(\rho)\sigma$*, and (ii) removing each atom from* $body(\texttt{query}_\Psi(\vec{c}, \vec{x}))$ *that is contained in* $\Pi_\Psi(D_{\Psi,\mathcal{K}})$.

If the body of the unfolding Π_Ψ *for* $\vec{c}$ *is* $tr(q)$ *for some CQ* q*, we call* $\vec{c}$ *the* Π_Ψ-encoding *of* q.

The idea behind the unfolding is simple. If q has a Π_Ψ-encoding, then it is derivable from Ψ w.r.t $\mathcal{K}$. Conversely, for each derivable q we can find a Π_Ψ-encoding.

Example 4.4. *Let* $\vec{c} = (\mathsf{MCI}, \mathsf{hasLocation}, \mathsf{Vienna})$*. The unfolding of* Π_Ψ *for* $\vec{c}$ *is*

$\texttt{query}_\Psi(\vec{c}, V_x) \leftarrow \texttt{uAtm}(\mathsf{MCI}, V_x), \texttt{bAtm}(\mathsf{hasLocation}, V_x, V_z), V_z = \mathsf{Vienna}.$

The body matches $tr(q_\Psi)$ *from Example 4.1, so* $\vec{c}$ *is a* Π_Ψ*-encoding for* q_Ψ.

We proceed next with showing that the Datalog encoding is correct.

Lemma 4.3. *A CQ* q *is derivable from* Ψ *using* $\mathcal{K}$ *iff there exists a* Π_Ψ*-encoding of* q.

Proof. The lemma follows from the following claims. Let $(\Psi, \mathcal{K})$ be an arbitrary pair consisting of a template $\Psi[\vec{x}] = \exists \vec{y} \tau_1 \wedge \cdots \wedge \tau_n$ and a set of reformulation axioms $\mathcal{K}$.

Claim 6. *For any CQ* q *such that* q *is derivable from* Ψ *using* $\mathcal{K}$ *there exists a* Π_Ψ*-encoding of* q.

$\mathcal{D}_{\mathcal{K}}$:	$A \mapsto \texttt{conc}(A)$,	$A \sqsubseteq A' \mapsto \texttt{cISA}(A, A')$	$A(A) \mapsto \texttt{uAssrt}(\mathsf{A}, a)$
	$r \mapsto \texttt{role}(r)$,	$\exists r \sqsubseteq A \mapsto \texttt{cISA}(r, A)$	$r(a,b) \mapsto \texttt{bAssrt}(\mathsf{r}, a, b)$
	$a \mapsto \texttt{const}(a)$,	$A \sqsubseteq \exists r \mapsto \texttt{cISA}(A, r)$	$\{s(a,b), r \circ s \sqsubseteq r\} \mapsto \texttt{ddn}(b, a, \mathsf{r}), \texttt{rup}(a, b, \mathsf{r})$
		$s \sqsubseteq r \mapsto \texttt{rISA}(\mathsf{s}, \mathsf{r})$	$\{A \sqsubseteq \exists s.A', r \circ s \sqsubseteq r\} \mapsto \texttt{ddn}(\mathsf{A}', \mathsf{A}, \mathsf{r})$
		$p^- \sqsubseteq r \mapsto \texttt{rISAinv}(\mathsf{p}, \mathsf{r})$	$\{A \sqsubseteq \exists s.A', r \circ s \sqsubseteq r\} \mapsto \texttt{rup}(\mathsf{A}, \mathsf{A}', \mathsf{r})$
$\mathcal{D}_{\Psi}$:	$x \notin \vec{x} \mapsto \texttt{nAns}(\mathsf{v}_x)$	$A^{\mathsf{x}}(x) \mapsto \texttt{atm}_{\mathsf{s}}(\mathsf{c}_\tau, \mathsf{A}, \mathsf{v}_x, \texttt{null})$	$A(x) \mapsto \texttt{refAt}(\mathsf{c}_\tau, \mathsf{A}, \mathsf{v}_x, \texttt{null})$
	$A \mapsto \texttt{conc}(A)$	$r^{\mathsf{x}}(x,y) \mapsto \texttt{atm}_{\mathsf{s}}(\mathsf{c}_\tau, \mathsf{r}, \mathsf{v}_x, \mathsf{v}_y)$	$r(x,y) \mapsto \texttt{refAt}(\mathsf{c}_\tau, \mathsf{r}, \mathsf{v}_x, \mathsf{v}_y)$
	$r \mapsto \texttt{role}(r)$	$x = a^{\mathsf{x}} \mapsto \texttt{atm}_{\mathsf{s}}(\mathsf{c}_\tau, a, \mathsf{v}_x, \texttt{null})$	$x = a \mapsto \texttt{refAt}(\mathsf{c}_\tau, a, \mathsf{v}_x, \texttt{null})$
	$a \mapsto \texttt{const}(a)$	with $\mathsf{x} \in \{\mathsf{s}, \mathsf{g}\}$	

Encoding of the reformulation rules Π_{rules}

(R1–3)	$\texttt{atm}_{\mathsf{s}}(V_\tau, U, Y, Z)$	$\leftarrow$	$\texttt{atm}_{\mathsf{s}}(V_\tau, U', Y, Z), \texttt{cISA}(U, U'), \texttt{conc}(U').$
	$\texttt{atm}_{\mathsf{g}}(V_\tau, U', Y, Z)$	$\leftarrow$	$\texttt{atm}_{\mathsf{g}}(V_\tau, U, Y, Z), \texttt{cISA}(U, U'), \texttt{conc}(U).$
	$\texttt{atm}_{\mathsf{s}}(V_\tau, U, Y, \texttt{null})$	$\leftarrow$	$\texttt{atm}_{\mathsf{s}}(V_\tau, U', Y, \texttt{null}), \texttt{cISA}(U, U'), \texttt{conc}(U), \texttt{role}(U').$
	$\texttt{atm}_{\mathsf{g}}(V_\tau, U', Y, \texttt{null})$	$\leftarrow$	$\texttt{atm}_{\mathsf{g}}(V_\tau, U, Y, \texttt{null}), \texttt{cISA}(U, U'), \texttt{conc}(U'), \texttt{role}(U).$
(R4), **(R5)**	$\texttt{atm}_{\mathsf{s}}(V_\tau, U, Y, Z)$	$\leftarrow$	$\texttt{atm}_{\mathsf{s}}(V_\tau, U', Y, Z), \texttt{rISA}(U, U').$
	$\texttt{atm}_{\mathsf{g}}(V_\tau, U', Y, Z)$	$\leftarrow$	$\texttt{atm}_{\mathsf{g}}(V_\tau, U, Y, Z), \texttt{rISA}(U, U').$
	$\texttt{atm}_{\mathsf{s}}(V_\tau, U, Y, Z)$	$\leftarrow$	$\texttt{atm}_{\mathsf{s}}(V_\tau, U', Z, Y), \texttt{rISAinv}(U, U').$
	$\texttt{atm}_{\mathsf{g}}(V_\tau, U', Y, Z)$	$\leftarrow$	$\texttt{atm}_{\mathsf{g}}(V_\tau, U, Z, Y), \texttt{rISAinv}(U, U').$
(R6), **(R9)**	$\texttt{atm}_{\mathsf{s}}(V_\tau, U', W, \texttt{null})$	$\leftarrow$	$\texttt{ddn}(U, U', Z), \texttt{atm}_{\mathsf{s}}(V_\tau, U, W, \texttt{null}), W \neq \texttt{null},$ $\texttt{refAt}(V_{\tau'}, Z, Y, W), V_\tau \neq V_{\tau'}, \texttt{nAns}(W).$
	$\texttt{atm}_{\mathsf{g}}(V_\tau, U', W, \texttt{null})$	$\leftarrow$	$\texttt{rup}(U, U', Z), \texttt{atm}_{\mathsf{g}}(V_\tau, U, W, \texttt{null}), W \neq \texttt{null},$ $\texttt{refAt}(V'_\tau, Z, Y, W), V_\tau \neq V_{\tau'}, \texttt{nAns}(W).$
(R7), **(R8)**	$\texttt{atm}_{\mathsf{s}}(V_\tau, V, Y, \texttt{null})$	$\leftarrow$	$\texttt{atm}_{\mathsf{s}}(V_\tau, U, Y, \texttt{null}), \texttt{uAssrt}(U, V).$
	$\texttt{atm}_{\mathsf{g}}(V_\tau, U, Y, \texttt{null})$	$\leftarrow$	$\texttt{atm}_{\mathsf{g}}(V_\tau, V, Y, \texttt{null}), \texttt{uAssrt}(U, V).$
	$\texttt{atm}_{\mathsf{g}}(V_\tau, U, \texttt{null}, Y)$	$\leftarrow$	$\texttt{atm}_{\mathsf{g}}(V_\tau, V', Y, \texttt{null}), \texttt{bAssrt}(U, V, V').$
	$\texttt{atm}_{\mathsf{g}}(V_\tau, U, Y, \texttt{null})$	$\leftarrow$	$\texttt{atm}_{\mathsf{g}}(V_\tau, V, Y, \texttt{null}), \texttt{bAssrt}(U, V, V').$
	$\texttt{atm}_{\mathsf{s}}(V_\tau, V, Y, \texttt{null})$	$\leftarrow$	$\texttt{atm}_{\mathsf{s}}(V_\tau, U, Y, \texttt{null}), \texttt{bAssrt}(U, V, V'), Y \neq \texttt{null}.$
	$\texttt{atm}_{\mathsf{s}}(V_\tau, V', \texttt{null}, Z)$	$\leftarrow$	$\texttt{atm}_{\mathsf{s}}(V_\tau, U, \texttt{null}, Z), \texttt{bAssrt}(U, V, V'), Z \neq \texttt{null}.$

Query space encoding Π_Q:

$$\texttt{refAt}(V_\tau, U, X, Y) \leftarrow \texttt{atm}_{\mathsf{s}}(V_\tau, U, X, Y).$$
$$\texttt{refAt}(V_\tau, U, X, Y) \leftarrow \texttt{atm}_{\mathsf{g}}(V_\tau, U, X, Y).$$
$$\texttt{refAtms}(U_1, \ldots, U_n) \leftarrow \texttt{refAt}(\mathsf{c}_{\tau_1}, U_1, \vec{V}_1), \ldots, \texttt{refAt}(\mathsf{c}_{\tau_n}, U_n, \vec{V}_n).$$
$$\texttt{query}_\Psi(\vec{U}, V_{x_1}, \ldots, V_{x_k}) \leftarrow \texttt{refAtms}(\vec{U}), \texttt{qAtm}_{\tau_1}(U_1, \vec{V}_1), \ldots, \texttt{qAtm}_{\tau_n}(U_n, \vec{V}_n).$$
$$\texttt{qAtm}_\tau(U, X, \texttt{null}) \leftarrow \texttt{uAtm}(U, X), \texttt{conc}(U).$$
$$\texttt{qAtm}_\tau(U, X, \texttt{null}) \leftarrow U = X, \texttt{const}(U).$$
$$\texttt{qAtm}_\tau(U, X, Y) \leftarrow \texttt{bAtm}(U, X, Y), \texttt{role}(U).$$

Table 4.2: Datalog encoding of $\Psi[\vec{x}]$, $\mathcal{K}$ and reformulation rules

.

Proof (Sketch). To prove this claim, we first argue that for each atom in q there exists an associated assertion $\texttt{refAt}(\vec{c}) \in \Pi_{rules} \cup \Pi_Q(D_\Psi \cup D_\mathcal{K})$. This follows almost immediately from the fact that there is some $\texttt{atom}_\texttt{x}$, with $\texttt{x} \in \{\texttt{s}, \texttt{g}\}$, for each specializing or generalizing atom in Ψ, Π_{rules} mimics each of the reformulation rules to derive queries from the templates, and that $\texttt{atom}_\texttt{x}$ implies $\texttt{refAt}$. Since this holds for any query atom in q, we obtain that there exists some tuple $(\texttt{c}_1, \ldots, \texttt{c}_\texttt{n})$ of Datalog constants denoting concept, relations or constants such that $\texttt{c}_\texttt{i}$ denotes the symbol appearing in atom i in q, and that $\texttt{refAx}(\texttt{c}_1, \ldots, \texttt{c}_\texttt{n}) \in \Pi_{rules} \cup \Pi_Q(D_\Psi \cup D_\mathcal{K})$. Lastly, since each of $\texttt{c}_\texttt{i}$ has exactly one type, i.e. it is either concept, role or constant, and that $\texttt{query}_\Psi$ preserves the structure of Ψ, we obtain that $(\texttt{c}_1, \ldots, \texttt{c}_\texttt{n})$ is the Π_Ψ-encoding of q. □

Claim 7. *For any CQ q such that there exists a Π_Ψ-encoding we have that q is derivable from Ψ using $\mathcal{K}$.*

Proof (Sketch). Let q be an arbitrary CQ and let $\vec{c} = (c_1, \ldots, c_n)$ be the Π_Ψ-encoding of q. By definition, we obtain that $\texttt{refAx}(\vec{c}) \in \Pi_\Psi(D_{\Psi,\mathcal{K}})$. Since each fact in $\Pi_\Psi(D_{\Psi,\mathcal{K}})$ is either present in $D_{\Psi,\mathcal{K}}$ or derived by the application of some rule in Π_Ψ, we obtain that there exists some $\texttt{refAt}(\texttt{c}_{\tau_i}, c_i, \texttt{c}_{\vec{y}_i}) \in \Pi_\Psi(D_{\Psi,\mathcal{K}})$, for each $\tau_i \in \Psi$. Following the same reasoning, we obtain a sequence of rules $\alpha_1, \ldots, \alpha_k$ in Π_{rules} which are sequentially applied starting from the facts in $D_{\Psi,\mathcal{K}}$ which account to the encoding of the template and of the reformulation axioms. Since rules in Π_{rules} simulate the derivation rules in Table 4.1, and $D_{\Psi,\mathcal{K}}$ encodes the structure of Ψ and each reformulation axiom, we directly obtain that q, modulo variable renaming, is derivable from Ψ using $\mathcal{K}$. □

From the above claims we obtain that the above lemma holds. □

4.3.3 Evaluation of the Datalog Translation for *DL-Lite*$_\mathcal{R}$ OMQs

We now fix an ABox $\mathcal{A}$ and a *DL-Lite*$_\mathcal{R}$ ontology $\mathcal{O}$, and consider the evaluation of the queries $(q, \mathcal{O})$ in $\mathcal{Q}_\Psi^\mathcal{K}$ over $\mathcal{A}$. The relevant queries are captured by $\langle \Pi_\Psi, D_{\Psi,\mathcal{K}} \rangle$, but we still need an *OMQ answering algorithm* for the DL of $\mathcal{O}$. In the case of *DL-Lite*$_\mathcal{R}$, one could call an external query rewriting engine for the encoded queries. However, we chose to partially complete the data w.r.t. $\mathcal{O}$, and then evaluate the Datalog encoding over the extended dataset.[1]

We say that a CQ template is *rooted* if each variable is either an answer variable or occurs in a join sequence of the form $r_1(x_0, x_1), \ldots, r_k(x_{k-1}, x_k)$, where x_1 is an answer variable and for $1 < i \leq k$, x_i occurs existentially in Ψ. We define the *join length* of a CQ template as the length of its longest join sequence, and let k be the join length of Ψ. In the following, B denotes either a concept name or $\exists r$. Then, we apply the following procedure:

1. For Σ denoting the signature of $(\Psi, \mathcal{K})$, we build a *k-bounded Σ-expansion $\mathcal{A}_\Sigma^{\mathcal{O},k}$ of $\mathcal{A}$ w.r.t. $\mathcal{O}$* constructed by taking for each $A \in \Sigma$ and each $r \in \Sigma$:

[1] Such a procedure is very easy to realize if an *existential rule* engine [CDG+19] is chosen instead of a plain Datalog, as we do in the next section.

- If $\mathcal{O} \vDash B \sqsubseteq A$ and $\mathcal{A} \vDash B(a)$, then $A(a) \in \mathcal{A}_\Sigma^{\mathcal{O},k}$.
- If $\mathcal{O} \vDash B \sqsubseteq \exists r_1 \sqsubseteq \ldots \sqsubseteq \exists r_n$, $\mathcal{O} \vDash \exists r_n^- \sqsubseteq A$ and $\mathcal{A} \vDash B(a)$, then $A(a_{r_1 \ldots r_n}) \in \mathcal{A}_\Sigma^{\mathcal{O},k}$, where $a_{r_1 \ldots r_n}$ is a fresh individual not in $\mathcal{A}$ and $n \leq k$.
- If $\mathcal{O} \vDash s \sqsubseteq r$ (with s possibly inverse) and $\mathcal{A} \vDash s(a,b)$, then $r(a,b) \in \mathcal{A}_\Sigma^{\mathcal{O},k}$.
- If $\mathcal{O} \vDash B \sqsubseteq \exists r_1 \sqsubseteq \ldots \sqsubseteq \exists r_n \sqsubseteq \exists r$ and $\mathcal{A} \vDash B(a)$ then $r(a_{r_1 \ldots r_n}, a_{r_1 \ldots r_n r}) \in \mathcal{A}_\Sigma^{\mathcal{O},k}$, where $k \geq n \geq 0$ and $a_{r_1 \ldots r_n}$ and $a_{r_1 \ldots r_n r}$ are fresh individuals.

2. Lastly, we translate $\mathcal{A}_\Sigma^{\mathcal{O},k}$ into the signature of Π_Ψ, similarly as before. For that, we assume that each individual in $\mathcal{A}_\Sigma^{\mathcal{O},k}$ is a constant in $\mathbf{K}$. We define $D_\mathcal{A} = tr(\mathcal{A}_\Sigma^{\mathcal{O},k})$, where tr translates each assertion in $\mathcal{A}_\Sigma^{\mathcal{O},k}$ as above.

We formulate a minor adaptation of a well known result in the OMQ literature [KLT+10, BOSX13].

Lemma 4.4. *Let $(q, \mathcal{O})$ be a rooted DL-Lite$_\mathcal{R}$ OMQ over the signature Σ. Then, $cert(q, \mathcal{O}, \mathcal{A}) = cert(q, \emptyset, \mathcal{A}_\Sigma^{\mathcal{O},k})$.*

If the template is rooted, then we can answer all $(q, \mathcal{O})$ in the space by evaluating Π_Ψ over $D_{\Psi,\mathcal{K}} \cup D_\mathcal{A}$. From Lemmas 4.3 and 4.4 we obtain:

Theorem 4.1. *Let Ψ be rooted. Let $q \in \mathcal{Q}_\Psi^\mathcal{K}$, and let $\vec{c}_q$ be the Π_Ψ-encoding of q. Then*

$$\vec{a} \in cert(q, \mathcal{O}, \mathcal{A}) \quad \textit{iff} \quad \mathtt{query}_\Psi(\vec{c}_q, \vec{a}) \in \Pi_\Psi(D_{\Psi,\mathcal{K}} \cup D_\mathcal{A}).$$

4.3.4 Datalog Program to Compute Query Reformulations

In order to compute within the same Datalog program which are the MGSSs and MSSGs of each query in the space, we additionally define a set rules Π_{ref}, presented in Table 4.3. We describe the rules for identifying the MGSS for all queries in the space that produce answers. For the generalizing case the intuition behind the defined rules is analogous.

Firstly, we collect all the reformulation axioms using predicate $\mathtt{refAx}$ to account for each one-step reformulation. For example, we have $\mathtt{refAx}(\mathsf{A}, \mathsf{A}')$ whenever $A' \sqsubseteq A \in \mathcal{K}$ or $\{A' \sqsubseteq \exists s.A, r \circ s \sqsubseteq r\} \subseteq \mathcal{K}$. Next, for each query in the space which has answers we identify the strict one-step specializations and generalizations respectively. For example, we have that $\mathtt{strAtm}_\mathsf{s}(\vec{c}_q, \mathsf{c}_{\tau_i}, \mathsf{c}') \in \Pi_{\Psi,\mathcal{K},ref}(D_{\Psi,\mathcal{K}} \cup D_\mathcal{A})$ whenever by reformulating one-step further the atom in q that is derivable from τ_i we obtain c' and via this one-step operation some answer is dropped. To realize this we use the fact that each query in the space and all of its answers are captured using predicate $\mathtt{query}_\Psi$. If some $\mathtt{refAx}(\mathsf{c}, \mathsf{c}')$ is applicable on $\mathtt{query}_\Psi(\mathsf{c}_1, \ldots, \mathsf{c}_{i-1}, \mathsf{c}, \ldots, \mathsf{c}_n, \vec{a})$ and there is no $\mathtt{query}_\Psi(\mathsf{c}_1, \ldots, \mathsf{c}_{i-1}, \mathsf{c}', \mathsf{c}_{i+1}, \ldots, \mathsf{c}_n, \vec{a})$, this means that such operation is strict for the query encoded as $(\mathsf{c}_1, \ldots, \mathsf{c}_{i-1}, \mathsf{c}, \mathsf{c}_{i+1}, \ldots, \mathsf{c}_n)$, meaning that $\vec{a}$ is not an answer to the derived query denoted by $(\mathsf{c}_1, \ldots, \mathsf{c}_{i-1}, \mathsf{c}', \mathsf{c}_{i+1}, \ldots, \mathsf{c}_n)$.

Similarly, we can identify the neutral one-step specializations and generalizations. If some $\mathtt{refAx}(\mathsf{c}, \mathsf{c}')$ is applicable on the query denoted by $(\mathsf{c}_1, \ldots, \mathsf{c}_{i-1}, \mathsf{c}, \mathsf{c}_{i+1} \ldots, \mathsf{c}_n)$, meaning that we

have some $\texttt{refAt}(\texttt{c}_{\tau_i}, \texttt{c}, \vec{\texttt{v}})$, and the fact $\texttt{strAtm}_\texttt{s}(\texttt{c}_1, \ldots, \texttt{c}_{i-1}, \texttt{c}, \texttt{c}_{i+1}, \ldots, \texttt{c}_\texttt{n}, \texttt{c}_{\tau_i}, \texttt{c}')$ cannot be derived, then we get $\texttt{nrtAtm}_\texttt{s}(\texttt{c}_1, \ldots, \texttt{c}_{i-1}, \texttt{c}, \texttt{c}_{i+1}, \ldots, \texttt{c}_\texttt{n}, \texttt{c}_{\tau_i}, \texttt{c}')$ and conclude that such operation produces a neutral one-step specialization of the query and we transitively close all such operations for $(\texttt{c}_1, \ldots, \texttt{c}_{i-1}, \texttt{c}, \texttt{c}_{i+1}, \ldots, \texttt{c}_\texttt{n})$ and $\texttt{c}_{\tau_i}$.

In the next step, we identify for each query the one-step neutral reformulations. Then, for each atom τ_i, and query, denoted by $\texttt{refAtms}(\vec{\texttt{c}})$, we define rules using predicates $\texttt{nrtAtm}_\texttt{s}$ and $\texttt{ntrAtm}_\texttt{g}$ to identify the non-strict operations on atom i in the query. For that we rely on the applicable reformulation axioms which do not trigger a strict specialization or generalization, i.e. negating $\texttt{strAtm}_\texttt{s}(\vec{\texttt{c}}, \texttt{c}_{\tau_i}, \texttt{c}')$ and $\texttt{strAtm}_\texttt{g}(\vec{\texttt{c}}, \texttt{c}_{\tau_i}, \texttt{c}')$, where $\texttt{c}'$ denotes the Datalog constant for the reformulated atom. Then we collect using predicate $\texttt{nrt}_\texttt{s}$ all possible application of one-step specializations and close them transitively: $\texttt{nrt}_\texttt{s}(\vec{\texttt{c}}, \vec{\texttt{c}})$ denotes that query with encoding $\vec{\texttt{c}}'$ is a neutral specialization of $\vec{\texttt{c}}$. Lastly, as according to the definition, a most specific neutral specialization is a specialization that cannot be further neutrally specialized, which is what the rule defining predicate $\texttt{msnSpe}$ states. The generalizing case is analogous.

We have now a Datalog encoding that is is sound and complete for navigating the query space.

Theorem 4.2. *Let Ψ be rooted, $\mathcal{Q} = (Q^{\mathcal{K}}_{\Psi}, \preceq_{\mathcal{K}})$ and let $D_{all} = D_{\Psi,\mathcal{K}} \cup D_{\mathcal{A}}$. For $q, q' \in \mathcal{Q}$ such that $cert(q, \emptyset, \mathcal{A}^{\mathcal{O},k}_{\Sigma}) \neq \emptyset$ and $cert(q', \emptyset, \mathcal{A}^{\mathcal{O},k}_{\Sigma}) \neq \emptyset$, let $\vec{c_q}$ be Π_Ψ-encoding of q and $\vec{c_{q'}}$ the Π_Ψ-encoding of q'.*

(a) $q' \in msnSpe_{\mathcal{A}}(q, \mathcal{Q})$ *iff* $\texttt{msnSpe}(\vec{c_q}, \vec{c_{q'}}) \in \Pi_{\Psi,\mathcal{K},ref}(D_{all})$,
(b) $q' \in mgsSpe_{\mathcal{A}}(q, \mathcal{Q})$ *iff* $\texttt{mgsSpe}(\vec{c_q}, \vec{c_{q'}}) \in \Pi_{\Psi,\mathcal{K},ref}(D_{all})$,
(c) $q' \in mgnGen_{\mathcal{A}}(q, \mathcal{Q})$ *iff* $\texttt{mgnGen}(\vec{c_q}, \vec{c_{q'}}) \in \Pi_{\Psi,\mathcal{K},ref}(D_{all})$,
(d) $q' \in mssGen_{\mathcal{A}}(q, \mathcal{Q})$ *iff* $\texttt{mssGen}(\vec{c_q}, \vec{c_{q'}}) \in \Pi_{\Psi,\mathcal{K},ref}(D_{all})$.

Proof. We proceed with a proof for the specializing cases. The proof is done analogously for the generalizing cases.

First, we want to show correctness of the rule defining $\texttt{strAtm}_\texttt{s}$. Recall that $\Psi \overset{\text{s}}{\rightsquigarrow} \Psi'$ denotes a specializing one-step and we reuse the same notation upon queries, i.e., $q_1 \overset{\text{s}}{\rightsquigarrow} q_2$ if $\Psi_{q_1} \overset{\text{s}}{\rightsquigarrow} \Psi_{q_1}$. For any $q_1, q_2 \in \mathcal{Q}$:

$$q_1 \overset{\text{s}}{\rightsquigarrow} q_2 \text{ s.t. } q_2 \prec_{\mathcal{K}} q_1 \text{ iff } \texttt{strAtm}_\texttt{s}(\vec{c_{q_1}}, \texttt{c}_{\tau_j}, c') \in \Pi_{\Psi,\mathcal{K},ref}(D_{all}) \text{ and } \vec{c_{q_2}} = \vec{c}_{q_1}[c_j \mapsto c'], \text{ for some } 1 \leq j \leq n, \tag{4.1}$$

where $\vec{c}_{q_1}[c_j \mapsto c']$ denotes that c' replaces constant on position j from tuple $\vec{c}_{q_1}$.

Direction $\Rightarrow$ follows immediately from the fact that the Datalog encoding captures all queries in the space and all their answers using predicate $\texttt{query}_\Psi$ (c.f. Theorem 4.1). This means that each query encoding is captured by $\texttt{refAt}$ in $\Pi_{\Psi,\mathcal{K}}(D_{all})$. From $q_2 \prec q_1$ we get that there is some answer $\vec{a}$ of q_1 which is not an answer of q_2, hence we also have that $\texttt{query}_\Psi(\vec{c}_{q_1}, \vec{a}) \in \Pi_{\Psi,\mathcal{K}}(D_{all})$ and $\texttt{query}_\Psi(\vec{c}_{q_2}, \vec{a}) \notin \Pi_{\Psi,\mathcal{K}}(D_{all})$. Therefore, rule defining $\texttt{strAtm}_\texttt{s}$ fires in this case, hence this strict specializing step is captured by $\texttt{strAtm}_\texttt{s}$ in $\Pi_{\Psi,\mathcal{K},ref}(D_{all})$. **Direction** $\Leftarrow$ follows from the

$\mathbf{\Pi}_{\text{ref}}$:

$\texttt{refAx}(U', U) \leftarrow \texttt{cISA}(U, U').$ $\quad \texttt{refAx}(U', U) \leftarrow \texttt{rISA}(U, U').$

$\texttt{refAx}(U, X) \leftarrow \texttt{uAtm}(U, X).$ $\quad \texttt{refAx}(U, X) \leftarrow \texttt{bAtm}(U, X, Y).$

$\texttt{refAx}(U, Y) \leftarrow \texttt{bAtm}(U, X, Y).$ $\quad \texttt{refAx}(U', U) \leftarrow \texttt{rISAinv}(U, U').$

$\texttt{refAx}(U', U) \leftarrow \texttt{rup}(U, U', Z).$ $\quad \texttt{refAx}(U, U') \leftarrow \texttt{ddn}(U, U', Z).$

$\texttt{refAx}(U, U) \leftarrow \texttt{conc}(U).$ $\quad \texttt{refAx}(U, U) \leftarrow \texttt{role}(U).$

$\texttt{refAx}(U, U) \leftarrow \texttt{const}(U).$

$\texttt{strAtm}_\text{s}(\vec{U}[U_i \mapsto V], \texttt{c}_{\tau_i}, V') \leftarrow \texttt{refAx}(V, V'), \texttt{refAt}(\texttt{c}_{\tau_i}, V, X, Y), \texttt{refAt}(\texttt{c}_{\tau_i}, V', X', Y'),$
$\texttt{query}_\Psi(\vec{U}[U_i \mapsto V], \vec{X}),\ \neg\texttt{query}_\Psi(\vec{U}[U_i \mapsto V'], \vec{X}).$

$\texttt{strAtm}_\text{g}(\vec{U}[U_i \mapsto V'], \texttt{c}_{\tau_i}, V) \leftarrow \texttt{refAx}(V, V'), \texttt{refAt}(\texttt{c}_{\tau_i}, V, X, Y), \texttt{refAt}(\texttt{c}_{\tau_i}, V', X', Y'),$
$\texttt{query}_\Psi(\vec{U}[U_i \mapsto V], \vec{X}),\ \neg\texttt{query}_\Psi(\vec{U}[U_i \mapsto V'], \vec{X}).$

$\texttt{nrtAtm}_\text{s}(\vec{U}[U_i \mapsto V], \texttt{c}_{\tau_i}, V') \leftarrow \texttt{refAx}(V, V'), \texttt{refAtms}(\vec{U}[U_i \mapsto V]), \neg\texttt{strAtm}_\text{s}(\vec{U}[U_i \mapsto V], \texttt{c}_{\tau_i}, V').$

$\texttt{ntrAtm}_\text{g}(\vec{U}[U_i \mapsto V], \texttt{c}_{\tau_i}, V') \leftarrow \texttt{refAx}(V', V), \texttt{refAtms}(\vec{U}[U_i \mapsto V]), \neg\texttt{strAtm}_\text{g}(\vec{U}[U_i \mapsto V], \texttt{c}_{\tau_i}, V').$

$\texttt{nrt}_\text{x}(\vec{U}, \vec{U}[U_i \mapsto V]) \leftarrow \texttt{refAtms}(\vec{U}), \texttt{nrt}_\text{x}(\vec{U}, \texttt{c}_{\tau_i}, V).$

$\texttt{nrt}_\text{x}(\vec{U}, \vec{V}') \leftarrow \texttt{nrt}_\text{x}(\vec{U}, \vec{V}), \texttt{nrt}_\text{x}(\vec{V}, \vec{V}').$

$\texttt{msnSpe}(\vec{U}, \vec{V}) \leftarrow \texttt{nrt}_\text{s}(\vec{U}, \vec{V}), \texttt{refAtms}(\vec{V}'), \neg\texttt{nrt}_\text{s}(\vec{V}, \vec{V}'), \vec{V} \neq \vec{V}'.$

$\texttt{mgnGen}(\vec{U}, \vec{V}) \leftarrow \texttt{nrt}_\text{g}(\vec{U}, \vec{V}), \texttt{refAtms}(\vec{V}'), \neg\texttt{nrt}_\text{g}(\vec{V}, \vec{V}'), \vec{V} \neq \vec{V}'.$

$\texttt{simeq}(\vec{U}[U_i \mapsto V], \vec{U}) \leftarrow \texttt{nrtAtm}_\text{s}(\vec{U}, \texttt{c}_{\tau_i}, V).$

$\texttt{simeq}(\vec{U}, \vec{U}[U_i \mapsto V]) \leftarrow \texttt{ntrAtm}_\text{g}(\vec{U}, \texttt{c}_{\tau_i}, V).$

$\texttt{simeq}(\vec{U}_1, \vec{U}_3) \leftarrow \texttt{simeq}(\vec{U}_1, \vec{U}_2), \texttt{simeq}(\vec{U}_2, \vec{U}_3).$

$\texttt{mgsSpe}(\vec{U}, \vec{V}'[V_i' \mapsto V'']) \leftarrow \texttt{simeq}(\vec{V}', \vec{U}), \texttt{strAtm}_\text{s}(\vec{V}', \texttt{c}_{\tau_i}, V'').$

$\texttt{mssGen}(\vec{U}, \vec{V}'[V_i' \mapsto V'']) \leftarrow \texttt{simeq}(\vec{U}, \vec{V}'), \texttt{strAtm}_\text{g}(\vec{V}', \texttt{c}_{\tau_i}, V'').$

Table 4.3: Datalog program to compute query reformulations

fact that all queries in $\texttt{refAt}$ are part of $\mathcal{Q}$ (c.f. Lemma 4.3) and that each answer of $\texttt{query}_\Psi$ over $\Pi_{\Psi,\mathcal{K}}(\mathcal{D}_{all})$ contains some query encoding and one of its answers. Since the rules in Π_{rules} capture all query derivation rules, we conclude that for $\texttt{strAtm}_\text{s}(\vec{c_{q_1}}, \texttt{c}_{\tau_k} c')$ s.t. $\vec{c_{q_2}} = \vec{c}_{q_1}[c_k \mapsto c']$ it must be that $q_2 \prec q_1$.

We proceed next with proving statement a. For direction $\Rightarrow$ let q, q' be such that q' is a MSNS of q in $\mathcal{Q}$ and $cert(q, \emptyset, \mathcal{A}_\Sigma^{\mathcal{O},k}) \neq \emptyset$. Let $\vec{c}_q = (\texttt{c}_1, \ldots, \texttt{c}_\text{n})$ and $\vec{c}_{q'} = (\texttt{c}_1', \ldots, \texttt{c}_\text{n}')$ be the Π_Ψ-encodings of q and q' respectively. Then there exists some sequence of one-step derivation such that $q \overset{\text{s}}{\rightsquigarrow}_\mathcal{K} q_1 \overset{\text{s}}{\rightsquigarrow}_\mathcal{K} \ldots \overset{\text{s}}{\rightsquigarrow}_\mathcal{K} q_m \overset{\text{s}}{\rightsquigarrow}_\mathcal{K} q'$ s.t. $q_k \simeq_\mathcal{K} q$, for $1 \leq k \leq m$. We argue that such sequence is captured by predicate $\texttt{nrtAtm}_\text{s}$: for $1 \leq k \leq m$ there is some $1 \leq i_k \leq n$ such that $\texttt{nrtAtm}_\text{s}(\vec{c}_{q_k}, \texttt{c}_{\tau_{i_k}}, c')$, where c' denotes the constant on position i_k in $\vec{c}_{q_{k+1}}$. If $k = 0$, it means that $q' = q$ and in this case, for each $1 \leq i \leq n$, there is no $\texttt{strAtm}_\text{s}(\vec{c}_q, \texttt{c}_{\tau_i}, c_i)$ in $\Pi_{\Psi,\mathcal{K},ref}(\mathcal{D}_{all})$. Moreover, from the fact that $\texttt{refAx}$ is reflexive, we obtain that $\texttt{nrtAtm}_\text{s}(\vec{c}_q, \texttt{c}_{\tau_i}, c_i)$ holds, hence also $\texttt{nrt}_\text{s}(\vec{c}_q, \vec{c}_{q'})$. If $k \geq 1$, since all derivation steps are captured by the Datalog encoding

Template Ψ_i [# atoms in Ψ_i]	Ψ_1[2]	Ψ_2[3]	Ψ_3[5]	Ψ_4[5]	Ψ_5[6]
Size of $\Pi_i = (\mathrm{D}_i)$ **(MB)**	0.45	5	0.2	18.8	0.5
# Queries in $Q^{\mathcal{K}}_{\Psi_i}$ with answers	4	82	56	316	119
Sum of answers over all queries	1774	3431	16	2758	113
$\Delta_s(\Psi_i)$ / $\Delta_g(\Psi_i)$	350.7 / 175.2	236 / 1005	1.32 / 1.28	139 / 510	3.9 / 40
			Time		
Computation of $\Pi_i(\mathrm{D}_i)$ **(s)**	2.4	4.04	2	10.7	5.6
Avg. retrieval of answers $q \in Q^{\mathcal{K}}_{\Psi_i}$ **(ms)**	0.5	0.17	0.3	0.09	0.16
Avg. retrieval $q' \in \mathsf{minStrG/S}(q)$ **(ms)**	0.3 / 0.4	0.10 / 0.09	0.14 / 0.1	0.04 / 0.03	0.10 / 0.09

Table 4.4: Experiment results over DBpedia. Here $\Pi_i = \Pi_{\Psi_i} \cup \Pi_{ref}$ and $\mathrm{D}_i = D_{\Psi_i,\mathcal{K}} \cup D_{\mathcal{A}}$.

Π_Ψ, and from (4.1) we get that all such neutral one-step specializations are not captured by $\mathtt{strAtm_s}$, we obtain that $\mathtt{nrt_s}(\vec{c}_{q_k}, \vec{c}_{q_{k+1}})$ holds and by the transitive closure rule, we get that also $\mathtt{nrt_s}(\vec{c}_q, \vec{c}_{q'})$ holds. Likewise, from (4.1) we can also conclude that there is no other $\mathtt{nrt_s}(\vec{c}_{q'}, \vec{c}_{q''})$ since otherwise will contradict the fact that q'' is most specific w.r.t. $\preceq_{\mathcal{K}}$.

For direction $\Leftarrow$, let $\mathtt{msnSpe}(\vec{c}_q, \vec{c}_{q'}) \in \Pi_{\Psi,\mathcal{K},ref}(\mathcal{D}_{all})$. Then there must be a sequence s.t. $\vec{c}_{q_1} = \vec{c}_q$ and $\vec{c}_{q_m} = \vec{c}_{q'}$ and for $1 \leq k \leq m$ there is some $1 \leq i_k \leq n$ and $\mathtt{nrtAtm_s}(\vec{c}_{q_k}, \mathtt{c}_{\tau_{i_k}}, c_k)$ such that $\vec{c}_{q_k} = \vec{c}_{q_{k-1}}[c_{i_{k-1}} \mapsto c_{k-1}]$. Hence we get that $\mathtt{refAx}(\vec{c}_{q_k})$ holds and from completeness of Π_Ψ regarding all queries in the space we get that each $q_k \in \mathcal{Q}$. Moreover, from (4.1), we get that $q_{k+1} \simeq q_k$. From definition of rule $\mathtt{msnSpe}$, we get that the sequence stops with q_m, i.e., there is no $\mathtt{nrtAtm_s}(\vec{c}_{q_{m+1}}, \mathtt{c}_{\tau_{i_{m+1}}}, c_{m+1})$ with $\vec{c}_{q_{m+1}} \neq \vec{c}_{q_m}$. Then it must be that if some $q'' \preceq_{\mathcal{K}} q'$ then $q'' \prec q'$, therefore $q'' \prec q$. We then conclude that q' is a MSNS of q.

We now focus on b. Recall that, a most general strict specialization of a query q is a query q' such that $q' \prec q$ and for each q' such that $q' \preceq q'' \preceq q$, we have that $q'' \simeq q$. Based on Observation 4.1, if we can identify all neutral specializations and all strict specializations, we can identify the most general strict specializations by applying a strict one-step specializing operation to some neutral specialization, reflected in the rule defining $mgsSpe_{\mathcal{A}}$. We then conclude that also b holds.

The proof of statements c and d is done analogously.

□

4.4 Implementation and Evaluation

We implemented our exploratory framework using Rulewerk Java API for VLog Datalog reasoner [CDG+19]. The implementation[2] consists of a template parser and a translator of the template and rules of the encoding into Rulewerk syntax. All experiments have been performed on a MacBook

[2] https://github.com/medinaandresel/DL2020

```
%Psi_1
X,Y :− GEN{dbo:FilmFestival}(X) AND dbo:startDate(X,Y)

%Psi_2
X :− GEN{dbo:FilmFestival}(X) AND SPE{dul:hasLocation}(X,Z) AND Z=GEN{dbr:Paris}

%Psi_3
X,Y :− GEN{dbo:Museum}(X) AND SPE{dbo:Museum}(Y,X) AND
       SPE{dbo:Artwork}(Y) AND SPE{dul:hasLocation}(X,Z) AND Z=GEN{dbr:Paris}

%Psi_4
X,Y :− GEN{dbo:FilmFestival}(X) AND SPE{dbo:Museum}(Y) AND
     SPE{dul:hasLocation}(X,Z) AND SPE{dul:hasLocation}(Y,Z) AND Z=GEN{dbr:Paris}

%Psi_5
X,Y,Z :− SPE{dbo:Organisation}(X) AND GEN{dbo:foundationPlace}(X,Y)
         AND GEN{dbo:headquarter}(X,Z) AND SPE{dbo:isPartOf}(Z,U)
         AND SPE{dbo:isPartOf}(Y,U) AND U=GEN{dbr:Paris}
```

Figure 4.2: The templates $\Psi_1, \ldots, \Psi_5$ (in the supported input syntax) used in the experimental evaluation.

Pro (2.7 GHz i5 8GB) using JavaSE 14.0.1 and Rulewerk version 0.5.0[3]. We used the DBpedia ontology and dataset, accessed via the endpoint[4]. We used one set $\mathcal{K}$ of reformulation axioms with 336 axioms and assertions extracted from DBpedia. With its signature and the DBpedia ontology, we computed the k-bounded Σ-extension $D_{\mathcal{A}}$ using existential rules in VLog. The resulting dataset was computed relatively fast, given that the data was remotely accessed, and it took in total about 3 minutes to materialize. The size of the extended dataset is around 700 MB. We have designed templates of various sizes and shapes over the ontology vocabulary containing classes such as: $\mathsf{Event}^{\mathsf{s}}$, $\mathsf{Museum}^{\mathsf{s}}$, $\mathsf{ArtWork}^{\mathsf{s}}$, $\mathsf{Organization}^{\mathsf{s}}$ etc., properties $\mathsf{startDate}^{\mathsf{g}}$, $\mathsf{museum}^{\mathsf{s}}$, $\mathsf{hasLocation}^{\mathsf{s}}$, $\mathsf{headquarter}^{\mathsf{g}}$ etc., and resources e.g., $\mathsf{Paris}^{\mathsf{g}}$. We have used 5 templates with sizes between 2 and 6 atoms, which are presented in Figure 4.2 in the supported input syntax (for further details regarding this syntax please consult the online repository).

The main goals of our evaluation were (a) to test the feasibility of our framework in practice, in particular the trade-off between the time to evaluate the Datalog program and the time to answer and compute query reformulations from the pre-computed model, and (b) to test if the template-generated query spaces ensure a gradual navigation of the answers. For each input Ψ_i we have measured: $|Q^{\mathcal{K}}_{\Psi_i}|$ - number of generated queries with answers, total number of answers captured by the query space, and as well as the computational time: to evaluate the Datalog program, the average time to read the answers to queries and the average time to

[3] https://github.com/knowsys/rulewerk

[4] SPARQL endpoint: https://dbpedia.org/sparql

compute reformulations from the compilation. Then, for each query q, we measure $\Delta_s(q)$ – the average number of answers that are dropped by some query in $mgsSpe_{\mathcal{A}}(q)$, respectively $\Delta_g(q)$ the average number of answers gained by some $mssGen_{\mathcal{A}}(q)$. Then, for the entire query space, $\Delta_s(\Psi_i) = (\sum_{q \in Q_{\Psi_i}} \Delta_s(q))/|Q_{\Psi_i}|$ measures the average discarded answers when navigating the query space and similarly $\Delta_g(\Psi_i) = (\sum_{q \in Q_{\Psi_i}} \Delta_g(q))/|Q_{\Psi_i}|$ measures the average gained answers.

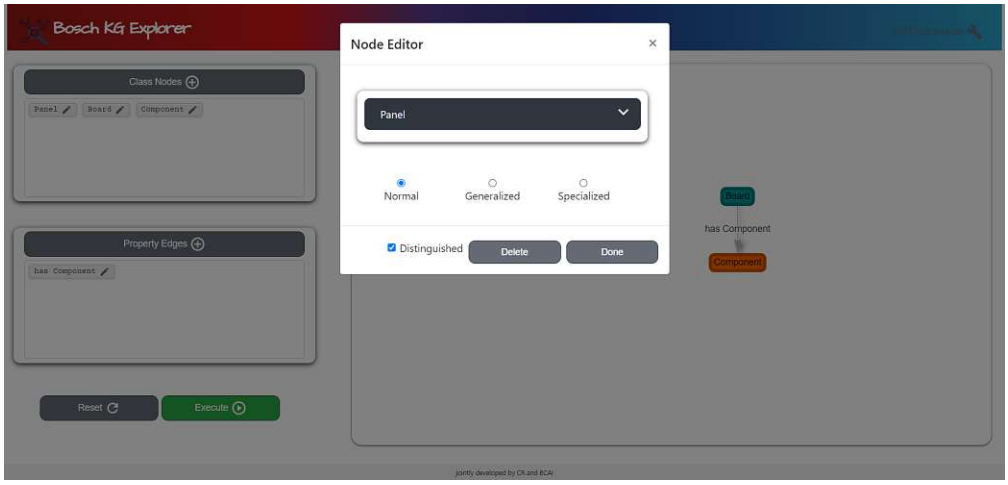
Table 4.4 summarizes our evaluation. Evaluating the Datalog program $\Pi_{\Psi,\mathcal{K},ref}$ over the materialized data $\mathcal{D}_{all}$ is done within seconds for all the query spaces and depends on the number of answers captured by the query space (i.e. the larger the number of answers the longer it took to compile). Reading the answers to queries and navigating the query space is done in less than 1 ms in all cases. Based on the inclusivity and exclusivity of answers, measured for of the strict reformulations, we can conclude that using most general strict specializations and most specific strict generalizations offer a reasonably gradual exploration of the data, relative to the number of answers captured by each query space.

4.5 Related Work and Discussion

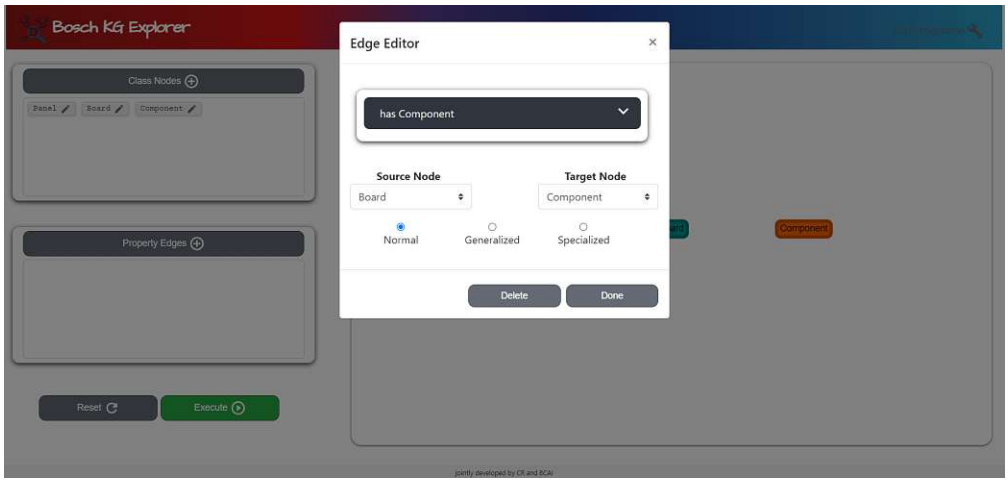
In recent years, several exploratory search engines have been proposed to support data access for different exploratory purposes. The basic idea at the core of many of them is to guide the query formulation process, in a step-by-step fashion. The many proposed techniques for exploring ontology-mediated data include similarity-based methods such as [SEI+18, YBRW16], and visual query languages [ACGK+14, SGKK17, SKZ+14], which include ontology reasoning with query language expressivity ranging between tree-shaped CQs and monadic positive existential queries. More recently, [VAHL19] is able to cover formulation of SPARQL queries, however without ontological reasoning. To abstract away the ontology reasoning step needed to obtain complete answers, in [BKPR14] a schema-agnostic approach to rewrite *DL-Lite*$_{\mathcal{R}}$ OMQs into SPARQL 1.1. is proposed.

Query generalizations have been proposed as a technique for interpreting null answers (i.e., empty answers) in cooperative database systems [Mot84]. The considered generalizations are similar to the ones we propose, however they also consider numeric comparisons, while we additionally consider the roll-up and drill-down operations. In line with, [MT14], we propose a query template language also designed to ease the process of constructing queries in which specializing or generalizing atoms are at the core. Our template language allows to describe a set of CQs that are semantically related via two types of taxonomies: concept and role hierarchies, and dimensions. In addition, we propose also solutions for automatic query reformulations that supports data exploration. In our approach this is achieved by moving from one query to another in a way that minimally changes the answers. One advantage of our approach is that it is implementable in Datalog and our preliminary evaluation results show that it is feasible in practice.

Evaluating the Datalog program is related to the problem of answering queries using views, which has been intensively studied for relational data [Hal01, Hal00]. As shown in [CGLR12], to answer OMQs using views, a different semantics to materialize the views is needed, however, for *DL-Lite*$_{\mathcal{R}}$ this can be done using existing techniques as discussed in this chapter.



(a) Unary atom creation



(b) Binary atom creation

Figure 4.3: Node and edge view modes used to create query templates

For future work, it would be interesting to implement a user interface that can support the creation of the query template, the query navigation process and the visualization of the answers in an informative manner. This is indeed a low-hanging fruit and an initial prototype, illustrated in Figure 4.3, has been implemented.

Another interesting extension is to study other derivation rules that rely on more expressive ontology languages. Moreover, the notion of query space for aggregating queries is another relevant direction as well as to exploit the information existing in the compilation for data analysis purposes.

Part II

Data Incompleteness

CHAPTER 5

Ontology-mediated Conditional Answers

In many scenarios traditional query answering is too restrictive when the data is either incomplete or only partially accessible to the end-users. For example, medical records or private information are in general inaccessible and thus not accounted in decision making.

Among the coping mechanisms for handling incompleteness is to answer queries considering possible hypothesis, problem denoted as *hypothetical query answering* [CA98], which allows answering questions of the form "which students are eligible for graduation if they pass a particular exam". In this formalism the assumptions are used as possible data extensions and in general they are very concrete and pre-defined by users. Another existing approach is to compute the conditions which guarantee the validity of the answer [Dem92, Dem98]. In this latter approach, the conditions are as well derived, thus not designed in advance.

In this chapter, we consider the problem of answering ontology-mediated queries under assumptions and present novel semantics that use the notion of a *conditional answer*, which, in our case, pairs tuples (which need not be certain answers) with sets of facts (that would make them true certain answers). Traditional certain answers are just assumptive answers where the assumption component is empty.

In order to allow some control in choosing the assumptions, we enrich OMQs with *assumption patterns*, which define the form of complex assertions that we want to use as conditions. This leads to a powerful formalism, but does not impact significantly the worst-case complexity compared to plain OMQs. Importantly, our OMQs with assumptions are *first-order rewritable* for *DL-Lite*$_{\mathcal{A}}$.

Our extension of OMQs turns out to be particularly powerful when complete and incomplete information coexist. Based on classical logic, standard OMQs make the *open-world assumption* (OWA), where a fact that cannot be inferred may be either true or false. Traditional databases, in contrast, make the *closed-world assumption (CWA)*: facts that cannot be inferred are assumed

false. The OWA makes OMQs useful for incomplete data, but at the same time, too weak to yield useful answers when the data is known to be partially complete. For example, a gluten-free restaurant implies there exists some gluten-free dish in the menu, and in the data all existing gluten-free dishes are also labeled healthy. If we know that r_1 is a gluten-free restaurant and in addition all existing gluten-free dishes in the menu are present in the data, then r_1 is expected as answer to the query "restaurants with some healthy dish in the menu" even if no explicit link between r_1 and some gluten-free dish exists in the data. Therefore leveraging the completeness assumption we can obtain additional answers that cannot be derived under the classical OWA.

A simple yet powerful way to leverage such meta-information is to explicitly declare the predicates that should be assumed complete, e.g., marking GlutenFreeDish as closed and restricting the semantics such that we rely only on models that map such predicates to the exact set of constants (or pair of constants in the case of roles) present in the ABox. Closed predicates are very useful to cope with partial incompleteness, but they can dramatically increase the complexity of reasoning. So far most results were negative, even for lightweight DLs [LSW13, LSW15, NOv16]. Existing results show that even for the most restricted OMQs languages, like CQs paired with *DL-Lite* ontologies, closed predicates make OMQ answering intractable in data complexity, and destroy FO-rewritability [LSW13]. Remarkably, in our OMQs, we can use closed predicates in our assumptions, for instance by using that each gluten-free restaurant has also a gluten-free dish in the menu as an *assumption* instead of a rule in the ontology, we can as well obtain r_1 as answer to the query "restaurants with some healthy dish in the menu" under the condition that r_1 has some gluten-free dish in the menu.

This chapter is structured as follows: in Section 5.1 we formalize the notion of an ABox extension, which is used to define the semantics of conditional answers, and differentiate it from the notion of ABox completion[1], in Section 5.2 we introduce the notion of assumption-based OMQA and propose concrete semantics in terms of conditional answers while in Section 5.3 we present a rewriting approach to construct them for FO-rewritable *DL-Lite*$_{\mathcal{A}}^{++}$ positive ontologies. The integration of closed predicates is presented in Section 5.4, while in Section 5.5 we propose a rewriting approach for integrating functionality and disjointness axioms. A preliminary empirical evaluation of the approach is presented in Section 5.6. It shows that our technique is not only feasible in practice but also is more efficient than evaluating federated queries. Based on this observation, assumption-based OMQA is a good alternative technique in cases when the data is not centralized and part of it is accessed via remote end-points. This chapter concludes with Section 5.7 in which we present an in-depth overview of the related work and discussion.

5.1 ABox Completion and Extension

In this section we motivate the need for expanding the ABox and analyze two perspectives on how to complete the ABox for obtaining plausible answers derived from using coherent assumptions about existing data.

[1] or KG completion as mostly known in the literature.

A natural way to complete the data is to add missing true facts involving the given signature of the ABox and ontology. For example the fact that the UN is located in Vienna is a true fact however the assertion locatedIn(UN, Vienna) might not be present in the data. In this case, under the assumption that missing facts involve only constants and predicates from the given signature, the challenge is then to identify the missing connections between them. This notion of completion is closely related to the problem of *link prediction over incomplete knowledge graphs*, which we will address more closely in the next chapter.

Definition 5.1 (ABox completion)**.** *Given an ABox $\mathcal{A}$ and an ontology $\mathcal{O}$, a* completion for $\mathcal{A}$ w.r.t. $\mathcal{O}$ *is an ABox $\mathcal{A}'$ that is consistent w.r.t. $\mathcal{O}$ such that $\mathcal{A} \subseteq \mathcal{A}'$ and which can contain any additional assertion α of the form $A(a)$ or $r(a, b)$, with $A, r \in sign(\mathcal{O})$ and $a, b \in cst(\mathcal{A})$, such that $(\mathcal{A}, \mathcal{O}) \not\models \alpha$.*

The above introduced notion of an ABox completion is different than the notion of an ABox model since it considers as relevant only missing assertions involving known individuals. For that reason, given that the signature of $\mathcal{O}$ and the set of ABox constant symbols are finite, each possible ABox completion is finite and moreover there exist only finitely many of them.

In our case, we want to identify only those which trigger *possible additional answers* to OMQs. There are however many sets of possible facts which do not make sense given the underlying domain ontology. For example, consider the following ABox, ontology and query:

$$\mathcal{A} = \{\text{MCI}(\mathsf{e}), \text{location}(\mathsf{e}, \mathsf{Praterstern}), \text{connectedTo}(\mathsf{Praterstern}, \mathsf{Taborstrasse})\},$$
$$\mathcal{O} = \{\text{SevereMCI} \sqsubseteq \text{MCI}, (\mathsf{funct\ \ location})\}, \qquad q(y) \leftarrow \exists x.\text{SevereMCI}(x) \wedge \text{location}(x, y).$$

For such an OMQ and ABox, there are no certain answers, however there are several additional sets of possible facts which either produce senseless answers, such as:

- e by adding the facts $\mathcal{A}_{\mathsf{e}} = \{\text{SevereMCI}(\mathsf{Praterstern}), \text{location}(\mathsf{Praterstern}, \mathsf{e})\}$,

or produce answers that require the addition of facts which do not make sense with respect to the ontology:

- Praterstern by adding any $\mathcal{A}_c = \{\text{location}(c, \mathsf{Praterstern}), \text{SevereMCI}(c)\}$, for $c \in \{\mathsf{e}, \mathsf{Praterstern}, \mathsf{Taborstrasse}\}$;
- Taborstrasse by adding any $\mathcal{A}_c = \{\text{SevereMCI}(c), \text{location}(c, \mathsf{Taborstrasse})\}$, for $c \in \{\mathsf{Taborstrasse}, \mathsf{Praterstern}\}$.

In the first case SevereMCI(e) is sufficient for obtaining e as answer and it is preferred instead of $\mathcal{A}_{\mathsf{e}}$, $\mathcal{A}_{\mathsf{Praterstern}}$ or $\mathcal{A}_{\mathsf{Taborstrasse}}$, while in the second case, we want to avoid obtaining such sets of possible facts. To avoid completions which do not make sense, we could add to the ontology all the necessary concept disjointness assertions with the effect of loosing Taborstrasse as a possible certain answer, since there will be no completion which makes this answer true w.r.t. the extended ontology.

While having meaningful approximations of the ground truth is important in many scenarios, it is typically difficult to obtain. What we propose instead is to identify possible answers to OMQs and

the assumptions that are needed to ensure the answer is certain. For testing if the assumptions hold or not, additional solutions are required which are beyond of the scope of this thesis. However, to avoid obtaining meaningless answers, we aim for identifying very specific assumptions which are in general easier to verify. To that end, we build on the notion of an ABox completion and allow $\mathcal{ELI}$ assertions as possible facts.

Definition 5.2 (ABox extension)**.** *Let $\mathcal{A}$ be an ABox, $\mathcal{O}$ an ontology. A set $\mathcal{E}$ of assertions of the form $r(c, c')$ or $C(c)$, where $c, c' \in cst(\mathcal{A})$, $r \in sign(\mathcal{O})$ and C is an $\mathcal{ELI}$ concept over $sign(\mathcal{O})$ is called* a set of possible assertion w.r.t. $(\mathcal{O}, \mathcal{A})$ *iff $\mathcal{A} \cup \mathcal{E}$ is consistent w.r.t. $\mathcal{O}$ and for each $\alpha \in \mathcal{E}$ we have that $(\mathcal{A}, \mathcal{O}) \not\models \alpha$.*

In this setting, by adding $\exists \text{location}^{-}.\text{SevereMCI}(\text{Taborstrasse})$ to $\mathcal{A}$ in the previous example, which states that there is a severe MCI located at station Taborstrasse, without naming a particular entity would allow us to obtain also Taborstrasse as additional answer, while completing the ABox in a manner that is coherent w.r.t. the ontology. Moreover, answers which do not make sense can be avoided by controlling on which positions in the query the assumptions should be applied. In the following section we formalize such problem.

5.2 Assumption-based Ontology-mediated Query Answering

In this section we present the problem of evaluating OMQs over ABox extensions to obtain *conditional answers*, which determine not only the possible certain answer tuple but also which set of possible assertions we need to add.

Definition 5.3 (Conditional answers semantics)**.** *Let $\mathcal{Q} = (\mathcal{O}, q(\vec{x}))$ be an OMQ and let $\mathcal{A}$ be an arbitrary ABox. A pair $(\vec{a}, \mathcal{E})$ consisting of a tuple of constants $\vec{a}$ and a set of possible assertions $\mathcal{E}$ w.r.t. $(\mathcal{O}, \mathcal{A})$ is a* conditional certain answer *to $\mathcal{Q}$ over $\mathcal{A}$ if $\vec{a}$ is a certain answer of $\mathcal{Q}$ over $\mathcal{A} \cup \mathcal{E}$.*

We denote by $cans(\mathcal{Q}, \mathcal{A})$ the set of all conditional answers to $\mathcal{Q}$ over $\mathcal{A}$. We call $(\vec{a}, \mathcal{E}) \in cans(\mathcal{Q}, \mathcal{A})$ a minimal conditional answer *if there is no $\mathcal{E}' \subsetneq \mathcal{E}$ such that $(\vec{a}, \mathcal{E}') \in cans(\mathcal{Q}, \mathcal{A})$, and denote the set thereof by $cans_{min}(\mathcal{Q}, \mathcal{A})$.*

A certain answer does not need any additional assumptions, i.e., $(\vec{a}, \emptyset)$ is a conditional answer iff $\vec{a}$ is a certain answer in the usual sense, however for a tuple which is a possible certain answer, the most relevant assertions are those participating in the matching of that tuple. Furthermore, we are also interested in *minimal* extensions, meaning that all facts in $\mathcal{E}$ are needed to ensure entailment which can be computed by identifying the minimal $\mathcal{E}$, with respect to set inclusion, for each tuple $\vec{a}$. Note that in general for some tuple $\vec{a}$ there can be several minimal extensions which are incomparable.

In order to facilitate some control over the preferred extensions and matching positions in the query, we introduce the notion of *assumption patterns*, which are $\mathcal{ELI}$ atoms that can include terms from the query. The intention is that only extensions of that shape are relevant on the indicated query atoms.

Definition 5.4 (Assumptive OMQs)**.** *An* assumption pattern *is an atom of the form $r(t, t')$ or $C(t)$ where t, t' are terms, r is a relation and C is an $\mathcal{ELI}$ concept.*

An assumptive ontology-mediated query *(AOMQ) is a triple $\mathcal{Q} = (q(\vec{x}), \mathcal{O}, \mathcal{H})$, where $(q(\vec{x}), \mathcal{O})$ is an OMQ, and $\mathcal{H}$ is a set of assumption patterns.*

For an AOMQ we consider as valid conditional answers only those pairs for which $\mathcal{E}$ is complying with the shape of the given assumption pattern. However sometimes part of the ground assumption is made true by the existing facts in the data therefore it is useful to identify also the minimal extensions that are necessary to construct a conditional answer. To make this more precise, we define the set of implied assumption patterns which is obtained as follows.

Definition 5.5. *Given $\mathcal{H}$, the set of* implicit assumption patterns*, denoted by $\overline{\mathcal{H}}$ contains $\mathcal{H}$ and in addition extensively applying the following rules:*

- *If $C_1 \sqcap C_2(t) \in \overline{\mathcal{H}}$ then add $C_1(t)$ and $C_2(t)$ to $\overline{\mathcal{H}}$.*
- *If $(\exists p.C)(t) \in \mathcal{H}$ and there are no $\{p(t, y), C(y)\} \subseteq \overline{\mathcal{H}}$, add $p(t, y)$ and $C(y)$ to $\overline{\mathcal{H}}$, where y is a fresh variable.*
- *If $r^-(t, t') \in \mathcal{H}$, then add $r(t', t)$ to $\overline{\mathcal{H}}$.*
- *If $r(t, t') \in \mathcal{H}$, then add $r^-(t', t)$ to $\overline{\mathcal{H}}$.*

When the set of assumption patterns is associated to an OMQ we assume that the fresh variables in the implicit set of assumption patterns are disjoint from the variables in the query.

Observation 5.1. *For any ABox $\mathcal{A}$, OMQ $\mathcal{Q} = (\mathcal{O}, q(\vec{x}))$ and conditional answer $(\vec{a}, \mathcal{E})$ of $\mathcal{Q}$ over $\mathcal{A}$, by definition, there exists a mapping $\pi : \vec{x} \mapsto cst(\mathcal{A})$ such that $\pi(\vec{x}) = \vec{a}$ and $\mathcal{O}, \mathcal{A} \cup \mathcal{E} \models q\pi$. We call π a* conditional match of $\mathcal{Q}$ given $(\mathcal{A}, \mathcal{E})$.

Next, we characterize the conditional answers which comply with the assumption patterns of choice. We are interested in set of possible assertions that are obtained from grounding the set of assumption patterns. If the assumption patterns refer to particular variables in the query, we ensure that the ground assumption patterns are used on the exact positions to match the query.

Definition 5.6 (Conditional answers AOMQs)**.** *Let $\mathcal{H}$ be a set of assumption patterns and $\mathcal{E}$ a set of possible assertions. If there exists a substitution $\pi : vars(\mathcal{H}) \mapsto cst(\mathcal{E})$ such that $\mathcal{H}'\pi = \mathcal{E}$ for some $\mathcal{H}' \subseteq \overline{\mathcal{H}}$, then we say that $\mathcal{E}$* is a π-instantiation of $\mathcal{H}$.

Let $\mathcal{Q} = (\mathcal{O}, q(\vec{x}), \mathcal{H})$ be an AOMQ, $\mathcal{A}$ an ABox and let $(\vec{a}, \mathcal{E})$ be a conditional answer of $(\mathcal{O}, q(\vec{x}))$ over $\mathcal{A}$. We say that $(\vec{a}, \mathcal{E})$ is a conditional answer of $\mathcal{Q}$ over $\mathcal{A}$ *if there is a partial mapping $\pi : \vec{x} \cup vars(\overline{\mathcal{H}}) \mapsto cst(\mathcal{A})$ such that $\pi(\vec{x}) = \vec{a}$, π is a conditional match of $\mathcal{Q}$ given $(\mathcal{A}, \mathcal{E})$, and $\mathcal{E}$ is a π-instantiation of $\mathcal{H}$.*

Example 5.1. *The following example illustrates that conditional answers for AOMQs can retrieve additional information that we would not obtain with standard OMQs. In Figure 5.1, we have the existing facts illustrated in black and solid edges, such as Anna is currently located at Karlsplatz*

Ontology $\mathcal{O}$:

$$\exists \text{location}^{-}.\text{SevereMCI} \sqsubseteq \text{ClosedStation} \qquad \text{disj}(\text{SevereMCI}, \text{Location})$$
$$\exists \text{line}^{-}.\text{ClosedStation} \sqsubseteq \text{DisruptedLine} \qquad (\text{funct location})$$

Query and assumption patterns:

$q(y) \leftarrow \exists xz\ \text{location}(\text{Anna}, x) \wedge \text{line}(x, y) \wedge \text{DisruptedLine}(y) \wedge \text{line}(z, y) \wedge \text{location}(\text{Anna\_home}, z)$,
$\mathcal{H} = \{\exists \text{location}^{-}.\text{SevereMCI}(x), \exists \text{location}^{-}.\text{SevereMCI}(z)\}$

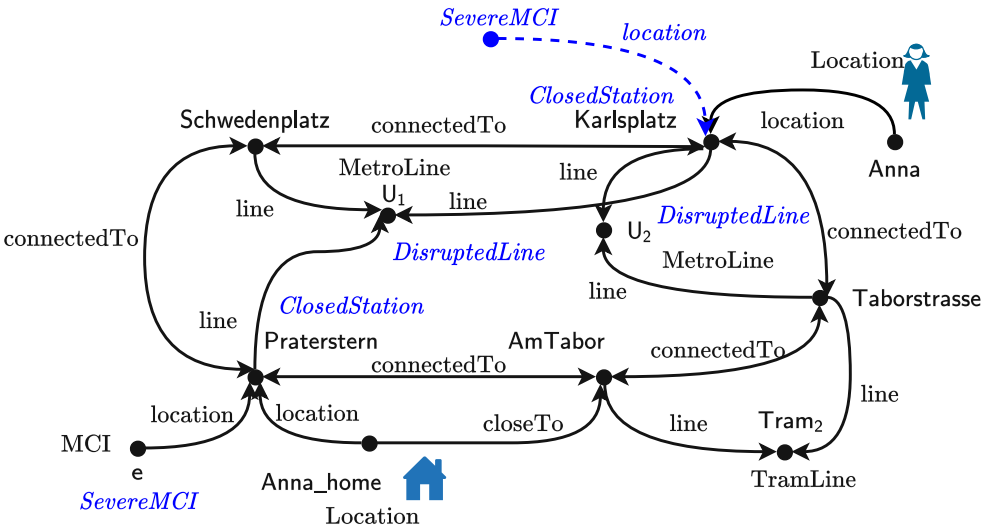


Figure 5.1: Evaluating an assumptive OMQ

and that there is a reported incident at station Praterstern, which is her usual stop to get home. In this situation, she wants to check which are the disrupted lines provided that the incident turns out to be severe.

We can encode this information request as an assumptive OMQ, e.g., $(q(y), \mathcal{O}, \mathcal{H})$ *in Figure 5.1, in which the ontology states that the stations at which some severe incident is reported are closed, and the lines which pass through those stations are disrupted. With the assumption pattern* $\exists \text{location}^{-}.\text{SevereMCI}(x)$ *we enable the assumption that there is some severe MCI reported where Anna is located, while with* $\exists \text{location}^{-}.\text{SevereMCI}(z)$ *we enable the assumption that there is some severe MCI located where Anna lives. The only groundings of the assumption patterns which produce answers are obtained when* $x \mapsto \text{Karlsplatz}$ *and* $z \mapsto \text{Praterstern}$*, as graphically shown (with blue color) in Figure 5.1.*

Applying ontology reasoning over the ground assumptions, we obtain that such stations are closed stations and hence lines U_1 *and* U_2 *would be disrupted. Then the (minimal) conditional answers*

for $(q, \mathcal{O}, \mathcal{H})$, which looks for the disrupted lines that connect Anna's location to her home, are:

U_1 *and* U_2 *for* $\mathcal{E}_1 = \{\exists \text{location}^-.\text{SevereMCI}(\mathsf{Karlsplatz})\}$,

U_1 *for* $\mathcal{E}_2 = \{\text{SevereMCI}(\mathsf{e})\}$ *or* $\mathcal{E}_3 = \{\exists \text{location}^-.\text{SevereMCI}(\mathsf{Praterstern})\}$.

The conditional answers offer the information that the disrupted lines which affect Anna's commute are metro line U_1, if the reported incident e is severe or if there is some severe incident at Praterstern, and both metro lines $\mathsf{U}_1, \mathsf{U}_2$ if there is some severe incident at Karlsplatz.

Since only these possible assertions produce possible answers and since none of them implies that tram line 2 could be disrupted, this means that Anna can safely take the alternative route: the tram line 2 from Taborstrasse to Am Tabor.

The associated decision problem is to test if a pair consisting of a tuple $\vec{a}$ and a set of possible assertions is a conditional answer to a given AOMQ. This problem, naturally depends on the ontology language. For a given ontology language $\mathcal{L}$, the *problem of evaluating an AOMQ in $\mathcal{L}$* is defined as follows:

$\mathcal{L}$-COND-ANSWERS	
Input:	An AOMQ $Q = (q, \mathcal{O}, \mathcal{H})$, ABox $\mathcal{A}$, $\vec{a}$ a tuple of constants and $\mathcal{E}$ a set of possible facts w.r.t. $(\mathcal{A}, \mathcal{O})$
Question:	Is $(\vec{a}, \mathcal{E}) \in \mathit{cans}(Q, \mathcal{A})$?

For OMQ languages where we can rely on existing algorithms for entailment of regular OMQs and consistency testing, it is not hard to obtain an algorithm for answering AOMQs. Given a candidate conditional answer $(\vec{a}, \mathcal{E})$, an ABox $\mathcal{A}$, and an AOMQ $Q = (q(\vec{x}), \mathcal{O}, \mathcal{H})$, we can decide whether $(\vec{a}, \mathcal{E}) \in \mathit{cans}(Q, \mathcal{A})$ in the following way:

1. Guess $\mathcal{H}' \subseteq \overline{\mathcal{H}}$ and a partial mapping π of $\mathit{vars}(q) \cup \mathit{vars}(\exp(\mathcal{H}'))$ to $\mathit{cst}(\mathcal{A})$ such that $\pi(\vec{x}) = \vec{a}$ and $\mathcal{E} = \mathcal{H}'\pi$.
2. Test if $\mathcal{O}, \mathcal{A} \cup \mathcal{E} \models q\pi$.

For typical OMQ languages that combine well known DLs with CQs, the combined complexity of OMQ answering falls into the classes NP, ExpTime, or 2-ExpTime. In such cases, this simple algorithm yields tight complexity bounds, as the cost of the additional steps is subsumed by the cost of answering OMQs.

Theorem 5.1. $\mathcal{L}$-COND-ANSWERS *is:*

- NP-*complete when $\mathcal{L}$ is DL-Lite$_\mathcal{R}$ [CDL+07], DL-Lite$_\mathcal{A}^{++(\text{rec-safe})}$ (cf. Theorem 3.7), DL-Lite$_\mathcal{A}^{++(\text{non-rec})}$ (cf. Theorem 3.4), and $\mathcal{EL}^\bot$ [Ros07],*
- ExpTime-*complete when $\mathcal{L}$ is $\mathcal{ELI}$, Horn-$\mathcal{SHIQ}$ [BO15] and Horn-$\mathcal{SHOIQ}$ [ORS11],*
- 2-ExpTime-*complete when $\mathcal{L}$ is $\mathcal{SHIQ}$ [GLHS08], $\mathcal{SHOQ}$ [GHS08].*

If we consider the ABox as the only component of the input, then we can consider $Q = (\mathcal{O}, q, \mathcal{H})$ to be an arbitrary but fixed AOMQ in $\mathcal{L}$. Then,

$\mathcal{L}$-COND-ANSWERS$(\mathcal{O}, q, \mathcal{H})$	
Input:	ABox $\mathcal{A}$, $\vec{a}$ a tuple of constants and $\mathcal{E}$ a set of possible facts w.r.t. $(\mathcal{A}, \mathcal{O})$
Question:	Is $(\vec{a}, \mathcal{E}) \in cans(\mathcal{Q}, \mathcal{A})$?

The latter problem determines the data complexity of answering AOMQs, and we show next that for ontology languages in $DL\text{-}Lite_{\mathcal{A}}^{++}$ which enjoy the FO-rewriting property we can rely on a rewriting approach to construct conditional answers.

5.3 Rewriting AOMQs

In this section we provide a *rewriting algorithm* for AOMQs $\mathcal{Q} = (q(\vec{x}), \mathcal{O}, \mathcal{H})$ where $\mathcal{O}$ is an ontology in a language that enjoys the FO-rewritabiity property. The algorithm takes such a $\mathcal{Q}$ as input, and it outputs a set of FO-queries $Rew(\mathcal{Q})$, whose answers over any ABox $\mathcal{A}$ are in one-to-one correspondence with the conditional answers of $\mathcal{Q}$ over $\mathcal{A}$.

Given an ontology $\mathcal{O}$, we denote by $neg(\mathcal{O})$ the set of all disjointness and functionality axioms in $\mathcal{O}$, and by $pos(\mathcal{O})$ the set $\mathcal{O} \setminus neg(\mathcal{O})$. Our procedure to obtain the full FO-rewriting for AOMQs has three steps:

(1) We rewrite q w.r.t. $pos(\mathcal{O})$.
(2) Each resulting query is rewritten w.r.t. $\overline{\mathcal{H}}$.
(3) Finally, we take into account $neg(\mathcal{O})$ to rule out ground assumptive patterns which can lead to inconsistency.

5.3.1 Rewriting w.r.t. $pos(\mathcal{O})$

An important observation is that for FO-rewritable ontology languages we can drop the ontology by simply rewriting the query and the resulting FO-query preserves the set of conditional answers.

Lemma 5.1. *Let $(q(\vec{x}), \mathcal{O})$ be an OMQ in $\mathcal{L}$ that enjoys the FO-rewritability property and let $\phi_{\mathcal{O}}(\vec{x})$ be the FO-rewriting of $q(\vec{x})$ w.r.t. $\mathcal{O}$.*

Then, for any ABox $\mathcal{A}$ that is consistent w.r.t. $\mathcal{O}$ and for any set of assumption patterns $\mathcal{H}$ we have:

$$cans((q(\vec{x}), \mathcal{O}, \mathcal{H}), \mathcal{A}) = cans((\phi_{\mathcal{O}}(\vec{x}), \emptyset, \mathcal{H}), \mathcal{A}).$$

Proof. We consider an arbitrary ABox $\mathcal{A}$ and set of assumption patterns $\mathcal{H}$. Let $(\vec{a}, \mathcal{E})$ be an arbitrary conditional answer of AOMQ $\mathcal{Q} = (q(\vec{x}), \mathcal{O}, \mathcal{H})$ over $\mathcal{A}$ and we want to show that it is also a conditional answer for $\mathcal{Q}' = (\phi_{\mathcal{O}}(\vec{x}), \emptyset, \mathcal{H})$. By definition we have that there exists a substitution π such that: (i) $\pi(\vec{x}) = \vec{a}$, (ii) $(\mathcal{O}, \mathcal{E} \cup \mathcal{A}) \models q\pi$, (iii) $\mathcal{E} = \mathcal{H}'\pi$ for some $\mathcal{H}' \subseteq \overline{\mathcal{H}}$. From the first two conditions we get that $\vec{a}$ is a certain answer of $(\mathcal{O}, q(\vec{x}))$ over $\mathcal{E} \cup \mathcal{A}$. Therefore, $\vec{a}$ is a certain answer of $(\emptyset, \phi_{\mathcal{O}}(\vec{x}))$ over $\mathcal{E} \cup \mathcal{A}$ hence $(\vec{a}, \mathcal{E})$ is a conditional answer for $(\phi_{\mathcal{O}}, \emptyset, \mathcal{H})$ over $\mathcal{A}$.

For the other direction, let $(\vec{a}, \mathcal{E})$ be an arbitrary conditional answer of $\mathcal{Q}' = (\phi_{\mathcal{O}}(\vec{x}), \emptyset, \mathcal{H})$ and we want to show that it is a conditional answer of $\mathcal{Q}$ over $\mathcal{A}$. Again, by definition we have that there exists a substitution π' such that: (i) $\pi'(\vec{x}) = \vec{a}$, (ii) $\mathcal{A} \cup \mathcal{E} \vDash \phi_{\mathcal{O}}\pi'$, (iii) $\mathcal{E} = \mathcal{H}'\pi'$, for some $\mathcal{H}' \subseteq \overline{\mathcal{H}}$. Firstly, since $\phi_{\mathcal{O}}$ is a FO-rewriting of $(\mathcal{O}, q(\vec{x}))$ we have that $vars(q) \subseteq vars(\phi_{\mathcal{O}})$. Secondly, from (i),(ii) we have that $\vec{a}$ is an answer to $\phi_{\mathcal{O}}(\vec{a})$ over $\mathcal{A} \cup \mathcal{E}$. From the fact that each answer of $\phi_{\mathcal{O}}$ is a certain answer of $(q, \mathcal{O})$, we obtain that and $\mathcal{O}, \mathcal{A} \cup \mathcal{E} \vDash q\pi'$, hence $(\vec{a}, \mathcal{E}) \in cans((q(\vec{x}), \mathcal{O}, \mathcal{H}), \mathcal{A})$. □

5.3.2 Rewriting w.r.t. Assumption Patterns

In this section we propose a technique to rewrite a given FO-query w.r.t. a set of assumption patterns such that each match of the rewriting stands for a conditional answer. The core idea is to identify subformulas which are made true by the assumptions, and drop them while carefully handling the free and join variables.

Definition 5.7. *Given an $\mathcal{ELI}$ atom α we can transform it into a set of unary and binary atoms using the following recursive function:*

- *If α is $A(t)$ or $r(t, t')$ then* $\exp_\alpha = \{\alpha\}$.
- *If α is $r^-(t, t')$ then* $\exp_\alpha = \{r(t', t)\}$.
- *If α of the form $(C_1 \sqcap C_2)(t)$ then* $\exp_\alpha = \exp_{C_1(t)} \cup \exp_{C_2(t)}$.
- *If α is of the form $(\exists p.C)(t)$ then* $\exp_\alpha = \exp_{p(t,y^\alpha)} \cup \exp_{C(y^\alpha)}$, *where y^α is a fresh variable.*

The expansion *of a set $\mathcal{H}$ of assumption patterns is defined as:* $\exp(\mathcal{H}) = \bigcup_{\alpha \in \mathcal{H}} \exp_\alpha$.

Intuitively, the expansion allows us to view a given set of assumption patterns as simply a set of query atoms. Note that the set of implicit assumption patterns $\overline{\mathcal{H}} \supseteq \exp(\mathcal{H})$ and that $\exp(\overline{\mathcal{H}})$ is identical to $\exp(\mathcal{H})$, modulo variable renamings.

From the definition of conditional answers in presence of assumption patterns, the set of possible assertions $\mathcal{E}$ that are part of conditional answers, represent groundings of the implicit assumption patterns (i.e., of some set of atoms in $\overline{\mathcal{H}}$). The rewriting idea is to unify some implicit assumption patterns with some part of the query, and make the variable equalities explicit in the rewriting in order to construct each $\mathcal{E}$ from the answer of the rewriting and $\overline{\mathcal{H}}$.

Given two sets of query atoms Γ_1 and Γ_2, a *unifier* of Γ_1 and Γ_2 is a partial mapping θ from $vars(\Gamma_1 \cup \Gamma_2)$ to $term(\Gamma_1 \cup \Gamma_2)$ such that $\Gamma_1\theta = \Gamma_2\theta$.

Let $q(\vec{x})$ be a CQ and Γ a subset of atoms in q. We denote by q_Γ the sub-formula of q containing all atoms in Γ and by $q_{\overline{\Gamma}}$ the sub-formula containing all the atoms not in Γ. We denote by $keep(q, \Gamma)$ the set of variables in $vars(\Gamma)$ that are in $\vec{x} \cup (vars(q) \setminus vars(\Gamma))$.

In order to ensure that the query entailment is correctly captured by the obtained rewriting, we need to disallow unifying free or join variables to existential variables.

Definition 5.8 (Rewriting w.r.t. $\mathcal{H}$)**.** *Let $\mathcal{H}$ be a set of assumption patterns and $q(\vec{x})$ a CQ. We write $q(\vec{x}) \rightsquigarrow_{\mathcal{H}} q'(\vec{x'})$ and call q' a* rewriting of q w.r.t. $\mathcal{H}$ *if it is obtained by choosing*

(i) a subset Γ *of the atoms of* q*, and a subset* Γ' *of* $\exp(\mathcal{H})$

(ii) a unifier h *of* Γ *and* Γ' *that verifies the following conditions: each variable in* $keep(q, \Gamma)$ *is not mapped to fresh variables in* $\exp(\mathcal{H})$*, and each fresh variable in* $\exp(\mathcal{H})$ *is not mapped to a variable in* $keep(q, \Gamma)$*,*

and then doing the following transformations on q*:*

1. *Discard* q_Γ *from* q
2. *Add* $q_h = \bigwedge\{x = h(x) \mid x \in keep(q, \Gamma) \cup vars(\mathcal{H})\}$.
3. *Drop existential quantification for each* $x \in vars(\mathcal{H})$.

The rewritten queries may have more free variables than the original q. After each rule application the resulting q' has $vars(\mathcal{H})$ as free variables, additionally to the original $\vec{x}$. These additional free variables and the equality atoms will allow us to read from each (ordinary) answer to some rewritten query, the variable assignment for $\mathcal{H}$ that gives the corresponding conditional answer.

Example 5.2. *Let us revisit the Example 5.1. The following query*

$$q_1(y) \leftarrow \exists xzu\ \mathrm{location}(\mathsf{Anna}, x) \wedge \mathrm{line}(x, y) \wedge \mathrm{location}(u, z) \wedge \mathrm{SevereMCI}(u) \wedge \mathrm{location}(\mathsf{Anna\_home}, z), \mathrm{line}(z, y)$$

is contained in the rewriting of q *w.r.t.* $\mathcal{O}$ *(obtained by firstly applying axiom* $\exists \mathrm{line}^{-}.\mathrm{ClosedStation} \sqsubseteq \mathrm{DisruptedLine}$ *on atom* $\mathrm{DisruptedLine}(y)$ *in* q *and, after a unification step, the axiom* $\exists \mathrm{location}^{-}.\mathrm{SevereMCI} \sqsubseteq \mathrm{ClosedStation}$ *is applied).*

If we consider $\mathcal{H} = \{\exists \mathrm{location}^{-}.\mathrm{SevereMCI}(z)\}$ *with* $\exp(\mathcal{H}) = \{\mathrm{location}(u', z), \mathrm{SevereMCI}(u')\}$*, a rewriting of* q_1 *w.r.t.* $\mathcal{H}$ *can be obtained by choosing:*

$$\Gamma = \{\mathrm{location}(u, z), \mathrm{SevereMCI}(u)\}, \quad \Gamma' = \{\mathrm{location}(u', z), \mathrm{SevereMCI}(u')\} \text{ and}$$
$$h = \{u' \mapsto u, z \mapsto z\}.$$

The conditions (ii) of Definition 5.8 are satisfied since $keep(\Gamma, q_1) = \{z\}$ *and* h *maps* z *to itself, and the fresh variable* u' *is mapped to existential variable* u*. The result of the transformations 1-3 as described in Definition 5.8 yields the following rewriting:*

$$q_1'(y, z) \leftarrow \exists xu\ \mathrm{location}(\mathsf{Anna}, x) \wedge \mathrm{line}(x, y) \wedge \mathrm{location}(\mathsf{Anna\_home}, z) \wedge \mathrm{line}(z, y) \wedge z = z.$$

The purpose of the equality atoms is to capture the variable assignment for $\mathcal{H}$ *from each match of the rewriting. For instance, given the match* π *of* $q_1'(y, z)$ *over the ABox* $\mathcal{A}$ *in Example 5.1 that maps* y *to* U_1 *and* z *to* $\mathsf{Praterstern}$*, we can construct the following conditional answer, by simply applying* $\pi \circ h$ *to* $\mathcal{H}$*. In this case, we get* $(\mathsf{U}_1, \exists \mathrm{location}^{-}.\mathrm{SevereMCI}(\mathsf{Praterstern}))$ *as conditional answer to* q_1*, hence also to* q*.*

The motivation behind applying implicit assumption patterns is to find more specific ground assumptions. For example, if we apply the implicit assumption patterns $\mathcal{H}' = \{\mathrm{SevereMCI}(u')\}$*, which is more specific, to rewrite* q_1 *we obtain:*

$$q_1''(y, u') \leftarrow \exists xzu\ \mathrm{location}(\mathsf{Anna}, x) \wedge \mathrm{line}(x, y) \wedge \mathrm{location}(u, z) \wedge \mathrm{location}(\mathsf{Anna\_home}, z) \wedge \mathrm{line}(z, y) \wedge \wedge u = u'.$$

which also gives U_1*, hence obtaining* $(\mathsf{U}_1, \text{SevereMCI}(\mathsf{e}))$ *as conditional answer.*

As shown in the above example, the use of $\overline{\mathcal{H}}$ is to automatically construct multiple related assumption patterns that minimizes the input of the user while maximizing the variety of ABox completions that can be relevant for the given OMQ. The full rewriting of a query w.r.t. a set of assumption patterns $\mathcal{H}$ is therefore obtained by rewriting w.r.t. each subset $\mathcal{H}'$ of $\overline{\mathcal{H}}$.

Definition 5.9 (Complete rewriting w.r.t. $\mathcal{H}$)**.** *Let q be a CQ and $\mathcal{H}$ a set of assumption patterns. The* complete rewriting of q w.r.t. $\mathcal{H}$ *is*

$$rew_{\mathcal{H}}(q) = \bigcup_{\mathcal{H}' \subseteq \overline{\mathcal{H}}} \{q' \mid q \rightsquigarrow_{\mathcal{H}'} q'\}.$$

The key to the correctness is that the unifier h for Γ and Γ' exists iff the atoms of Γ are made true in a grounding of a subset of $\overline{\mathcal{H}}$. Hence each rewriting step drops precisely atoms that would be made true by some grounding of a subset of $\overline{H}$. We show that the rewriting is correct by using the following claim:

Claim 8. *Let $\mathcal{Q} = (q(\vec{x}), \emptyset, \mathcal{H})$ be an AOMQ, $\mathcal{A}$ an ABox and $\mathcal{E}$ a set of possible assertions given $\mathcal{A}$. For any partial mapping $\pi : vars(q) \cup (vars(\overline{\mathcal{H}})) \mapsto cst(\mathcal{A})$ such that $\mathcal{E}$ is a π-instantiation of $\mathcal{H}'$ for some $\mathcal{H}' \subseteq \overline{\mathcal{H}}$, we have that the following are equivalent:*

1) π is a conditional match of q given $(\mathcal{A}, \mathcal{E})$, and
2) there exist unifier h of $\Gamma \subseteq q$ and $\Gamma' \subseteq \exp(\mathcal{H}')$ satisfying (ii) in Definition 5.8 and π is a match over $\mathcal{A}$ for query $q'(\vec{x})$ obtained by applying steps 1-3 in Definition 5.8.

Proof. Given a query q and a set of assumption patterns $\mathcal{H}$, a rewriting of q w.r.t. $\mathcal{H}$ has the following form:

$$q'(\vec{x} \cup vars(\mathcal{H})) \leftarrow q_{\overline{\Gamma}} \wedge q_h$$

where $q_{\overline{\Gamma}}$ is the query obtained by discarding q_Γ from q (step 1 in Definition 5.8), and q_h are the equality atoms added according to the unifier h (step 2). Using this notation, we proceed with the proof of the claim.

2) ⇒ 1): Let π be a match of an arbitrary q' such that $q \rightsquigarrow_{\mathcal{H}'} q'$ over $\mathcal{A}$. This means that there exist: $\Gamma \subseteq q, \Gamma' \subseteq \exp(\mathcal{H}')$ and unifier h complying with conditions in Definition 5.8. W.l.o.g., we assume that π is defined over all variables in q'. Since each join or free variable must be mapped by h to a variable in $\mathcal{H}'$, each existential variable in Γ' cannot be mapped to a free variable in q, and π matches such variables according to h, we obtain that $\mathcal{H}'\pi \vDash q_\Gamma\pi$. Since π is also a match of $q_{\overline{\Gamma}}$ over $\mathcal{A}$ we obtain that $\mathcal{A} \cup \mathcal{H}'\pi \vDash q\pi$. Clearly $\mathcal{H}'\pi$ is a π-instantiation of $\mathcal{H}$.

1) ⇒ 2). Let π be an arbitrary conditional match of q such that $\mathcal{E}$ is a π-instantiation of $\mathcal{H}' \subseteq \overline{H}$. By definition, we have that $\mathcal{A} \cup \mathcal{E} \vDash q\pi$ and $\mathcal{E} = \mathcal{H}'\pi$. This implies that there exists a set of atoms Γ in q such that for sub-query q_Γ we have $\mathcal{E} \vDash q_\Gamma\pi$ and for the remainder part $q_{\overline{\Gamma}}$ we have that $\mathcal{A} \vDash q_{\overline{\Gamma}}\pi$. W.l.o.g. for $\overline{\Gamma}$ denoting the set of atoms in $q_{\overline{\Gamma}}$, we assume that $\overline{\Gamma}\pi \subseteq \mathcal{A}$. It follows that $\mathcal{H}'\pi \vDash q_\Gamma\pi$ and let Γ' be the minimal set of atoms in $\exp(\mathcal{H}')$ such that $\Gamma'\pi \vDash q_{\overline{\Gamma}}\pi$.

Then there exists a mapping θ that extends π by mapping remaining variables in Γ to terms in $\Gamma'\pi$ and fresh variables in Γ' to themselves such that $\Gamma\theta = \Gamma'\theta$. Based on the set of equalities $E = \{x = y \mid \theta(x) = \theta(y)\}$, using the same procedure to obtain the most general unifier, we can construct a set of variable renamings R by iteratively taking some $x = y$ and: (1) adding $x \mapsto y$ to R, (2) replace all occurrences of x with y in E, (3) remove atoms of the form $x = x$ from E. Based on R we can construct a substitution h such that $h(x) = y$ if $x \mapsto y \in R$.

We show next that h is a unifier of Γ and Γ', and satisfies conditions in Definition 5.8. We proceed with showing that $\theta = \pi \circ h$: for some variable x let $h(x) = y$. Then there exists a series of equality atoms $x = x_1 = \cdots = x_n = y$ in E hence $\theta(x) = \theta(y)$ and since θ extends π to all variables in $\Gamma \cup \Gamma'$, we obtain: $\pi(h(x)) = \pi(y) = \theta(y) = \theta(x)$, thus clearly $\theta = \pi \circ h$. Moreover, since $\Gamma\theta = \Gamma'\theta$, we obtain that also $\Gamma h = \Gamma' h$, thus h is a unifier of Γ and Γ'.

Next, we show that h does not map (i) any join or free variable in q to an existential one in $\exp(\mathcal{H}')$ or (ii) fresh variables in $\exp(\mathcal{H}')$ to free variables in q. We proof by contradiction: assume that for some $x \in \mathit{keep}(\Gamma, q)$ we have $h(x) = y$ with y fresh variable in $\exp(\mathcal{H}')$. By definition it must be that $\theta(x) = \theta(y) = y$, thus we can construct a model $\mathcal{I}$ of $\mathcal{A} \cup \mathcal{H}'\pi$ by mapping y to any arbitrary constant in the domains such that $\mathcal{I} \not\models q_{\overline{\Gamma}}\pi$, hence a contradiction with the fact that π is a conditional match. Similarly, for (ii) let x be a free variable in $\exp(\mathcal{H}')$ such that $h(x) = y$ and $y \in \mathit{keep}(\Gamma, q)$. Since $\overline{\Gamma}\pi \subseteq \mathcal{A}$ and since $y \in \mathit{keep}$ it occurs in Γ and we can easily see that $\mathcal{H}'\pi \not\models q_\Gamma\pi$ which is a direct contradiction, therefore we conclude that h satisfies the conditions in Definition 5.8. Lastly, from $\overline{\Gamma}\pi \subseteq \mathcal{A}$ and properties of h it follows that π is indeed a match of the rewriting denoted by q'. □

The following Lemma states that the rewriting w.r.t. assumption patterns is complete.

Lemma 5.2. *Let $\mathcal{Q} = (q(\vec{x}), \emptyset, \mathcal{H})$ be an AOMQ. For every ABox $\mathcal{A}$ and conditional answer $(\vec{a}, \mathcal{E})$ of $\mathcal{Q}$ over $\mathcal{A}$, there exists some $\mathcal{H}' \subseteq \overline{\mathcal{H}}$, with $\mathit{vars}(\mathcal{H}') = \vec{y}$ and $q(\vec{x}) \rightsquigarrow_{\mathcal{H}'} q'(\vec{x} \cup \vec{y})$ such that $(\vec{a} \cup \vec{b}) \in \mathit{cert}(q'(\vec{x} \cup \vec{y}), \emptyset, \mathcal{A})$ and $\mathcal{E} = \mathcal{H}'(\vec{b})$.*

Proof. We take an arbitrary ABox $\mathcal{A}$ and arbitrary conditional answer $(\vec{a}, \mathcal{E})$ of $\mathcal{Q}$ over $\mathcal{A}$. By definition, there exists some $\mathcal{H}' \subseteq \overline{\mathcal{H}}$ and a partial mapping $\pi : \mathit{vars}(q) \cup \mathit{vars}(\mathcal{H}') \mapsto \mathit{cst}(\mathcal{A})$ such that $\pi(\vec{x}) = \vec{a}$, $\mathcal{A} \cup \mathcal{E} \models q\pi$ and $\mathcal{E} = \mathcal{H}'\pi$ and suppose that $\pi(\vec{y}) = \vec{b}$. From Claim 8 it follows that there exists a unifier h of some $\Gamma \subseteq q$ and some $\Gamma' \subseteq \exp(\mathcal{H}')$ that complies with variable conditions according to Definition 5.8 and π is a match for the obtained rewriting q'. Since $\mathit{vars}(\mathcal{H}')$ are free in q' be obtain that $\pi(\vec{x} \cup \vec{y}) = (\vec{a} \cup \vec{b})$ and since π is a match of q' over $\mathcal{A}$, we obtain $(\vec{a} \cup \vec{b}) \in \mathit{cert}(q'(\vec{x} \cup \vec{y}), \emptyset, \mathcal{A})$. □

The remaining step in proving the correctness of the rewriting is to show that it is a sound procedure. The following Lemma states exactly that.

Lemma 5.3. *Let $\mathcal{Q} = (q(\vec{x}), \emptyset, \mathcal{H})$ be an AOMQ. For any ABox $\mathcal{A}$, $\mathcal{H}' \subseteq \overline{\mathcal{H}}$ and $q(\vec{x}) \rightsquigarrow_{\mathcal{H}'} q'(\vec{x} \cup \vec{y})$, if $(\vec{a} \cup \vec{b}) \in \mathit{cert}(q'(\vec{x} \cup \vec{y}), \emptyset, \mathcal{A})$ then $(\vec{a}, \mathcal{H}'(\vec{b})) \in \mathit{cans}(\mathcal{Q}, \mathcal{A})$.*

Proof. Let $\mathcal{A}$ be an arbitrary ABox, $\mathcal{H}' \subseteq \overline{\mathcal{H}}$ and $q'(\vec{x} \cup \vec{y})$ be a rewriting of q w.r.t. $\mathcal{H}'$, where $vars(\mathcal{H}') = \vec{y}$ and take an arbitrary $(\vec{a} \cup \vec{b}) \in cert(q'(\vec{x} \cup \vec{y}), \emptyset, \mathcal{A})$. Then there is some mapping $\pi : vars(q') \mapsto cst(\mathcal{A})$ such that $\pi(\vec{x}) = \vec{a}$, $\pi(\vec{y}) = \vec{b}$ and $\mathcal{A} \vDash q'\pi$. Again, from Claim 8 we obtain that π is a conditional match of q given $(\mathcal{A}, \mathcal{H}'\pi)$. Therefore $(\vec{a}, \mathcal{H}'\pi) \in cans(\mathcal{Q}, \mathcal{A})$. □

The following definition captures the full rewriting of a given AOMQ, when the ontology does not contain any disjointness or functionality axiom since they can restrict the number of consistent ground assumptions (a case which is captured in a later section). The definition captures any AOMQ such that the ontology belongs to a language $\mathcal{L}$ that is FO-rewritable and which has UCQ as target rewriting query language. In this category falls *DL-Lite*$_{\mathcal{A}}$ and any of the FO-rewritable *DL-Lite*$_{\mathcal{A}}^{++}$ extensions.

Definition 5.10 (Perfect rewriting of AOMQ over $pos(\mathcal{O})$)**.** *Let $\mathcal{Q} = (q(\vec{x}), pos(\mathcal{O}), \mathcal{H})$ be an AOMQ where $\mathcal{O}$ is a FO-rewritable DL-Lite$_{\mathcal{A}}^{++}$ ontology. Then its* perfect rewriting consists of *the following set of queries*

$$Rew(\mathcal{Q}) = \{q'' \mid q'' \in rew_{\mathcal{H}}(q') \text{ and } q' \in rew_{\mathcal{O}}(q)\}.$$

Note that the queries in $Rew(Q)$ may not all share the same answer variables, hence we write it as a set of CQs rather than a UCQ.

From Lemmas 5.1 to 5.3, evaluating $Rew(Q)$ over the ABox alone is sound and complete for AOMQs over positive ontologies and it is a special case of Definition 5.14 below. We provide the full correctness theorem after we discuss the incorporation of closed predicates and of $neg(\mathcal{O})$.

5.4 AOMQs with Closed Predicates

The usual open-world semantics of ontologies is not suitable when part of the data is known to be complete, and there are many application domains which require the combination of both open and closed world assumptions. For example the information about the transportation system in a city is in general known to the users, such as the metro lines in Vienna, while concepts like "functioning or out of service metro stations" are more prone to updates and therefore incomplete to the users.

In the literature, a successful mechanism for combining open and closed-world semantics is to declare some predicates as *closed* and to strengthen the usual certain answer semantics of OMQs by considering only particular models that interpret each marked predicate exactly as described in the data. To be more precise, if for a given set of predicate symbols Σ (i.e., a set of concept, role or feature names) we know the ABox $\mathcal{A}$ contains *all* the (tuples of) constants for each predicate symbol in Σ then, to answer OMQs, we rely only on models $\mathcal{I}$ such that they employ the standard name assumption and that for each $P \in \Sigma$ we have that $P^{\mathcal{I}} = \{\vec{c} \mid P(\vec{c}) \in \mathcal{A}\}$.

Unfortunately, allowing closed predicates in the ontology comes with a cost: *DL-Lite*-CERTAIN ANSWERS$(\mathcal{O}, q)$ is CONP-hard, and thus not FO-rewritable [LSW13]. In this section, we extend AOMQs to allow declarations of closed predicates in a different way while remarkably preserving the FO-rewritability for ontology languages which extend *DL-Lite*.

5.4.1 OMQs with Closed Predicates (OMQC)

In this subsection, the preliminaries for evaluating OMQs in presence of closed predicates are presented.

Definition 5.11 (OMQCs)**.** *A set of closed predicates Σ, is a subset of predicate symbols,i.e., $\Sigma \subseteq \mathbf{C} \cup \mathbf{R} \cup \mathbf{F}$. An interpretation $\mathcal{I}$ is a* Σ-model *of $(\mathcal{O}, \mathcal{A})$, written $\mathcal{I} \models_\Sigma (\mathcal{O}, \mathcal{A})$, if $\mathcal{I} \models (\mathcal{O}, \mathcal{A})$ and additionally $\vec{a} \in P^\mathcal{I}$ implies $P(\vec{a}) \in \mathcal{A}$ for all $P \in \Sigma$. We say that $\mathcal{A}$ is Σ-consistent w.r.t. $\mathcal{O}$ if there exists a Σ-model of $(\mathcal{O}, \mathcal{A})$. An* OMQ with closed predicates (OMQC) *is a tuple $\mathcal{Q} = (q(\vec{x}), \mathcal{O}, \Sigma)$ where $(q, \mathcal{O})$ is an OMQ.*

The notion of certain answers for OMQCs is lifted in the usual way from OMQs and the notation $cert(\mathcal{Q}, \mathcal{A})$ remains valid for OMQCs.

Definition 5.12 (OMQCs certain answers)**.** *A* certain answer *for an OMQC $\mathcal{Q}$ is a tuple $\vec{a}$ such that $\vec{a}$ is an answer to q in every Σ-model of $(\mathcal{O}, \mathcal{A})$. The set $cert(\mathcal{Q}, \mathcal{A})$ denotes the set of all certain answers of $\mathcal{Q}$ over $\mathcal{A}$.*

Given a certain answer $\vec{a}$, we call a substitution π such that $\pi(\vec{x}) = \vec{a}$ and $(\mathcal{O}, \mathcal{A}) \models_\Sigma q\pi$ a Σ-match of $\vec{a}$ given $(\mathcal{Q}, \mathcal{A})$.

To exemplify the use of OMQCs, let us consider a minor adaptation of Example 1 in [LSW15], which nicely illustrate the need for allowing closed predicates.

Example 5.3. *Suppose the ABox $\mathcal{A}$ contains the following facts:*

$$\text{SkodaModel}(\mathsf{sm17}),\ \text{SkodaEngine}(\mathsf{se1}),\ \text{SkodaEngine}(\mathsf{se2}),\ \text{DieselEngine}(\mathsf{se1}),\ \text{PetrolEngine}(\mathsf{se2}),$$

and we know that there is no additional Skoda model other than sm17 *and no other Skoda engine other than* se1 *and* se2*. Hence we consider* SkodaModel *and* SkodaEngine *as closed, i.e.,* $\Sigma = \{\text{SkodaModel}, \text{SkodaEngine}\}$.

Moreover, in the ontology $\mathcal{O}$ we have the following axioms:

$$\text{DieselEngine} \sqsubseteq \text{ICEngine}, \qquad \text{PetrolEngine} \sqsubseteq \text{ICEngine},$$
$$\text{SkodaModel} \sqsubseteq \exists \text{hasEngine}.\text{SkodaEngine}.$$

Considering the query

$$q(x) \leftarrow \exists y.\text{SkodaModel}(x) \wedge \text{hasEngine}(x, y) \wedge \text{ICEngine}(y)$$

from first two ontology axioms we know that each Diesel and petrol engine is a internal combustion (IC) engine, hence also se1 *and* se2 *are implicitly IC engines. Using the last axiom, each Skoda model must have a Skoda engine and since it has to be precisely one of* se1 *or* se2*, we deduce, that* sm17 *is a certain answer to the OMQC $(q, \mathcal{O}, \Sigma)$. Thus, a desirable additional answer is retrieved which could not be obtained under the classical certain answer semantics of OMQs.*

As previously stated, the OMQ in this example is no longer FO-rewritable. The underlying problem is the occurrence of closed predicate SkodaEngine on the right-hand-side of the last axiom, which creates a form of non-determinism, thus an increase in the complexity of reasoning. This motivates the extension of our formalism to include close predicates in the assumption patterns and redefine the semantics of answering AOMQs to take into account the use of closed-world semantics. The main advantage of AOMQs with closed predicates is that they preserve FO-rewritability.

5.4.2 AOMQs with Closed Predicates

In this subsection, we introduce AOMQs with closed predicates and present a rewriting technique that can be used to compute conditional answers, under the clopen world assumption, without the complexity increase. In our formalism, we disallow closed predicates in the ontology, for reasons previously discussed, and allow them in the assumption patterns. The difference in this case is that we cannot take arbitrary groundings $\mathcal{E}$ of $\mathcal{H}$ in conditional answers, but only ground assumptions that are valid w.r.t. the semantics of closed predicates.

Definition 5.13 (AOMQ with closed predicates (AOMQC))**.** *An* AOMQ with closed predicates (AOMQC) *is a tuple* $\mathcal{Q} = (q, \mathcal{H}, \mathcal{O}, \Sigma)$*, where* $(q, \mathcal{H}, \mathcal{O})$ *is an AOMQ and* Σ *is a set of closed predicates, disjoint from sign*$(\mathcal{O})$*.*

Let $\mathcal{A}$ *be an ABox. A pair* $(\vec{a}, \mathcal{E})$ *of a tuple of constants* $\vec{a}$ *and a set of possible assertions* $\mathcal{E}$ *w.r.t.* $(\mathcal{O}, \mathcal{A})$*, is called a* conditional answer to $\mathcal{Q}$ over $\mathcal{A}$ *if (i) there exists a* Σ*-model of* $(\mathcal{O}, \mathcal{A} \cup \mathcal{E})$*, and (ii) a substitution* π *that is a* Σ*-match of* $\vec{a}$ *given* $(\mathcal{Q}, \mathcal{A} \cup \mathcal{E})$*, and* $\mathcal{E}$ *is a* π*-instantiation of* $\mathcal{H}$*.*

Conditional answers of AOMQCs can capture typical scenarios where closed predicates are desirable for OMQs. Based on Example 5.3, we can discard the problematic axiom SkodaModel $\sqsubseteq$ $\exists$hasEngine.SkodaEng from the ontology and rely on the assumption patterns

$$\mathcal{H} = \{\exists \text{hasEngine.SkodaEng}(x)\}$$

and the fact that they are applicable to atom SkodaModel(x) in q. Answering such AOMQC, yields (sm17, hasEngine.SkodaEng(sm17)) as a conditional answer, hence we obtain that sm17 is a Skoda model that has an IC engine, assuming that sm17 has a Skoda engine (which is known to be true).

This shows that we can model examples that motivate the use of closed predicates in the first place, and remarkably, as we show next, this formalism remains FO-rewritable for $\mathit{DL\text{-}Lite}_{\mathcal{A}}^{++(\mathit{non\text{-}rec})}$ and $\mathit{DL\text{-}Lite}_{\mathcal{A}}^{++(\mathit{rec\text{-}safe})}$.

Rewriting AOMQCs. The rewriting process is similar to the previous case: we first apply the rewriting w.r.t. the ontology, and then we rewrite w.r.t. the assumption patterns. This step, however, is different if $\mathcal{H}$ has closed predicates.

Intuitively, now we cannot assume arbitrary groundings of $\mathcal{H}$, but only groundings that respect the extensions of Σ in the input ABox. There is no canonical way to ground $\mathcal{H}$, and instead we

need to iterate over all groundings and exclude those that are not valid w.r.t. Σ and the input data. However, they are determined by the finite number of possible groundings of the closed predicates, so we use an FO-query with universal quantification to iterate over them.

We denote by $term_\Sigma(\mathcal{H})$ all terms $\vec{z} \in vars(\exp(\mathcal{H}))$ such that $P(\vec{z}) \in q$ and $P \in \Sigma$.

Definition 5.14 (Rewriting w.r.t. $(\mathcal{H}, \Sigma)$)**.** *Let $\mathcal{H}$ be a set of assumption patterns, Σ a set of closed predicates and q a CQ. A* rewriting of q w.r.t. $(\mathcal{H}, \Sigma)$*, written $q \rightsquigarrow_{\mathcal{H},\Sigma} \varphi$, has the following form:*

$$\forall \vec{u}.(q_\Sigma(\vec{u}) \rightarrow q'(\vec{x'}))$$

where $q_\Sigma(\vec{u})$ and $q'(\vec{x'}))$ are CQs obtained as follows:

(i) Choose a set of atoms Γ_1 of q, and a set of atoms Γ_2 of $\exp(\mathcal{H})$,
(ii) a unifier h of Γ_1 and Γ_2 that satisfies the following: no variable in $keep(q, \Gamma_1)$ is mapped to a fresh variables in $\exp(\mathcal{H}) \setminus \text{term}_\Sigma(\mathcal{H})$, and no fresh variable in $\exp(\mathcal{H}) \setminus \text{term}_\Sigma(\mathcal{H})$ is mapped to a variable in $keep(q, \Gamma_1)$.

Then q_Σ consists of $P_1(\vec{y}_1) \wedge \cdots \wedge P_k(\vec{y}_k)$, where $P_i(\vec{y}_i) \in \exp(\mathcal{H})$ such that $P_i \in \Sigma$ and $h(\vec{z}) = \vec{y}_1$ or $h(\vec{y}_i) = \vec{z}$ for $z \in vars(\Gamma_1)$, and q' is obtained by the following steps:

1. *Replace by $\top$ every atom in Γ_1.*
2. *Add $\bigwedge\{\vec{z} = \vec{y}_i \mid \vec{z} \in vars(\Gamma_1), P_i(\vec{y}_i)) \in \exp(\mathcal{H})$ and $h(\vec{z}) = \vec{y}_1$ or $h(\vec{y}_i) = \vec{z}\}$.*
3. *Add $\bigwedge\{x = h(x) \mid x \in keep(q, \Gamma_1) \cup vars(\mathcal{H})\}$.*
4. *Drop from the existential quantification all $x \in vars(\mathcal{H})$.*
5. *Rename each universally quantified x if $x = x \in q'$ and $x \in free(q) \cup vars(\mathcal{H})$, and apply the renaming to one part of the equality.*

The rewriting generalizes Definition 5.8, but the main difference is that we add to the universally quantified precondition each closed $P(x)$ in $\exp(\mathcal{H})$ that contributes the assumptions that make Γ_1 true.

Example 5.4. *Considering ABox in Example 5.1, it is reasonable to assume that the information regarding the stations and transportation lines is complete. Therefore we take $\Sigma = \{\text{Station}, \text{line}\}$ as a set of closed predicates. Moreover, we consider the following additional axioms:*

$$\text{location} \sqsubseteq \text{accessible} \qquad \text{accessible} \circ \text{connectedTo} \sqsubseteq \text{accessible}.$$

Suppose that we are interested in checking if Anna's home is accessible from the her current location, under the assumption that she is located near a station. Thus the following $\mathcal{H} = \{\exists \text{location}.\text{Station}(\text{Anna})\}$ and query:

$$q() \leftarrow \exists xy\ \text{location}(\text{Anna}, x) \wedge \text{accessible}(x, y) \wedge \text{location}(\text{Anna\_home}, y).$$

The rewriting of q w.r.t. $(\mathcal{H}, \Sigma)$ is:

$$\varphi() \leftarrow \forall u.(\text{Station}(u) \rightarrow u = x \wedge \text{accessible}(x, y) \wedge \text{location}(\text{Anna\_home}, y)).$$

Since the assumption does not point to any particular station, and under the closed-world semantics all stations are known, the empty tuple is an answer to the OMQ $(q, \mathcal{O}, \Sigma)$ since all existing stations have an accessible connection to Anna's home.

However, in order to actually obtain this conditional answer, the ontology rewriting is also required. Since the ontology rewriting is applicable only on the atom accessible(x, y) *and the ABox in the example is k-bounded for $k = 5$, we obtain that the full rewriting is equivalent to the following FO-query:*

$$\varphi'() \leftarrow \forall u.(\text{Station}(u) \rightarrow \exists x, x_0\ u = x \wedge (\text{location}(x, x_0) \vee \text{accessible}(x, x_0)) \wedge \bigvee_{i \in [0..5]} \exists x_1 \ldots x_i\ (\text{connectedTo}(x_0, x_1) \wedge \cdots \wedge \text{connectedTo}(x_{i-1}, x_i) \wedge \text{location}(\mathsf{Anna\_home}, x_i))).$$

The full rewriting, when evaluated over the ABox returns the empty tuple, since all stations are connected to the station where Anna's home is located.

We show next that the rewriting w.r.t. $(\mathcal{H}, \Sigma)$ is correct, analogously to the previous case. For that we rely on a finite set of minimal Σ-models of $\mathcal{A} \cup \mathcal{E}$ which are sufficient for obtaining the set of conditional answers.

Definition 5.15 (Representative Σ-models of $(\mathcal{A}, \mathcal{E})$)**.** *Let Σ be a set of closed predicates and $\mathcal{A}$ an ABox. Given a set of possible assertions $\mathcal{E}$ w.r.t. $\mathcal{A}$ let $\theta : vars(\exp(\mathcal{E})) \mapsto \mathbf{C}$ be a mapping such that for each $P(\vec{x}) \in \exp(\mathcal{E})$ we have that (i) $P(\vec{x})\theta \in \mathcal{A}$, if $P \in \Sigma$, (ii) $\vec{x}$ are mapped to fresh constants $\vec{c}_{\vec{x}}$, otherwise. For some θ, A* representative Σ-model *of $(\mathcal{A}, \mathcal{E})$ is the minimal interpretation $\mathcal{I}^{\theta}_{\mathcal{A}}$ that models $\mathcal{A} \cup \exp(\mathcal{E})\theta$.*

Given a pair $(\mathcal{A}, \mathcal{E})$, the set of representative Σ-models is finite given that each θ depends on the mapping of variables occurring in closed predicate atoms. In the following we show that each Σ-model of $(\mathcal{A}, \mathcal{E})$ is structurally *represented* by some $\mathcal{I}^{\theta}_{\mathcal{A}}$, hence we can rely on the set of representative models to compute conditional answers.

Claim 9. *Let $\mathcal{A}$ be an ABox and $\mathcal{E}$ a set of possible assertions w.r.t. $\mathcal{A}$. For each Σ-model $\mathcal{I}$ of $(\mathcal{A}, \mathcal{E})$ there exists a mapping θ such that $\mathcal{I}^{\theta}_{\mathcal{A}} \rhd \mathcal{I}$.*

Proof. Given $\mathcal{A}$ and $\mathcal{E}$, let $\mathcal{I}$ be an arbitrary Σ-model of $(\mathcal{A}, \mathcal{E})$. We construct θ as in Definition 5.15. Since both $\mathcal{I}$ and $\mathcal{I}^{\theta}_{\mathcal{A}}$ are Σ-models of $(\mathcal{A}, \mathcal{E})$, and by definition $\mathcal{I}^{\theta}_{\mathcal{A}}$ is minimal, we easily obtain that there exists a mapping $\ell : \Delta^{\mathcal{I}^{\theta}_{\mathcal{A}}} \mapsto \Delta^{\mathcal{I}}$ such that $\ell(c) = c$, for each $c \in cst(\mathcal{A})$ and for each additional fresh constant a there must be some b such that $\ell(a) = b$ and if $\mathcal{I}^{\theta}_{\mathcal{A}} \vDash A(a)$ then $\mathcal{I} \vDash A(b)$ and if $\mathcal{I}^{\theta}_{\mathcal{A}} \vDash p(a, d)$ then $\mathcal{I} \vDash p(b, e)$ and $\ell(d) = e$. Mapping ℓ is clearly a homomorphism, hence $\mathcal{I}^{\theta}_{\mathcal{A}} \rhd \mathcal{I}$. □

The next property follows immediately.

Corollary 2. *Let $\mathcal{Q}$ be an AOMQC. For each ABox $\mathcal{A}$ we have that $(\vec{a}, \mathcal{E}) \in cans(\mathcal{Q}, \mathcal{A})$ iff $\vec{a} \in ans(q, \mathcal{I}^{\theta}_{\mathcal{A}})$ for each minimal Σ-model $\mathcal{I}^{\theta}_{\mathcal{A}}$ of $(\mathcal{A}, \mathcal{E})$.*

We proceed now by showing that the rewriting is correct.

Lemma 5.4. *Let $\mathcal{Q} = (q(\vec{x}), \emptyset, \mathcal{H}, \Sigma)$ be an AOMQC and for $\mathcal{H}' \subseteq \overline{\mathcal{H}}$ let $\varphi(\vec{x} \cup \vec{x'})$ be a rewriting w.r.t. $(\mathcal{H}', \Sigma)$. For any ABox $\mathcal{A}$, if $(\vec{a} \cup \vec{b}) \in cert(\varphi(\vec{x} \cup \vec{x'}), \emptyset, \mathcal{A})$ then $(\vec{a}, \mathcal{H}'(\vec{b})) \in cans(\mathcal{Q}, \mathcal{A})$.*

Proof. Let $\mathcal{A}$ be an arbitrary ABox and let $(\vec{a} \cup \vec{b}) \in cert(\varphi(\vec{x} \cup \vec{x'}), \emptyset, \mathcal{A})$. By definition, we have that $\mathcal{A} \models \varphi(\vec{a} \cup \vec{b})$, hence $\mathcal{A} \models \forall \vec{u}.(q_\Sigma(\vec{u}) \rightarrow q'(\vec{a} \cup \vec{b}))$. This means that for each $\vec{c}$, a tuple of constants from $\mathcal{A}$ of the same arity as $\vec{u}$, there exists a substitution $\pi : \vec{u} \cup \vec{x} \cup \vec{x'} \mapsto cst(\mathcal{A})$ such that $\pi(\vec{u}) = \vec{c}$, $\pi(\vec{x}) = \vec{a}$ and $\pi(\vec{x'}) = \vec{b}$, and $\mathcal{A} \models (q_\Sigma(\vec{u}) \rightarrow q'(\vec{x} \cup \vec{x'}))\pi$. If $\mathcal{A} \not\models q_\Sigma(\vec{u})\pi$ then $\exp(q_\Sigma\pi) \subsetneq \mathcal{A}$, therefore $\mathcal{I}^\pi_\mathcal{A}$ is not a Σ-model of $\mathcal{A}$. It remains to show that for π such that $\mathcal{A} \models q_\Sigma\pi$ we have that $\mathcal{I}^\pi_\mathcal{A} \models q\pi$.

Let π be an arbitrary such mapping. Then, we can construct θ such that $\theta(\vec{u}) = \pi(\vec{u})$ (since π is defined on $\vec{u}$ as they appear in q'), and other variables in $\exp(\mathcal{H}')$ as in Definition 5.15. By applying Claim 8 on the fact that $\mathcal{A} \models q'\pi$ and that q' is obtained by the identification of a unifier h that respects the conditions in Definition 5.8, we obtain that $\mathcal{I}^\theta_\mathcal{A} \models q\pi$. Since π is arbitrarily chosen we conclude that for each representative $\mathcal{I}^\theta_\mathcal{A}$ we have that $\mathcal{I}^\theta_\mathcal{A} \models q(\vec{a})$. Lastly from Corollary 2 we obtain that $\vec{a} \in cans(\mathcal{Q}, \mathcal{A})$. □

We show next that the rewriting w.r.t. $(\mathcal{H}, \Sigma)$ is complete.

Lemma 5.5. *Let $\mathcal{Q} = (q(\vec{x}), \emptyset, \mathcal{H}, \Sigma)$ be an AOMQC. For any ABox $\mathcal{A}$, $\mathcal{H}' \subseteq \overline{\mathcal{H}}$ and tuples $\vec{a}$, $\vec{b}$ such that $(\vec{a}, \mathcal{H}'(\vec{b})) \in cans(\mathcal{Q}, \mathcal{A})$, then there is some φ such that $q(\vec{x}) \rightsquigarrow_{\mathcal{H},\Sigma} \varphi(\vec{x} \cup \vec{x'})$ and $(\vec{a} \cup \vec{b}) \in ans(\varphi, \mathcal{I}_\mathcal{A})$.*

Proof. We arbitrarily take ABox $\mathcal{A}$, $\mathcal{H}' \subseteq \overline{\mathcal{H}}$ and $\vec{a}, \vec{b}$ such that $(\vec{a}, \mathcal{H}'(\vec{b})) \in cans(\mathcal{Q}, \mathcal{A})$. From Corollary 2 we obtain that for each representative Σ-model $\mathcal{I}^\theta_\mathcal{A}$ of $(\mathcal{A}, \mathcal{H}'(\vec{b}))$ we have that $\mathcal{I}^\theta_\mathcal{A} \models q(\vec{a})$. It follows then that there is a mapping π such that $\pi(\vec{x}) = \vec{a}$ and $\pi(vars(\mathcal{H}')) = \vec{b}$ that is a conditional match of q w.r.t. $(\mathcal{A}, \exp(\mathcal{H}')\theta)$. Applying Claim 8 for each θ, we obtain that there exists h a unifier of Γ in q and Γ' in $\exp(\mathcal{H}')$ satisfying conditions in Definition 5.14 (since conditions in Definition 5.14 comply with those in Definition 5.8), therefore the rewriting φ as in Definition 5.14 exists.

Lastly, for mapping $\theta' = \pi \cup \theta$ we obtain that $\mathcal{A} \models q'\theta'$ for each θ as in Definition 5.15. Therefore, clearly $\mathcal{A} \models \varphi\pi$, hence $\vec{a} \in ans(\varphi, \mathcal{A})$. □

The full rewriting is the set of all FO-queries obtain by rewriting w.r.t. any subset of $\overline{H}$ and Σ, and, similarly to the case without closed predicates, the perfect rewriting is obtained by firstly rewriting w.r.t. $\mathcal{O}$ and then further w.r.t. $(\mathcal{H}, \Sigma)$.

Definition 5.16 (Perfect Rewriting of AOMQC over $pos(\mathcal{O})$)**.** *Let $\mathcal{Q} = (q(\vec{x}), pos(\mathcal{O}), \Sigma, \mathcal{H})$ be a FO-rewritable DL-Lite$^{++}_\mathcal{A}$ AOMQ with closed predicates. The* complete rewriting of q w.r.t. $(\mathcal{H}, \Sigma)$ *is*

$$rew_{\mathcal{H},\Sigma}(q) = \bigcup_{\mathcal{H}' \subseteq \overline{H}} \{\varphi(\vec{x'}) \mid q(\vec{x}) \rightsquigarrow_{\mathcal{H}',\Sigma} \varphi(\vec{x'})\}.$$

The perfect rewriting of $\mathcal{Q}$, *is*

$$Rew(\mathcal{Q}) = \{\varphi'(\vec{x'}) \mid \varphi'(\vec{x'}) \in rew_{\mathcal{H},\Sigma}(q') \text{ and } q'(\vec{x}) \in rew_{\mathcal{O}}(q)\}.$$

Correctness of this rewriting follows from Lemma 5.1 (using the fact that the closed predicates do not occur in the ontology), Lemma 5.5 and Lemma 5.4. We will state the theorem after the inclusion of disjointness and functionality axioms.

Let us return to the initial example of the section.

Example 5.5. *For $\mathcal{Q}$ of Example Example 5.3, the $rew_{\mathcal{O}}(\mathcal{Q})$ is equivalent to*

$$\varphi(x) \leftarrow \exists y.(\text{SkodaModel}(x) \wedge \text{hasEngine}(x, y) \wedge (\text{ICEngine}(y) \vee \text{PetrolEngine}(y) \vee \text{DieselEngine}(y)).$$

Then, $Rew(\mathcal{Q})$ is equivalent to:

$$\begin{aligned}\varphi'(x) \leftarrow &\text{SkodaModel}(x) \wedge \forall y'.(\text{SkodaEngine}(y') \rightarrow \exists y\ y = y' \wedge \\ &(\text{ICEngine}(y) \vee \text{PetrolEngine}(y) \vee \text{DieselEngine}(y))).\end{aligned}$$

Evaluating such query over the data, will allow us to easily construct the expected conditional answer.

5.5 Incorporating Disjointness and Functionality Axioms

Note that an AOMQ is a special case of an AOMQC and in this section we focus on adapting the rewriting of AOMQCs when negative axioms are part of the given ontology. The obtained rewriting can be applied also to the case without closed predicates.

The last step in obtaining the full rewriting of a given AOMQC is to apply the disjointness and functionality axioms for discarding tuples $(\vec{a}, \mathcal{E})$ for which $\mathcal{E} \cup \mathcal{A}$ is inconsistent w.r.t. $\mathcal{O}$.

In standard OMQs, to evaluate an OMQ $(q, \mathcal{O})$ one can usually check first if the given ABox $\mathcal{A}$ is consistent with $\mathcal{O}$, and then answer q assuming consistency of $\mathcal{A}$ w.r.t. $\mathcal{O}$. Unfortunately, this approach is not sufficient for AOMQCs, since we would need to verify consistency for each ABox extension. For that reason, we incorporate the inconsistency check as part of the rewritten query.

For that we will make use of inequality atoms $x \neq y$ for checking violations of functionality constraints. Queries with inequalities are in general problematic, however due to the restriction regarding relations participating in functionality assertions for *DL-Lite*$_{\mathcal{A}}^{++}$, violations of such axioms can occur only in the ABox (see upper-bound proof of Theorem 3.5). Furthermore, due to the SNA, query answering is a reliable procedure for checking consistency of ABoxes over *DL-Lite*$_{\mathcal{A}}^{++(non\text{-}rec)}$ and *DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$ ontologies.

Definition 5.17 (Inconsistency CQs)**.** *For any given disjointness or functional axiom α, we define the (Boolean) CQ $q_\alpha()$ as follows:*

- *if* $\alpha = \mathsf{disj}(A, A')$, *then* $q_\alpha() \leftarrow \exists x\ A(x) \wedge A'(x)$,
- *if* $\alpha = \mathsf{disj}(A, \exists r)$, *then* $q_\alpha() \leftarrow \exists xy\ A(x) \wedge r(x, y)$,
- *if* $\alpha = \mathsf{disj}(r_1, r_2)$, *then* $q_\alpha() \leftarrow \exists xy\ r_1(x, y) \wedge r_2(x, y)$,
- *if* $\alpha = (\mathsf{funct}\ \ p)$, *then* $q_\alpha() \leftarrow \exists xyz\ p(x, y) \wedge p(x, z) \wedge y \neq z$.

Let $\mathcal{O}$ be a FO-rewritable DL-Lite$_\mathcal{A}^{++}$ ontology. For a given set of assumption patterns $\mathcal{H}$ and a set of closed predicates Σ, the set of inconsistent queries for $(\mathcal{H}, \mathcal{O}, \Sigma)$, *denoted by* $Rew_\perp(\mathcal{H}, \mathcal{O}, \Sigma)$ *is*

$$\bigvee_{\alpha \in neg(\mathcal{O})} \{\varphi \mid \varphi \in rew_{\mathcal{H},\Sigma}(q'),\ \text{and}\ q' \in rew_{\mathcal{O}}(q_\alpha)\}.$$

We then show that each grounding of some set of assumption patterns which causes any inconsistency is captured by some query in $Rew_\perp(\mathcal{H}, \mathcal{O})$. For a tuple $\vec{y}$ from $vars(\mathcal{H})$, we denote by $Rew(\mathcal{Q}_{pos}^{\vec{y}})$ the UCQ whose disjuncts are the queries $q \in Rew(\mathcal{Q}_{pos})$ with $free(q) = \vec{y}$, and similarly, $Rew_\perp{}^{\vec{y}}(\mathcal{H}, \mathcal{O})$ is a UCQ with all queries $q \in Rew_\perp{}^{\vec{y}}(\mathcal{H}, \mathcal{O})$ with $free(q) = \vec{y}$.

Proposition 5.1. *Let $\mathcal{O}$ be a FO-rewritable DL-Lite$_\mathcal{A}^{++}$ ontology and ABox $\mathcal{A}$ consistent with $\mathcal{O}$. For any $\mathcal{E}$ a set of possible assertions w.r.t. $(\mathcal{A}, pos(\mathcal{O}))$, we have that $\mathcal{A} \cup \mathcal{E}$ is Σ-inconsistent w.r.t. $\mathcal{O}$ iff $() \in cert((\bigvee_{\alpha \in neg(\mathcal{O})} q_\alpha, pos(\mathcal{O}), \Sigma), \mathcal{A} \cup \mathcal{E})$.*

Proof. **Direction** $\Leftarrow$: Let $\mathcal{E}$ be such that $() \in cert((\bigvee_{\alpha \in neg(\mathcal{O})} q_\alpha, pos(\mathcal{O}), \Sigma), \mathcal{A} \cup \mathcal{E})$. This means that for each $\mathcal{I}$ a Σ-model of $\mathcal{A} \cup \mathcal{E}$ w.r.t. $pos(\mathcal{O})$ we have that there exists some q_α and a variable assignment θ such that $\mathcal{I} \vDash q_\alpha\theta$, hence $\mathcal{I} \nvDash \alpha$. Since this holds for each such $\mathcal{I}$ we conclude that $\mathcal{A} \cup \mathcal{E}$ is Σ-inconsistent w.r.t. $\mathcal{O}$.

Direction $\Rightarrow$: Consider a set $\mathcal{E}$ of possible assertions w.r.t. $(\mathcal{A}, pos(\mathcal{O}))$ such that $\mathcal{E}$ is Σ-inconsistent w.r.t. $\mathcal{O}$. This means that for each $\mathcal{I}$ a Σ-model of $(\mathcal{A}, \mathcal{E})$ w.r.t. $pos(\mathcal{O})$ there is some $\alpha \in neg(\mathcal{O})$ such that $\mathcal{I} \nvDash \alpha$. Therefore, there is some variable assignment θ such that $\mathcal{I} \vDash q_\alpha\theta$, hence $() \in cert(\bigvee_{\alpha \in neg(\mathcal{O})} q_\alpha, pos(\mathcal{O}), \Sigma)$. □

Note that, again, we obtain a set of queries with possibly different subsets of $vars(\mathcal{H})$ as free variables. We next prove that each answer of any query in $Rew_\perp(\mathcal{H}, \mathcal{O}, \Sigma)$ produces an ABox extension which is inconsistent w.r.t. $\mathcal{O}$.

Claim 10. *Let $\mathcal{O}$ be a FO-rewritable DL-Lite$_\mathcal{A}^{++}$ ontology and $\mathcal{H}$ a set of assumption patterns. For any ABox $\mathcal{A}$ consistent with $\mathcal{O}$ we have that $\vec{c} \in ans(Rew_\perp{}^{\vec{y}}(\mathcal{H}, \mathcal{O}, \Sigma), \mathcal{I}_\mathcal{A})$ iff for each $\mathcal{H}' \subseteq \overline{H}$ such that $vars(\mathcal{H}') = \vec{y}$ we have that $\mathcal{A} \cup \mathcal{H}'(\vec{c})$ is (Σ-)inconsistent with $\mathcal{O}$.*

Proof. **Direction** $\Rightarrow$. Let $\mathcal{A}$ be an arbitrary ABox Σ-consistent w.r.t. $\mathcal{A}$. We take an arbitrary $\vec{c} \in ans(\varphi_\perp, \mathcal{I}_\mathcal{A})$ such that for some arbitrary $\alpha \in neg(\mathcal{O})$ and $\mathcal{H}' \subseteq \overline{H}$ we have that $q_\alpha \rightsquigarrow_\mathcal{O}^* q'_\alpha \rightsquigarrow_{\mathcal{H}',\Sigma} \varphi_\perp$ and $\vec{c} \in ans(\varphi_\perp, \mathcal{I}_\mathcal{A})$. From Lemma 5.4 we obtain that $((), \mathcal{H}'(\vec{c})) \in cans((q'_\alpha, \emptyset, \mathcal{H}, \Sigma), \mathcal{A})$ and from Lemma 5.1 and the fact that the ontology does not contain symbols in Σ, we obtain that

$((), \mathcal{H}'(\vec{c})) \in cans((q_\alpha, pos(\mathcal{O}), \mathcal{H}, \Sigma), \mathcal{A})$ and using Proposition 5.1 we conclude that $\mathcal{A} \cup \mathcal{H}'(\vec{c})$ is Σ-inconsistent w.r.t. $\mathcal{O}$.

Direction $\Leftarrow$. Let $\vec{c}$ be a tuple of constants from $\mathcal{A}$ such that $\mathcal{A} \cup \mathcal{H}'(\vec{c})$ is Σ-inconsistent w.r.t. $\mathcal{O}$. Therefore, for each Σ-model of $\mathcal{A} \cup \mathcal{H}'(\vec{c})$ that satisfies $pos(\mathcal{O})$ we have that there exists some variable assignment θ such that $\mathcal{I} \vDash \bigvee_{\alpha \in neg(\mathcal{O})} q_\alpha\theta$, hence $() \in cert((\bigvee_{\alpha \in neg(\mathcal{O})} q_\alpha, pos(\mathcal{O}), \Sigma), \mathcal{A} \cup \mathcal{H}'(\vec{c}))$ from which, using Proposition 5.1, we get that $((), \mathcal{H}'(\vec{c})) \in cans((q_\alpha, pos(\mathcal{O}), \mathcal{H}, \Sigma), \mathcal{A})$. From Lemma 5.5 we obtain that there exists some $\varphi_\perp \in rew_{\mathcal{H},\Sigma}(q_\alpha)$ such that $\vec{c} \in ans(\varphi_\perp, \mathcal{I}_\mathcal{A})$. Since $\varphi_\perp \in Rew_\perp{}^{vars(\mathcal{H}')}(\mathcal{H}, \mathcal{O}, \Sigma)$ we obtain that $\vec{c} \in ans(Rew_\perp{}^{vars(\mathcal{H}')}(\mathcal{H}, \mathcal{O}, \Sigma), \mathcal{I}_\mathcal{A})$. □

A direct result form this claim is that the negation of $Rew_\perp(\mathcal{H}, \mathcal{O}, \Sigma)$ correctly ensures that the possible groundings of the assumption patterns do not produce inconsistencies.

Corollary 3. *Let $\mathcal{O}$ be a FO-rewritable DL-Lite$_\mathcal{A}^{++}$ ontology and $\mathcal{H}$ a set of assumption patterns. For each ABox $\mathcal{A}$, each $\vec{c} \in ans(\neg Rew_\perp{}^{\vec{y}}(\mathcal{H}, \mathcal{O}, \Sigma), \mathcal{I}_\mathcal{A})$ and $\mathcal{H}' \subseteq \overline{\mathcal{H}}$ such that $vars(\mathcal{H}') = \vec{y}$, we have that $\mathcal{A} \cup \mathcal{H}'(\vec{c})$ is Σ-consistent w.r.t $\mathcal{O}$.*

Therefore, to prevent inconsistency we can negate $Rew_\perp(\mathcal{H}, \mathcal{O}, \Sigma)$ and add it to the previous rewriting. The only minor issue to take care of is that the queries have different free variables.

Then the rewriting of general AOMQCs is obtained as follows.

Definition 5.18 (Perfect Rewriting of general AOMQCs)**.** *Let $\mathcal{Q} = (q, \mathcal{O}, \mathcal{H}, \Sigma)$ be a FO-rewritable DL-Lite$_\mathcal{A}^{++}$ AOMQC and let $\mathcal{Q}_{pos} = (q, pos(\mathcal{O}), \mathcal{H}, \Sigma)$. The* perfect rewriting of $\mathcal{Q}$, *$Rew(\mathcal{Q})$ is as follows:*

$$\bigcup_{\vec{y} \subseteq vars(\overline{\mathcal{H}})} \{\neg Rew_\perp{}^{\vec{y}}(\mathcal{H}, \mathcal{O}, \Sigma) \wedge Rew(Q_{pos}^{(\vec{x},\vec{y})})\}.$$

The following theorem follows from Corollary 3 and the fact that perfect rewriting for $\mathcal{Q}_{pos}$ is correct.

Theorem 5.2. *Let $\mathcal{Q} = (q(\vec{x}), \mathcal{O}, \mathcal{H})$ be a FO-rewritable DL-Lite$_\mathcal{A}^{++}$ AOMQ. For every ABox $\mathcal{A}$, the following are equivalent:*

a) $(\vec{a}, \mathcal{E}) \in cans(\mathcal{Q}, \mathcal{A})$.
b) There exists $\mathcal{H}' \subseteq \overline{\mathcal{H}}$, $\varphi(\vec{x} \cup \vec{y}) \in Rew(\mathcal{Q})$, a substitution π such that $\pi(\vec{x}) = \vec{a}$, $\mathcal{H}'\pi = \mathcal{E}$ and $\pi(\vec{x} \cup \vec{y})$ is an answer to the query φ over $\mathcal{I}_\mathcal{A}$.

5.6 Empirical Evaluation

To demonstrate the potential usefulness of our approach, we developed a prototype implementation[2] of the AOMQ rewriting. It was done in Java using Apache Jena 2.11 and Jena ARQ as SPARQL query engine, and tested on a MacBook Pro i5 2.7, Sierra OS.

[2] https://github.com/medinaandresel/conditionalAnswers

Size of AOMQ $\mathcal{Q}$ and $Rew(\mathcal{Q})$				Average Time (sec)			
$\mathcal{Q}$ [$\#\mathcal{H}$]	$\#rew_{\mathcal{O}}(\mathcal{Q})$	$\#Rew(\mathcal{Q})$	$\#cans_{min}(\mathcal{Q})$	**Evaluate** $Rew(\mathcal{Q})$	**Construct** $cans_{min}(\mathcal{Q})$	**Test** $\mathcal{E}$	**FedSPARQL**
q_1[3]	12	54	14415	0,6	1,1	40,4	42,7
q_2[4]	44	47	1603	0,1	0,2	4,8	6,6
q_3[4]	65	69	980	0,3	0,5	3,5	5,4
q_4[3]	14	64	60501	2,1	4,6	145,6	163,9
q_5[1]	3	6	8742	0,08	0,2	22	32,2

Table 5.1: Evaluation results for AOMQs over MyITS dataset.

We used the ontology, data, and a data generation tool from the MyITS project ([EKS13, EPS+15]). The tool creates ABoxes with assertions for spatial relations like locatedNext out of *OpenStreetMap* data, using parameters such as distance to create large sets of facts, in addition to other 'local' data (e.g., crowd-sourced restaurant data). Then one can pose queries that need both parts of data, as well as ontological reasoning, to get answers (e.g., hotels in residential areas close to a subway station).

Integrating geospatial data is relevant to many applications, however usually such datasets are incomplete or noisy. Instead of ingesting such data into our ABox, we might better keep such data as a remote source and query it on demand. In many application scenarios, however, instant answer retrieval is important, therefore assumption-based query answering can be a suitable option as only boolean queries need to be evaluated remotely. To simulate this scenario, we extracted some spatial relations and outsourced their access via a SPARQL endpoint (using Jena Fuseki). This resulted in two sources: the local datasets with 227634 RDF triples, and a remote one with more than 2 million triples. We created 5 AOMQs based on test queries of [EPS+15], and treated spatial atoms as assumption patterns. In this way, we can query the local datasets and verify at the remote access point whether the geospatial relations hold, only for the relevant candidates.

Table 5.1 shows for these queries the sizes of rewritings w.r.t. $\mathcal{O}$, and $(\mathcal{O}, \mathcal{H})$, and the size of $cans_{min}$, which gives a bound on the number of remote tests. We evaluated the time needed to answer the full rewriting over the local dataset, the time to construct the set $cans_{min}$, and the time to test remotely the spatial atoms (using SPARQL "ask queries"). The results show that evaluating $Rew(\mathcal{Q})$ and constructing $cans_{min}(\mathcal{Q})$ is very efficient, while testing the assumptions remotely was more expensive, as expected. In practice, this delay may be amortized in many cases, e.g., if many queries share remote tests. As a sanity check, we compared the total time needed by our approach to posing a federated SPARQL query [W3C13] using both data sources. The latter approach was slower, even despite the fact that we disregarded ontological reasoning. This implies that naively posing the result of the ontology rewriting as a federated query seems infeasible in practice at least based on current engines for evaluating federated queries.

5.7 Related Work and Discussion

Conditional query answering problem was studied since the early nineties in order to cope with incompleteness in disjunctive deductive databases [Dem92]. They are related to the problem of evaluating queries in hypothetically updated states of databases [GGMO95, CA98]. In [GH97] hypothetical queries are rewritten into equivalent usual queries which are evaluated using standard techniques. Recently, [tCCST15] define so-called *why-not queries*, where an ontology is leveraged to obtain explanations of why tuples are not an answer, and study the complexity of obtaining most general explanations. We consider the shape of the explanations to be part of the input, while focusing on preserving worst-case (data) complexity.

This work is most in line with the work of [COvS13], where *negative answers* are employed for describing a tuple of individuals and an associated explanation to why it is not an answer to the given query. Explanations there are ABoxes with assertions over concept and role names, while here this is generalized to sets of $\mathcal{EL}\mathcal{I}$ atoms. Additionally, we allow for closed predicates.

We have introduced AOMQs, which are extensions of OMQs with assumption patterns, designed for leveraging information when querying incomplete databases. Answering AOMQs consists of computing conditional answers, which generally extend the answers of OMQs with tuples that are made true by the assumptions. In the case of *DL-Lite*$_{\mathcal{R}}$ AOMQs, they remain FO-rewritable even in the presence of closed predicates. A simple prototype for constructing and testing minimal conditional answers shows promising results, and suggests that this approach may be useful in scenarios when answering OMQs over all relevant data at once is costly or infeasible.

For future work, it would be interesting to consider different shapes of assumption patterns, and to extend the evaluation with closed predicates.

CHAPTER 6

Neural-Symbolic Ontology-mediated Query Answering

Knowledge Graphs are relevant to many applications such as natural question answering, web search and data analytics, and due to their semi-automatic construction, they are prone to be incomplete. Since in a KG, each fact is represented as a triple (s, p, o), where s denotes the subject entity, p the predicate or relation, and o the object entity, the goal of *link prediction* is that of identifying the missing edges in the graph, and has received increasingly more attention since the seminal work [BUG+13]. The proposed model, TransE, aims at learning a representation of the graph in a low-dimensional vector space by translating each entity and relation in the KG into the vector space. Such translation is denoted as the *embedding function* of the model. In order to cover higher semantical expressivity, such as reflexivity or symmetry, other embedding models have been proposed in recent years, see [WQW21] for an in-depth overview.

The general idea of KG embedding models is that semantically related entities in the KG are embedded closely together in the vector space. Based on a scoring function that takes as input any possible triple over the KG signature and that depends on an embedding function, each possible triple in the KG has an associated score. If the score is higher than a threshold, then one concludes it is a true fact. The goal of the link prediction model is to learn an embedding function based on which accurate predictions are obtained.

Missing links are problematic when answering queries over KGs and several coping mechanisms are required to obtain the desired answers to queries. For example, the query *Who works for Amazon and has a degree from MIT?* over the KG in Figure 6.1 enhanced with ontology $\mathcal{O}$, can be formulated as $q(x) \leftarrow \text{degreeFrom}(X, \mathsf{mit}) \wedge \text{worksFor}(X, \mathsf{amazon})$. Evaluated in the traditional way such OMQ does not produce answers, however, there exist *plausible answers* to such query which may be true, such as entity mary.

Standard query answering is not sufficient in this case but neither is the use of out-of-the-box link prediction models. For instance, mary is a missing but desired answer of q and it is clear

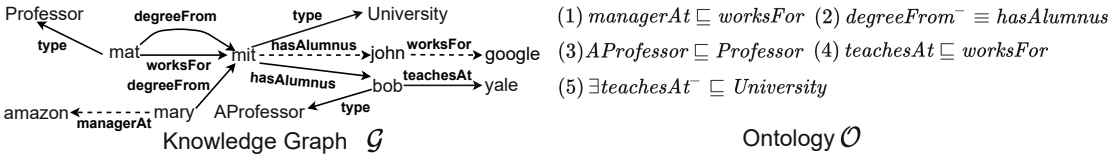


Figure 6.1: An example KG in which solid edges illustrate existing facts in the KG, while dashed edges indicate missing facts. The rules in $\mathcal{O}$ state that (1) managers at companies also work there; (2) the inverse of relation degreeFrom is hasAlumnus; (3) assistant professors are professors; (4) teachers at organizations also work there; (5) the range of the relation teachesAt is University.

that it cannot be obtained in the traditional way since fact worksFor(mary, amazon) cannot be derived by means of ontological reasoning. Similarly, due to the data distribution in the KG, link prediction models might learn that managerAt(mary, amazon) is a missing true fact, however it is not sufficient since in order to derive mary as answer to q, further domain knowledge has to be applied. The only way to obtain such answer is to use the fact that managerAt implies worksFor in ontology $\mathcal{O}$ of Figure 6.1, and derive the new fact worksFor(mary, amazon) which leads to the retrieval of mary as an answer to q. As demonstrated, obtaining complete answers for such a query require combining link prediction models and ontological reasoning.

A naive approach is to apply link prediction before query answering, however this is infeasible in practice as it requires the scoring and enumeration of all possible subgraphs that can satisfy a query. Recently, *Knowledge Graph Embedding* (KGE) techniques [NMTG16, WMWG17] that are able to predict missing answers to queries have been proposed. These models operate in the embedding space and produce scores for each entity and each possible query over the existing signature. The existing methods can be broadly divided into two categories: *query-based* [RHL20, RL20, LDJ+21, CRK+21, KLN21] and *atom-based* [ADMC21]. The former compute continuous query embedding representations, and use them for answering queries, while the latter compute answers to a query by identifying the most likely answers to all its atoms using *neural link predictors* [NMTG16], and then aggregating those answers using t-norms.

While being promising, such existing embedding-based query answering methods do not account for ontologies. On the one hand, the use of ontologies requires *deductive reasoning*, i.e., inferring new facts by applying ontology rules to existing facts, but it cannot derive true facts that do not directly follow from the existing knowledge. On the other hand, embedding methods are essentially tailored towards *inductive reasoning*, i.e., learning from examples: Given a number of queries and their answers, they are used to predict answers to other similar queries, but they typically cannot perform ontology reasoning. Since large portions of expert knowledge can be conveniently encoded using ontologies, the benefits of coupling ontology reasoning and embedding methods for KG completion are evident, and have been acknowledged [BRC+20, ZCZ+20, GS18, KLYH19]. However, to the best of our knowledge, such coupling has not been studied for OMQA.

Similarly to the naive approach to predict answers, a natural attempt is to interchangeably complete the KG using ontology reasoning and embedding methods, and then perform query answering on top of the result. This naive procedure, as mentioned before, comes with a big scalability challenge: In practice, we need to restrict ourselves to computing merely small subsets of likely

fact predictions required for answering a given query; thus more sophisticated techniques are required. To this end, we investigate three open questions: (1) How can we adapt existing OMQA techniques to the setting of Knowledge Graph Embeddings (KGE)? (2) How do different data augmentation strategies impact the accuracy of existing embedding models on the OMQA task? and (3) Does the enforcement of ontology axioms in the embedding space, via the loss function, help in improving the inductive and deductive reasoning performance? We answer these questions by making the following contributions:

- We formally define the task of *Embedding-Based OMQA* (E-OMQA) and empirically show that existing off-the-shelf KGE models applied naively perform poorly on this task.
- We propose novel ontology-driven strategies for sampling training queries as well as loss function modifications to enforce the ontology within the embedding space, and demonstrate the effectiveness of these proposals on popular representatives of query-based and atom-based KGE models.
- Since no previous benchmarks exist for E-OMQA, we design two datasets using LUBM and NELL, which are well-known benchmarks for OMQA and embedding models, respectively.
- Extensive evaluation for the E-OMQA task shows that enforcing the ontology via the loss function improves the deductive power of the KGE models, while data augmentation seems to be much more crucial for optimal performance. We obtain improvements, ranging from 20% to 55% in HITS@3.

The chapter is structured as follows: In Section 6.1, we present the preliminaries regarding the KG completion problem and we focus on two representative embedding techniques for QA. Next, in Section 6.2 we introduce the problem of E-OMQA. The proposed solutions for E-OMQA are presented in Section 6.3 – where we introduce several ontology-driven data augmentation strategies, and in Section 6.4 – where we present techniques for ontology injection into the loss function of the representative models. Section 6.5 presents the evaluation procedure and results on two novel benchmarks. We conclude the chapter with the related work and discussion in Section 6.6.

6.1 Query Answering over Knowledge Graph Embeddings

In this section we present a general overview on knowledge graph embeddings and two state-of-the-art models that rely on a learned embedding function to answer queries.

6.1.1 Knowledge Graphs and Embedding Models

In a broad sense, a KG contains data represented as a multi-relational graph. In particular, a knowledge graph (KG) is a set of triples of the form (s, p, o) in which s and o denote *entities*, such as Mary and Amazon, while p denotes the *relation* between s and o, like worksFor. This resembles to the already familiar notion of an ABox and it is relatively easy to view any given ABox as a KG and vice-versa. However, given the already defined ABox signature, there is

another distinction we need to make: concept assertions of the form $A(a)$ are given as triples of the form (a, type, A) in a KG, using the special relation "type", thus the set of entities captures symbols from $\mathbf{C} \cup \mathbf{K}$, i.e. concept names and constants. Given a KG $\mathcal{G}$, the set of certain answers for an OMQ $\mathcal{Q} = (q, \mathcal{O})$, denoted by $cert(\mathcal{Q}, \mathcal{G})$ is the set of certain answers obtained over the corresponding ABox $\mathcal{A}$ obtained by transforming each triple into an ABox assertion.

An important task is that of *link prediction* in which the goal is to predict missing relations between existing entities in the graph. This amounts to predict plausible answers to atomic queries (having exactly one answer variable) such as $q(x) \leftarrow \text{worksFor}(\mathsf{Mary}, x)$. Statistical approaches that use *embeddings* are the most promising, in which, roughly speaking, entities and relations in the signature of the KG are encoded as points and regions in a low-dimensional vector space with the goal of creating semantically meaningful vector representation of the data. This implies that entities that are (possibly) related are closely located within the embedding space. By leveraging the underlying structure of the data, the embedding-based model could potentially learn additional *data patterns* which are then leveraged for predicting missing edges in the graph. Among the most popular such models are TransE [BUG+13], DistMult [YYH+15] and ComplEx [TWR+16].

Recall Definition 5.1, which defines a general notion of ABox completion as an addition of a set of possible assertions which do not directly follow from the data and the ontology. A natural attempt in identifying sets of possible assertions is by using link predictors, however in addition, for the link prediction task the goal is to obtain sets of possible assertions that approximate the ground truth, thus a link predictor aims at identifying the most relevant sets of possible assertions.

Recently, embedding-based models are able to predict answers to more complex queries. Depending on the underlying approach, they can be categorized into *query-based* and *atom-based*. In the former category the general idea is to embed arbitrary queries (with more than one atom in the body) into the vector space by means of learning various geometrical operators such as projection and intersection. The main representatives of such category are models such as Query2Box [RHL20] and GQE [HBZ+18]. The atom-based models try to answer any general query by firstly decomposing it into query atoms which are then answered using existing atomic KGEs models, and later aggregating all answers using t-norms. The main representative of atomic-based QA model is CQD [ADMC21]. Given that none of the existing models are tailored for answering OMQs, we want to adapt such models for predicting missing answers to OMQs. For that we chose Query2Box and CQD as our front-runners for which we provide next the conceptual description behind such models.

We define next the type of queries currently supported by Query2Box and CQD.

Definition 6.1 (Anchor queries)**.** *Given a (U)CQ q, the* dependency graph *of q is a graph in which the nodes correspond to variables or entities in q and edges correspond to relations in atoms of q. A source node in the graph is a node with no incoming edges and a target node is a node with no outgoing edges.*

An anchor (U)CQ *is a (U)CQ for which the dependency graph is a directed acyclic graph in which each source node is an entity–denoted as* anchors*, and which has exactly one target node corresponding to the unique free variable.*

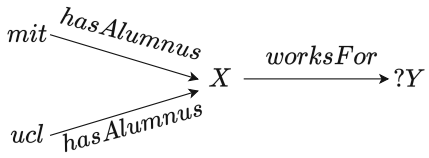


Figure 6.2: Dependency Graph Example

An example of an anchor query is $q(y) \leftarrow \exists x\ \text{hasAlumnus}(\mathsf{ucl}, x) \wedge \text{hasAlumnus}(\mathsf{mit}, x) \wedge \text{worksFor}(x, y)$ for which the dependency graph is illustrated in Figure 6.2.

We focus next on each representative embedding model for query answering.

6.1.2 Query2Box

In order to answer queries with conjunction it is required to have a corresponding operation in the embedding space. That motivates the design choice of Query2Box to embed queries as box regions given that the intersection of boxes is also a box. This model can also answer queries with disjunction by firstly transforming each such query into disjunctive normal form, secondly answering each conjunctive query by performing intersections of boxes, and lastly aggregating those answers.

Definition 6.2 (Box embeddings)**.** *Let $E \subset \mathbf{C} \cup \mathbf{K}$ be a finite set of entities and $R \subset \mathbf{R} \cup \mathbf{F}$ a finite set of relations. A d-*dimensional box*, denoted as $\mathbf{p}$=($\mathbf{cen_p}, \mathbf{off_p}$) $\in \mathbb{R}^d \times \mathbb{R}^d_{\geq 0}$ where $\mathbf{cen_p}$ is the* center of the box*, and $\mathbf{off_p}$ is the positive* offset of the box *modeling its size, is defined by:*

$$box_{\mathbf{p}} = \{\mathbf{v} \in \mathbb{R}^d \mid \mathbf{cen_p} - \mathbf{off_p} \preceq \mathbf{v} \preceq \mathbf{cen_p} + \mathbf{off_p}\},$$

where $\preceq$ is the element-wise inequality.

*A d-*dimensional box embedding *is a function that maps each $e \in E$ to a box $\mathbf{e}$ with offset $\mathbf{0} \in \mathbb{R}^d$ (i.e., a point), each $r \in R$ to a box $\mathbf{r}$, and each anchor query q to a box $\mathbf{q}$.*

In order to obtain the embedding of a query, based on the dependency graph, we can derive two geometric operators: projection and intersection which are then used to compute embeddings to queries. The idea behind is to learn embeddings to entities and relations such that by performing projection and intersection operations in the embedding space, the probable answer entities are contained in the box embedding of each query.

Projection. Assume a KG $\mathcal{G}$, a set of entities $E \subset \mathbf{C} \cup \mathbf{K}$ and a relation $r \in \mathbf{R} \cup \mathbf{F}$. The projection operator provides the set $\{v' \in E \mid v \in E, r(v, v') \in \mathcal{A}\}$. If we are given the embedding $\mathbf{r} = (\mathbf{cen_r}, \mathbf{off_r})$, then the *projection* of a box $\mathbf{v} = (\mathbf{cen_v}, \mathbf{off_v})$ by applying element-wise summation $\mathbf{v} + \mathbf{r} = (\mathbf{cen_v} + \mathbf{cen_r}, \mathbf{off_v} + \mathbf{off_r})$. This relational translation [BUG+13] operation corresponds to the translation and enlargement of the box $\mathbf{v}$.

Intersection. Given a set of entity sets $\{S_1, \ldots, S_n\}$, this operator computes the intersection of the given sets. We model the intersection $\mathbf{w} = (\mathbf{cen}_\mathbf{w}, \mathbf{off}_\mathbf{w})$ of a set of boxes by applying the following operations:

$$\mathbf{cen}_\mathbf{w} = \underset{i=1}{n} \phi(\mathbf{NN}(\mathbf{cen}_{\mathbf{v}_1}), \ldots, \mathbf{NN}(\mathbf{cen}_{\mathbf{v}_n}))_i \odot \mathbf{cen}_{\mathbf{v}_i}$$
$$\mathbf{off}_\mathbf{w} = min(\mathbf{off}_{\mathbf{v}_1}, \ldots, \mathbf{off}_{\mathbf{v}_n}) \odot \sigma(\psi(\mathbf{off}_{\mathbf{v}_1}), \ldots, \mathbf{off}_{\mathbf{v}_n}),$$

where $\odot$ and min denote the element-wise multiplication and minimum, respectively. $\mathbf{NN} : \mathbb{R}^d \mapsto \mathbb{R}^d$ is a 2-layer feed-forward neural network having the same dimensionality for the hidden layers as for the input layer. ϕ and σ stand for softmax and sigmoid functions, respectively, applied in a dimension-wise manner. ψ is a permutation invariant function composed of a 2-layer feed-forward network followed by element-wise mean operation and a linear transformation. The center $\mathbf{cen}_\mathbf{w}$ is calculated as the weighted mean of the box centers $\mathbf{cen}_{\mathbf{v}_1}, \ldots, \mathbf{cen}_{\mathbf{v}_n}$. This geometric intersection provides a smaller box that lies inside the set of boxes.

Intuitively, the projection and intersection operators are used to perform graph traversal in the vector space. Note that there is a set of parameters, corresponding to the parameters of the projection and intersection operators, that need to be optimized in order to learn the embedding function. For more technical details, we invite the reader to consult the original work – [RHL20].

Scoring Function. The score for an entity v being an answer to q is computed based on the Euclidean distance from $\mathbf{v}$ to $\mathbf{q}$:

$$d(\mathbf{q}, \mathbf{v}) = d_{\text{outside}}(\mathbf{q}, \mathbf{v}) + \alpha \cdot d_{\text{inside}}(\mathbf{q}, \mathbf{v}),$$

where d_{outside} represents the distance between $\mathbf{v}$ and the margins of the box $\mathbf{q}$, while d_{inside} represents the distance between $\mathbf{v}$ and the center of $\mathbf{q}$, and α is a weight which penalizes if $\mathbf{v}$ is inside the box but far from the center.

Training Objective. The goal is to learn the (box) embeddings of entities and relations such that for each query embedding, computed as a sequence of projection and intersection operations in the embedding space, the embedding of each answer entity is located inside the box embedding of the query. This is achieved by minimizing the following loss function:

$$L = -\log \sigma(\gamma - d(\mathbf{q}, \mathbf{v})) - \sum_{i=1}^{k} \frac{1}{k} \log \sigma(d(\mathbf{v}'_i, \mathbf{q}) - \gamma)$$

where γ is a scalar margin, v is an answer of q (i.e., positive entity) and v'_i is a non-answer entity (i.e., negative entity) and k is the number of negative examples. The σ function (called sigmoid function) is being used to transform the distance between the embedding of a query and the embedding of some entity into the interval $(0, 1)$. The intuition behind the loss function is that with each training step, the distance between each positive entity and the query box is minimized, and the distance between each negative entity and the query box is maximized.

6.1.3 Continuous Query Decomposition (CQD)

Another embedding model which we consider is *Continuous Query Decomposition* (CQD) which relies on neural link predictors for answering atomic subqueries, and aggregates the resulting scores via t-norms.

Neural Link Predictor. A neural link predictor is a differentiable model in which assertions are mapped into a d-dimensional vector space and then used for obtaining a score for the atom. More precisely, given an assertion $r(a, b)$ or $A(a)$, the score for $r(a, b)$ or $A(a)$, is computed as $f_r(\mathbf{a}, \mathbf{b})$, resp. $f_{\text{type}}(\mathbf{a}, \mathbf{A})$, where $\mathbf{a}, \mathbf{b}, \mathbf{A} \in \mathbb{R}^d$ are the embedding vectors of entities a, b, A, and f_r, respectively f_{type} take as input any two entity embeddings and produce a score in interval $[0, 1]$ that denotes the likelihood of the triple $r(a, b)$, respectively $A(a)$.

Current implementation of CQD is based on neural link predictor ComplEx-N3 [TWR+16] and the technical details of such model are not relevant for understanding how to answer queries. Moreover, any neural link predictor can be in principle used for the continuous query decomposition method.

T-Norms. A t-norm is a generalization of the conjunction logical operator, and the ones that are being used by this technique are the product t-norm $\top_{\text{prod}}(x, y) = x \cdot y$, and the Gödel t-norm $\top_{\text{min}}(x, y) = min\{x, y\}$. For handling disjunction, the complementary t-conorm is defined as $\bot(x, y) = 1 - \top(1 - x, 1 - y)$.

Query Answering via Combinatorial Optimization. Given a query q, in this approach we search for a set of variable substitutions that maximizes the query score. This is done by traversing the dependency graph of the query and for each atom $r(a, x)$ where a is an entity and x a variable, x is replaced by top-k entities c that maximize $f_r(\mathbf{a}, \mathbf{c})$, i.e. the most likely entities that are predicted as answers to the atomic query $q(x) \leftarrow r(a, x)$ by the neural link predictor. This procedure is similar to the *beam search*. The results are then aggregated using t-norms as described above.

6.2 Embedding-based Ontology-mediated Query Answering

Current embedding-based QA methods, like the ones presented before, only focus on *inductive reasoning*, i.e. answering queries by predicting missing facts based on patterns learned from the data, and lack the ability of *deductive reasoning*, i.e. inferring new triples derived from existing triples and predefined logical axioms in ontologies. Inductive and deductive reasoning complement each other, and thus yield more complete answers to a given query. Therefore, our goal is to develop an *Embedding-Based Ontology-Mediated Query Answering* method in which both types of reasoning are combined.

Following the common settings for KG completion[ACLS20, ADMC21, DNPR13], we define the *ideal completion* $\mathcal{G}^i$ of a knowledge graph $\mathcal{G}$ as a super-set of $\mathcal{G}$ that contains as many true triples, constructed using the signature of $\mathcal{G}$, as possible. For example, $\mathcal{G}^i$ for $\mathcal{G}$ in Figure 6.1 contains in addition the triples denoted by the dashed edges. The goal is then to approximate certain answers to OMQs over $\mathcal{G}^i$. Given that $\mathcal{G}^i$ is difficult to obtain in practice, in order to evaluate the accuracy

of an embedding method, the standard procedure is to consider some $\mathcal{G}^i$ typically fixed at the beginning (testing graph) and $\mathcal{G}$ (training graph) is created by removing facts from $\mathcal{G}^i$.

For a given KG $\mathcal{G}$, based on the particular scoring function, and the learned embedding function, *the embedding-based QA function* $f_{\mathcal{G}}$ takes as input a query and returns answers to that query. We rely on such an abstract notion to define the problem of answering OMQs over KGEs.

Definition 6.3 (Embedding-based OMQA)**.** *Let $\mathcal{G}$ be a KG, $\mathcal{O}$ an ontology, and $\mathcal{G}^i$ be the ideal completion of $\mathcal{G}$. An embedding function $f_{\mathcal{G}}$ is* reliable *if for any query q and entity a we have $a \in cert(q, \emptyset, \mathcal{G}^i)$ iff $a \in f_{\mathcal{G}}(q)$. Moreover, $f_{\mathcal{G}}$ is also* ontology-aware *if for any query q and entity a we have $a \in cert(q, \mathcal{O}, \mathcal{G}^i)$ iff $a \in f_{\mathcal{G}}(q)$. The* problem of embedding-based OMQA *is to obtain a reliable and ontology-aware embedding function.*

While we do not have access to the ideal completion $\mathcal{G}^i$ of $\mathcal{G}$, our goal is to approximate it using embedding methods. Note that, $cert(q, \mathcal{O}, \mathcal{G}^i)$ subsumes both $cert(q, \emptyset, \mathcal{G}^i)$, the answers that can be approximated by an embedding QA method via inductive reasoning, and $cert(q, \mathcal{O}, \mathcal{G})$, the answers computed by OMQA methods via deductive reasoning. In fact the most challenging answers are those that require deductive reasoning over missing facts (so-called *hard answers*).

Inspired by classical approaches for OMQA, we suggest the following options for embedding-based OMQA.

Query Rewriting. One of the most natural approaches is to first apply the ontology rewriting and then to evaluate each rewriting using the embedding-based QA function $f_{\mathcal{G}}$. In practice this amounts to train any embedding-based QA model using only information from $\mathcal{G}$, and then take the union of the set of answers of each query in the rewriting. This setting is similar to the rewriting techniques which are sound and complete procedures for OMQA. However, as our experiments demonstrate, this method enhances the effectiveness of identifying hard answers only by little compared to the evaluation of the original query using $f_{\mathcal{G}}$ (more details in Section 6.5).

Ontology-Aware Models. An alternative to query rewriting is to develop an embedding QA function that accounts for axioms in $\mathcal{O}$. While several ontology-aware models have been proposed for the standard link prediction task [MDRR17, ACLS20], to the best of our knowledge, none of the existing models addresses the problem of embedded-based OMQA. Thus, we propose two different approaches:

1. To train existing embedding models for logical QA on the data derived from the deductive closure of $\mathcal{G}$ using $\mathcal{O}$.
2. To develop an *ontology-aware* embedding model that has special terms in the training objective structurally enforcing in the embedding space the axioms in $\mathcal{O}$.

While the proposed approaches can be realized on top of any embedding model for complex QA, in this work we verify their effectiveness on *Query2Box* and *CQD*. Regarding (1), in Section 6.3 we present several methods for effective ontology-driven training. As for (2), building on *Query2Box*, in Section 6.4 we develop an ontology-aware embedding model. Moreover, as an ontology-aware extension of *CQD*, we build it on top of the neural link predictor using *adversarial sets*

regularization (ASR) [MDRR17] to enforce the ontology axioms. We chose this approach, since it is general, and allows us to incorporate rules into any off-the-shelf neural link predictor. In our experiments, we use ComplEx-N3 as it requires minimal modification to CQD and it outperforms other neural link predictors (see [LUO18]).

6.3 Ontology-driven Data Sampling

In order to obtain a reliable and ontology-aware QA function, the training set has to be carefully designed. The existing sampling procedure in the literature [HBZ+18, RHL20], arbitrarily chooses entities and edges in the graph to construct queries of various shapes. The training dataset consists of arbitrary queries and their answers over the KG. Since for Query2Box the projection and intersection operators have to be trained, queries of different shapes must be considered, while for CQD, since it relies on a neural link predictor, atomic queries are sufficient. For checking how well the model generalizes, the testing queries consist of more complex shapes and their answers are taken over $\mathcal{G}^i$.

The set of training queries does not take the ontology into account as they are randomly generated and answered over the KG alone, as positive entities. The negative entities, meaning some of the entities which are not answers over the KG, are as well randomly generated during training. This random query sampling technique is clearly not optimal for the case of OMQs since queries which might not have answers over the ABox alone cannot be generated following such procedure, while implicit answers can be selected as negative entities during training.

There are also other limitations which we need to consider. In practice it is not feasible to consider all possible queries of some shape over a given signature as they can be exponentially many. In the following, we discuss various options for sampling queries to train ontology-aware KGE models.

Certain Answer and Query Rewriting-Based Sampling. The first natural training approach is to use the random query sampling procedure described above but take their certain answers instead of the answers over the KG alone. For ontology languages such *DL-Lite*$_{\mathcal{A}}^{++(non\text{-}rec)}$ and *DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$ introduced in Chapter 3 computing certain answers can be done efficiently.

A second approach is to incorporate the ontology axioms in the training set. To do that, for each query shape we can randomly sample queries over the KG, using the standard procedure described above, transform each query into a query templates by marking atoms for specializing and generalizing, and generate their generalizations and specializations obtained using the reformulation rules in Table 4.1 presented in Chapter 4. The last step is to incorporate the reformulations into the training set alongside their certain answers. Intuitively, the specializations of a given query q incorporate additional more specific information regarding the answers of q, while the generalizations of q incorporate additional related entities, which can reinforce the inductive bias of the model.

Example 6.1. *Let* $q_1(x) \leftarrow \text{University}(x)$, $q_2(x) \leftarrow \exists z\ \text{teachesAt}(z, x)$ *be two queries. Using* ***R3*** *in Table 4.1 and (5) in Figure 6.1 we get that* q_2 *is a specialization of* q_1 *and vice-versa,* q_1 *is a generalization of* q_2.

In general, there are exponentially many rewritings thus to keep the training size reasonable, we fix a rewriting depth, up to which the respective training queries are generated, via a dedicated parameter.

Strategic Ontology-Based Sampling. While adding generalizations and specializations of randomly selected queries should capture some parts of the domain knowledge, many relevant queries might still be missed. For example, if query $q(x) \leftarrow \text{teachesAt}(x, \mathsf{yale})$ is not sampled during the random procedure neither is going to be $q'(x) \leftarrow \text{worksFor}(x, \mathsf{yale})$ since yale has no incoming worksFor edges. Based on this observation that the random sampling could still miss relevant part of implicit information, another training approach that we propose is to leverage the ontology to strategically generate the train queries.

For that first, we formalize the set of target queries by means of a *query shape*, i.e., a directed acyclic graph (DAG) (N, E), where N is a set of nodes and $E \subseteq N \times N$ is a set of directed edges. Such DAG captures the underlying structure of each query having that shape. The set of target queries is then obtained by applying a labeling function to assign symbols in Σ to nodes and edges.

Definition 6.4 (Query Shape)**.** *A* query shape S *is a tuple* (N, E, n) *such that* (N, E) *is a DAG and* $n \in N$ *is the* distinguished node *of* S *(i.e., the node corresponding to the answer variable). For a given set of relations and constants in* Σ*,* a labeling function $\ell : N \cup E \mapsto \Sigma \cup \mathbf{V}$ *maps each node to either a variable or an entitiy and each edge to a relation symbol in* Σ*.*

Our goal is to exploit the ontology to label query shapes and create queries that are semantically meaningful. For that we create query templates for each particular shape and then use the derivation rules introduced in Table 4.1 from Chapter 4 to obtain semantically meaningful training queries. The query sampling process is then defined for shapes that belong to the anchor queries, thus no cycles are involved and it is sufficient to only look at the neighboring edges and nodes to label each query shape.

Towards that, for any relation p in the signature of the ontology we define:

$$
\begin{aligned}
inv(p) &= \{p' \mid p \sqsubseteq p'^{-} \in \mathcal{O}\},\\
dom(p) &= \{A \mid \exists p' \sqsubseteq A' \in \mathcal{O} \text{ s.t. } p \sqsubseteq^* p', A \sqsubseteq^* A' \text{ or } A' \sqsubseteq^* A\},\\
range(p) &= \{A \mid \exists p'^{-} \sqsubseteq A' \in \mathcal{O} \text{ s.t. } p \sqsubseteq^* p', A \sqsubseteq^* A' \text{ or } A' \sqsubseteq^* A\},\\
follows(p) &= \{p' \mid range(p) \cap dom(p') \neq \emptyset \text{ or } p \circ p' \sqsubseteq s \in \mathcal{O}\},\\
inter_r(p) &= \{p' \mid range(p) \cap range(p') \neq \emptyset \text{ or } p_1 \in inv(p), p_2 \in inv(p') \text{ and } dom(p_1) \cap dom(p_2) \neq \emptyset\},\\
inter_d(p) &= \{p' \mid dom(p) \cap dom(p') \neq \emptyset \text{ or } p_1 \in inv(p), p_2 \in inv(p') \text{ and } range(p_1) \cap range(p_2) \neq \emptyset\}.
\end{aligned}
$$

Intuitively, for a given relation p, the set $inv(p)$ contains all inverse relations of p, $dom(p)$ contains all domain types for p, $range(p)$ all range types for p, $follows(p)$ stores all relations p' which can follow p, and $inter_r(p)$, $inter_d(p)$ contain respectively all relations p' which can intersect p on range and domain positions. Then, for each shape, starting from the anchor nodes, we label nodes and edges to create query templates that are valid w.r.t. $\mathcal{O}$ as illustrated in Figure 6.3. This

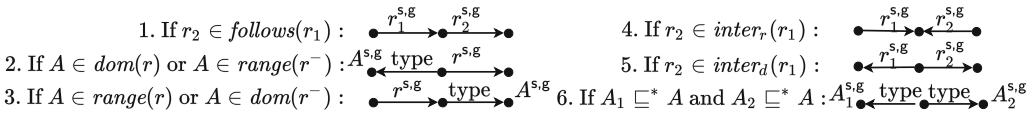


Figure 6.3: Ontology-driven rules to label query template atoms; s, g denote that the atom can be either specialized or generalized.

query sampling process uses only the ontology, thus it is data independent, however to create anchor UCQs, we also randomly choose a small percentage of entities as anchors provided that the obtained anchored UCQ has answers.

6.4 Ontology-Aware Models

In this section we present the modifications we made to Query2Box and CQD in order to enforce the ontology axioms in the embedding space. For CQD we build it on top of the neural link predictor using *adversarial sets regularization* (ASR) [MDRR17] to enforce the ontology axioms. We chose this approach, since it is general, and allows us to incorporate rules into any off-the-shelf neural link predictor. In our experiments, we use ComplEx-N3 as it requires minimal modification to CQD and it outperforms other neural link predictors (see [LUO18]).

6.4.1 Ontology-aware CQD

In this subsection we briefly describe how we inject the ontology axioms into the neural link predictor employed by CQD. For that we rely on the FO translation of the DL axioms. Following [MDRR17], for each rule the goal is to identify the entity embeddings which maximize an *inconsistency loss*, meaning the entities for which the scoring of the head is much lower compared to the scoring of the body. For example, given the rule $\text{teachesAt}(X, Y) \rightarrow \text{type}(Y, \text{University})$, the goal is to find a mapping from variables to d-dimensional embeddings, i.e. $\phi : \mathbf{V} \mapsto \mathbb{R}^d$, such that $[f_{\text{teachesAt}}(\phi(X), \phi(Y)) - f_{\text{type}}(\phi(Y), \textbf{University})]_+$ is maximal, where $[x]_+ = max([x], 0)$ and $[x]$ is the integral part of x. Such mapping determines a so-called *adversarial input set*, which is used as an adaptive regulariser for the neural link predictor. This inconsistency loss is then incorporated into the final loss function of the ontology-aware model which tries to minimize the maximal inconsistency while learning to predict the target graph over the given sets of correct triples. In our experiments we rely on the existing implementation of the adversarial sets regularisation method into ComplEx-N3, which is the default neural link predictor for CQD.

6.4.2 Ontology-aware Query2Box

In this section, we present our novel training objective function employed by Query2Box. Recall that when embedding a query, the Query2Box model defines a box in an embedding space, such that the answer entities of the given query are mapped to points located inside of the box. The

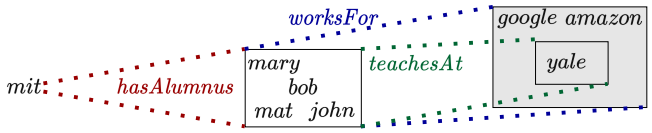


Figure 6.4: Our extension of Query2Box, where query embeddings capture the ontology axiom teachesAt $\sqsubseteq$ worksFor, represented by the inclusion of the respective boxes.

general idea is to treat each ontological axiom as an inclusion of queries and then injecting them into the model by ensuring in the vector space the inclusion of the boxes corresponding to the respective queries.

Example 6.2. *In Figure 6.4, the entities and relations are embedded into the vector space as points and projection operators, respectively. The embedding of* $q(Y) \leftarrow \exists X.\mathsf{hasAlumnus}(\mathsf{mit}, X) \wedge \mathsf{worksFor}(X, Y)$ *is represented by the larger grey box, obtained by applying the projection* hasAlumnus *to the embedding of entity* mit *followed by the projection on* worksFor. *To enforce* teachesAt $\sqsubseteq$ worksFor *we ensure that the box corresponding to* $q'(Y) \leftarrow \exists X.\mathsf{hasAlumnus}(\mathsf{mit}, X) \wedge \mathsf{teachesAt}(X, Y)$*, is contained in the box corresponding to* q.

The goal is to learn the embedding of queries, such that the *distance* between the box, corresponding to the query, and its answers is minimized, while the *distance* to this box from other negative samples is maximized. Similarly to [RHL20], we define the distance between $\mathbf{q} \in \mathbb{R}^d \times \mathbb{R}^d_{\geq 0}$ and $\mathbf{v} \in \mathbb{R}^d$ as $d(\mathbf{q}, \mathbf{v}) = \|\mathbf{cen}_q - \mathbf{v}\|_1$, namely the L_1 distance from the entity $\mathbf{v}$ to the center of the box. Using the sigmoid function we transform the distance into the $(0, 1)$ interval, that is, $p(\mathbf{v} \mid \mathbf{q}) = \sigma\big(-(d(\mathbf{q}, \mathbf{v}) - \gamma)\big)$, where $\gamma > 0$ is a margin, which denotes the probability of $v \in cert(q, \mathcal{O}, \mathcal{G}^i)$.

For a query q, let $Gen(q) = \{q_1 \dots q_n\}$ be the set of all generalizations of q based on $\mathcal{O}$. Given a train query q and its certain answer $v \in cert(q, \mathcal{O}, \mathcal{G})$, we aim at maximizing $\prod_{i=1}^n p(\mathbf{v} \mid \mathbf{q}_i)^{\beta_i}$, where $\beta_i \geq 0$ is a weighting parameter for all $i = 1, \dots, n$. This is achieved by minimizing the negative log-likelihood:[1] $-\log\Big(\prod_{i=1}^n p(\mathbf{v} \mid \mathbf{q}_i)^{\beta_i}\Big) = -\sum_{i=1}^n \beta_i \log\big(p(\mathbf{v} \mid \mathbf{q}_i)\big)$. By exploiting the fact that $\sigma(x) = 1 - \sigma(-x)$, for any $\mathbf{v}'_j \notin cert(q, \mathcal{O}, \mathcal{G})$, we have that $p(\mathbf{v}' \mid \mathbf{q}) = 1 - p(\mathbf{v} \mid \mathbf{q}_i) = \sigma(d(\mathbf{q}, \mathbf{v}) - \gamma)$.

Our goal is to ensure that if q' is a generalization of a given train query q w.r.t. $\mathcal{O}$, then the box of q' contains the box of q. In other words, if a is an answer to the query q then the distance not only between a and q should be minimized, but also between a and q' as well as between a and all other generalizations of q. The following training objective reflects our goal:

$$L = -\sum_{i=1}^{n} \beta_i \log \sigma\big(\gamma - d(\mathbf{v}, \mathbf{q}_i)\big) - \sum_{j=1}^{k} \frac{1}{k} \log \sigma(d(\mathbf{v}'_j; \mathbf{q}) - \gamma),$$

where $\mathbf{v}'_j \notin q[\mathcal{G}, \mathcal{O}]$ is a random entity for all $j = 1, \dots, k$ obtained via negative sampling. In our experiments, we use $\beta_i = |Gen(q)|^{-1} = 1/n$.

[1] The log is strictly monotonically increasing, thus, it will not change the maximization. It only changes the product to a summation. During training we consider a minimization, which motivates the negative sign.

Example 6.3. *Consider* $q(Y) \leftarrow \exists X.\text{hasAlumnus}(\text{mit}, X) \wedge \text{type}(X, \text{AProfessor}) \wedge \text{teachesAt}(X, Y)$. *We have* $Gen(q) = \{q_1, q_2, q_3\}$, *where* q_1 *is obtained from* q *by substituting* teachesAt *with* worksAt, *while* q_2 *is* q *with* $\text{type}(X, \text{Professor})$ *instead of* $\text{type}(X, \text{AProfessor})$. *In* q_3 *the first, second and third atoms are resp. the same as in* q, q_1 *and* q_2. *It holds that* $q[\mathcal{G}, \mathcal{O}] = \{\text{yale}\}$, *hence our training objective is to minimize the distance between* **yale** *(the embedding of* yale*), and* $\mathbf{q}$ *as well as the distance between* **yale** *and the boxes of* q_1, q_2 *and* q_3 *(denoted by* $\mathbf{q_1}$, $\mathbf{q_2}$ *and* $\mathbf{q_3}$*).*

Note that conceptually, our training data sampling techniques and the loss function modifications are flexible in terms of the DL in which the ontology is encoded. The only restriction is that for this DL there exist meaningful reformulation rules to generalize queries and efficient algorithms to construct them. Moreover, using this technique one can enforce more expressive rules: For example, consider the pair of queries: $q_1(y) \leftarrow \exists x\ \text{nationality}(x, \text{Canadian}) \wedge \text{hasAward}(x, \text{TuringAward}) \wedge \text{graduatedFrom}(x, y)$ and $q_2(y) \leftarrow \exists x\ \text{nationality}(x, \text{American}) \wedge \text{worksFor}(x, \text{Google}) \wedge \text{hasPhDFrom}(x, y)$. Using our method one can enforce the following complex rule: "The set of universities, from which Canadian Turing Awardees are graduated is included in the set of universities from which American Google employees got their PhDs degrees". To the best of our knowledge, existing embedding methods cannot capture this kind of complex rules involving constants.

6.5 Evaluation

In this section, we evaluate the proposed training strategies on the two recent embedding models for QA: Query2Box model [RHL20] and Continuous Query Decomposition [ADMC21] as well as our ontology-aware adaptations $O2B$ and CQD^{ASR}. All models are evaluated in different settings to measure their ability to perform inductive reasoning, deductive reasoning, and their combination.[2]

Query and Answers Sampling. We use the same type of queries as [RHL20] (see Figure 6.5). We consider each input KG $\mathcal{G}$ to be the ideal completion (i.e. $\mathcal{G}^i$) and then partition it into $\mathcal{G}_{valid}$ for validation and $\mathcal{G}_{train}$ for training by discarding 10% of edges at each step; this yields $\mathcal{G}_{train} \subsetneq \mathcal{G}_{valid} \subsetneq \mathcal{G}$. We then create several training sets of queries according to our ontology-aware data sampling strategies from Section 6.3. More specifically, these include:

plain: the training queries are randomly sampled based on the signature of $\mathcal{G}_{train}$, and their plain answers, i.e. over $\mathcal{G}_{train}$ alone ignoring the ontology.

gen: queries from *plain* augmented with their ontology-based generalizations [3]; all answers are certain, i.e. over $\mathcal{G}_{train}$ by taking into account the ontology.

spec: queries from *gen* augmented with their ontology-based specializations; all answers are certain answers as well.

[2] Code and data are available at `https://tinyurl.com/66hbhppc`.

[3] This setting is similar to random sampling over the deductive closure of $\mathcal{G}_{train}$ w.r.t. $\mathcal{O}$, but our procedure is guaranteed to terminate. We used the rewriting depth of up to 10.

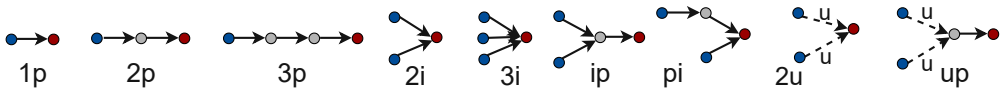


Figure 6.5: Query shapes considered in our experiments, where blue nodes correspond to anchor entities and red ones to answer variables; p stands for projection, i for intersection and u for union. The first five shapes are used in training.

onto: queries constructed relying on the ontology axioms as introduced in section 6.3, for which we randomly choose a percentage of valid entities as anchors; all answers are certain.

Following [RL20], the training query shapes are the first five ones in Figure 6.5 (1p–3i); non-compliant specializations and generalizations are discarded. The $Q2B$ and $O2B$ are trained on all five query shapes, while CQD and CQD^{ASR} are trained only on 1p queries [ADMC21].

Evaluation Procedure. For each trained model we measure its performance using standard metric HITS at K for K=3 (HITS@3), which indicates the frequency that the correct answer is ranked among the top-3 results (the higher, the better). We use such metric for measuring the reliability and ontology-awareness of the resulting models (as in Definition 6.3):

Inductive reasoning (I). Is the model able to predict missing answers to queries over the ideal completion $\mathcal{G}^i$?

Deductive reasoning (D). Is the model able to predict answers that can be inferred from the known triples in $\mathcal{G}_{train}$ using ontology axioms?

Inductive + Deductive reasoning (I+D). The combination of **I** and **D**: Is the model able to predict missing answers that are inferred from the ideal completion $\mathcal{G}^i$ using axioms from $\mathcal{O}$?

For test case **I** , respectively **I+D** , test queries are randomly sampled over $\mathcal{G}$, respectively over deductive closure of $\mathcal{G}$ w.r.t. $\mathcal{O}$, while for **D** they are randomly sampled over the deductive closure of $\mathcal{G}_{train}$ w.r.t. $\mathcal{O}$. All test queries are sampled in such a way that they cannot be trivially answered over $\mathcal{G}_{train}$, and they are unseen during training. In each test case the validation queries are generated similarly but over $\mathcal{G}_{valid}$. For each test and validation query, we measure the accuracy based on so-called *hard answers*, i.e., those that cannot be trivially answered over $\mathcal{G}_{train}$ (or $\mathcal{G}_{valid}$ for test queries) and require prediction of missing edges and/or ontology application. The statistics regarding the train, validation and test queries is presented in Table 6.2.

Models and Datasets. We consider $Q2B$, $O2B$, CQD and CQD^{ASR} trained in each described setting: i.e., M_x, where $M \in \{Q2B, O2B, CQD, CQD^{ASR}\}$ and $x \in \{\text{plain, gen, spec, onto}\}$; $Q2B_{plain}$ and CQD_{plain} are taken as baselines. We consider two datasets: NELL [CBK+10], a general purpose real world KG, and LUBM [GPH05], a domain specific synthetic dataset describing the university domain. We chose these KGs, as they are among few large KGs that have ontologies (see Table 6.1 for statistics).

Table 6.1: The number of axioms in the ontology $|\mathcal{O}|$, the number of each type of axiom, the size of the input KG $|\mathcal{G}|$, the number of entities $|\mathbf{E}|$, the number of relations $|\mathbf{R}|$, and the number of materialized triples ($|\mathcal{O}^\infty(\mathcal{G})|$).

Dataset	Ontology $\mathcal{O}$						KG $\mathcal{G}$			
	$\lvert\mathcal{O}\rvert$	$A \sqsubseteq A'$	$p \sqsubseteq s$	$p^- \sqsubseteq s$	$\exists p \sqsubseteq A$	$\exists p^- \sqsubseteq A$	$\lvert\mathcal{G}\rvert$	$\lvert\mathbf{E}\rvert$	$\lvert\mathbf{R}\rvert$	$\lvert\mathcal{O}^\infty(\mathcal{G})\rvert$
LUBM	68	13	5	28	11	11	284k	55684	28	565k
NELL	307	–	92	215	–	–	285k	63361	400	497k

Table 6.2: Number of queries per query shape for each sampling case.

Dataset	Train/Test	Query Shape								
		1p	2p	3p	2i	3i	ip	pi	2u	up
LUBM	*Plain*	110000	110000	110000	110000	110000	–	–	–	–
	Gen	117124	136731	150653	181234	208710	–	–	–	–
	Spec	117780	154851	173678	271532	230085	–	–	–	–
	Onto	116893	166159	333406	212718	491707	–	–	–	–
	I	8000	8000	8000	8000	8000	8000	8000	8000	8000
	D	1241	4701	6472	3829	4746	7393	7557	4986	7122
	I+D	8000	8000	8000	8000	8000	8000	8000	7986	8000
NELL	*Plain*	107982	107982	107982	107982	107982	–	–	–	–
	Gen	174310	408842	864268	398412	930787	–	–	–	–
	Spec	174310	419664	906609	401954	936537	–	–	–	–
	Onto	114614	542923	864268	629144	930787	–	–	–	–
	I	15688	3910	3918	3828	3786	3932	3895	3940	3966
	D	346	4461	4294	4842	5996	7295	5862	5646	6894
	I+D	8000	8000	8000	8000	8000	8000	8000	7990	8000

We have configured both $Q2B$ and $O2B$ systems as follows: The size of the embedding dimension was set to 400, and the models were trained for 15×10^4 steps using Adam optimizer with an initial learning rate of 10^{-4} and the batch size of 512. We evaluated the models periodically and reported the test results of the models which have the best performance on the validation dataset. For CQD and CQD^{ASR} we have used the following configuration: we used ComplEx-N3 [LUO18] as the underlying neural link predictor, where the embedding size was set to 1000, and the regularisation weight was selected based on the validation set by searching in $\{10^{-3}, 5 \times 10^{-3}, \ldots, 10^{-1}\}$.

6.5.1 Evaluation Results

We present the results in the settings **D** and **I+D** for LUBM and NELL in Tables 6.3–6.6 (aggregated in Table 6.7 which in addition contains also the setting *spec*) and Figures 6.6–6.7, and

Table 6.3: HITS@3 scores in the deductive setting (**D**) for LUMB

Model	Avg.	1p	2p	3p	2i	3i	ip	pi	2u	up
$Q2B_{plain}$	0.253	0.318	0.12	0.103	0.464	0.588	0.181	0.242	0.160	0.104
$O2B_{plain}$	0.276	0.317	0.113	0.094	0.512	0.63	0.189	0.263	0.257	0.11
CQD_{plain}	0.174	0.101	0.051	0.100	0.364	0.509	0.133	0.199	0.076	0.040
CQD^{ASR}_{plain}	**0.685**	**0.806**	**0.727**	**0.442**	**0.811**	**0.777**	**0.710**	**0.601**	**0.649**	**0.646**
$Q2B_{gen}$	0.506	0.619	0.242	0.113	**0.887**	**0.936**	0.426	0.333	0.671	0.327
$O2B_{gen}$	0.493	0.641	0.221	0.100	0.876	0.921	0.399	0.317	0.66	0.301
CQD_{gen}	0.427	0.460	0.150	0.079	0.770	0.830	0.343	0.342	0.618	0.252
CQD^{ASR}_{gen}	**0.778**	**0.879**	**0.794**	**0.477**	0.883	0.871	**0.788**	**0.726**	**0.835**	**0.749**
$Q2B_{onto}$	0.818	0.929	0.760	0.482	0.988	0.994	0.877	0.646	0.932	0.751
$O2B_{onto}$	0.838	**0.960**	0.771	0.514	**0.991**	**0.996**	0.879	0.697	**0.963**	0.768
CQD_{onto}	**0.861**	0.901	0.857	**0.552**	0.961	0.979	**0.896**	**0.879**	0.942	0.788
CQD^{ASR}_{onto}	0.830	0.902	**0.860**	0.493	0.920	0.915	0.853	0.818	0.908	**0.808**

Table 6.4: HITS@3 scores in the inductive and deductive setting (**I+D**) for LUMB

Model	Avg.	1p	2p	3p	2i	3i	ip	pi	2u	up
$Q2B_{plain}$	0.218	0.173	0.101	0.107	0.433	0.546	0.167	0.200	0.133	0.100
$O2B_{plain}$	0.245	0.235	0.109	0.095	0.488	0.584	0.176	0.218	0.2	0.103
CQD_{plain}	0.179	0.109	0.058	0.104	0.384	0.502	0.130	0.187	0.092	0.046
CQD^{ASR}_{plain}	**0.56**	**0.682**	**0.589**	**0.393**	**0.659**	**0.664**	**0.547**	**0.488**	**0.509**	**0.509**
$Q2B_{gen}$	0.458	0.592	0.267	0.129	**0.789**	**0.870**	0.360	0.282	0.552	0.279
$O2B_{gen}$	0.447	0.577	0.257	0.114	0.777	0.859	0.359	0.27	0.546	0.264
CQD_{gen}	0.408	0.539	0.214	0.098	0.710	0.791	0.304	0.302	0.513	0.208
CQD^{ASR}_{gen}	**0.628**	**0.733**	**0.640**	**0.413**	0.717	0.720	**0.598**	**0.599**	**0.653**	**0.582**
$Q2B_{onto}$	0.687	0.762	0.617	0.447	0.868	0.915	0.693	0.555	0.732	0.600
$O2B_{onto}$	0.707	**0.771**	0.629	0.476	**0.878**	**0.927**	0.694	0.619	**0.752**	**0.618**
CQD_{onto}	**0.723**	0.752	**0.681**	**0.481**	0.870	0.924	**0.735**	**0.728**	0.738	0.604
CQD^{ASR}_{onto}	0.664	0.753	**0.681**	0.421	0.744	0.755	0.643	0.666	0.704	0.615

briefly discuss the setting **I** (results illustrated in Figure 6.8). Our main observation in the test case **I** is that baseline $Q2B_{plain}$ performs best on LUBM, while CQD_{plain} outperforms the other models and configurations on NELL. The reason is that ontologies are not effective when coping with missing edges and facts in a KG beyond those that they can deductively infer. In fact, if statistically, the patterns reflected by ontologies do not hold. The second observation is that the query rewriting over pre-trained embedding models only slightly improves the prediction accuracy, resulting in at most 10% enhancement on test cases **D** and **I+D** over $Q2B_{plain}$, and CQD_{plain} respectively (see Table 6.8). These limited improvements are likely due to the incompleteness of the rewriting procedure caused by the restriction of the queries supported by the models. The

Table 6.5: HITS@3 scores in the deductive setting (**D**) for NELL

Model	Avg.	1p	2p	3p	2i	3i	ip	pi	2u	up
$Q2B_{plain}$	0.521	0.763	0.401	0.325	0.776	0.832	0.452	0.535	0.337	0.272
$O2B_{plain}$	0.664	0.816	0.483	**0.413**	**0.961**	**0.975**	0.535	0.648	**0.792**	0.355
CQD_{plain}	0.598	0.710	0.412	0.341	0.891	0.929	0.522	0.593	0.649	0.331
CQD^{ASR}_{plain}	**0.708**	**0.942**	**0.647**	0.403	0.880	0.909	**0.666**	**0.663**	0.700	**0.570**
$Q2B_{gen}$	0.734	0.974	0.559	0.466	0.99	0.99	0.622	0.685	0.94	0.377
$O2B_{gen}$	0.744	0.962	0.572	0.492	0.989	0.990	0.639	0.712	0.944	0.396
CQD_{gen}	**0.953**	**1.000**	0.996	**0.601**	**1.000**	**1.000**	0.997	**0.999**	**1.000**	0.988
CQD^{ASR}_{gen}	0.949	**1.000**	**0.998**	0.570	**1.000**	**1.000**	**0.998**	0.994	0.997	**0.989**
$Q2B_{onto}$	0.725	0.973	0.567	0.466	0.985	0.986	0.606	0.654	0.909	0.384
$O2B_{onto}$	0.748	0.968	0.598	0.496	0.989	0.988	0.646	0.697	0.941	0.412
CQD_{onto}	0.591	0.659	0.404	0.329	0.896	0.943	0.515	0.611	0.663	0.302
CQD^{ASR}_{onto}	**0.902**	**0.979**	**0.921**	**0.542**	**0.995**	**0.995**	**0.942**	**0.921**	**0.976**	**0.852**

Table 6.6: HITS@3 scores in the inductive and deductive setting (**I+D**) for NELL

Model	Avg.	1p	2p	3p	2i	3i	ip	pi	2u	up
$Q2B_{plain}$	0.458	0.516	0.343	0.286	0.747	0.81	0.404	0.447	0.325	0.241
$O2B_{plain}$	**0.596**	**0.79**	0.409	**0.359**	**0.904**	**0.936**	0.479	**0.521**	**0.666**	0.303
CQD_{plain}	0.555	0.664	0.383	0.304	0.853	0.903	0.471	0.512	0.599	0.306
CQD^{ASR}_{plain}	0.592	0.716	**0.518**	0.337	0.807	0.831	**0.547**	0.513	0.614	**0.445**
$Q2B_{gen}$	0.642	0.858	0.485	0.397	0.928	0.95	0.538	0.539	0.768	0.312
$O2B_{gen}$	0.652	0.859	0.494	0.420	0.928	0.953	0.552	0.559	0.77	0.329
CQD_{gen}	**0.809**	**0.903**	**0.775**	**0.473**	**0.957**	**0.969**	**0.821**	**0.757**	**0.886**	**0.743**
CQD^{ASR}_{gen}	0.787	0.9	0.771	0.467	0.919	0.924	0.793	0.723	0.846	0.741
$Q2B_{onto}$	0.636	0.858	0.472	0.398	0.927	0.948	0.529	0.524	0.747	0.317
$O2B_{onto}$	0.655	0.862	0.498	0.423	**0.933**	**0.953**	0.557	0.555	0.773	0.340
CQD_{onto}	0.545	0.667	0.368	0.293	0.848	0.904	0.453	0.506	0.595	0.275
CQD^{ASR}_{onto}	**0.77**	**0.9**	**0.741**	**0.456**	0.922	0.923	**0.77**	**0.701**	**0.851**	**0.672**

results on **I** and query-rewriting over embeddings are presented in Figure 6.8 and Table 6.8 respectively. Moreover, the results on certain answer prediction (**D** and **I+D**) show that none of the baselines is able to capture the domain knowledge expressed in the ontology, and thus cannot be used directly for OMQA. Next, we discuss our main observations based on the results reported on test cases **D** and **I+D** . Overall, the effectiveness of the proposed solutions is evident: for **I+D** on LUBM the improvements are of almost 50% for Query2Box and 54% for CQD, while for NELL of almost 20% for Query2Box and 25% for CQD.

Table 6.7: HITS@3 metric per query shape for deductive (**D**) and inductive+deductive (**I+D**)

Model	Test Case D										Test Case I+D									
	Avg.	**1p**	**2p**	**3p**	**2i**	**3i**	**ip**	**pi**	**2u**	**up**	**Avg.**	**1p**	**2p**	**3p**	**2i**	**3i**	**ip**	**pi**	**2u**	**up**
LUBM																				
$Q2B_{plain}$	0.253	0.318	0.12	0.103	0.464	0.588	0.181	0.242	0.160	0.104	0.218	0.173	0.101	0.107	0.433	0.546	0.167	0.200	0.133	0.100
$O2B_{plain}$	0.276	0.317	0.113	0.094	0.512	0.63	0.189	0.263	0.257	0.11	0.245	0.235	0.109	0.095	0.488	0.584	0.176	0.218	0.2	0.103
CQD_{plain}	0.174	0.101	0.051	0.100	0.364	0.509	0.133	0.199	0.076	0.040	0.179	0.109	0.058	0.104	0.384	0.502	0.130	0.187	0.092	0.046
CQD^{ASR}_{plain}	**0.685**	**0.806**	**0.727**	**0.442**	**0.811**	**0.777**	**0.710**	**0.601**	**0.649**	**0.646**	**0.56**	**0.682**	**0.589**	**0.393**	**0.659**	**0.664**	**0.547**	**0.488**	**0.509**	**0.509**
$Q2B_{gen}$	0.506	0.619	0.242	0.113	**0.887**	**0.936**	0.426	0.333	0.671	0.327	0.458	0.592	0.267	0.129	**0.789**	**0.870**	0.360	0.282	0.552	0.279
$O2B_{gen}$	0.493	0.641	0.221	0.100	0.876	0.921	0.399	0.317	0.66	0.301	0.447	0.577	0.257	0.114	0.777	0.859	0.359	0.27	0.546	0.264
CQD_{gen}	0.427	0.460	0.150	0.079	0.770	0.830	0.343	0.342	0.618	0.252	0.408	0.539	0.214	0.098	0.710	0.791	0.304	0.302	0.513	0.208
CQD^{ASR}_{gen}	**0.778**	**0.879**	**0.794**	**0.477**	0.883	0.871	**0.788**	**0.726**	**0.835**	**0.749**	**0.628**	**0.733**	**0.640**	**0.413**	0.717	0.720	**0.598**	**0.599**	**0.653**	**0.582**
$Q2B_{spec}$	0.506	0.677	0.229	0.107	**0.893**	**0.936**	0.408	0.327	0.66	0.313	0.456	0.590	0.263	0.122	**0.791**	**0.872**	0.359	0.286	0.548	0.275
$O2B_{spec}$	0.497	0.646	0.228	0.104	0.873	0.919	0.407	0.317	0.666	0.310	0.447	0.576	0.258	0.113	0.776	0.857	0.360	0.27	0.544	0.265
CQD_{spec}	0.436	0.459	0.193	0.097	0.764	0.825	0.378	0.342	0.616	0.252	0.414	0.539	0.240	0.114	0.705	0.787	0.330	0.298	0.511	0.210
CQD^{ASR}_{spec}	**0.793**	**0.881**	**0.806**	**0.483**	0.890	0.878	**0.803**	**0.745**	**0.878**	**0.774**	**0.639**	**0.732**	**0.652**	**0.412**	0.718	0.727	**0.609**	**0.620**	**0.682**	**0.598**
$Q2B_{onto}$	0.818	0.929	0.760	0.482	0.988	0.994	0.877	0.646	0.932	0.751	0.687	0.762	0.617	0.447	0.868	0.915	0.693	0.555	0.732	0.600
$O2B_{onto}$	0.838	**0.960**	0.771	0.514	**0.991**	**0.996**	0.879	0.697	**0.963**	0.768	0.707	**0.771**	0.629	0.476	**0.878**	**0.927**	0.694	0.619	**0.752**	**0.618**
CQD_{onto}	**0.861**	0.901	0.857	**0.552**	0.961	0.979	**0.896**	**0.879**	0.942	0.788	**0.723**	0.752	**0.681**	**0.481**	0.870	0.924	**0.735**	**0.728**	0.738	0.604
CQD^{ASR}_{onto}	0.830	0.902	**0.860**	0.493	0.920	0.915	0.853	0.818	0.908	**0.808**	0.664	0.753	**0.681**	0.421	0.744	0.755	0.643	0.666	0.704	0.615
NELL																				
$Q2B_{plain}$	0.521	0.763	0.401	0.325	0.776	0.832	0.452	0.535	0.337	0.272	0.458	0.516	0.343	0.286	0.747	0.81	0.404	0.447	0.325	0.241
$O2B_{plain}$	0.664	0.816	0.483	**0.413**	**0.961**	**0.975**	0.535	0.648	**0.792**	0.355	**0.596**	**0.79**	0.409	**0.359**	**0.904**	**0.936**	0.479	**0.521**	**0.666**	0.303
CQD_{plain}	0.598	0.710	0.412	0.341	0.891	0.929	0.522	0.593	0.649	0.331	0.555	0.664	0.383	0.304	0.853	0.903	0.471	0.512	0.599	0.306
CQD^{ASR}_{plain}	**0.708**	**0.942**	**0.647**	0.403	0.880	0.909	**0.666**	**0.663**	0.700	**0.570**	0.592	0.716	**0.518**	0.337	0.807	0.831	**0.547**	0.513	0.614	**0.445**
$Q2B_{gen}$	0.734	0.974	0.559	0.466	0.99	0.99	0.622	0.685	0.94	0.377	0.642	0.858	0.485	0.397	0.928	0.95	0.538	0.539	0.768	0.312
$O2B_{gen}$	0.744	0.962	0.572	0.492	0.989	0.990	0.639	0.712	0.944	0.396	0.652	0.859	0.494	0.420	0.928	0.953	0.552	0.559	0.77	0.329
CQD_{gen}	**0.953**	**1.000**	0.996	**0.601**	**1.000**	**1.000**	0.997	**0.999**	**1.000**	0.988	**0.809**	**0.903**	**0.775**	**0.473**	**0.957**	**0.969**	**0.821**	**0.757**	**0.886**	**0.743**
CQD^{ASR}_{gen}	0.949	**1.000**	**0.998**	0.570	**1.000**	**1.000**	**0.998**	0.994	0.997	**0.989**	0.787	0.9	0.771	0.467	0.919	0.924	0.793	0.723	0.846	0.741
$Q2B_{spec}$	0.734	0.974	0.559	0.464	0.99	0.991	0.622	0.684	0.940	0.377	0.641	0.859	0.483	0.397	0.927	0.950	0.538	0.538	0.766	0.313
$O2B_{spec}$	0.745	0.967	0.573	0.493	0.988	0.990	0.639	0.711	0.944	0.397	0.651	0.859	0.494	0.42	0.928	0.954	0.551	0.558	0.771	0.329
CQD_{spec}	**0.953**	**1.000**	0.996	**0.599**	**1.000**	**1.000**	**0.998**	**0.999**	**1.000**	0.988	**0.808**	**0.902**	**0.774**	**0.474**	**0.958**	**0.968**	**0.820**	**0.755**	**0.885**	**0.741**
CQD^{ASR}_{spec}	0.949	**1.000**	**0.998**	0.566	**1.000**	**1.000**	**0.998**	0.993	0.997	**0.990**	0.784	0.899	0.768	0.461	0.919	0.921	0.790	0.721	0.841	0.738
$Q2B_{onto}$	0.725	0.973	0.567	0.466	0.985	0.986	0.606	0.654	0.909	0.384	0.636	0.858	0.472	0.398	0.927	0.948	0.529	0.524	0.747	0.317
$O2B_{onto}$	0.748	0.968	0.598	0.496	0.989	0.988	0.646	0.697	0.941	0.412	0.655	0.862	0.498	0.423	**0.933**	**0.953**	0.557	0.555	0.773	0.340
CQD_{onto} *	0.591	0.659	0.404	0.329	0.896	0.943	0.515	0.611	0.663	0.302	0.545	0.667	0.368	0.293	0.848	0.904	0.453	0.506	0.595	0.275
CQD^{ASR}_{onto}	**0.902**	**0.979**	**0.921**	**0.542**	**0.995**	**0.995**	**0.942**	**0.921**	**0.976**	**0.852**	**0.77**	**0.9**	**0.741**	**0.456**	0.922	0.923	**0.77**	**0.701**	**0.851**	**0.672**

* Trained using 34% fewer queries than CQD_{gen}.

Evaluation of Data Augmentation Strategies The first observation is that incorporating the ontology in the training data is crucial as both $Q2B$ and CQD trained in settings *gen* and *onto* yields significant improvements over the baselines.

However the additional incorporation of specializations (setting *spec*) does not seem to have a major impact (see Table 6.7). On LUBM, for all models, the advantage of the ontology-driven query sampling (i.e. *onto* setting) is significant compared to *gen* setting. Remarkably, for LUBM CQD_{onto}, respectively CQD^{ASR}_{onto} trained on less data than CQD_{gen}, respectively CQD^{ASR}_{gen} results

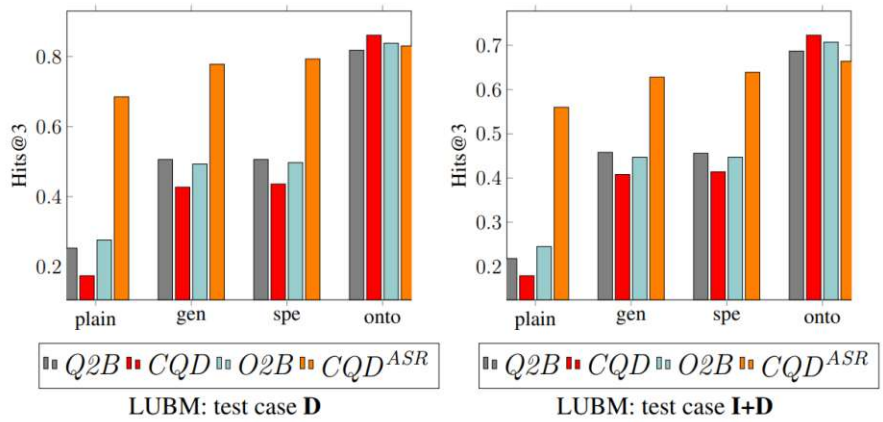


Figure 6.6: Comparison of $Q2B$,$O2B$, CQD and CQD^{ASR} in each training setting for test cases (**D**) and (**I+D**) for LUBM

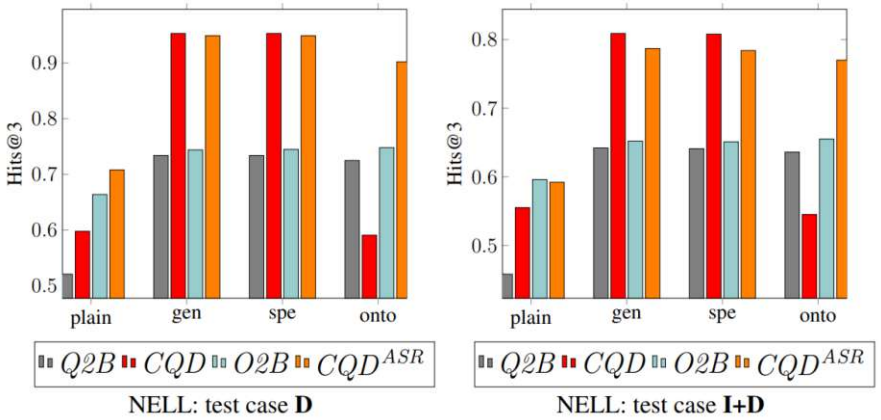


Figure 6.7: Comparison of $Q2B$, $O2B$, CQD and CQD^{ASR} in each training setting for test cases (**D**) and (**I+D**) for NELL

in higher accuracy. This shows that random sampling is not adequate for E-OMQA. For NELL, since the ontology is not expressive enough to create meaningful queries, for *onto* we proceeded in a bottom-up fashion: we randomly labeled query shapes which produce answers, and construct their generalizations as before; thus the settings *gen* and *onto* are similar. As the total number of queries constructed is very large, in order to keep the training set reasonable we chose a significantly smaller number of anchors, which explains why CQD_{gen} outperforms CQD_{onto}.

Evaluation of the Ontology-Aware Training Objective The results of all models on setting *plain* show that the ontology-aware models $O2B$ and CQD^{ASR} show significant improvement over the baselines, meaning that enforcing the rules in the embedding space already leads to better results. However, it is not sufficient, as by simply training on *onto* we obtain significantly better results across both datasets ($Q2B_{onto}$ outperforms $O2B_{plain}$, and CQD_{onto} and CQD_{gen}

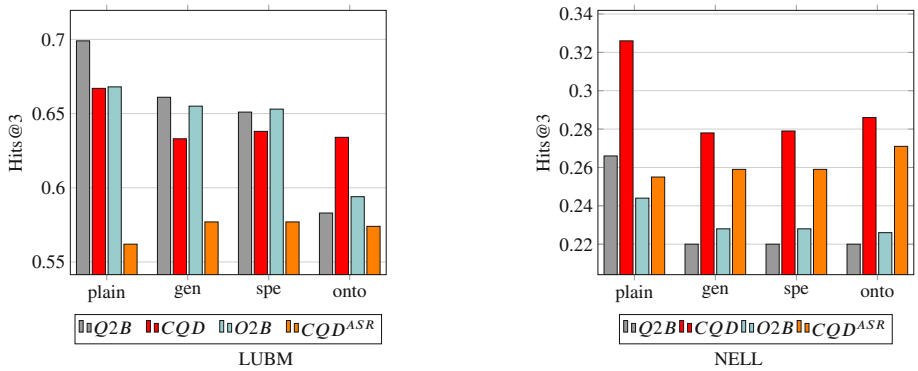


Figure 6.8: Comparison of $Q2B$, $O2B$, CQD and CQD^{ASR} in each train setting for inductive test case (**I**)

Table 6.8: Avg. HITS@3 metric on answering queries of shapes 1p, 2p, 3p, 2i, 3i using rewriting on top of pre-trained *plain* model versus the *plain* model alone.

Models	**Test Case D**		**Test Case I+D**	
	LUBM	NELL	LUBM	NELL
$Q2B_{plain}$	0.189	0.617	0.193	0.539
$Q2B^{rew}_{plain}$	0.248	0.683	0.261	0.639
Gain	+0.059	+0.066	+0.068	+ 0.1
CQD_{plain}	0.225	0.656	0.231	0.621
CQD^{rew}_{plain}	0.228	0.743	0.249	0.708
Gain	+0.003	+0.087	+0.018	+ 0.087

outperform CQD^{ASR}_{plain}). Therefore, the best training strategy for E-OMQA depends on the chosen model, and the ontology language. For complex ontologies, relying on the ontology-driven training strategies yields the best performance, while for less expressive ontologies, generalization-based training strategy is already effective. Moreover, we observe that for query-based models, the rule enforcement offers a significant advantage, while for the atom-based models using ontology-driven training strategy is already sufficient.

6.6 Related Work and Discussion

The task of answering queries that involve multiple atoms using embedding techniques has recently received a lot of attention. The existing proposals can be divided into *query-based* [RHL20, RL20, LDJ+21, CRK+21, KLN21, SAB+20] and *atom-based* [ADMC21]. [FdB20] and [BCL19] study the relation between the problem of conjunctive QA in the embedding space and over probabilistic databases. Our work is different from the above proposals in that along with the data we also rely on ontologies to answer queries.

Integration of ontologies into KG embeddings has been studied by e.g. [KBT15, MDRR17, HCY+19, GWW+16, RSR15, DRR16, KP18, FRP19, ACLS20], but these works do not capture all supported axioms and focus on link prediction rather than QA. The capability of embeddings to model hierarchical data has been explored by [PDB+20, IKA19, GS18]. In particular, [IKA19] aim at interpreting embeddings by finding concept spaces in node embeddings and linking them to a simple external type hierarchy; this is different from our method for OMQA over embeddings. In [GS18], conceptual space representations of known concepts are learned by associating a Gaussian distribution with each concept over a learned vector space. Constructing models for $\mathcal{EL}$ ontologies in the embedding space [KLYH19] is another relevant direction. While [GS18, KLYH19, ÖLW20] are related to our work, they do not touch upon the problem of performing OMQA in the embedding space. The OMQA problem has been actively studied from the data management perspective (see e.g. [SS20] for an overview), but available theoretical and practical methods in this setting only focus on purely logic-based deductive reasoning, without aiming at simultaneously handling missing links.

In this chapter, we have presented methods for ontology-mediated query answering that operate in the embedding space to enable simultaneous inductive and deductive reasoning over the incomplete data. To the best of our knowledge, this is the first work on embedding-based OMQA. We have empirically demonstrated that embedding-based methods for QA applied naively or combined with query rewriting techniques are not effective. In our work, we have proposed alternative solutions for making the existing models ontology-aware via ontology-driven training sampling strategies and loss function modifications. The improvements in the accuracy on prominent query-based and atom-based models range from 20% to 50% compared to the baselines. We believe that this work opens interesting perspectives for combining OMQA methods, with roots in knowledge representation, and embedding techniques originating from the machine learning domain.

CHAPTER 7

Summary and Conclusions

In this thesis we focus on the problem of query answering over incomplete graph-structured data that is mediated by ontologies. In particular, we focus on the querying process from a user's perspective, which can be challenging for non-experts, and on the need to facilitate interactive querying solutions.

In Chapter 1 we motivate, based on a concrete application scenario, the need for more sophisticated tools that make the query process more interactive and help the exploration and analysis of the existing data. In particular we advocate for the need to support the user in formulating queries and exploring the data by means of navigation from one query to another. Furthermore, we also motivate the adaptation of OLAP functionalities to the OBDA context, for which part-of hierarchies and aggregation are crucial.

Based on this envisioned goal, in Chapter 3 we study how to model part-of hierarchies. For that, we need to allow complex role inclusions (CRIs) in the ontology language which can easily lead to undecidability. However their addition to lightweight DLs such as *DL-Lite*$_{\mathcal{A}}$ has not been previously studied. Our results show that, unsurprisingly, without syntactical restrictions, the extension of *DL-Lite*$_{\mathcal{A}}$ with CRIs, denoted as *DL-Lite*$_{\mathcal{A}}^{++}$, is undecidable. Following this result, we consider several restrictions to ensure decidability. Among the decidable fragments are *DL-Lite*$_{\mathcal{A}}^{++(non\text{-}rec)}$ which disallows recursion involving complex relations, and *DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$ which restrict it to involve only known objects or constants. For each we study the data and combined complexity for consistency testing and query answering. The obtained positive results include that the data complexity for such languages can remain AC^0, meaning that OMQs are FO-rewritable.

In Chapter 4, we present an exploratory framework that leverages *DL-Lite*$_{\mathcal{A}}^{++(rec\text{-}safe)}$ ontologies to facilitate OLAP-like functionalities and to support query navigation. The framework involves three main components: a) a query template which allows the user to mark query atoms for specializing or generalizing, b) reformulation rules which apply the ontology axioms to generate a large set of queries that are semantically related and c) a Datalog encoding of the framework that

compiles all queries, their minimal specializations and generalizations as well as their complete answers. We also implemented a prototype using VLog engine and provided an initial evaluation in practice, which shows promising results. An interface to support the creation of the query template and the visualization of answers has also been proposed.

In order to cope with missing information at query time, in Chapter 5 consider conditional query answering and extend the ontology-mediated query formalism to include assumption patterns which when instantiated over existing data, they can produce more informative answers to the users. This problem is related to hypothetical query answering which is a flexible access mechanism in the context of traditional databases. For that, we introduce novel semantics for answering *DL-Lite* OMQs under assumptions, and show that such formalism is FO-rewritable also in when considering closed-predicates in the assumption patterns. Also in this case, we implemented and tested our solution in practice for which we considered that part of the data is remotely available and we rely on assumption patterns to construct ground assumptions that require remote data access. Our results show that such approach is more efficient than evaluating each OMQ as a federated query.

Another proposed solution for dealing with incomplete data, is to rely on existing state-of-the-art KG embedding models to predict answers to OMQs. However, such models are not tailored to KG that are mediated by ontologies, therefore in Chapter 6 we introduce the task of embedding-based ontology-mediated query answering and study how to adapt existing KG embedding models for this task. In contrast to the answer prediction for plain queries, for our novel task it is required that the embedding models can apply both inductive and deductive reasoning, which is much challenging and requires more sophisticated solutions. As proposed approaches, we consider several training strategies for incorporating the ontology into the training set, and as well as adaptation of the loss function to enforce the ontology axioms in the embedding space. We evaluate them on two novel datasets, based on LUBM and NELL KGs. The achieved improvements for the chosen models are from 20% to 55% in HITS@3.

Further Research

The work presented in this thesis is foundational and there are several interesting directions for future work. One of the goals is to create a fully-fledged implementation of the exploratory framework which can support all the described functionalities. This would require intensive work in the design and implementation of such system as well as user studies to assess the usefulness of having such exploratory capabilities.

Another future work regarding the exploratory framework is to consider other ontology languages and queries. For instance investigating other reformulation rules relying on more expressive ontology axioms and as well as extending the framework to support reformulations of regular path queries, while ensuring the feasibility of constructing and navigating the query space.

A possible interesting further research is to combine both techniques to query knowledge graphs, that is ontology-based query answering and embedding-based query answering. Currently, embedding-based models are not able to support arbitrary conjunctive queries or queries with aggregation and classical ontology-based methods are not able to cope with missing links, however

the combination of both is desirable in many scenarios, such as the drug discovery use-case. For instance, using an embedding model we can predict the side effects of a particular substance and filter based on their score, then using classical query answering methods, return the known symptoms associated to the top 5 side effects.

Conditional query answering is another relevant direction for future work and there are several potential applications that could benefit from having a system that can support this. For example, in domains such as military and criminal investigations, "what if" queries are very relevant in assessing targets and identify threats. Therefore more research as well as engineering efforts are required in order to create a system that can be used in practice.

With the increasing relevance of ontologies and knowledge graphs in many applications, having flexible means to access graph-structured data is important to create value from the large amounts of available information. We believe this work contributes to this purpose and opens new perspective and ideas to advance this area of research.

List of Figures

List of Tables

List of Algorithms

Bibliography

[ABW88] Krzysztof R. Apt, Howard A. Blair, and Adrian Walker. Towards a theory of declarative knowledge. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 89–148. Morgan Kaufmann, 1988.

[ACGK+14] Marcelo Arenas, Bernardo Cuenca Grau, Evgeny Kharlamov, Sarunas Marciuska, and Dmitriy Zheleznyakov. Faceted search over ontology-enhanced rdf data. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 939–948, 2014.

[ACKZ09] Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The DL-Lite family and relations. *J. Artif. Intell. Res.*, 36:1–69, 2009.

[ACLS20] Ralph Abboud, İsmail İlkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. BoxE: A box embedding model for knowledge base completion. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[ADMC21] Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. Complex query answering with neural link predictors. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[AGMR15] Elham Akbari Azirani, François Goasdoué, Ioana Manolescu, and Alexandra Roatis. Efficient OLAP operations for RDF analytics. In *ICDE Workshops*, pages 71–76. IEEE Computer Society, 2015.

[AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[AIO20] Medina Andresel, Yazmín Ibáñez-García, and Magdalena Ortiz. A framework for exploratory query answering with ontologies. In *Proceedings of the 33rd International Workshop on Description Logics (DL 2020) co-located with the 17th*

International Conference on Principles of Knowledge Representation and Reasoning (KR 2020), Online Event [Rhodes, Greece], September 12th to 14th, 2020, volume 2663 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.

[AIOS18] Medina Andresel, Yazmín Angélica Ibáñez-García, Magdalena Ortiz, and Mantas Simkus. Taming complex role inclusions for DL-Lite. In *Proceedings of the 31st International Workshop on Description Logics co-located with 16th International Conference on Principles of Knowledge Representation and Reasoning (KR 2018), Tempe, Arizona, US, October 27th - to - 29th, 2018*, volume 2211 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018.

[AIOS19] Medina Andresel, Yazmín Ibáñez-García, Magdalena Ortiz, and Mantas Simkus. Relaxing and restraining queries for OBDA. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 2654–2661. AAAI Press, 2019.

[ANS12] Stefan Anderlik, Bernd Neumayr, and Michael Schrefl. Using domain ontologies as semantic dimensions in data warehouses. In *ER*, volume 7532 of *Lecture Notes in Computer Science*, pages 88–101. Springer, 2012.

[AOS20] Medina Andresel, Magdalena Ortiz, and Mantas Simkus. Query rewriting for ontology-mediated conditional answers. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 2734–2741. AAAI Press, 2020.

[ATD+23] Medina Andresel, Trung-Kien Tran, Csaba Domokos, Pasquale Minervini, and Daria Stepanova. Combining inductive and deductive reasoning for query answering over incomplete knowledge graphs. In Ingo Frommholz, Frank Hopfgartner, Mark Lee, Michael Oakes, Mounia Lalmas, Min Zhang, and Rodrygo L. T. Santos, editors, *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023*, pages 15–24. ACM, 2023.

[Bal03] Matteo Baldoni. *Normal multimodal logics: Automatic deduction and logic programming extension*. PhD thesis, Dipartimento di Informatica — Universit'a degli Studi di Torino, 2003.

[BBL05] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the EL envelope. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005*, pages 364–369. Professional Book Center, 2005.

[BCL19] Stefan Borgwardt, İsmail İlkan Ceylan, and Thomas Lukasiewicz. Ontology-mediated query answering over log-linear probabilistic data. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 2711–2718. AAAI Press, 2019.

[BCM+03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, New York, NY, USA, 2003.

[BGM98] Matteo Baldoni, Laura Giordano, and Alberto Martelli. A tableau for multimodal logics and some (un)decidability results. In *TABLEAUX*, volume 1397 of *Lecture Notes in Computer Science*, pages 44–59. Springer, 1998.

[BHLS17] Franz Baader, Ian Horrocks, Carsten Lutz, and Uli Sattler. *An Introduction to Description Logic*. Cambridge University Press, United Kingdom, 2017.

[BKPR14] Stefan Bischof, Markus Krötzsch, Axel Polleres, and Sebastian Rudolph. Schema-agnostic query rewriting in SPARQL 1.1. In *International Semantic Web Conference (1)*, volume 8796 of *Lecture Notes in Computer Science*, pages 584–600. Springer, 2014.

[BLB08] Franz Baader, Carsten Lutz, and Sebastian Brandt. Pushing the EL envelope further. In *OWLED (Spring)*, volume 496 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[BLMS11] Jean-François Baget, Michel Leclère, Marie-Laure Mugnier, and Eric Salvat. On rules with existential variables: Walking the decidability line. *Artif. Intell.*, 175(9-10):1620–1654, 2011.

[BO15] Meghyn Bienvenu and Magdalena Ortiz. Ontology-mediated query answering with data-tractable description logics. In Wolfgang Faber and Adrian Paschke, editors, *Reasoning Web. Web Logic Rules - 11th International Summer School 2015, Berlin, Germany, July 31 - August 4, 2015, Tutorial Lectures*, volume 9203 of *Lecture Notes in Computer Science*, pages 218–307. Springer, 2015.

[BOS15] Meghyn Bienvenu, Magdalena Ortiz, and Mantas Simkus. Regular path queries in lightweight description logics: Complexity and algorithms. *J. Artif. Intell. Res.*, 53:315–374, 2015.

[BOSX13] Meghyn Bienvenu, Magdalena Ortiz, Mantas Simkus, and Guohui Xiao. Tractable queries for lightweight description logics. In Francesca Rossi, editor, *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, pages 768–774. IJCAI/AAAI, 2013.

[BRC+20] Federico Bianchi, Gaetano Rossiello, Luca Costabello, Matteo Palmonari, and Pasquale Minervini. Knowledge graph embeddings and explainable AI. In *Knowledge Graphs for eXplainable Artificial Intelligence: Foundations, Applications and Challenges*, pages 49–72. 2020.

[BS85] Ronald J. Brachman and James G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cogn. Sci.*, 9(2):171–216, 1985.

[BT20] Franz Baader and Clément Théron. Role-value maps and general concept inclusions in the minimal description logic with value restrictions or revisiting old skeletons in the DL cupboard. *Künstliche Intell.*, 34(3):291–301, 2020.

[BUG+13] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795, 2013.

[CA98] Henning Christiansen and Troels Andreasen. A practical approach to hypothetical database queries. In Burkhard Freitag, Hendrik Decker, Michael Kifer, and Andrei Voronkov, editors, *Transactions and Change in Logic Databases*, pages 340–355, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[CBK+10] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. Toward an architecture for never-ending language learning. In Maria Fox and David Poole, editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*, pages 1306–1313. AAAI Press, 2010.

[CDG+19] David Carral, Irina Dragoste, Larry González, Ceriel J. H. Jacobs, Markus Krötzsch, and Jacopo Urbani. Vlog: A rule engine for knowledge graphs. In *ISWC (2)*, volume 11779 of *Lecture Notes in Computer Science*, pages 19–35. Springer, 2019.

[CDL+07] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning*, 39(3):385–429, 2007.

[CGL+05] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Dl-lite: Tractable description logics for ontologies. In *AAAI*, pages 602–607. AAAI Press / The MIT Press, 2005.

[CGL+07] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning*, 39(3):385–429, 2007.

[CGL09a] Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general datalog-based framework for tractable query answering over ontologies. In *PODS*, pages 77–86. ACM, 2009.

[CGL+09b] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, and Riccardo Rosati. Ontologies and databases: The dl-lite approach. In *Reasoning Web*, volume 5689 of *Lecture Notes in Computer Science*, pages 255–356. Springer, 2009.

[CGL12] Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general datalog-based framework for tractable query answering over ontologies. *J. Web Semant.*, 14:57–83, 2012.

[CGLR12] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. View-based query answering in description logics: Semantics and complexity. *J. Comput. Syst. Sci.*, 78(1):26–46, 2012.

[CGMR14] Dario Colazzo, François Goasdoué, Ioana Manolescu, and Alexandra Roatis. RDF analytics: lenses over semantic graphs. In Chin-Wan Chung, Andrei Z. Broder, Kyuseok Shim, and Torsten Suel, editors, *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014*, pages 467–478. ACM, 2014.

[CGP10] Andrea Calì, Georg Gottlob, and Andreas Pieris. Query answering under non-guarded rules in datalog+/-. In *RR*, volume 6333 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2010.

[CHH10] Catherine Comparot, Ollivier Haemmerlé, and Nathalie Hernandez. An easy way of expressing conceptual graph queries from keywords and query patterns. In *ICCS*, volume 6208 of *Lecture Notes in Computer Science*, pages 84–96. Springer, 2010.

[CKNT08] Diego Calvanese, Evgeny Kharlamov, Werner Nutt, and Camilo Thorne. Aggregate queries over ontologies. In *Proceedings of the 2nd International Workshop on Ontologies and Information Systems for the Semantic Web, ONISW 2008, Napa Valley, California, USA, October 30, 2008*, pages 97–104, 2008.

[COvS13] Diego Calvanese, Magdalena Ortiz, Mantas Šimkus, and Giorgio Stefanoni. Reasoning about explanations for negative query answers in dl-lite. *J. Artif. Intell. Res.*, 48:635–669, 2013.

[CR15] Cristina Civili and Riccardo Rosati. On the first-order rewritability of conjunctive queries over binary guarded existential rules. In Davide Ancona, Marco Maratea, and Viviana Mascardi, editors, *Proceedings of the 30th Italian Conference on Computational Logic, Genova, Italy, July 1-3, 2015*, volume 1459 of *CEUR Workshop Proceedings*, pages 25–30. CEUR-WS.org, 2015.

[CRK+21] Nurendra Choudhary, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan K. Reddy. Self-supervised hyperboloid representations from logical queries over knowledge graphs. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 1373–1384, 2021.

[dCP88] Luis Fariñas del Cerro and Martti Penttonen. Grammar logics. *Logique et Analyse*, 31(121–122):123–134, 1988.

[Dem92] Robert Demolombe. A strategy for the computation of conditional answers. In *ECAI*, pages 134–138, 1992.

[Dem98] Robert Demolombe. Answers about validity and completeness of data: Formal definitions, usefulness and computation technique. In Troels Andreasen, Henning Christiansen, and Henrik Legind Larsen, editors, *Flexible Query Answering Systems*, pages 138–147. Springer Berlin Heidelberg, 1998.

[Dem01] Stéphane Demri. The complexity of regularity in grammar logics and related modal logics. *J. Log. Comput.*, 11(6):933–960, 2001.

[DNPR13] Fariz Darari, Werner Nutt, Giuseppe Pirrò, and Simon Razniewski. Completeness statements about RDF data sources and their use for query answering. In *ISWC (1)*, volume 8218 of *Lecture Notes in Computer Science*, pages 66–83. Springer, 2013.

[DRR16] Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. Lifted rule injection for relation embeddings. In *EMNLP*, pages 1389–1399. The Association for Computational Linguistics, 2016.

[EKS13] Thomas Eiter, Thomas Krennwallner, and Patrik Schneider. Lightweight spatial conjunctive query answering using keywords. In *The Semantic Web: Semantics and Big Data*, pages 243–258. Springer Berlin Heidelberg, 2013.

[EPS+15] Thomas Eiter, Jeff Z. Pan, Patrik Schneider, Mantas Šimkus, and Guohui Xiao. A rule-based framework for creating instance data from openstreetmap. In *RR*, volume 9209 of *Lecture Notes in Computer Science*, pages 93–104. Springer, 2015.

[FdB20] Tal Friedman and Guy Van den Broeck. Symbolic querying of vector spaces: Probabilistic databases meets relational embeddings. In Ryan P. Adams and Vibhav Gogate, editors, *UAI*, pages 1268–1277, 2020.

[FGTT11] Enrico Franconi, Paolo Guagliardo, Marco Trevisan, and Sergio Tessaris. Quelo: an ontology-driven query interface. In Riccardo Rosati, Sebastian Rudolph, and Michael Zakharyaschev, editors, *Proceedings of the 24th International Workshop on Description Logics (DL 2011), Barcelona, Spain, July 13-16, 2011*, volume 745 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.

[FRP19] Bahare Fatemi, Siamak Ravanbakhsh, and David Poole. Improved knowledge graph embedding using background taxonomic information. In *IAAI*, pages 3526–3533, 2019.

[FS99] Enrico Franconi and Ulrike Sattler. A data warehouse conceptual data model for multidimensional aggregation. In *In Proceedings of the Workshop on Design and Management of Data Warehouses (DMDW'99*, 1999.

[GGMO95] D. Gabbay, L. Giordano, A. Martelli, and N. Olivetti. Hypothetical updates, priority and inconsistency in a logic programming language. In *Logic Programming and Nonmonotonic Reasoning*, pages 203–216. Springer Berlin Heidelberg, 1995.

[GGR+18] Enrico Gallinucci, Matteo Golfarelli, Stefano Rizzi, Alberto Abelló, and Oscar Romero. Interactive multidimensional modeling of linked data for exploratory OLAP. *Inf. Syst.*, 77:86–104, 2018.

[GH97] Timothy Griffin and Richard Hull. A framework for implementing hypothetical queries. *SIGMOD Rec.*, 26(2):231–242, June 1997.

[GHM+14] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. Hermit: An owl 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[GHS08] Birte Glimm, Ian Horrocks, and Ulrike Sattler. Unions of conjunctive queries in SHOQ. In *KR*, pages 252–262. AAAI Press, 2008.

[GIKK15] Víctor Gutiérrez-Basulto, Yazmin Angélica Ibáñez-García, Roman Kontchakov, and Egor V. Kostylev. Queries with negation and inequalities over lightweight ontologies. *J. Web Semant.*, 35:184–202, 2015.

[GLHS08] Birte Glimm, Carsten Lutz, Ian Horrocks, and Ulrike Sattler. Conjunctive query answering for the description logic SHIQ. *J. Artif. Intell. Res.*, 31:157–204, 2008.

[GPH05] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. LUBM: A benchmark for OWL knowledge base systems. *J. Web Semant.*, 3(2-3):158–182, 2005.

[GS18] Víctor Gutiérrez-Basulto and Steven Schockaert. From knowledge graph embedding to ontology embedding? An analysis of the compatibility between vector space representations and rules. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona, 30 October - 2 November 2018*, pages 379–388. AAAI Press, 2018.

[GWW+16] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. Jointly embedding knowledge graphs and logical rules. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 192–202, 2016.

[Hal00] Alon Y. Halevy. Theory of answering queries using views. *SIGMOD Rec.*, 29(4):40–47, 2000.

[Hal01] Alon Y. Halevy. Answering queries using views: A survey. *VLDB J.*, 10(4):270–294, 2001.

[HBZ+18] William L. Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. Embedding logical queries on knowledge graphs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 2030–2041, 2018.

[HCY+19] Junheng Hao, Muhao Chen, Wenchao Yu, Yizhou Sun, and Wei Wang. Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 1709–1719, 2019.

[HKS06] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The even more irresistible sroiq. In *Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'06, pages 57–67. AAAI Press, 2006.

[HS04] Ian Horrocks and Ulrike Sattler. Decidability of shiq with complex role inclusion axioms. *Artif. Intell.*, 160(1-2):79–104, December 2004.

[HYY05] Orland Hoeber, Xue Dong Yang, and Yiyu Yao. Conceptual query expansion. In *AWIC*, volume 3528 of *Lecture Notes in Computer Science*, pages 190–196. Springer, 2005.

[IKA19] Maximilian Idahl, Megha Khosla, and Avishek Anand. Finding interpretable concept spaces in node embeddings using knowledge bases. *CoRR*, abs/1910.05030, 2019.

[JLW20] Jean Jung, Carsten Lutz, and Frank Wolter. Least general generalizations in description logic: Verification and existence. In *AAAI 2020, Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, Februrary 7 -12, 2020, New York, New York, US*, 2020.

[Jon75] Neil D. Jones. Space-bounded reducibility among combinatorial problems. *J. Comput. Syst. Sci.*, 11(1):68–85, 1975.

[Kaz08] Yevgeny Kazakov. RIQ and SROIQ are harder than SHOIQ. In *KR*, pages 274–284. AAAI Press, 2008.

[Kaz10] Yevgeny Kazakov. An extension of complex role inclusion axioms in the description logic SROIQ. In *Proc. IJCAR 2010*, 2010.

[KBT15] Denis Krompaß, Stephan Baier, and Volker Tresp. Type-constrained representation learning in knowledge graphs. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*, pages 640–655, 2015.

[KKM+16] Evgeny Kharlamov, Yannis Kotidis, Theofilos Mailis, Christian Neuenstadt, Charalampos Nikolaou, Özgür L. Özçep, Christoforos Svingos, Dmitriy Zheleznyakov, Sebastian Brandt, Ian Horrocks, Yannis E. Ioannidis, Steffen Lamparter, and Ralf Möller. Towards analytics aware ontology based access to static and streaming data. In *International Semantic Web Conference (2)*, volume 9982 of *Lecture Notes in Computer Science*, pages 344–362, 2016.

[KLN21] Bhushan Kotnis, Carolin Lawrence, and Mathias Niepert. Answering complex queries in knowledge graphs with bidirectional sequence encoders. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 4968–4977, 2021.

[KLT+10] Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev. The combined approach to query answering in DL-Lite. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, Toronto, Ontario, Canada, May 9-13, 2010*. AAAI Press, 2010.

[KLYH19] Maxat Kulmanov, Wang Liu-Wei, Yuan Yan, and Robert Hoehndorf. EL embeddings: Geometric construction of models for the description logic EL ++. *CoRR*, abs/1902.10499, 2019.

[KP18] Seyed Mehran Kazemi and David Poole. SimplE embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 4289–4300, 2018.

[KR13] Egor V. Kostylev and Juan L. Reutter. Answering counting aggregate queries over ontologies of the DL-Lite family. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*. AAAI Press, 2013.

[LDJ+21] Lihui Liu, Boxin Du, Heng Ji, ChengXiang Zhai, and Hanghang Tong. Neural-answering logical queries on knowledge graphs. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 1087–1097, 2021.

[LP82] Harry R. Lewis and Christos H. Papadimitriou. Symmetric space-bounded computation. *Theor. Comput. Sci.*, 19:161–187, 1982.

[LSW13] Carsten Lutz, Inanç Seylan, and Frank Wolter. Ontology-based data access with closed predicates is inherently intractable(sometimes). In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, pages 1024–1030. IJCAI/AAAI, 2013.

[LSW15] Carsten Lutz, Inanç Seylan, and Frank Wolter. Ontology-mediated queries with closed predicates. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 3120–3126. AAAI Press, 2015.

[LSW19] Carsten Lutz, Inanç Seylan, and Frank Wolter. The data complexity of ontology-mediated queries with closed predicates. *Log. Methods Comput. Sci.*, 15(3), 2019.

[LUO18] Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. Canonical tensor decomposition for knowledge base completion. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 2869–2878. PMLR, 2018.

[MDRR17] Pasquale Minervini, Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. Adversarial sets for regularising neural link predictors. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*. AUAI Press, 2017.

[Mot84] Amihai Motro. Query generalization: A method for interpreting null answers. In *Expert Database Workshop*, pages 597–616. Benjamin/Cummings, 1984.

[MT14] Davide Martinenghi and Riccardo Torlone. Taxonomy-based relaxation of query answering in relational databases. *VLDB J.*, 23(5):747–769, 2014.

[NAS12] Bernd Neumayr, Stefan Anderlik, and Michael Schrefl. Towards ontology-based OLAP: datalog-based reasoning over multidimensional ontologies. In *DOLAP 2012, ACM 15th International Workshop on Data Warehousing and OLAP, Maui, HI, USA, November 2, 2012, Proceedings*, pages 41–48. ACM, 2012.

[NMTG16] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proc. IEEE*, 104(1):11–33, 2016.

[NOv16] Nhung Ngo, Magdalena Ortiz, and Mantas Šimkus. Closed predicates in description logics: Results on combined complexity. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016*, pages 237–246. AAAI Press, 2016.

[Nut11] Premchand Nutakki. Specializing conjunctive queries in the el-family for better comprehension of result sets. Master's thesis, 2011.

[ÖLW20] Özgür Lütfü Özçep, Mena Leemhuis, and Diedrich Wolter. Cone semantics for logics with negation. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 1820–1826. ijcai.org, 2020.

[ORS10] Magdalena Ortiz, Sebastian Rudolph, and Mantas Simkus. Worst-case optimal reasoning for the horn-dl fragments of OWL 1 and 2. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, Toronto, Ontario, Canada, May 9-13, 2010*. AAAI Press, 2010.

[ORS11] Magdalena Ortiz, Sebastian Rudolph, and Mantas Simkus. Query answering in the horn fragments of the description logics SHOIQ and SROIQ. In *IJCAI 2011,*

Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011, pages 1039–1044. IJCAI/AAAI, 2011.

[Ort10] Magdalena Ortiz. *Query Answering in Expressive Description Logics - Techniques and Complexity Results*. PhD thesis, TU Wien, 2010.

[Ort13] Magdalena Ortiz. Ontology based query answering: The story so far. In *Proceedings of the 7th Alberto Mendelzon International Workshop on Foundations of Data Management, Puebla/Cholula, Mexico, May 21-23, 2013*, volume 1087 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2013.

[OS12] Magdalena Ortiz and Mantas Simkus. Reasoning and query answering in description logics. In *Reasoning Web*, volume 7487 of *Lecture Notes in Computer Science*, pages 1–53. Springer, 2012.

[Pap94] Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

[PDB+20] Dhruvesh Patel, Shib Sankar Dasgupta, Michael Boratko, Xiang Li, Luke Vilnis, and Andrew McCallum. Representing joint hierarchies with box embeddings. In *Conference on Automated Knowledge Base Construction, AKBC 2020, Virtual, June 22-24, 2020*, 2020.

[PHH11] Camille Pradel, Ollivier Haemmerlé, and Nathalie Hernandez. Expressing conceptual graph queries from patterns: How to take into account the relations. In *Conceptual Structures for Discovering Knowledge - 19th International Conference on Conceptual Structures, ICCS 2011, Derby, UK, July 25-29, 2011. Proceedings*, volume 6828 of *Lecture Notes in Computer Science*, pages 229–242. Springer, 2011.

[PLC+08] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Linking data to ontologies. In Stefano Spaccapietra, editor, *Journal on Data Semantics X*, volume 4900 of *Lecture Notes in Computer Science*, pages 133–173. Springer Berlin Heidelberg, 2008.

[RHL20] Hongyu Ren, Weihua Hu, and Jure Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[RKZ13] Mariano Rodriguez-Muro, Roman Kontchakov, and Michael Zakharyaschev. Ontology-based data access: Ontop of databases. In *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I*, pages 558–573, 2013.

[RL20] Hongyu Ren and Jure Leskovec. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[Ros07] Riccardo Rosati. On conjunctive query answering in EL. In *Proceedings of the 2007 International Workshop on Description Logics (DL2007), Brixen-Bressanone, near Bozen-Bolzano, Italy, 8-10 June, 2007*, volume 250 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[RSR15] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting logical background knowledge into embeddings for relation extraction. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 1119–1129, 2015.

[SAB+20] Haitian Sun, Andrew O. Arnold, Tania Bedrax-Weiss, Fernando Pereira, and William W. Cohen. Faithful embeddings for knowledge base queries. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[Sat00] Ulrike Sattler. Description logics for the representation of aggregated objects. In *ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, Germany, August 20-25, 2000*, pages 239–243. IOS Press, 2000.

[SBN+21] Christoph G. Schuetz, Loris Bozzato, Bernd Neumayr, Michael Schrefl, and Luciano Serafini. Knowledge graph OLAP. *Semantic Web*, 12(4):649–683, 2021.

[SEI+18] Marta Sabou, Fajar J. Ekaputra, Tudor B. Ionescu, Juergen Musil, Daniel Schall, Kevin Haller, Armin Friedl, and Stefan Biffl. Exploring enterprise knowledge graphs: A use case in software engineering. In *ESWC*, volume 10843 of *Lecture Notes in Computer Science*, pages 560–575. Springer, 2018.

[SGKK17] Evgeny Sherkhonov, Bernardo Cuenca Grau, Evgeny Kharlamov, and Egor V. Kostylev. Semantic faceted search with aggregation and recursion. In *International Semantic Web Conference (1)*, volume 10587 of *Lecture Notes in Computer Science*, pages 594–610. Springer, 2017.

[SKZ+14] Ahmet Soylu, Evgeny Kharlamov, Dmitriy Zheleznyakov, Ernesto Jimenez-Ruiz, Martin Giese, and Ian Horrocks. Optiquevqs: Visual query formulation for obda. In *Proceedings of the 27th International Workshop on Description Logics (DL 2014)*, volume 1193, pages 725–728, Vienna, Austria, 2014.

[SN07] P. K. Srimani and S. F. B. Nasir. *Context-Free Grammars and Context-Free Languages*, page 304–376. Foundation Books, 2007.

[SPG+07] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical owl-dl reasoner. *Web Semant.*, 5(2):51–53, June 2007.

[SS89] Manfred Schmidt-Schaubß. Subsumption in KL-ONE is undecidable. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 421–431, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[SS91] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with complements. *Artif. Intell.*, 48(1):1–26, 1991.

[SS20] Thomas Schneider and Mantas Simkus. Ontologies and data management: A brief survey. *Künstliche Intell.*, 34(3):329–353, 2020.

[SSS09] Dimitrios Skoutas, Alkis Simitsis, and Timos K. Sellis. Ontology-driven conceptual design of ETL processes using graph transformations. *J. Data Semant.*, 13:120–146, 2009.

[tCCST15] Balder ten Cate, Cristina Civili, Evgeny Sherkhonov, and Wang-Chiew Tan. High-level why-not explanations using ontologies. In *PODS*, pages 31–43. ACM, 2015.

[TWR+16] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org, 2016.

[VAHL19] Hernán Vargas, Carlos Buil Aranda, Aidan Hogan, and Claudia López. RDF explorer: A visual SPARQL query builder. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part I*, volume 11778 of *Lecture Notes in Computer Science*, pages 647–663. Springer, 2019.

[Var82] Moshe Y. Vardi. The complexity of relational query languages (extended abstract). In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, pages 137–146. ACM, 1982.

[W3C13] W3C. Sparql 1.1 federated query. w3c recommendation 21 march 2013, 2013. https://www.w3.org/TR/sparql11-federated-query/.

[WMWG17] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*, 29(12):2724–2743, 2017.

[WQW21] Meihong Wang, Linling Qiu, and Xiaoli Wang. A survey on knowledge graph embeddings for link prediction. *Symmetry*, 13(3):485, 2021.

[YBRW16] Mohamed Yahya, Klaus Berberich, Maya Ramanath, and Gerhard Weikum. Exploratory querying of extended knowledge graphs. *Proc. VLDB Endow.*, 9(13):1521–1524, 2016.

[YYH+15] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[ZCZ+20] Jing Zhang, Bo Chen, Lingxi Zhang, Xirui Ke, and Haipeng Ding. Neural-symbolic reasoning on knowledge graphs. *CoRR*, abs/2010.05446, 2020.

[ZGN+15] Yujiao Zhou, Bernardo Cuenca Grau, Yavor Nenov, Mark Kaminski, and Ian Horrocks. Pagoda: Pay-as-you-go ontology query answering using a datalog reasoner. *J. Artif. Intell. Res. (JAIR)*, 54:309–367, 2015.