IOWA STATE UNIVERSITY
Digital Repository

Graduate Theses and Dissertations

Iowa State University Capstones, Theses and Dissertations

2016

Secrecy-preserving reasoning in simple description logic knowledge bases

Gopalakrishnan Krishnasamy Sivaprakasam
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/etd

Part of the Artificial Intelligence and Robotics Commons

Recommended Citation
Krishnasamy Sivaprakasam, Gopalakrishnan, "Secrecy-preserving reasoning in simple description logic knowledge bases" (2016). *Graduate Theses and Dissertations*. 15026.
https://lib.dr.iastate.edu/etd/15026

This Dissertation is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

Secrecy-preserving reasoning in simple description logic knowledge bases

by

Gopalakrishnan Krishnasamy Sivaprakasam

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Science

Program of Study Committee:

Giora Slutzki, Major Professor

Les Miller

Carl K Chang

Samik Basu

Ananda Weerasinghe

Iowa State University

Ames, Iowa

2016

Copyright © Gopalakrishnan Krishnasamy Sivaprakasam, 2016. All rights reserved.

TABLE OF CONTENTS

LIST OF FIGURES iv

ACKNOWLEDGEMENTS v

ABSTRACT vi

CHAPTER 1. INTRODUCTION 1

1.1 Description Logics 2

1.2 Secrecy-preserving Query Answering Problem 3

CHAPTER 2. SECRECY-PRESERVING REASONING IN ACYCLIC $DL\text{-}Lite_{\mathcal{R}}$ **KNOWLEDGE BASES** 7

2.1 Introduction 7

2.2 Preliminaries - Syntax and semantics of $DL\text{-}Lite_{\mathcal{R}}$ 8

2.2.1 Syntax 8

2.2.2 Semantics 9

2.3 Computation of $\mathcal{A}^*$ 11

2.4 Graph representation of ABoxes and BCQs over $DL\text{-}Lite_{\mathcal{R}}$ KBs 18

2.5 Secrecy-Preserving Reasoning in $DL\text{-}Lite_{\mathcal{R}}$ KBs 22

2.6 Answering Queries 31

2.7 Complexities of computing $\mathcal{A}^*$, $\mathbb{E}$ and Query Answering 32

2.8 Conclusions 33

CHAPTER 3. SECRECY-PRESERVING QUERY ANSWERING IN $\mathcal{ELH}$ **KNOWLEDGE BASES** 34

3.1 Introduction 34

3.2 Syntax and Semantics 36

3.3 Computation of $\mathcal{A}^*$ and $\mathcal{T}^*$ 37

3.4 Secrecy-Preserving Reasoning 46

3.5 Query Answering 52

3.6 Conclusions 57

CHAPTER 4. KEEPING SECRETS IN MODALIZED DL KNOWLEDGE BASES 58

4.1 Introduction 58

4.2 Syntax and Semantics of $\mathcal{ELH}^{\Diamond}$ 60

4.3 Computation of a Model for $\mathcal{ELH}^{\Diamond}$ KB Σ and $\mathcal{A}^*$ 61

4.4 Secrecy-Preserving Reasoning in $\mathcal{ELH}^{\Diamond}$ KBs 70

4.5 Query Answering 75

4.6 Conclusions 78

CHAPTER 5. SUMMARY AND DISCUSSION 79

APPENDIX A. ADDITIONAL MATERIAL 81

A.1 Additional Material for Chapter 2 81

A.1.1 Proof of Lemma 2.3.1 81

A.1.2 Proof of Lemma 2.3.2 83

A.1.3 Proof of Lemma 2.3.3 85

BIBLIOGRAPHY 89

LIST OF FIGURES

Figure 2.1 Expansion rules for computing $\mathcal{A}_1^*$ 13

Figure 2.2 Expansion rules for computing $\mathcal{A}_{12}^*$ 13

Figure 2.3 Expansion rules for computing $\mathcal{A}^*$ 14

Figure 2.4 The ABox and query graphs 19

Figure 2.5 Secrecy closure rules obtained by inverting rules in Figures 2.1 and 2.2 23

Figure 2.6 Secrecy closure rules obtained by inverting rules in Figure 2.3 24

Figure 2.7 Secrecy closure rules for $q \in S_{CQ}$ 25

Figure 2.8 The graphs of $\mathcal{A}^*$ and $\mathcal{A}^* \setminus \mathbb{E}$ 26

Figure 2.9 The graphs of $\mathcal{A}^* \setminus \mathbb{E}$ and queries 31

Figure 3.1 TBox Tableau expansion rules 38

Figure 3.2 ABox Tableau expansion rules. 42

Figure 3.3 Inverted ABox Tableau expansion rules 47

Figure 3.4 Inverted TBox Tableau expansion rules. 49

Figure 3.5 Query answering algorithm for assertional queries 51

Figure 3.6 Query answering algorithm for GCI queries 55

Figure 4.1 Local expansion rules 63

Figure 4.2 Global expansion rules 64

Figure 4.3 Completed constraint tree $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ 64

Figure 4.4 Inverted local expansion rules 71

Figure 4.5 Inverted global expansion rule 72

Figure 4.6 Secrecy-preserving tree $\mathbb{T}_E = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L}_E \rangle$ 73

Figure 4.7 Query answering algorithm for assertional queries 76

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and thanks to my advisor Dr. Giora Slutzki, for everything that he taught me with kindness and patience. Without his insight, guidance, effort and encouragement, this dissertation would not have been possible. I am also grateful to him for his effort to help me to get the financial support during the entire period that I worked with him.

I would also like to thank my committee members for their encouragement, insightful comments and time: Dr. Les Miller, Dr. Carl K Chang, Dr. Samik Basu and Dr. Ananda Weerasinghe.

I would additionally like to thank Dr. Jin Tian for his encouragement and help to continue to pursue this program.

ABSTRACT

In this dissertation, we study the problem of secrecy-preserving query answering (SPQA) against knowledge bases (KBs) under the open world assumption (OWA) - the assumption that typical KBs are incomplete. Protection of secret information is a critical requirement for the design of information systems in semantic web applications. Recently, semantic web technologies are widely used in many application domains like healthcare, bioinformatics, intelligence and national security. So, there is a pressing need for developing robust secret protection mechanisms suitable for ontology-based information systems. In our work, we use a logical approach to enforce secrecy where the domain knowledge is represented in an appropriate description logic (DL). In particular, to protect secret information we take advantage of OWA. Under OWA, a querying agent cannot distinguish whether a query is being protected or it cannot be inferred from the KB. The central idea in our approach to protect the secret information is to build a logical shield called "envelope" around the confidential information and answers queries correctly as much as possible without compromising the secrecy.

We have chosen lightweight DL languages like $DL\text{-}Lite_{\mathcal{R}}$ and $\mathcal{ELH}$ for studying SPQA problem with single querying agent in the first half of this dissertation. We have considered $DL\text{-}Lite_{\mathcal{R}}$ KB with acyclic TBox and the secrecy set containing both assertional queries and Boolean Conjunctive Queries (BCQs). By computing a suitable envelope, we protect the secrets in the secrecy set. We have used Kleenes 3-valued semantics to prove the correctness of the query answering procedure. We have also performed a detailed analysis of computational complexities of various algorithms used in this dissertation. In $\mathcal{ELH}$ logic, we define a secrecy set that contains both assertional and general concept inclusion queries. A new strategy has been employed to construct the SPQA system for the given $\mathcal{ELH}$ KB. This includes designing efficient query answering algorithms based on recursive decomposition of queries and have shown that the query answering algorithms are sound and complete, thus providing correctness

proof. In the second half of this dissertation, we have studied the SPQA problem in $\mathcal{ELH}^{\Diamond}$ ($\mathcal{ELH}$ augmented with modal operator $\Diamond$). Given a $\mathcal{ELH}^{\Diamond}$ KB and a finite secrecy set, we compute a SPQA system in the form of a tree, called secrecy-preserving tree. In this case the secrecy set contains only assertions. Since the information available in secrecy-preserving tree is not sufficient to answer all the queries, we further augment the query answering procedure with a recursive procedure. The recursive procedure is based on th idea of breaking the query into smaller assertions all the way until the information in the secrecy-preserving tree can be used.

CHAPTER 1. INTRODUCTION

The explosive growth in online banking activities, social networks, web based travel services and other internet based business and homeland security applications contain massive amounts of private data of users, administrators, service providers and governmental agencies. This contributes, on one hand, to unprecedented levels of data sharing and, on the other hand, to grave concerns about privacy and confidentiality of communication between WWW users. It will be an indispensable aspect of future web based service industry that private data while being shared must remain inviolate. In these applications, automated reasoning plays a central role in processing a large volume of data in an automated way for (a) storing data in a particular format; and (b) protecting the private data and sharing the public data. Automated reasoning (Knowledge representation and reasoning) is a branch of artificial intelligence that is concerned with representing knowledge in some well known logical languages and manipulating the represented knowledge, by reasoning, to generate new knowledge Brachman et al. (1992); Fagin et al. (2003). One of the crucial tasks in automated reasoning is selecting a suitable logical language that has nice modeling and computability properties to represent the domain knowledge. The well known logical languages like propositional logic and first order logic (FOL) are not good candidates to represent knowledge for the many application domains because they lack either good modeling or efficient computability properties. In order to represent the domain knowledge for wide range of applications and to study the reasoning tasks efficiently, a family of formal languages called description logics have been considered, see Baader (2003); Hitzler et al. (2009); Calvanese et al. (2007).

1.1 Description Logics

Description Logics (DLs) Baader (2003) are a decidable fragments of FOL. DLs are a family of logic based knowledge representation formal languages which have an adequate modeling and good computability properties. That is, DLs have the reasonable balance between expressivity and efficiency of reasoning. Further, DLs are considered to be underlying logics of Web Ontology Languages (OWLs) Krötzsch (2012) which are recommended by the WWW consortium (W3C) as a knowledge representation languages for the web.

Generally, DLs have been classified based on their expressivity and computational complexity of reasoning tasks Hitzler et al. (2009); Ortiz and Šimkus (2012); Krötzsch (2012). There are *expressive* DLs and *lightweight* DLs. Expressive DLs are very expressive, and the complexity of the reasoning tasks in these DLs tends to be high. The prototypical $\mathcal{ALC}$ language and its extensions are examples of expressive DLs. On the other hand, lightweight DLs have been designed to have limited expressive power in order to achieve lower computational complexity for the reasoning tasks. The important and widely used lightweight DLs are $DL\text{-}Lite_{\mathcal{R}}$, $\mathcal{EL}$, $\mathcal{ELH}$, $\mathcal{EL}^{+}$ and their extensions.

In this dissertation, we choose to use lightweight description logic to represent the domain knowledge and study secrecy related reasoning problems. The reason we have chosen lightweight description logic languages like DL - $Lite_{\mathcal{R}}$, $\mathcal{EL}$ and $\mathcal{ELH}$ for representing knowledge is that, on one hand, they have sufficient expressive power for many applications and, on the other hand, the corresponding reasoning tasks can be done efficiently. There are two more practical reasons for selecting these languages: (a) reasoning systems are available for these languages. QuOnto Acciarri et al. (2005) is the standard reasoning system for the language $DL\text{-}Lite_{\mathcal{R}}$, and CEL Mendez and Suntisrivaraporn (2009) is the reasoning system for $\mathcal{EL}$ family of languages, and (b) the popular ontologies like Systemized Nomenclature of Medicine (SNOMED) and Galen Medical Knowledge Base (GALEN) for medical domain, and the Gene Ontology (GO) for bioinformatics domain are designed and developed using the DLs $\mathcal{EL}$ or its extensions, for details see Baader et al. (2006). Further, we have considered a language with modal operator $\mathcal{ELH}^{\Diamond}$ to represent specific domain information. For instance, the statement 'It is possible

that Joe is a teacher' can be modeled as $\Diamond teacher(Joe)$ in $\mathcal{ELH}^{\Diamond}$ language. $\mathcal{ELH}^{\Diamond}$ is the DL $\mathcal{ELH}$ augmented with the modal operator $\Diamond$. The following are some of the reported works in modalized DLs. Lutz et al. (2001) studied the satisfiability problem in Modalized $\mathcal{ALC}$, and Tao et al. (2012) studied the query answering problem in Modalized $\mathcal{ALC}$.

1.2 Secrecy-preserving Query Answering Problem

Preserving secrecy in a database setting is a problem of paramount importance and it has been studied for a long time, see Biskup and Weibert (2008); Biskup et al. (2010); Sicherman et al. (1983). With the advent of the semantic web and its increasingly pervasive usage, there is a lot of interest in studying this problem in knowledge base (KB) setting, see Bao et al. (2007); Grau et al. (2013); Tao et al. (2010, 2014). The concern here is that in view of the fundamental assumption that KBs possess incomplete knowledge, a situation could arise in which logical reasoning (used to produce implicit knowledge from the explicit one stored in the KB) may possibly lead to disclosure of secret information, see Grau et al. (2013). Some approaches dealing with "information protection" are based on access control mechanisms Bell and LaPadula (1973), defining appropriate policy languages to represent obligation, provision and delegation policies Kagal et al. (2003), and based on theory of knowledge (epistemology) Halpern and O'Neill (2005).

One approach to secrecy in incomplete database was presented in Biskup and Weibert (2008); Biskup et al. (2010); Biskup and Tadros (2012) in the form of controlled query evaluation (CQE). The idea behind CQE is that rather than providing strict access control to data, the CQE approach enforces secrecy by checking (at run time) whether from a truthful answer to a query a user can deduce secret information. In this case the answer is distorted by either simply refusing to answer or by outright lying. For a study of confidentiality in a setting that is an adaptation of CQE framework to ontologies over OWL 2 RL profile of OWL 2, see Grau et al. (2013).

In response to concerns raised in Weitzner et al. (2008), the authors in Bao et al. (2007); Tao et al. (2010, 2014), have developed a secrecy framework that attempts to satisfy the following competing goals: (a) it protects secret information and (b) queries are answered as informatively

as possible (subject to satisfying property (a)). The notion of an *envelope* to hide secret information against logical inference was first defined and used in Tao et al. (2010). In Tao et al. (2014), the authors introduced a more elaborate conceptual framework for secrecy-preserving query answering (SPQA) under Open World Assumption (OWA) with multiple querying agents. This approach is based on OWA and (so far) it has been restricted to instance-checking queries. Specifically, in Bao et al. (2007); Tao et al. (2010, 2014) the main idea was to utilize the secret information within the reasoning process, but then answering "Unknown" whenever the answer is truly unknown or in case the true answer could compromise confidentiality. Further, in Krishnasamy Sivaprakasam and Slutzki (2016), the authors extended the work of Tao et al., reported in Tao et al. (2010), to the $\mathcal{ELH}$ language and studied secrecy in the context of assertions as well as general concept inclusions (GCIs).

An approach of SPQA system adapted in this dissertation is explained as follows:

(a) given a KB Σ in the language of our choice, we design a sound and complete tableau algorithm with some suitable restrictions to compute a finite set of assertional consequences ($\mathcal{A}^*$) or GCI consequences ($\mathcal{T}^*$) of the KB Σ;

(b) given Σ and a secrecy set $\mathbb{S}$, we compute an envelope $\mathbb{E}$ of $\mathbb{S}$ by inverting the rules of tableau algorithm considered in (a), for more details see Tao et al. (2010, 2014). The idea behind the envelope concept is that no expression in the envelope can be logically deduced from information outside the envelope. Once such envelopes are computed, the answers to the queries are censored whenever the queries belong to the envelopes. Since, generally, an envelope for a given secrecy set is not unique, the developer has some freedom to output an envelope from the available choices, depending on his/her needs or preferences; and

(c) given the consequences of Σ (here we consider $\mathcal{A}^*$) and the envelope $\mathbb{E}$ for the secrecy set $\mathbb{S}$, we answer queries without revealing secrets. Usually in SPQA framework queries are answered by checking their membership in $\mathcal{A}^* \setminus \mathbb{E}$. Since, generally, $\mathcal{A}^*$ does not contain all the statements entailed by Σ, we need to extend the query answering procedure from just membership checking. Towards that end we have designed recursive algorithms to answer

more complicated queries. To answer an assertion query q, the algorithm first checks if $q \in \mathcal{A}^* \setminus \mathbb{E}$ in which case the answer is "Yes"; otherwise, the given query is broken into subqueries based on the constructors, and the algorithm is applied recursively on the subqueries based on the constructors defined in the language.

In this dissertation, we have studied the SPQA problem for single querying agent in four different languages. In Chapter 2, we consider the lightweight DL *DL-Lite*$_{\mathcal{R}}$ to study SPQA problem with secrecy set consisting of both assertional queries and Boolean conjunctive queries (BCQs). In this case, we assume that *TBox* $\mathcal{T}$, a finite set of GCIs, is acyclic. Using a tableau algorithm, we construct $\mathcal{A}^*$, an inferential closure of the given *ABox* $\mathcal{A}$, a finite set of assertions, which includes both positive as well as negative assertions. Note that $\mathcal{A}^*$ contains all the consequences of Σ. We use a notational variant of Kleene 3-valued semantics, which we call OW-semantics as it fits nicely with OWA. This allows us to answer queries, including Boolean Conjunctive Queries (BCQs) with "Yes", "No" or "Unknown", as opposed to answering just "Yes" or "No" as in Ontology Based Data Access (OBDA) framework, see Calvanese et al. (2007), thus improving the informativeness of the query-answering procedure. One of the main contributions of this work is a study of answering BCQs without compromising secrecy. Using the idea of secrecy envelopes, we give a precise characterization of when BCQs should be answered "Yes", "No" or "Unknown". We prove the correctness of the secrecy-preserving query-answering algorithm and briefly discuss its computational complexity.

In Chapter 3, the SPQA problem under OWA for $\mathcal{ELH}$ KBs has been presented. In this work, we allow the querying agents to answer both assertional and GCI queries. We employ two efficient tableau procedures designed to compute some consequences of ABox ($\mathcal{A}$) and TBox ($\mathcal{T}$) denoted by $\mathcal{A}^*$ and $\mathcal{T}^*$ respectively. A secrecy set of a querying agent is subset $\mathbb{S}$ of $\mathcal{A}^* \cup \mathcal{T}^*$ which the agent is not allowed to access. Once envelopes are computed, they are used to efficiently answer assertional and GCI queries without compromising the secret information in $\mathbb{S}$. Answering GCI queries while preserving secrecy has not been studied in the current literature. Since we are not computing all the consequences of the KB, answers to the queries based on just $\mathcal{A}^*$ and $\mathcal{T}^*$ could be erroneous. To fix this problem, we further augment our algorithms to make the query answering procedure foolproof. The augmented query answering

procedures are designed based on the idea of breaking the queries into smaller assertions or GCIs all the way until the information in the sets $\mathcal{A}^*$ and $\mathcal{T}^*$ can be used. Further, we prove that the query answering algorithm is correct and show that it is efficient.

In Chapter 4, we have studied SPQA problem under OWA for $\mathcal{ELH}^{\Diamond}$ KBs. Here $\mathcal{ELH}^{\Diamond}$ is a description logic $\mathcal{ELH}$ augmented with a modal operator $\Diamond$. We employ a tableau procedure designed to compute a rooted labeled tree $\mathbb{T}$ which contains information about some assertional consequences of the given KB. Given a secrecy set $\mathbb{S}$, which is a finite set of assertions, we compute a function E, called an envelope of $\mathbb{S}$, which assigns a set of assertions to each node of $\mathbb{T}$. E provides logical protection to the secrecy set $\mathbb{S}$ against the reasoning of a querying agent. Once the tree $\mathbb{T}$ and an envelope E are computed, we define the secrecy-preserving tree $\mathbb{T}_E$. Based on the information available in $\mathbb{T}_E$, assertional queries with modal operator $\Diamond$ can be answered efficiently while preserving secrecy. To the best of our knowledge, this work is the first one studying secrecy-preserving reasoning in description logic augmented with a modal operator. Since we are not computing all the consequences of the knowledge base, answers to the queries based on just secrecy-preserving tree $\mathbb{T}_E$ could be erroneous. To fix this problem, we further augment our algorithms by providing recursive query decomposition algorithm to make the query answering procedure foolproof.

CHAPTER 2. SECRECY-PRESERVING REASONING IN ACYCLIC $DL\text{-}Lite_{\mathcal{R}}$ KNOWLEDGE BASES

2.1 Introduction

Recently, the World Wide Web Consortium (W3C) has proposed OWL 2 profiles which have limited modeling features, but provide substantial improvements in scalability as well as a significant reduction in the complexity of various reasoning tasks. Based on this proposal, there has been a lot of work done on developing languages tailored to specific applications, in particular those that involve massive amount of data, i.e., large ABoxes. In addition, a lot of work has dealt with answering conjunctive queries over these data sets, see Ortiz and Šimkus (2012). The goal is to provide just enough expressive power to deal with those applications, while keeping low complexity of reasoning, see Calvanese et al. (2007); Krötzsch (2012). $DL-Lite$ family is one such family of languages designed with an eye towards precisely these kinds of applications, see Artale et al. (2009); Calvanese et al. (2007); Ortiz and Šimkus (2012).

In this chapter we continue the work begun in Bao et al. (2007); Tao et al. (2010). The framework introduced in Tao et al. (2014), which we use here as well, was illustrated on very simple examples: the Propositional Horn Logic and the Description Logic $\mathcal{AL}$. Here we consider the SPQA problem under OWA for $DL\text{-}Lite_{\mathcal{R}}$ *acyclic* KBs. In contrast to previous work, an important contribution here is that we allow Boolean Conjunctive Assertions/Queries (BCQs) both in the specification of secrets as well as in the queries. Given a $DL\text{-}Lite_{\mathcal{R}}$ KB (consisting of an ABox $\mathcal{A}$ and an acyclic TBox $\mathcal{T}$) and a secrecy set $\mathbb{S}$ consisting of both instance-assertions as well as BCQs, the querying agent is allowed to ask queries of both kinds. Moreover, we allow the ABox of the KB to contain both positive and negative assertions, see Artale et al. (2009). By OWA, the answer to a query against a KB can be "Yes", "No" or "Unknown". As the first

step in constructing our SPQA system, we use a tableau algorithm to compute a finite set $\mathcal{A}^*$ which consists of the consequences of the KB (with respect to the TBox), both positive and negative. To prove the completeness of this algorithm, we use the 3-valued OW-semantics as introduced in Tao et al. (2014), see also Section 2.2.2. Next, starting from the secrecy set $\mathbb{S}$ we compute a finite set of assertions, viz., the *envelope* $\mathbb{E} \subseteq \mathcal{A}^*$ of the secrecy set $\mathbb{S}$, whose goal is to provide a "logical shield" against reasoning launched from $\mathcal{A}^* \setminus \mathbb{E}$ (outside the envelope) and whose aim is to "penetrate" the secrecy set $\mathbb{S}$ (i.e., to figure out which assertions are in $\mathbb{S}$). Computation of the envelope is based on the ideas given in Tao et al. (2010, 2014), viz., inversion of the tableau expansion rules used in computing $\mathcal{A}^*$. Moreover, we add two special expansion rules to deal with BCQs. The details are presented in Section 2.5.

The answer to the instance-checking queries posed to the KB is based on membership of those queries in the set $\mathcal{A}^* \setminus \mathbb{E}$. To answer BCQs, we use graph terminology: we express both the ABox $\mathcal{A}^* \setminus \mathbb{E}$ and the BCQ q as node-edge labeled graphs, see also Ortiz and Šimkus (2012). The answer is based on the existence or non-existence of specific mappings between these two graphs. In more detail, if there is a (labeled) homomorphism from the query graph $G[q]$ (for q) to the ABox graph $G[\mathcal{A}^* \setminus \mathbb{E}]$ (for $\mathcal{A}^* \setminus \mathbb{E}$) then the answer to the query is "Yes"; if there are no such homomorphisms and there is a 'non-clashy' mapping from $G[q]$ to $G[\mathcal{A}^* \setminus \mathbb{E}]$ then the answer to the query is "Unknown"; otherwise the answer is "No", see Section 2.6 for details. Based on the OW-semantics, we are able to provide an exact characterization of all answers.

2.2 Preliminaries - Syntax and semantics of $DL\text{-}Lite_{\mathcal{R}}$

2.2.1 Syntax

A vocabulary of $DL\text{-}Lite_{\mathcal{R}}$ is a triple $< N_O, N_C, N_R >$ of countably infinite, pairwise disjoint sets. The elements of N_O are called *objects* or *individual names*, the elements of N_C are called *concept names* (unary relation symbols) and the elements N_R are called *role names* (binary relation symbols). The set of *basic concepts* and the set of *basic roles*, respectively denoted by $\mathcal{BC}$ and $\mathcal{BR}$, are defined below by the grammar (a) where $A \in N_{\mathcal{C}}$, $P \in N_{\mathcal{R}}$ and P^- stands for the inverse of the role name P. The set of *concept expressions* and *role expressions*

in $DL\text{-}Lite_{\mathcal{R}}$, denoted by $\mathcal{C}$ and $\mathcal{R}$, is defined by the grammar (b) where $B \in \mathcal{BC}$, and $R \in \mathcal{BR}$.

$$(a) \quad B ::= A \mid \exists R \qquad\qquad (b) \quad C ::= B \mid \neg B$$
$$R ::= P \mid P^{-} \qquad\qquad E ::= R \mid \neg R$$

Note that $\mathcal{BC} \subseteq \mathcal{C}$, and $\mathcal{BR} \subseteq \mathcal{R}$. For $C \in \mathcal{C}$ and $D \in \mathcal{BC}$, we write $\neg C$ to stand for D if $C = \neg D$ and for $\neg D$ if $C = D$. Similarly, for $E \in \mathcal{R}$ and $R \in \mathcal{BR}$, $\neg E$ denotes R if $E = \neg R$ and $\neg R$ if $E = R$. *Assertions* in $DL\text{-}Lite_{\mathcal{R}}$ are expressions of the form $C(a)$ and $E(a, b)$ where $a, b \in N_O$, $C \in \mathcal{C}$ and $E \in \mathcal{R}$; these are called *basic assertions* if $C \in \mathcal{BC}$ and $E \in \mathcal{BR}$.

There are two types of subsumptions in $DL\text{-}Lite_{\mathcal{R}}$,

a) *concept subsumptions* of the form $B \sqsubseteq C$ with $B \in \mathcal{BC}$ and $C \in \mathcal{C}$, and

b) *role subsumptions* of the form $R \sqsubseteq E$ with $R \in \mathcal{BR}$ and $E \in \mathcal{R}$.

Note the asymmetry between the left-hand side and the right-hand side of subsumptions in $DL\text{-}Lite_{\mathcal{R}}$.

2.2.2 Semantics

In this section we reformulate Kleene's 3-valued logic so as to provide semantics for $DL\text{-}Lite_{\mathcal{R}}$ which we feel is particularly well-suited in the context of OWA, see also Tao et al. (2014). It allows us to give an "epistemic separation" between "known that Yes", "known that No" and "Unknown". We use the idea of *weak 3-partition* [1], defined as follows. Let X be a non-empty set, and A_1, A_2, A_3 (possibly empty) subsets of X. The ordered triple (A_1, A_2, A_3) is a *weak 3-partition* of X if

1. $A_1 \cup A_2 \cup A_3 = X$ and

2. $\forall i, j$ with $i \neq j$, $A_i \cap A_j = \varnothing$.

An *OW-interpretation* of the language $DL\text{-}Lite_{\mathcal{R}}$ is a tuple $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ where Δ is a non-empty domain and $\cdot^{\mathcal{I}}$ is an *interpretation function* such that

[1] It is weak in that we do not require that the sets A_i are non-empty.

- $\forall a \in N_O$, $a^{\mathcal{I}} \in \Delta$,
- $\forall A \in N_C$, $A^{\mathcal{I}} = (A^{\mathcal{I}}_N,\ A^{\mathcal{I}}_U,\ A^{\mathcal{I}}_Y)$ is a weak 3-partition of Δ, and
- $\forall P \in N_R$, $P^{\mathcal{I}} = (P^{\mathcal{I}}_N,\ P^{\mathcal{I}}_U,\ P^{\mathcal{I}}_Y)$ is a weak 3-partition of $\Delta \times \Delta$.

We extend the interpretation function $\cdot^{\mathcal{I}}$ inductively to all concept and role expressions as follows. Let $C \in \mathcal{BC}$, $P \in N_R$, $R \in \mathcal{BR}$ and suppose that $C^{\mathcal{I}} = (C^{\mathcal{I}}_N,\ C^{\mathcal{I}}_U,\ C^{\mathcal{I}}_Y)$, $P^{\mathcal{I}} = (P^{\mathcal{I}}_N,\ P^{\mathcal{I}}_U,\ P^{\mathcal{I}}_Y)$ and $R^{\mathcal{I}} = (R^{\mathcal{I}}_N,\ R^{\mathcal{I}}_U,\ R^{\mathcal{I}}_Y)$. Then,

- $(\neg C)^{\mathcal{I}} = (C^{\mathcal{I}}_Y,\ C^{\mathcal{I}}_U,\ C^{\mathcal{I}}_N)$ and $(\neg R)^{\mathcal{I}} = (R^{\mathcal{I}}_Y,\ P^{\mathcal{I}}_U,\ R^{\mathcal{I}}_N)$,
- $(P^-)^{\mathcal{I}} = ((P^-)^{\mathcal{I}}_N,\ (P^-)^{\mathcal{I}}_U,\ (P^-)^{\mathcal{I}}_Y)$, where $(P^-)^{\mathcal{I}}_X = \{(a,b) |\ (b,a) \in P^{\mathcal{I}}_X\}$, $X \in \{N, U, Y\}$,
- $(\exists R)^{\mathcal{I}} = ((\exists R)^{\mathcal{I}}_N,\ (\exists R)^{\mathcal{I}}_U,\ (\exists R)^{\mathcal{I}}_Y)$, where $(\exists R)^{\mathcal{I}}_Y = \{a |\ \exists b \in \Delta[(a,b) \in R^{\mathcal{I}}_Y]\}$, $(\exists R)^{\mathcal{I}}_N = \{a |\ \forall b \in \Delta[(a,b) \in R^{\mathcal{I}}_N]\}$ and $(\exists R)^{\mathcal{I}}_U = \Delta \setminus ((\exists R)^{\mathcal{I}}_Y \cup (\exists R)^{\mathcal{I}}_N)$.

Remark 1: The subscripts "N", "U" and "Y" stand for "No", "Unknown" and "Yes", which represent the possible dispositions of a domain element with respect to a given OW-interpretation of a concept. Similarly, for roles. In addition, all the weak 3-partitions in this chapter are ordered: First the N-component, second the U-component and third the Y-component.

Let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be an OW-interpretation, $B \in \mathcal{BC}$, $C \in \mathcal{C}$, $R \in \mathcal{BR}$, $E \in \mathcal{R}$ and $a, b \in N_{\mathcal{O}}$. We say that

- $\mathcal{I}$ *satisfies* $C(a)$, notation $\mathcal{I} \models C(a)$, if $a^{\mathcal{I}} \in C^{\mathcal{I}}_Y$;
- $\mathcal{I}$ *satisfies* $E(a,b)$, notation $\mathcal{I} \models E(a,b)$, if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in E^{\mathcal{I}}_Y$;
- $\mathcal{I}$ *satisfies* $B \sqsubseteq C$, notation $\mathcal{I} \models B \sqsubseteq C$, if $B^{\mathcal{I}}_Y \subseteq C^{\mathcal{I}}_Y$ and $C^{\mathcal{I}}_N \subseteq B^{\mathcal{I}}_N$, and
- $\mathcal{I}$ *satisfies* $R \sqsubseteq E$, notation $\mathcal{I} \models R \sqsubseteq E$, if $R^{\mathcal{I}}_Y \subseteq E^{\mathcal{I}}_Y$ and $E^{\mathcal{I}}_N \subseteq R^{\mathcal{I}}_N$.

DL-Lite$_{\mathcal{R}}$ KB is a pair $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$, where $\mathcal{A}$, called the ABox [2], is a finite, non-empty set of assertions of the form $A(a)$, $\neg A(a)$, $P(a,b)$ and $\neg P(a,b)$ with $A \in N_C$, $P \in N_R$ and $a, b \in N_O$,

[2]Note that we do not allow assertions of the form $\exists R(a)$ in the ABox $\mathcal{A}$

and $\mathcal{T}$ is a finite set of concept and role subsumptions, called *TBox*. An OW-interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ is an *OW-model* of Σ, notation $\mathcal{I} \models \Sigma$, if for all $\alpha \in \mathcal{A} \cup \mathcal{T}$, $\mathcal{I} \models \alpha$. Let α be an assertion or a concept/role subsumption. We say that Σ *entails* α, notation $\Sigma \models \alpha$, if all OW-models of Σ satisfy α.

Remark 2: OW-interpretations have many applications in Computer Science, see Avron (1991); Fitting (1985). There are several reasons why OW-interpretations are of particular interest in our work.

- OW-interpretations naturally reflect the Open World Assumption (OWA) which applies to most KB applications. Thus, one way think of an OW-interpretation is as an agent of sorts, who, when presented with a particular scenario, may not possess complete information regarding the various memberships of domain elements in the interpretation of some concepts and roles.

- Since the classical (the usual 2-valued) interpretations are special kind of OW-interpretations, no generality is lost. Actually, some flexibility is gained in that OW-interpretations can, if needed, interpret some concepts (or roles) classically while others using the OW-approach.

- Another, more technical, reason is the following. As part of the completeness proof, we need to show that $\Sigma \models \neg A(a)$ implies $\neg A(a) \in \mathcal{A}^*$ where $\mathcal{A}^*$ is a completed ABox (i.e., no assertion expansion rules are applicable). This can be easily shown using OW-interpretations, see Section 2.2.2. This result is very important because completeness proof plays a central role in proving the correctness of the query answering procedure. However, since generally, classical interpretations satisfy $(\neg A)^{\mathcal{I}} \equiv \Delta \setminus A^{\mathcal{I}} \neq \{a^{\mathcal{I}} | \neg A(a) \in \mathcal{A}^*\}$, the above implication does hot hold.

2.3 Computation of $\mathcal{A}^*$

In this section, we will use a tableau-style procedure to construct a set of consequences of the given KB $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$, denoted by $\mathcal{A}^*$. Since our main interest in this chapter is studying secrecy-preserving query answering, we shall henceforth assume that all TBoxes are acyclic

which will guarantee that $\mathcal{A}^*$ is finite. Finding the set of all assertions entailed by an $\mathcal{EL}^+$ KB with acyclic TBox has been considered by Mei et al., see Mei et al. (2009).

Given $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$, before we start computing $\mathcal{A}^*$, first we arrange the individual names occurring in Σ, assertions in $\mathcal{A}$ and subsumptions in $\mathcal{T}$ in lexicographic order. Also, we program the algorithm which computes $\mathcal{A}^*$ in a way that selects these individual names, assertions and subsumptions in lexicographic order. This ordering will enable us to get a unique $\mathcal{A}^*$, see Calvanese et al. (2007). The computation of $\mathcal{A}^*$ proceeds in several stages. In the first stage, $\mathcal{A}^*$ is initialized as $\mathcal{A}$ and expanded by exhaustively applying expansion rules listed in Figure 2.1. The resulting ABox is denoted by $\mathcal{A}_1^*$. The sets of all the individual names appearing in $\mathcal{A}$ and $\mathcal{A}_1^*$ are denoted by $\mathcal{O}_\Sigma$ and $\mathcal{O}^*$, respectively. $\mathcal{O}^*$ is initialized as $\mathcal{O}_\Sigma$ and expanded with applications of the $\sqsubseteq_{N\exists}$- and $\sqsubseteq_{\exists\exists}$-rules. An individual a is said to be *fresh* if $a \in \mathcal{O}^* \setminus \mathcal{O}_\Sigma$. It is important to note that all the fresh individuals are added in the first stage (Figure 2.1) and no new individuals are added in the following stages. This can be easily seen by inspecting the rules in Figures 2.1, 2.2 and 2.3. The rules are designed based on subsumptions present in the TBox $\mathcal{T}$. To name the rules in Figure 2.1, we adopt the following conventions. The first subscript of $\sqsubseteq$ represents the type of the symbol on the left hand side of the subsumption, and the second represents the type of the symbol on the right hand side. For example, the $\sqsubseteq_{N\exists}$ -rule has a concept name on the left hand side of the subsumption and existential restriction on the right hand side.

In order to write the rules more succinctly, we define two functions $\underline{inv}$ (standing for $\underline{inv}$*erse*) and $\underline{neg}$ (standing for $\underline{neg}$*ation*) as follows:

- for $P \in N_R$, $\quad \underline{inv}(R,a,b) = \begin{cases} P(a,b) & \text{if } R = P, \\ P(b,a) & \text{if } R = P^- \end{cases}$

- for $R \in \mathcal{BR}$, $\quad \underline{neg}(E,a,b) = \begin{cases} \underline{inv}(R,a,b) & \text{if } E = R, \\ \neg\underline{inv}(R,a,b) & \text{if } E = \neg R \end{cases}$

For instance, $\underline{neg}(\neg P^-, a, b) = \neg\underline{inv}(P^-, a, b) = \neg P(b, a)$. In addition, we use L to denote either a concept name or a negation of concept name. We write $\neg L$ with the intended meaning that $\neg L = \neg A$ if $L = A$, and $\neg L = A$ if $L = \neg A$.

$\sqsubseteq_{NL}$ − rule : if $A(a) \in \mathcal{A}^*,\ A \sqsubseteq L \in \mathcal{T}$ and $L(a)\ \notin \mathcal{A}^*$,
then $\mathcal{A}^* := \mathcal{A}^* \cup \{L(a)\}$;

$\sqsubseteq_{N\exists}$ − rule : if $A(a) \in \mathcal{A}^*,\ A \sqsubseteq \exists R \in \mathcal{T},$ and $\forall d \in \mathcal{O}^*,\ \underline{inv}(R,a,d)\ \notin \mathcal{A}^*$,
then $\mathcal{A}^* := \mathcal{A}^* \cup \{\underline{inv}(R,a,b)\}$ where b is fresh, and $\mathcal{O}^* := \mathcal{O}^* \cup \{b\}$;

$\sqsubseteq_{\exists L}$ − rule : if $\underline{inv}(R,a,b) \in \mathcal{A}^*,\ \exists R \sqsubseteq L\ \in \mathcal{T},$ and $L(a)\ \notin \mathcal{A}^*$,
then $\mathcal{A}^* := \mathcal{A}^* \cup \{L(a)\}$;

$\sqsubseteq_{\exists\exists}$ − rule : if $\underline{inv}(R,a,b) \in \mathcal{A}^*,\ \exists R \sqsubseteq \exists S\ \in \mathcal{T},$ and $\forall d \in \mathcal{O}^*,\ \underline{inv}(S,a,d)\ \notin \mathcal{A}^*$,
then $\mathcal{A}^* := \mathcal{A}^* \cup \{\underline{inv}(S,a,c)\}$ where c is fresh, and $\mathcal{O}^* := \mathcal{O}^* \cup \{c\}$;

$\sqsubseteq_{RE}$ − rule : if $\underline{inv}(R,a,b) \in \mathcal{A}^*,\ R \sqsubseteq\ E\ \in \mathcal{T}$ and $\underline{neg}(E,a,b)\ \notin \mathcal{A}^*$,
then $\mathcal{A}^* := \mathcal{A}^* \cup \{\underline{neg}(E,a,b)\}$.

We use the following conventions not stated explicitly within the individual rules:
$A \in N_C,\ L \in \{A, \neg A \mid A \in N_C\},\ \ R,S \in \mathcal{BR}$ and $E \in \mathcal{R}$.

Figure 2.1 Expansion rules for computing $\mathcal{A}_1^*$

$\sqsubseteq_{N\nexists}$ − rule : Let $A(a) \in \mathcal{A}^*$ and $A \sqsubseteq \neg\exists R \in \mathcal{T}$.
$\forall c \in \mathcal{O}^*$: if $\neg\underline{inv}(R,a,c)\ \notin \mathcal{A}^*$, then $\mathcal{A}^* := \mathcal{A}^* \cup \{\neg\underline{inv}(R,a,c)\}$;

$\sqsubseteq_{\exists\nexists}$ − rule : Let $\underline{inv}(R,a,b) \in \mathcal{A}^*$ and $\exists R \sqsubseteq \neg\exists S\ \in \mathcal{T}$.
$\forall c \in \mathcal{O}^*$: if $\neg\underline{inv}(S,a,c)\ \notin \mathcal{A}^*$, then $\mathcal{A}^* := \mathcal{A}^* \cup \{\neg\underline{inv}(S,a,c)\}$.

Computing $\mathcal{A}_{12}^*$: An application of each rule adds negation of role assertions for all $c \in \mathcal{O}^*$

Figure 2.2 Expansion rules for computing $\mathcal{A}_{12}^*$

In the second stage, $\mathcal{A}_1^*$ is expanded by applying expansion rules listed in Figure 2.2. The resulting ABox is denoted as $\mathcal{A}_{12}^*$. Observe that every application of a rule in Figure 2.2 adds at most $|\mathcal{O}^*|$ new assertions to $\mathcal{A}_1^*$. To name the rules in Figure 2.2, we adopt the same naming conventions as for the rules in Figure 2.1 except that the second symbol in the subscript represents the right hand side of $\sqsubseteq$: $\nexists$ stands for a negated unqualified existential restriction.

In the third stage, $\mathcal{A}_{12}^*$ is expanded by applying rules listed in Figure 2.3. The resulting final ABox is denoted as $\mathcal{A}^*$. To name the rules in Figure 2.3, we follow the previously adopted conventions. Additionally, negation in the subscript (see Figure 2.3) should be thought of as follows: For each rule in Figures 2.1 and 2.2, e.g. $\sqsubseteq_{NL}$-rule with $A \sqsubseteq L$, we have a corresponding $\sqsubseteq_{NL\neg}$-rule, which captures the effect of the subsumption $\neg L \sqsubseteq \neg A$ (which is not allowed in

$\sqsubseteq_{NL\neg}$ – rule : if $\neg L(a) \in \mathcal{A}^*,\ A \sqsubseteq L \in \mathcal{T}$ and $\neg A(a)\ \notin \mathcal{A}^*$,
then $\mathcal{A}^* := \mathcal{A}^* \cup \{\neg A(a)\}$;

$\sqsubseteq_{N\exists\neg}$ – rule : if $\forall b \in \mathcal{O}^*,\ \neg\underline{inv}(R,a,b) \in \mathcal{A}^*,\ A \sqsubseteq \exists R \in \mathcal{T},$ and $\neg A(a)\ \notin \mathcal{A}^*$,
then $\mathcal{A}^* := \mathcal{A}^* \cup \{\neg A(a)\}$;

$\sqsubseteq_{\exists L\neg}$ – rule : Let $\neg L(a) \in \mathcal{A}^*$ and $\exists R \sqsubseteq L\ \in \mathcal{T}$.
$\forall c \in \mathcal{O}^*:$ if $\neg\underline{inv}(R,a,c)\ \notin \mathcal{A}^*,$ then $\mathcal{A}^* := \mathcal{A}^* \cup \{\neg\underline{inv}(R,a,c)\}$;

$\sqsubseteq_{\exists\exists\neg}$ – rule : Let $\forall b \in \mathcal{O}^*,\ \neg\underline{inv}(S,a,b) \in \mathcal{A}^*$ and $\exists R \sqsubseteq \exists S\ \in \mathcal{T}$.
$\forall c \in \mathcal{O}^*:$ if $\neg\underline{inv}(R,a,c)\ \notin \mathcal{A}^*,$ then $\mathcal{A}^* := \mathcal{A}^* \cup \{\neg\underline{inv}(R,a,c)\}$;

$\sqsubseteq_{RE\neg}$ – rule : if $\neg\underline{neg}(E,a,b) \in \mathcal{A}^*,\ R \sqsubseteq\ E\ \in \mathcal{T}$ and $\neg\underline{inv}(R,a,b)\ \notin \mathcal{A}^*$,
then $\mathcal{A}^* := \mathcal{A}^* \cup \{\neg\underline{inv}(R,a,b)\}$;

$\sqsubseteq_{N\nexists\neg}$ – rule : if $\underline{inv}(R,a,b) \in \mathcal{A}^*,\ A \sqsubseteq \neg\exists R \in \mathcal{T}$ and $\neg A(a)\ \notin \mathcal{A}^*$,
then $\mathcal{A}^* := \mathcal{A}^* \cup \{\neg A(a)\}$.

$\sqsubseteq_{\exists\nexists\neg}$ – rule : Let $\underline{inv}(S,a,b) \in \mathcal{A}^*$ and $\exists R \sqsubseteq \neg\exists S\ \in \mathcal{T}$.
$\forall c \in \mathcal{O}^*:$ if $\neg\underline{inv}(R,a,c)\ \notin \mathcal{A}^*,$ then $\mathcal{A}^* := \mathcal{A}^* \cup \{\neg\underline{inv}(R,a,c)\}$.

We use the same conventions as in Figure 2.1

Figure 2.3 Expansion rules for computing $\mathcal{A}^*$

our syntax). It is easy to see that during the execution of rules in Figure 2.3 none of the rules in Figures 2.1 and 2.2 becomes applicable.

We say that $\mathcal{A}^*$ is *completed* or that it is an *assertional closure of* $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$ if no assertion expansion rule is applicable. We denote by Λ the *tableau algorithm* which (lexicographically) applies assertion expansion rules, first those in Figure 2.1 then those in Figure 2.2 and finally those in Figure 2.3, until no further applications are possible. Since, as explained previously, Λ works in a lexicographic fashion, for a given KB $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$, it outputs a unique $\mathcal{A}^*$.

Since some of the expansion rules can in some cases be applied exponentially many times in the size of the KB, the size of $\mathcal{A}^*$ can be exponential in the size of the KB. As an example consider a $DL{-}Lite_R$ KB $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$, where $\mathcal{A} = \{A(a)\}$ and $\mathcal{T} = \{A \sqsubseteq \exists P_1, A \sqsubseteq \exists Q_1, \exists P_i^- \sqsubseteq \exists P_{i+1}, \exists P_i^- \sqsubseteq \exists Q_{i+1}, Q_i \sqsubseteq P_{i+1}, 1 \leq i \leq n\}$. Clearly, TBox $\mathcal{T}$ is acyclic and the size of the KB is linear in n. To compute $\mathcal{A}^*$ for this KB, the $\sqsubseteq_{\exists\exists}$-rule has to be applied exponentially many times. It follows that $\mathcal{A}^*$ is exponential in the size of the Σ, implying that the computation of $\mathcal{A}^*$ could require exponential time as well.

Example 2.3.1. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$ be a DL-Lite$_{\mathcal{R}}$ KB, where $\mathcal{A}$ is defined by 1 and 2, and $\mathcal{T}$ is defined by 3, 4, 5 and 6,*

1	$A(a)$	*3*	$A \sqsubseteq B$	*5*	$\exists P^- \sqsubseteq \neg\exists R$
2	$D(b)$	*4*	$A \sqsubseteq \exists P$	*6*	$C \sqsubseteq \neg D$

Applying the assertion expansion rules in Figure 2.1, we can derive the following conclusions.

7	$B(a)$	$\sqsubseteq_{NL}$ *on 1,3*
8	$P(a, c)$, *c is fresh*	$\sqsubseteq_{N\exists}$ *on 1,4*

Therefore $\mathcal{A}_1^* = \mathcal{A} \cup \{B(a), P(a, c)\}$. Now applying the assertion expansion rules in Figure 2.2 on $\mathcal{A}_1^*$, we calculate $\mathcal{A}_{12}^*$.

9	$\neg R(c, a), \neg R(c, b), \neg R(c, c)$	$\sqsubseteq_{\exists\sharp}$ *on 8,5*

Thus $\mathcal{A}_{12}^* = \mathcal{A}_1^* \cup \{\neg R(c, a), \neg R(c, b), \neg R(c, c)\}$. Finally, using the assertion expansion rules in Figure 2.3 on $\mathcal{A}_{12}^*$, we get $\mathcal{A}^*$.

10	$\neg C(b)$	$\sqsubseteq_{NL\neg}$ *on 2,6*

Hence, $\mathcal{A}^* = \mathcal{A}_{12}^* \cup \{\neg C(b)\}$.

Observe that if we restrict the application of expansions rules in Figure 2.1, 2.2 and 2.3 to those ABox assertions involving only non-fresh individual names then we get $\{A(a), B(a), D(b), \neg C(b)\}$. $\square$

In general, if the computation is restricted to ABox assertions involving non-fresh individual names, then it is easy to see that the size of $\mathcal{A}^*$ is polynomial in the size of Σ and that it can be computed in polynomial time.

Soundness: The proof of the soundness of the tableau procedure Λ is split into two parts, dealing separately with rules in Figures 2.1 and 2.2 and Figure 2.3. The proof of Lemma 2.3.1 is standard and given in Appendix A.1.1.

Lemma 2.3.1 (Soundness of Λ, Part A). *Let $\mathcal{A}_{12}^*$ be a completed ABox obtained from Σ by first applying the rules listed in Figure 2.1 and then the rules of Figure 2.2. Then for every OW-model $\mathcal{I}$ of Σ, there is a OW-model $\mathcal{I}_{12}^*$ of Σ such that $\mathcal{I}_{12}^* \models \mathcal{A}_{12}^*$, where the domain of*

$\mathcal{I}^*_{12}$ is same as the domain of $\mathcal{I}$ and $\mathcal{I}^*_{12}$ remains same as $\mathcal{I}$ except for the interpretation of fresh individuals.

Let $\mathcal{O}^*$ be the set of individual names that occur in the completed ABox $\mathcal{A}^*_{12}$. We define a new OW-interpretation $\mathcal{I}^* = \langle \Delta^*, \cdot^{\mathcal{I}^*} \rangle$, where $\Delta^* = \mathcal{I}^*_{12}(\mathcal{O}^*)$, i.e., Δ^* is precisely the set of those elements of Δ that are interpretations of individuals in $\mathcal{O}^*$. The interpretation function $\cdot^{\mathcal{I}^*}$ is defined as a restriction of $\mathcal{I}^*_{12}$ to Δ^*:

(i) $\forall a \in \mathcal{O}^*[a^{\mathcal{I}^*} = a^{\mathcal{I}^*_{12}}]$;

(ii) $\forall A \in N_C[(A_N^{\mathcal{I}^*} = A_N^{\mathcal{I}^*_{12}} \cap \Delta^*, A_U^{\mathcal{I}^*} = A_U^{\mathcal{I}^*_{12}} \cap \Delta^*, A_Y^{\mathcal{I}^*} = A_Y^{\mathcal{I}^*_{12}} \cap \Delta^*)]$;

(iii) $\forall P \in N_R[P_N^{\mathcal{I}^*} = P_N^{\mathcal{I}^*_{12}} \cap (\Delta^* \times \Delta^*), P_U^{\mathcal{I}^*} = P_U^{\mathcal{I}^*_{12}} \cap (\Delta^* \times \Delta^*), P_Y^{\mathcal{I}^*} = P_Y^{\mathcal{I}^*_{12}} \cap (\Delta^* \times \Delta^*)]$ and

(iv) $\mathcal{I}^*$ is extended to compound concepts and roles as in Section 2.2.2.

Since every weak 3-partition of Δ induces a weak 3-partition of Δ^*, we have the following consequence of Lemma 2.3.1,

Corollary 2.3.1. *$\mathcal{I}^*$ is an OW-model of $\langle \mathcal{A}^*_{12}, \mathcal{T} \rangle$.*

The proof of the next lemma is standard and given in Appendix A.1.2.

Lemma 2.3.2 (Soundness of Λ, Part B). *Let $\mathcal{A}^*$ be the completed ABox obtained from $\mathcal{A}^*_{12}$ by applying the rules listed in Figure 2.3. For any OW-model $\mathcal{I}$ of Σ, let $\mathcal{I}^* = \langle \Delta^*, \cdot^{\mathcal{I}^*} \rangle$ be an OW-interpretation as defined above. Then, $\mathcal{I}^*$ is an OW-model of Σ and $\mathcal{I}^* \models \mathcal{A}^*$.*

In summary, given an OW-model $\mathcal{I}$ of Σ, using the proof of Lemma 2.3.1, we transform $\mathcal{I}$ to another OW-model $\mathcal{I}^*_{12}$ of Σ such that $\mathcal{I}^*_{12} \models \mathcal{A}^*_{12}$, where the domain of $\mathcal{I}^*_{12}$ is same as the domain of $\mathcal{I}$. In fact, $\mathcal{I}^*_{12}$ remains the same as $\mathcal{I}$ except for the interpretation of fresh individuals. Moreover, $\mathcal{I}^*_{12}$ is constructed in a canonical fashion, i.e., it is uniquely determined from $\mathcal{I}$. Having obtained $\mathcal{I}^*_{12}$, using Lemma 2.3.2, we modify $\mathcal{I}^*_{12}$ to obtain yet another OW-model $\mathcal{I}^*$ of Σ such that $\mathcal{I}^* \models \mathcal{A}^*$, where the domain of $\mathcal{I}^*$ was defined to be $\mathcal{I}^*_{12}(\mathcal{O}^*)$.

We use the notation $\Sigma \models^* \alpha$, where α is a concept (or role) name assertion or negation of a concept (or role) name assertion, to represent the following statement: For every OW-model

$\mathcal{I}$ of Σ, $\mathcal{I}^*$ is an OW-model of Σ and $\mathcal{I}^* \models \alpha$. We can combine Lemma 2.3.1 and Lemma 2.3.2 into a single theorem.

Theorem 2.3.1. *(**Soundness of** Λ): Let $\mathcal{A}^*$ be a completed ABox obtained from Σ by first applying the rules listed in Figure 2.1, then rules listed in Figure 2.2, and finally the rules listed in Figure 2.3. Then $\Sigma \models^* \mathcal{A}^*$, i.e., for every $\alpha \in \mathcal{A}^*$, $\Sigma \models^* \alpha$.*

Completeness: To prove the completeness of Λ, we first define a *canonical OW-interpretation* $\mathcal{J} = \langle \Delta, \cdot^{\mathcal{J}} \rangle$ for a completed ABox $\mathcal{A}^*$ as follows:

- $\Delta = \mathcal{O}^* = \{a \in N_O |\ a \text{ occurs in } \mathcal{A}^*\}$;
- $a^{\mathcal{J}} = a$, for each individual name $a \in \mathcal{O}^*$;
- for $A \in N_C$, $A^{\mathcal{J}} = (A_N^{\mathcal{J}},\ A_U^{\mathcal{J}},\ A_Y^{\mathcal{J}})$, where

  $A_Y^{\mathcal{J}} = \{a |\ A(a) \in \mathcal{A}^*\}$,

  $A_N^{\mathcal{J}} = \{a |\ \neg A(a) \in \mathcal{A}^*\}$ and

  $A_U^{\mathcal{J}} = (\Delta \setminus A_Y^{\mathcal{J}}) \setminus A_N^{\mathcal{J}}$;
- for $P \in N_R$, $P^{\mathcal{J}} = (P_N^{\mathcal{J}},\ P_U^{\mathcal{J}},\ P_Y^{\mathcal{J}})$, where

  $P_Y^{\mathcal{J}} = \{(a,b) |\ P(a,b) \in \mathcal{A}^*\}$,

  $P_N^{\mathcal{J}} = \{(a,b) |\ \neg P(a,b) \in \mathcal{A}^*\}$ and

  $P_U^{\mathcal{J}} = ((\Delta \times \Delta) \setminus P_Y^{\mathcal{J}}) \setminus P_N^{\mathcal{J}}$;
- $\mathcal{J}$ is extended to compound concepts and roles as in Section 2.2.2

The proof that $\mathcal{J}$ is a OW-model of Σ is standard and given in the Appendix A.1.3.

Lemma 2.3.3. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$ be a DL-Lite$_{\mathcal{R}}$ KB. Then $\forall \alpha \in \mathcal{A} \cup \mathcal{T},\ \mathcal{J} \models \alpha$.*

Theorem 2.3.2 (**Completeness of** Λ)**.** *Let $\mathcal{A}^*$ be a completed ABox obtained from Σ by applying Λ. Let α be a concept (or role) name assertion or negation of a concept (or role) name assertion [3]. Then $\Sigma \models^* \alpha \ \Rightarrow\ \alpha \in \mathcal{A}^*$.*

[3] *Recall that assertions of the form $\exists R(a)$ do not belong to $\mathcal{A}^*$*

Proof. *Let $\mathcal{J}$ be the canonical model of Σ as defined above, and let α be an assertion as in the statement of the theorem. Suppose $\Sigma \models^* \alpha$. By Lemma 2.3.3, $\mathcal{J} \models \Sigma$ and hence $\mathcal{J}^* \models \alpha$. Since $\mathcal{A}^*$ is completed, $\mathcal{J}^* = \mathcal{J}$, and so $\mathcal{J} \models \alpha$. In the following, we argue by cases for different α.*

- $\alpha = A(a)$, $A \in N_C$. *Then,* $\mathcal{J} \models A(a) \Rightarrow a \in A_Y^{\mathcal{J}} \Rightarrow A(a) \in \mathcal{A}^*$.

- $\alpha = \neg A(a)$, $A \in N_C$. *Then,* $\mathcal{J} \models \neg A(a) \Rightarrow a \in A_N^{\mathcal{J}} \Rightarrow \neg A(a) \in \mathcal{A}^*$.

- $\alpha = P(a,b)$, $P \in N_R$. *Then,* $\mathcal{J} \models P(a,b) \Rightarrow (a,b) \in P_Y^{\mathcal{J}} \Rightarrow P(a,b) \in \mathcal{A}^*$.

- $\alpha = \neg P(a,b)$, $P \in N_R$. *Then,* $\mathcal{J} \models \neg P(a,b) \Rightarrow (a,b) \in P_N^{\mathcal{J}} \Rightarrow \neg P(a,b) \in \mathcal{A}^*$. $\square$

2.4 Graph representation of ABoxes and BCQs over $DL\text{-}Lite_{\mathcal{R}}$ KBs

In this section, we will use node-edge labeled directed graph to represent the completed ABox $\mathcal{A}^*$ as well as Boolean conjunctive queries (BCQs), see Ortiz and Šimkus (2012) for similar representations. This helps "visualize" reasoning about such queries as well as being useful in formulating precise conditions for answering BCQs with 'Yes', 'No' and 'Unknown'.

The ABox graph for $\mathcal{A}^*$ is node-edge labeled digraph $G[\mathcal{A}^*] = (V[\mathcal{A}^*], E[\mathcal{A}^*], L[\mathcal{A}^*])$ with nodes $V[\mathcal{A}^*] = \mathcal{O}^*$ and edges $E[\mathcal{A}^*] = \{(a,b) \mid R(a,b) \in \mathcal{A}^*, \text{ for some } R \in \mathcal{R}\}$, where each node $a \in V[\mathcal{A}^*]$ is labeled with the set of literals $L[\mathcal{A}^*](a) = \{ L \mid L(a) \in \mathcal{A}^*\}$ and each directed edge $(a,b) \in E[\mathcal{A}^*]$ is labeled with a set of roles $L[\mathcal{A}^*](a,b) = \{ R \mid R(a,b) \in \mathcal{A}^*\}$.

Example 2.4.1. *Let $\mathcal{A}^* = \{A(a), \neg D(a), B(b), F(b), H(d), P(a,b), Q(a,b), P(b,c), Q(b,c), R(a,d), \neg S(a,d), \neg Q(c,c)\}$. Then ABox graph $G[\mathcal{A}^*]$ for $\mathcal{A}^*$ is given in Figure 2.4.*

We next define the syntax and semantics of Boolean conjunctive queries. Let N_V denote a countably infinite set of variables.

Definition 2.4.1. *A Boolean conjunctive query over $DL\text{-}Lite_{\mathcal{R}}$ is a finite expression of the form $\exists y_1, y_2, ..., y_n[\ \bigwedge_{i=1}^{k} A_i(\zeta_i)\ \wedge\ \bigwedge_{j=1}^{m} P_j(\eta_j, \mu_j)]$, where*

- $A_i \in N_C$ *for* $1 \leq i \leq k$, $P_j \in N_R$ *for* $1 \leq j \leq m$ *and* $y_l \in N_V$, $1 \leq l \leq n$,

- $\zeta_i, \eta_j, \mu_j \in \{y_1, y_2, ..., y_n\} \cup N_O$ *for* $1 \leq i \leq k$ *and* $1 \leq j \leq m$.

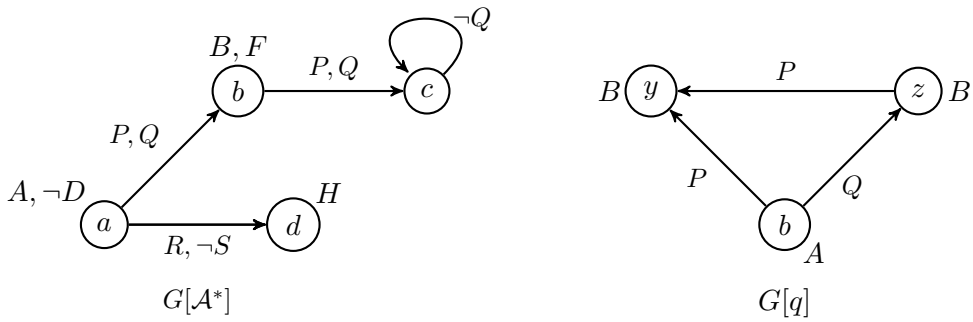

Figure 2.4 The ABox and query graphs

Query atoms of a BCQ q are of two sorts: *concept atoms* $A(v)$, and *role atoms* $P(u, v)$, where $u, v \in N_V \cup N_O$, $A \in N_C$ and $P \in N_R$. By $Atoms(q)$ we denote the set of concept and role atoms occurring in q. For instance the concept atoms in the BCQ $q = \exists y, z[A(b) \wedge B(y) \wedge B(z) \wedge P(b, y) \wedge Q(b, z) \wedge P(z, y)]$ are: $A(b)$, $B(y)$ and $B(z)$ and the role atoms are: $P(b, y)$, $Q(b, z)$ and $P(z, y)$.

As was the case with the ABox, we can represent the BCQ as a node-edge labeled directed graph capturing the syntactic structure of the query. The *query graph* of a BCQ q is the node-edge labeled directed graph $G[q] = (V[q], E[q], L[q])$ with nodes $V[q] = \{v \in N_V \cup N_O \mid v \text{ occurs in } q\}$ and edges $E[q] = \{(u, v) \mid \text{for some role name } P,\ P(u, v) \in Atoms(q)\}$; each node $v \in V[q]$ is labeled with the set of concept names $L[q](v) = \{A \mid A(v) \in Atoms(q)\}$ and each edge $(u, v) \in E[q]$ is labeled with the set of role names $L[q](u, v) = \{P \mid P(u, v) \in Atoms(q)\}$.

Example 2.4.2. *The query graph $G[q]$ of the BCQ $q = \exists y, z[A(b) \wedge B(y) \wedge B(z) \wedge P(b, y) \wedge Q(b, z) \wedge P(z, y)]$ mentioned above is given in Figure 2.4.*

An interpretation of a BCQ q is provided by an OW-interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ together with a *valuation* which is a function $\pi : V[q] \to \Delta$ such that $\pi(a) = a^{\mathcal{I}}$ for each individual $a \in V[q] \cap N_O$. We say that $(\mathcal{I}, \pi)$ *satisfies* $A(v)$, notation $(\mathcal{I}, \pi) \models A(v)$, if $\pi(v) \in A^{\mathcal{I}}_Y$. $(\mathcal{I}, \pi)$ *falsifies* $A(v)$, notation $(\mathcal{I}, \pi) \models \neg A(v)$, if $\pi(v) \in A^{\mathcal{I}}_N$. Similarly, $(\mathcal{I}, \pi)$ *satisfies* $P(u, v)$, notation $(\mathcal{I}, \pi) \models P(u, v)$, if $(\pi(u), \pi(v)) \in P^{\mathcal{I}}_Y$ and $(\mathcal{I}, \pi)$ *falsifies* $P(u, v)$, notation $(\mathcal{I}, \pi) \models \neg P(u, v)$, if $(\pi(u), \pi(v)) \in P^{\mathcal{I}}_N$. We say that $(\mathcal{I}, \pi)$ *satisfies* q, notation $(\mathcal{I}, \pi) \models q$, if $(\mathcal{I}, \pi) \models \alpha$

for every $\alpha \in Atoms(q)$. $(\mathcal{I}, \pi)$ *falsifies* q, notation $(\mathcal{I}, \pi) \mid\!\models q$, if $(\mathcal{I}, \pi)$ falsifies some atom $\alpha \in Atoms(q)$. $\mathcal{I}$ *satisfies* q, notation $\mathcal{I} \models q$, if there exists a valuation $\pi : V[q] \to \Delta$ such that $(\mathcal{I}, \pi) \models q$. In this case, we say that $\mathcal{I}$ is an OW-model of q. $\mathcal{I}$ *falsifies* q, notation $\mathcal{I} \mid\!\models q$, if for all valuations $\pi : V[q] \to \Delta$, $(\mathcal{I}, \pi) \mid\!\models q$.

Recall that given any OW-model $\mathcal{I}$ of Σ we have defined (a uniquely determined) OW-model $\mathcal{I}^*$ and we introduced the notation $\Sigma \models^* \alpha$ to mean that for any OW-model $\mathcal{I}$ of Σ, $\mathcal{I}^*$ is an OW-model of Σ and $\mathcal{I}^* \models \alpha$. Finally, a BCQ q is *entailed* from Σ, notation $\Sigma \models^* q$, if for every OW-model $\mathcal{I}$ of Σ, $\mathcal{I}^* \models q$. A BCQ q is *disentailed* from Σ, notation $\Sigma \mid\!\models^* q$, if for every OW-model $\mathcal{I}$ of Σ, $\mathcal{I}^* \mid\!\models q$.

Notation: We write $h : V[q] \underset{c}{\to} V[\mathcal{A}^*]$ to denote the fact that h is a mapping $h : V[q] \to V[\mathcal{A}^*]$ which "respects constants", i.e. $h(a) = a$, for every individual $a \in V[q] \cap N_O$.

Definition 2.4.2. *Mapping $h : V[q] \underset{c}{\to} V[\mathcal{A}^*]$ is a labeled graph homomorphism, if*

- *for every node v in $V[q]$, $L[q](v) \subseteq L[\mathcal{A}^*](h(v))$, and*

- *for every edge (u, v) in $E[q]$, $L[q](u, v) \subseteq L[\mathcal{A}^*](h(u), h(v))$.*

In the next two theorems we provide a complete characterization of entailment and disentailment of BCQs in terms of properties of mappings $h : V[q] \underset{c}{\to} V[\mathcal{A}^*]$.

Theorem 2.4.1. *Let q be a BCQ and Σ a DL-Lite$_{\mathcal{R}}$ KB. Then, $\Sigma \models^* q$ iff there exists a labeled graph homomorphism $h : V[q] \underset{c}{\to} V[\mathcal{A}^*]$.*

Proof. $(\Rightarrow)$ *Suppose $\Sigma \models^* q$ and let $\mathcal{J} = \langle \Delta, \cdot^{\mathcal{J}} \rangle$ be the canonical OW-model of Σ. Then, $\mathcal{J}^* = \mathcal{J}$, and by hypothesis, $\mathcal{J} \models q$. Hence, for some valuation $\pi : V[q] \to \mathcal{O}^* = V[\mathcal{A}^*]$, $(\mathcal{J}, \pi) \models \alpha$, for every $\alpha \in Atoms(q)$. Note that $\pi : V[q] \underset{c}{\to} V[\mathcal{A}^*]$. Now, let $v \in V[q]$ and $A(v) \in Atoms(q)$. Then, $(\mathcal{J}, \pi) \models A(v) \Rightarrow \pi(v) \in A_Y^{\mathcal{J}} \Rightarrow A(\pi(v)) \in \mathcal{A}^* \Rightarrow A \in L[\mathcal{A}^*](\pi(v))$. Similarly, for $u, v \in V[q]$ with $P(u, v) \in Atoms(q)$: $(\mathcal{J}, \pi) \models P(u, v) \Rightarrow (\pi(u), \pi(v)) \in P_Y^{\mathcal{J}} \Rightarrow P(\pi(u), \pi(v)) \in \mathcal{A}^* \Rightarrow P \in L[\mathcal{A}^*](\pi(u), \pi(v))$. It follows that π is a labeled graph homomorphism.*

$(\Leftarrow)$ *Assume that $h : V[q] \underset{c}{\to} V[\mathcal{A}^*]$ is a labeled graph homomorphism and let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be an arbitrary OW-model of Σ. By Lemma 2.3.2, $\mathcal{I}^* = \langle \Delta^*, \cdot^{\mathcal{I}^*} \rangle$, with $\Delta^* = \mathcal{I}^*(\mathcal{O}^*)$, is an OW-*

model of Σ and $\mathcal{I}^* \models \mathcal{A}^*$. Since $(\mathcal{I}^* \circ h) : V[q] \to \Delta^*$, we have $(\mathcal{I}^* \circ h)(a) = \mathcal{I}^*(h(a)) = a^{\mathcal{I}^*}$ for all $a \in V[q] \cap N_O$. I.e. $\mathcal{I}^* \circ h$ is a valuation. It remains to show that $\mathcal{I}^*$ is an OW-model of q. Let $v \in V[q]$ and $A \in L[q](v)$. Then, by the definition of labeled homomorphism, $A \in L[\mathcal{A}^*](h(v)) \Rightarrow A(h(v)) \in \mathcal{A}^* \Rightarrow h(v)^{\mathcal{I}^*} \in A_Y^{\mathcal{I}^*} \Rightarrow (\mathcal{I}^* \circ h)(v) \in A_Y^{\mathcal{I}^*} \Rightarrow (\mathcal{I}^*, (\mathcal{I}^* \circ h)) \models A(v)$. Similarly, for $u, v \in V[q]$ with $P \in L[q](u, v)$: $P \in L[\mathcal{A}^*](h(u), h(v)) \Rightarrow P(h(u), h(v)) \in \mathcal{A}^* \Rightarrow (h(u)^{\mathcal{I}^*}, h(v)^{\mathcal{I}^*}) \in P_Y^{\mathcal{I}^*} \Rightarrow ((\mathcal{I}^* \circ h)(u), (\mathcal{I}^* \circ h)(v)) \in P_Y^{\mathcal{I}^*} \Rightarrow (\mathcal{I}^*, (\mathcal{I}^* \circ h)) \models P(u, v)$. Thus, $\Sigma \models^* q$. $\square$

Next we define mappings that cannot be extended to labeled homomorphisms and prove a tight connection between such mappings and disentailment.

Definition 2.4.3. *A mapping $f : V[q] \underset{c}{\to} V[\mathcal{A}^*]$ is said to be clashy, if*

- *there exist $v \in V[q]$ and $A \in L[q](v)$ such that $\neg A \in L[\mathcal{A}^*](f(v))$, or*
- *there exist $u, v \in V[q]$ and $P \in L[q]((u, v))$ such that $\neg P \in L[\mathcal{A}^*]((f(u), f(v)))$.*

Theorem 2.4.2. *Let q be a BCQ and Σ a DL-Lite$_{\mathcal{R}}$ KB. Then, $\Sigma \mid\!\models^* q$ iff every mapping $f : V[q] \underset{c}{\to} V[\mathcal{A}^*]$ is clashy.*

Proof. ($\Rightarrow$) *Assume $\Sigma \mid\!\models^* q$ and let $\mathcal{J} = \langle \Delta, \cdot^{\mathcal{J}} \rangle$ be the canonical OW-model of Σ. Then, $\mathcal{J}^* = \mathcal{J}$ and so for every valuation $\tau : V[q] \to \Delta^*$, there is an $\alpha \in Atoms(q)$ such that $(\mathcal{J}, \tau) \models \neg\alpha$. Since $\Delta^* = \mathcal{J}(\mathcal{O}^*) = \mathcal{O}^* = V(\mathcal{A}^*)$ and $\tau(a) = a^{\mathcal{J}} = a$ for all $a \in V[q] \cap N_O$, $\tau : V[q] \underset{c}{\to} V[\mathcal{A}^*]$ and it follows that τ is clashy. Moreover, since τ is arbitary the conclusion follows.*

($\Leftarrow$) *Suppose now that every mapping $f : V[q] \underset{c}{\to} V[\mathcal{A}^*]$ is clashy. Let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be an arbitrary OW-model of Σ. By Lemma 2.3.2, $\mathcal{I}^* = \langle \Delta^*, \cdot^{\mathcal{I}^*} \rangle$ with $\Delta^* = \mathcal{I}^*(\mathcal{O}^*)$ is an OW-model of Σ such that $\mathcal{I}^* \models \mathcal{A}^*$. Let $\pi : V[q] \to \Delta^*$ be an arbitrary valuation and define the mapping $g_\pi : V[q] \to V[\mathcal{A}^*]$ by*

$$g_\pi(v) = \begin{cases} a & \text{if } v = a \in N_O \cap V[q] \\ c & \text{if } v \in N_V \cap V[q], \end{cases}$$

where $\pi(v) = c^{\mathcal{I}^*}$ and c be the first constant that satisfies in some arbitrary (but fixed) total ordering of $\mathcal{O}^*$. It is easy to check that $\pi = \mathcal{I}^* \circ g_\pi$ (in other words, π factors via $V[\mathcal{A}^*]$). Since, by

assumption, g_π *is clashy, for some* $A(v) \in Atoms(q)$, $\neg A \in L[\mathcal{A}^*](g_\pi(v))$ *or for some* $P(u,v) \in Atoms(q)$, $\neg P \in L[\mathcal{A}^*]((g_\pi(u), g_\pi(v)))$. *In the first case,* $\neg A(g_\pi(v)) \in \mathcal{A}^* \Rightarrow g_\pi(v)^{\mathcal{I}^*} \in A_N^{\mathcal{I}^*} \Rightarrow \pi(v) \in A_N^{\mathcal{I}^*}$ *implying,* $(\mathcal{I}^*, \pi) \models \neg A(v)$. *In the second case,* $\neg P((g_\pi(u), g_\pi(v))) \in \mathcal{A}^* \Rightarrow (g_\pi(u)^{\mathcal{I}^*}, g_\pi(v)^{\mathcal{I}^*}) \in P_N^{\mathcal{I}^*} \Rightarrow (\pi(u), \pi(v)) \in P_N^{\mathcal{I}^*}$ *implying,* $(\mathcal{I}^*, \pi) \models \neg P(u,v)$. *It follows that,* $\Sigma \mid\!\models^* q$. $\square$

2.5 Secrecy-Preserving Reasoning in $DL\text{-}Lite_{\mathcal{R}}$ KBs

Given a knowledge base Σ and a finite secrecy set $\mathbb{S}$ consisting of assertions in $\mathcal{A}^*$ and BCQs, the goal is to answer queries while preserving secrecy. Here we assume that $\mathcal{A}^*$ has been computed previously. Our approach is to compute a subset $\mathbb{E} \subseteq \mathcal{A}^*$, called the *secrecy envelope* for $\mathbb{S}$, so that by protecting $\mathbb{E}$, the querying agent cannot logically infer any assertions in $\mathbb{S}$, see Tao et al. (2010, 2014). It is interesting to note that, though the BCQs in $\mathbb{S}$ are not in $\mathbb{E}$, we can store the information pertinent to answering BCQs in $\mathbb{E}$. The OWA plays a vital role in protecting secret information when query answering is the main objective. When answering a query with "Unknown", the querying agent cannot differentiate between the following cases: (1) the case that the answer to the query is actually unknown to the KB reasoner and (2) the case that the answer is being protected in order to maintain secrecy.

Formally, the secrecy set is made of two parts, $\mathbb{S} = S_\Sigma \cup S_{CQ}$, where $S_\Sigma \subseteq \mathcal{A}_0^* \subseteq \mathcal{A}^*$ with $\mathcal{A}_0^*$ the subset of assertions which do not involve fresh individuals, and S_{CQ} is a finite set of BCQs. Clearly, the size of $\mathcal{A}_0^*$ is polynomial to the size of the input KB.

Definition 2.5.1. *Given a knowledge base* $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$ *and a finite secrecy set* $\mathbb{S} = S_\Sigma \cup S_{CQ}$, *where* $S_\Sigma \subseteq \mathcal{A}^*$ *and* S_{CQ} *is a finite set of BCQs, a secrecy envelope for* $\mathbb{S}$, *denoted by* $\mathbb{E}$, *is a set of assertions having the following properties:*

1 $S_\Sigma \subseteq \mathbb{E} \subseteq \mathcal{A}^*$,

2 for every $\alpha \in \mathbb{E}$, $\mathcal{A}^* \setminus \mathbb{E} \not\models^* \alpha$, *and*

3 for every $q \in S_{CQ}$, $\mathcal{A}^* \setminus \mathbb{E} \not\models^* q$ *and* $\mathcal{A}^* \setminus \mathbb{E} \mid\!\not\models^* q$.

$\sqsubseteq^{\leftarrow}_{NL}$ – rule : if $L(a) \in \mathbb{E},\ A \sqsubseteq L \in \mathcal{T}$ and $A(a)\ \in \mathcal{A}^* \setminus \mathbb{E}$,
then $\mathbb{E} := \mathbb{E} \cup \{A(a)\}$;

$\sqsubseteq^{\leftarrow}_{\exists L}$ – rule : if $L(a) \in \mathbb{E},\ \exists R \sqsubseteq L\ \in \mathcal{T}$ and $\underline{inv}(R, a, c)\ \in \mathcal{A}^* \setminus \mathbb{E}$, for some $c \in \mathcal{O}^*$
then $\mathbb{E} := \mathbb{E} \cup \{\underline{inv}(R, a, c)\}$;

$\sqsubseteq^{\leftarrow}_{RE}$ – rule : if $\underline{neg}(E, a, b) \in \mathbb{E},\ R \sqsubseteq\ E\ \in \mathcal{T}$ and $\underline{inv}(R, a, b)\ \in \mathcal{A}^* \setminus \mathbb{E}$,
then $\mathbb{E} := \mathbb{E} \cup \{\underline{inv}(R, a, b)\}$;

$\sqsubseteq^{\leftarrow}_{N\nexists}$ – rule : if $\neg\underline{inv}(R, a, b) \in \mathbb{E},\ A \sqsubseteq \neg\exists R \in \mathcal{T}$ and $A(a)\ \in \mathcal{A}^* \setminus \mathbb{E}$,
then $\mathbb{E} := \mathbb{E} \cup \{A(a)\}$;

$\sqsubseteq^{\leftarrow}_{\exists\nexists}$ – rule : if $\neg\underline{inv}(S, a, b) \in \mathbb{E},\ \exists R \sqsubseteq \neg\exists S\ \in \mathcal{T}$ and $\underline{inv}(R, a, c)\ \in \mathcal{A}^* \setminus \mathbb{E}$,
for some $c \in \mathcal{O}^*$, then $\mathbb{E} := \mathbb{E} \cup \{\underline{inv}(R, a, c)\}$.

Figure 2.5 Secrecy closure rules obtained by inverting rules in Figures 2.1 and 2.2

.

Property 2 says that no information in $\mathbb{E}$ can be entailed from $\mathcal{A}^* \setminus \mathbb{E}$. Property 3 makes sure that BCQs in S_{CQ} can neither be entailed nor disentailed from $\mathcal{A}^* \setminus \mathbb{E}$. To compute an envelope, we use the idea of inverting assertion expansion rules (see Tao et al. (2010), where this approach was first utilized). Induced by the tableau expansion rules in Figure 2.1 (except for the rules $\sqsubseteq_{N\exists}$ and $\sqsubseteq_{\exists\exists}$) and in Figure 2.2, we have the corresponding "inverted" secrecy closure rules in Figure 2.5. The reason for the omission of secrecy closure rules corresponding to the rules $\sqsubseteq_{N\exists}$ and $\sqsubseteq_{\exists\exists}$ is that an application of these rules results in adding assertions with fresh individual names. By the *hidden name assumptions* (HNA), the querying agent is barred from asking any queries that involve fresh individual names, see also Tao et al. (2010).

As an illustration of a secrecy closure rules in Figure 2.5, consider the $\sqsubseteq^{\leftarrow}_{N\nexists}$-rule. Let $\neg P(a, b) \in \mathbb{E},\ A \sqsubseteq \neg\exists P \in \mathcal{T}$ and $A(a) \in \mathcal{A}^* \setminus \mathbb{E}$. If the querying agent asks the query $q = \neg P(a, b)$, then the reasoner $\mathfrak{R}$ could answer "Yes". This is because of the $\sqsubseteq_{N\nexists}$-rule and the fact that $A(a) \notin \mathbb{E}$. So, to protect $\neg P(a, b)$, we have to put $A(a)$ in $\mathbb{E}$. Similarly, in Figure 2.6 the secrecy closure rules are given corresponding to the rules in Figure 2.3. For instance, we consider the $\sqsubseteq^{\leftarrow}_{\exists L\neg}$-rule. Let $\neg P(a, b) \in \mathbb{E}$, $\exists P \sqsubseteq B \in \mathcal{T}$ and $\neg B(a) \in \mathcal{A}^* \setminus \mathbb{E}$. If the querying agent asks the query $q = \neg P(a, b)$, then the reasoner $\mathfrak{R}$ could answer "Yes". This is because of the $\sqsubseteq_{\exists L\neg}$-rule and the fact that $\neg B(a) \notin \mathbb{E}$. So, to protect $\neg P(a, b)$, we have to put $\neg B(a)$

$\sqsubseteq^{\leftarrow}_{NL\neg}$ – rule : if $\neg A(a) \in \mathbb{E},\ A \sqsubseteq L \in \mathcal{T}$ and $\neg L(a)\ \in \mathcal{A}^* \setminus \mathbb{E}$,
then $\mathbb{E} := \mathbb{E} \cup \{\neg L(a)\}$;

$\sqsubseteq^{\leftarrow}_{N\exists\neg}$ – rule : if $\neg A(a) \in \mathbb{E},\ A \sqsubseteq \exists R \in \mathcal{T}$ and $\forall b \in \mathcal{O}^*,\ \neg\underline{inv}(R,a,b) \in \mathcal{A}^* \setminus \mathbb{E}$,
then pick a $c \in \mathcal{O}^*$ such that $\mathbb{E} := \mathbb{E} \cup \{\neg\underline{inv}(R,a,c)\}$;

$\sqsubseteq^{\leftarrow}_{\exists L\neg}$ – rule : if $\neg\underline{inv}(R,a,b) \in \mathbb{E},\ \exists R \sqsubseteq L\ \in \mathcal{T}$ and $\neg L(a)\ \in \mathcal{A}^* \setminus \mathbb{E}$,
then $\mathbb{E} := \mathbb{E} \cup \{\neg L(a)\}$;

$\sqsubseteq^{\leftarrow}_{\exists\exists\neg}$ – rule : if $\neg\underline{inv}(R,a,b) \in \mathbb{E},\ \exists R \sqsubseteq \exists S\ \in \mathcal{T}$ and $\forall c \in \mathcal{O}^*, \neg\underline{inv}(S,a,c)$
$\in \mathcal{A}^* \setminus \mathbb{E}$, then pick a $d \in \mathcal{O}^*$ such that $\mathbb{E} := \mathbb{E} \cup \{\neg\underline{inv}(S,a,d)\}$;

$\sqsubseteq^{\leftarrow}_{RE\neg}$ – rule : if $\neg\underline{inv}(R,a,b) \in \mathbb{E},\ R \sqsubseteq\ E\ \in \mathcal{T}$ and $\neg\underline{neg}(E,a,b)\ \in \mathcal{A}^* \setminus \mathbb{E}$,
then $\mathbb{E} := \mathbb{E} \cup \{\neg\underline{neg}(E,a,b)\}$;

$\sqsubseteq^{\leftarrow}_{N\not\exists\neg}$ – rule : if $\neg A(a) \in \mathbb{E},\ A \sqsubseteq \neg\exists R \in \mathcal{T}$ and $\underline{inv}(R,a,c) \in \mathcal{A}^* \setminus \mathbb{E}$,
for some $c \in \mathcal{O}^*$, then $\mathbb{E} := \mathbb{E} \cup \{\underline{inv}(R,a,b)\}$;

$\sqsubseteq^{\leftarrow}_{\exists\not\exists\neg}$ – rule : if $\neg\underline{inv}(R,a,b) \in \mathbb{E},\ \exists R \sqsubseteq \neg\exists S\ \in \mathcal{T}$ and $\underline{inv}(S,a,c)\ \in \mathcal{A}^* \setminus \mathbb{E}$,
for some $c \in \mathcal{O}^*$, then $\mathbb{E} := \mathbb{E} \cup \{\underline{inv}(S,a,c)\}$

Figure 2.6 Secrecy closure rules obtained by inverting rules in Figure 2.3

in $\mathbb{E}$. In both cases, these secrecy closure rules are named by adding the superscript $\leftarrow$ in the name of the corresponding assertion expansion rules.

Rules that specifically deal with BCQs are given in Figure 2.7. Few words of explanation may be helpful in understanding BCQ-rules. These rules have been designed to protect BCQ's in S_{CQ}. Let $q \in S_{CQ}$ be a BCQ. To protect q, we use BCQ$_h$-rule which "disrupts" each homomorphism $h : G[q] \to G[\mathcal{A}^* \setminus \mathbb{E}]$ and adds to $\mathbb{E}$ one of the atoms of q (whose variables are evaluated under h). Similarly, in the BCQ$_c$-rule, we pick an arbitrary clashy mapping $g : G[q] \to G[\mathcal{A}^* \setminus \mathbb{E}]$ and make it into a non-clashy mapping: This can be done by considering all the clashy atoms of q under g ($A \in L[q](v)$ and $\neg A \in L[\mathcal{A}^* \setminus \mathbb{E}](g(v))$, or $P \in L[q]((u,v))$ and $\neg P \in L[\mathcal{A}^* \setminus \mathbb{E}](g(u), g(v))$) and adding them to $\mathbb{E}$.

The computation of $\mathbb{E}$ proceeds in two stages. In the first step, $\mathbb{E}$ is initialized as S_Σ and expanded by using secrecy closure rules listed in Figures 2.5 and 2.6. In the second stage, $\mathbb{E}$ is expanded by using BCQ$_h$ and BCQ$_c$-rules. We denote by Λ_S the tableau algorithm which computes the envelope $\mathbb{E}$ by using secrecy closure rules listed in Figures 2.5, 2.6 and 2.7 until

$\text{BCQ}_h - \text{rule}$: if $q \in S_{CQ}$, and there is a labeled homomorphism
$h : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ such that
$\{A_1(h(\zeta_1)), .., \ A_k(h(\zeta_k)), \ P_1(h(\eta_1), h(\mu_1)), ..,$
$P_m(h(\eta_m), h(\mu_m))\} \cap \mathbb{E} = \varnothing$
then $\mathbb{E} := \mathbb{E} \cup \{A_p(h(\zeta_p))\}$ for some $1 \leq p \leq k$ or
$\mathbb{E} := \mathbb{E} \cup \{P_r(h(\eta_r), h(\mu_r))\}$ for some $1 \leq r \leq m$;

$\text{BCQ}_c - \text{rule}$: if $q \in S_{CQ}$, and every $f : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ is clashy, then
pick one such clashy mapping g. Then,

- $\forall p, 1 \leq p \leq k$, if $\neg A_p(g(\zeta_p)) \in \mathcal{A}^* \setminus \mathbb{E}$ then
  $\mathbb{E} := \mathbb{E} \cup \{\neg A_p(g(\zeta_p))\}$, and
- $\forall r, 1 \leq r \leq m$, if $\neg P_r(g(\eta_r), g(\mu_r)) \in \mathcal{A}^* \setminus \mathbb{E}$ then
  $\mathbb{E} := \mathbb{E} \cup \{\neg P_r(g(\eta_r), g(\mu_r))\}$.

$q = \exists y_1, ., y_n[A_1(\zeta_1) \wedge ... \wedge A_k(\zeta_k) \wedge P_1(\eta_1, \mu_1) \wedge .... \wedge P_m(\eta_m, \mu_m)]$

Figure 2.7 Secrecy closure rules for $q \in S_{CQ}$

no more rules are applicable. Due to non-determinism in applying the BCQ-rules, different executions of Λ_S may result different envelopes. Since $\mathcal{A}^*$ is finite, the computation of Λ_S terminates. Let $\mathbb{E}$ be the output of Λ_S. By the assumption that $S_\Sigma \subseteq \mathcal{A}^*$, and by the BCQ_h- and BCQ_c-rules, it is easy to see that $\mathbb{E} \subseteq \mathcal{A}^*$.

Example 2.5.1. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$ be a DL-Lite$_\mathcal{R}$ KB, where $\mathcal{A} = \{A(a), B(a), E(a), \neg F(a)\}$ and $\mathcal{T} = \{A \sqsubseteq D, \ A \sqsubseteq \neg C, \ A \sqsubseteq \exists P, \ B \sqsubseteq \exists P, \ \exists P^- \sqsubseteq \neg C, \ \exists P^- \sqsubseteq \neg F, \ P \sqsubseteq Q\}$. Also let $\mathbb{S} = \{D(a), \exists y_1, y_2[A(y_1) \wedge P(y_1, y_2)], \exists y_1, y_2[P(y_1, y_2) \wedge C(y_2)]\}$ be the secrecy set. Using the assertion expansion rules in Figures 2.1, 2.2 and 2.3, we get $\mathcal{A}^* = \{A(a), B(a), \neg C(a), \neg C(b), \neg C(c), D(a), E(a), \neg F(a), \ \neg F(b), \neg F(c), P(a, b), P(a, c), Q(a, b), Q(a, c)\}$. Using the secrecy closure rules in Figures 2.5, 2.6 and 2.7, we get $\mathbb{E} = \{A(a), D(a), \neg C(b)\}$. Then graphs for $\mathcal{A}^*$ and $\mathcal{A}^* \setminus \mathbb{E}$ are given in Figure 2.8.*

The following results show that no assertion in the envelope $\mathbb{E}$ is "logically reachable" from outside the envelope.

Lemma 2.5.1. *Let $\mathcal{A}^*$ be a completed ABox obtained from Σ by first applying the rules in Figure 2.1, then in Figure 2.2 and then rules in Figure 2.3 as specified in Section 2.3. Also,*

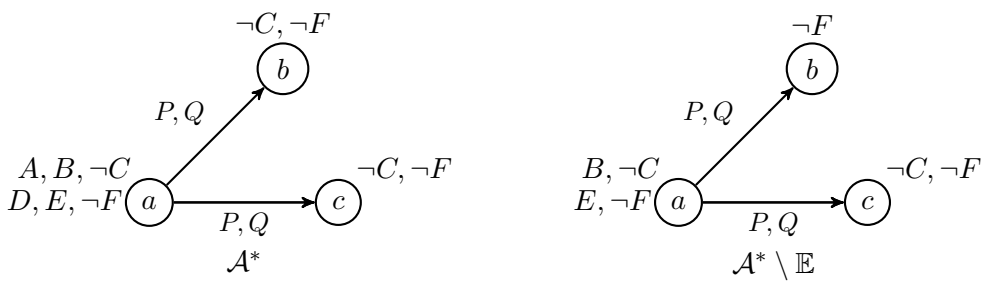

Figure 2.8 The graphs of $\mathcal{A}^*$ and $\mathcal{A}^* \setminus \mathbb{E}$

let $\mathbb{E}$ be a set of assertions which is completed by first using rules in Figures 2.5 and 2.6, and then rules in Figure 2.7. Then, the ABox $\mathcal{A}^* \setminus \mathbb{E}$ is completed.

Proof. *We have to show that no rule in Figures 2.1, 2.2 or 2.3 is applicable to $\mathcal{A}^* \setminus \mathbb{E}$. The proof is by contradiction according to cases. In each case, there are several sub cases. To avoid repetition, we give proof for one sub case.*

- *If $\sqsubseteq_{NL}$-rule is applicable, then there is an assertion $A(a) \in \mathcal{A}^* \setminus \mathbb{E}$ and a subsumption $A \sqsubseteq L \in \mathcal{T}$ such that $L(a) \notin \mathcal{A}^* \setminus \mathbb{E}$. Since $\mathcal{A}^*$ is completed, $L(a) \in \mathcal{A}^*$. Hence, $L(a) \in \mathbb{E}$. This makes the $\sqsubseteq^{\leftarrow}_{NL}$-rule applicable, contrary to the assumption that $\mathbb{E}$ is completed.*

- *If $\sqsubseteq_{N\exists}$-rule is applicable, we have two cases (i) $R = P$, and (ii) $R = P^-$. We argue the latter case. By assumption, $A(a) \in \mathcal{A}^* \setminus \mathbb{E}$, $A \sqsubseteq \exists P^- \in \mathcal{T}$ and there is no $b \in \mathcal{O}^*$ such that $P(b, a) \in \mathcal{A}^* \setminus \mathbb{E}$. Since $\mathcal{A}^*$ is completed, there is a $c \in \mathcal{O}^*$ such that $P(c, a) \in \mathcal{A}^*$. Hence $P(c, a) \in \mathbb{E}$. This makes the $\sqsubseteq^{\leftarrow}_{N\exists}$-rule applicable, which contradicts to the assumption that $\mathbb{E}$ is completed.*

- *If $\sqsubseteq_{\exists L}$-rule is applicable, we have two cases (i) $R = P$, and (ii) $R = P^-$. We consider the latter case. By assumption, $P(b, a) \in \mathcal{A}^* \setminus \mathbb{E}$, $\exists P^- \sqsubseteq L \in \mathcal{T}$ and $L(a) \notin \mathcal{A}^* \setminus \mathbb{E}$. Since $\mathcal{A}^*$ is completed, $L(a) \in \mathcal{A}^*$. Hence, $L(a) \in \mathbb{E}$. This makes the $\sqsubseteq^{\leftarrow}_{\exists L}$-rule applicable, a contradiction.*

- *If $\sqsubseteq_{\exists\exists}$-rule is applicable, we have four cases (i) $R = P, S = Q$, (ii) $R = P, S = Q^-$, (iii) $R = P^-, S = Q$ and (iv) $R = P^-, S = Q^-$. We consider case (ii). By assumption, $P(a, b) \in \mathcal{A}^* \setminus \mathbb{E}$, $\exists P \sqsubseteq \exists Q^- \in \mathcal{T}$ and for all $d \in \mathcal{O}^*$ such that $Q(d, a) \notin \mathcal{A}^* \setminus \mathbb{E}$. Since $\mathcal{A}^*$*

is completed, there is a $c \in \mathcal{O}^*$ *such that* $Q(c,a) \in \mathcal{A}^*$. *Hence* $Q(c,a) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{\exists\exists}$*-rule applicable, contrary to the assumption that* $\mathbb{E}$ *is completed.*

- *If* $\sqsubseteq_{RE}$*-rule is applicable, we have eight cases (i)* $R = P, E = Q$, *(ii)* $R = P, E = Q^-$, *(iii)* $R = P^-, E = Q$, *(iv)* $R = P^-, E = Q^-$, *(v)* $R = P, E = \neg Q$, *(vi)* $R = P, E = \neg Q^-$, *(vii)* $R = P^-, E = \neg Q$ *and (viii)* $R = P^-, E = \neg Q^-$. *We consider case (vii). By assumption,* $P(b,a) \in \mathcal{A}^* \setminus \mathbb{E}$, $P^- \sqsubseteq \neg Q \in \mathcal{T}$ *and* $\neg Q(a,b) \notin \mathcal{A}^* \setminus \mathbb{E}$. *Since* $\mathcal{A}^*$ *is completed,* $\neg Q(a,b) \in \mathcal{A}^*$. *Hence,* $\neg Q(a,b) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{RE}$*-rule applicable, a contradiction.*

- *If* $\sqsubseteq_{N\nexists}$*-rule is applicable, we have two cases (i)* $R = P$, *and (ii)* $R = P^-$. *We consider the latter case. By assumption,* $A(a) \in \mathcal{A}^* \setminus \mathbb{E}$, $A \sqsubseteq \neg\exists P^- \in \mathcal{T}$ *and* $\neg P(b,a) \notin \mathcal{A}^* \setminus \mathbb{E}$ *for some* $b \in \mathcal{O}^*$. *Since* $\mathcal{A}^*$ *is completed,* $\neg P(b,a) \in \mathcal{A}^*$. *Hence,* $\neg P(b,a) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{N\nexists}$*-rule applicable, a contradiction.*

- *If* $\sqsubseteq_{\exists\nexists}$*-rule is applicable, we have four cases (i)* $R = P, S = Q$, *(ii)* $R = P, S = Q^-$, *(iii)* $R = P^-, S = Q$ *and (iv)* $R = P^-, S = Q^-$. *We consider case (iii). By assumption,* $P(b,a) \in \mathcal{A}^* \setminus \mathbb{E}$, $\exists P^- \sqsubseteq \neg\exists Q \in \mathcal{T}$ *and* $\neg Q(a,c) \notin \mathcal{A}^* \setminus \mathbb{E}$ *for some* $c \in \mathcal{O}^*$. *Since* $\mathcal{A}^*$ *is completed,* $\neg Q(a,c) \in \mathcal{A}^*$. *Hence,* $\neg Q(a,c) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{\exists\nexists}$*-rule applicable, a contradiction.*

- *If* $\sqsubseteq_{NL\neg}$*-rule is applicable, then there is an assertion* $\neg L(a) \in \mathcal{A}^* \setminus \mathbb{E}$, $A \sqsubseteq L \in \mathcal{T}$ *and* $\neg A(a) \notin \mathcal{A}^* \setminus \mathbb{E}$. *Since* $\mathcal{A}^*$ *is completed,* $\neg A(a) \in \mathcal{A}^*$. *Hence,* $\neg A(a) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{NL\neg}$*-rule applicable, a contradiction.*

- *If* $\sqsubseteq_{N\exists\neg}$*-rule is applicable, we have two cases (i)* $R = P$, *and (ii)* $R = P^-$. *We argue the first case. By assumption,* $\forall b \in \mathcal{O}^*$, $\neg P(a,b) \in \mathcal{A}^* \setminus \mathbb{E}$, $A \sqsubseteq \exists P \in \mathcal{T}$ *and* $\neg A(a) \notin \mathcal{A}^* \setminus \mathbb{E}$. *Since* $\mathcal{A}^*$ *is completed,* $\neg A(a) \in \mathcal{A}^*$. *Hence,* $\neg A(a) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{N\exists\neg}$*-rule applicable, a contradiction.*

- *If* $\sqsubseteq_{\exists L\neg}$*-rule is applicable, we have two cases (i)* $R = P$, *and (ii)* $R = P^-$. *We argue the second case. By assumption,* $\neg L(a) \in \mathcal{A}^* \setminus \mathbb{E}$, $\exists P^- \sqsubseteq L \in \mathcal{T}$ *and* $\neg P(b,a) \notin \mathcal{A}^* \setminus \mathbb{E}$, *for*

some $b \in \mathcal{O}^*$. *Since* $\mathcal{A}^*$ *is completed,* $\neg P(b,a) \in \mathcal{A}^*$. *Hence,* $\neg P(b,a) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{\exists L\neg}$*-rule applicable, a contradiction.*

- *If* $\sqsubseteq_{\exists\exists\neg}$*-rule is applicable, we have four cases (i)* $R = P, S = Q$, *(ii)* $R = P, S = Q^-$, *(iii)* $R = P^-, S = Q$ *and (iv)* $R = P^-, S = Q^-$. *We consider case (i). By assumption,* $\forall b \in \mathcal{O}^*\ \neg Q(a,b) \in \mathcal{A}^* \setminus \mathbb{E}$, $\exists P \sqsubseteq \exists Q \in \mathcal{T}$ *and* $\neg P(a,c) \notin \mathcal{A}^* \setminus \mathbb{E}$ *for some* $c \in \mathcal{O}^*$. *Since* $\mathcal{A}^*$ *is completed,* $\neg P(a,c) \in \mathcal{A}^*$. *Hence,* $\neg P(a,c) \in \mathbb{E}$. *This makes the* $\sqsubseteq_{\exists\exists\neg}$*-rule applicable, a contradiction.*

- *If* $\sqsubseteq_{RE\neg}$*-rule is applicable, we have eight cases (i)* $R = P, E = Q$, *(ii)* $R = P, E = Q^-$, *(iii)* $R = P^-, E = Q$, *(iv)* $R = P^-, E = Q^-$, *(v)* $R = P, E = \neg Q$, *(vi)* $R = P, E = \neg Q^-$, *(vii)* $R = P^-, E = \neg Q$ *and (viii)* $R = P^-, E = \neg Q^-$. *We consider case (v). By assumption,* $Q(a,b) \in \mathcal{A}^* \setminus \mathbb{E}$, $P \sqsubseteq \neg Q \in \mathcal{T}$ *and* $\neg P(a,b) \notin \mathcal{A}^* \setminus \mathbb{E}$. *Since* $\mathcal{A}^*$ *is completed,* $\neg P(a,b) \in \mathcal{A}^*$. *Hence,* $\neg P(a,b) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{RE\neg}$*-rule applicable, a contradiction.*

- *If* $\sqsubseteq_{N\not\exists\neg}$*-rule is applicable, we have two cases (i)* $R = P$, *and (ii)* $R = P^-$. *We argue the second case. By assumption,* $P(b,a) \in \mathcal{A}^* \setminus \mathbb{E}$, $A \sqsubseteq \neg\exists P^- \in \mathcal{T}$ *and* $\neg A(a) \notin \mathcal{A}^* \setminus \mathbb{E}$. *Since* $\mathcal{A}^*$ *is completed,* $\neg A(a) \in \mathcal{A}^*$. *Hence,* $\neg A(a) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{N\not\exists\neg}$*-rule applicable, a contradiction.*

- *If* $\sqsubseteq_{\exists\not\exists\neg}$*-rule is applicable, we have four cases (i)* $R = P, S = Q$, *(ii)* $R = P, S = Q^-$, *(iii)* $R = P^-, S = Q$ *and (iv)* $R = P^-, S = Q^-$. *We consider case (iv). By assumption,* $Q(b,a) \in \mathcal{A}^* \setminus \mathbb{E}$ *and* $\exists P^- \sqsubseteq \neg\exists Q^- \in \mathcal{T}$ *such that* $\neg P(c,a) \notin \mathcal{A}^* \setminus \mathbb{E}$ *for some* $c \in \mathcal{O}^*$. *Since* $\mathcal{A}^*$ *is completed,* $\neg P(c,a) \in \mathcal{A}^*$. *Hence,* $\neg P(c,a) \in \mathbb{E}$. *This makes the* $\sqsubseteq^{\leftarrow}_{\exists\not\exists\neg}$*-rule applicable, a contradiction.* $\square$

The following corollary states, roughly, that the secret BCQs are not logically reachable from $\mathcal{A}^* \setminus \mathbb{E}$.

Corollary 2.5.1. *Let* $\mathbb{E}'$ *be any subset of* $\mathcal{A}^*$ *which is completed with respect to secrecy closure rules listed in Figure 2.7. Then, for every* $q \in S_{CQ}$,

- *there is no labeled graph homomorphism* $h : V[q] \xrightarrow[c]{} V[\mathcal{A}^* \setminus \mathbb{E}']$, *and*

- *there exists at least one mapping* $f : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}']$ *which is not clashy.*

Proof. *Let* $\mathbb{E}'$ *be completed with respect to secrecy closure rules listed in Figure 2.7. This implies that for every* $q \in S_{CQ}$*, no* BCQ_h*-rule is applicable to* q*. Hence, by the conditions of* BCQ_h*-rule, there is no labeled graph homomorphism* $h : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}']$*, for any* $q \in S_{CQ}$*. Similarly, no* BCQ_c*-rule is applicable to* q*. It follows that for each* $q \in S_{CQ}$*, there exist at least one mapping* $f : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}']$ *which is not clashy.* $\square$

We now show that the completed set $\mathbb{E}$ (an output of Λ_S), is in fact an envelope.

Theorem 2.5.1. $\mathbb{E}$ *is an envelope for* $\mathbb{S}$*.*

Proof. *We must show that the set* $\mathbb{E}$ *satisfies the properties of Definition 2.5.1. Clearly,* $S_\Sigma \subseteq \mathbb{E}$*. First we show that, for every* $\alpha \in \mathbb{E}$*,* $\mathcal{A}^* \setminus \mathbb{E} \not\models^* \alpha$*. Suppose* $\mathcal{A}^* \setminus \mathbb{E} \models^* \alpha$*, for some* $\alpha \in \mathbb{E}$*. By Theorem 2.3.2, we have* $\alpha \in (\mathcal{A}^* \setminus \mathbb{E})^*$ *and by Lemma 2.5.1,* $\alpha \in \mathcal{A}^* \setminus \mathbb{E}$*, a contradiction.*

Next we show that for each $q \in S_{CQ}$*,* $\mathcal{A}^* \setminus \mathbb{E} \not\models^* q$ *and* $\mathcal{A}^* \setminus \mathbb{E} \mid\not\models^* q$*.*

- *Assume* $\mathcal{A}^* \setminus \mathbb{E} \models^* q$*. Then, for every OW-model* $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ *of* $(\mathcal{A}^* \setminus \mathbb{E}, \mathcal{T})$*,* $\mathcal{I}^* \models q$ *where* $\mathcal{I}^* = (\Delta^* = \mathcal{I}^*(\mathcal{O}^*), \cdot^{\mathcal{I}^*})$*. Let* $\mathcal{J}$ *be the canonical model of* $\mathcal{A}^* \setminus \mathbb{E}$*. Then,* $\mathcal{J}^* = \mathcal{J}$*, and* $\pi_{\mathcal{J}} : V[q] \underset{c}{\rightarrow} \Delta^* = \mathcal{J}(\mathcal{O}^*) = \mathcal{O}^*$ *and* $(\mathcal{J}, \pi_{\mathcal{J}}) \models \beta$*, for every* $\beta \in Atoms(q)$*.*

  Now, let $v \in V[q]$ *and* $A(v) \in Atoms(q)$*. Then,* $(\mathcal{J}, \pi_{\mathcal{J}}) \models A(v) \Rightarrow \pi_{\mathcal{J}}(v) \in A_Y^{\mathcal{J}} \Rightarrow A(\pi_{\mathcal{J}}(v)) \in \mathcal{A}^* \setminus \mathbb{E} \Rightarrow A \in L[\mathcal{A}^* \setminus \mathbb{E}](\pi_{\mathcal{J}}(v))$*. Similarly, let* $u, v \in V[q]$ *and* $P(u, v) \in Atoms(q)$*. Then,* $(\mathcal{J}, \pi_{\mathcal{J}}) \models P(u, v) \Rightarrow (\pi_{\mathcal{J}}(u), \pi_{\mathcal{J}}(v)) \in P_Y^{\mathcal{J}} \Rightarrow P((\pi_{\mathcal{J}}(u), \pi_{\mathcal{J}}(v))) \in \mathcal{A}^* \setminus \mathbb{E} \Rightarrow P \in L[\mathcal{A}^* \setminus \mathbb{E}]((\pi_{\mathcal{J}}(u), \pi_{\mathcal{J}}(v)))$*. It follows that,* $\pi_{\mathcal{J}} : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ *is a labeled graph homomorphism contradicting Corollary 2.5.1.*

- *Assume* $\mathcal{A}^* \setminus \mathbb{E} \mid\models^* q$*. Then, for every OW-model* $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ *of* $(\mathcal{A}^* \setminus \mathbb{E}, \mathcal{T})$*,* $\mathcal{I}^* \mid\models q$ *where* $\mathcal{I}^* = (\Delta^* = \mathcal{I}^*(\mathcal{O}^*), \cdot^{\mathcal{I}^*})$*. Let* $\mathcal{J}$ *be the canonical model of* $\mathcal{A}^* \setminus \mathbb{E}$*. Then,* $\mathcal{J}^* = \mathcal{J}$ *and for each valuation* $\pi : V[q] \underset{c}{\rightarrow} \Delta^* = \mathcal{J}(\mathcal{O}^*) = \mathcal{O}^*$*,* $(\mathcal{J}, \pi) \models \neg\beta$*, for some* $\beta \in Atoms(q)$*. Let* k *be any such valuation. Then,* $(\mathcal{J}, k) \models \neg A(v)$ *for some* $A(v) \in Atoms(q)$ *or* $(\mathcal{J}, k) \models \neg P(u, v)$ *for some* $P(u, v) \in Atoms(q)$*. In the first case,* $k(v) \in A_N^{\mathcal{J}} \Rightarrow \neg A(k(v)) \in \mathcal{A}^* \setminus \mathbb{E} \Rightarrow \neg A \in L[\mathcal{A}^* \setminus \mathbb{E}](k(v))$ *and in the second case,*

$(k(u), k(v)) \in P_N^{\mathcal{J}} \Rightarrow \neg P((k(u), k(v))) \in \mathcal{A}^* \setminus \mathbb{E} \Rightarrow \neg P \in L[\mathcal{A}^* \setminus \mathbb{E}]((k(u), k(v)))$. Hence, $k : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ is clashy. Since k was arbitrary, it follows that all valuations are clashy. However, $\mathbb{E}$ is completed, so by Corollary 2.5.1 there exist at least one mapping $k : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ which is not clashy. This is a contradiction. Hence, $\mathcal{A}^* \setminus \mathbb{E} \not\models^* q$.
□

Ideally, we would like to compute a minimum envelope $\mathbb{E}$ which makes query answering as informative as possible without compromising the secrecy. But, computing minimum envelope appears to be hard, see Tao et al. (2014). So, our focus now is to compute a minimal envelope with the property that removing any one of the assertions in $\mathbb{E}$ would reveal some of the secrets. We call such an envelope a *tight envelope*.

Definition 2.5.2. *An envelope $\mathbb{E}$ is said to be tight if for every $\alpha \in \mathbb{E}$, $\mathbb{E} \setminus \{\alpha\}$ is not an envelope.*

Next, we observe that an envelope computed using the rules in Figures 2.5, 2.6 and 2.7 need not be tight.

Example 2.5.2. *Consider a DL-Lite$_{\mathcal{R}}$ KB, where $\mathcal{A} = \{W(a,b), W(a,c)\}$ and $\mathcal{T} = \{\exists W \sqsubseteq A, \exists W^- \sqsubseteq B\}$. Let $\mathbb{S} = \{\exists y, z[A(y) \wedge W(y,z) \wedge B(z)]\}$ be the secrecy set. Using the rules in Figure 1, we compute $\mathcal{A}^* = \{A(a), B(b), B(c), W(a,b), W(a,c)\}$. Since Λ_S is a non-deterministic algorithm, Λ_S may output different envelopes. For illustration purposes, we considered two envelopes namely $\mathbb{E}_1 = \{A(a), W(a,b), W(a,c)\}$ and $\mathbb{E}_2 = \{W(a,b), W(a,c)\}$. It is easy to see that $\mathbb{E}_2$ is tight, whereas $\mathbb{E}_1$ is not.*

We now present a naive approach to compute a tight envelope. Given a precomputed $\mathcal{A}^*$ and a secrecy set $\mathbb{S} = S_\Sigma \cup S_{CQ}$, we can compute an envelope $\mathbb{E}$ of $\mathbb{S}$ as explained in the beginning of this section. An assertion $\alpha \in \mathbb{E} \setminus \mathbb{S}$ is said to be *redundant* if $\mathbb{E} \setminus \{\alpha\}$ is an envelope, i.e., $((\mathcal{A}^* \setminus \mathbb{E}) \cup \{\alpha\})^* \cap (\mathbb{E} \setminus \{\alpha\}) = \emptyset$. To compute a tight envelope, for each $\beta \in \mathbb{E} \setminus \mathbb{S}$ we check whether β is redundant in which case it is moved from $\mathbb{E}$ to $\mathcal{A}^* \setminus \mathbb{E}$. Otherwise, β remains in $\mathbb{E}$. Updating the resulting ABox $\mathcal{A}^* \setminus \mathbb{E}$ could possibly take exponential time.

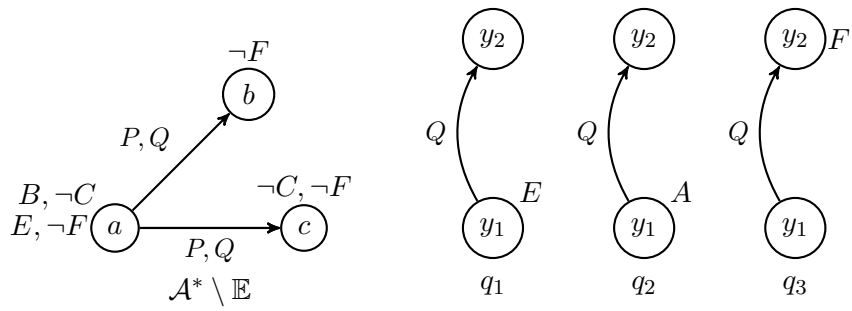

Figure 2.9 The graphs of $\mathcal{A}^* \setminus \mathbb{E}$ and queries

2.6 Answering Queries

At this point we assume that $\mathcal{A}^*$ and $\mathbb{E}$ have been precomputed. From an algorithmic point of view, answering queries may be based on checking membership in the set $\mathcal{A}^* \setminus \mathbb{E}$ or searching for specific graph structures in the graph $G[\mathcal{A}^* \setminus \mathbb{E}]$. Suppose that the agent poses query q of the form $C(a)$ or $E(a, b)$. Then, the reasoner checks for the membership of q and $\neg q$ in the set $\mathcal{A}^* \setminus \mathbb{E}$. If $q \in \mathcal{A}^* \setminus \mathbb{E}$, then the reasoner should answer "Yes" by Theorem 2.4.1. If $\neg q \in \mathcal{A}^* \setminus \mathbb{E}$, then the reasoner should answer "No" by Theorem 2.4.2. If neither q nor $\neg q$ is in $\mathcal{A}^* \setminus \mathbb{E}$, then the reasoner should answer "Unknown".

Now suppose that the agent poses BCQ q. Then, the reasoner considers the mappings $V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$. If there exists a labeled homomorphism $h : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$, then the reasoner should answer "Yes" by Theorem 2.4.1. If every such mapping is clashy, then the reasoner should answer "No", see Theorem 2.4.2. Otherwise, the reasoner should answer "Unknown".

Example 2.6.1. *We use the KB, the secrecy set $\mathbb{S}$ and the envelope $\mathbb{E}$ considered in Example 2.5.1. Answers for the BCQs q_1, q_2 and q_3 whose query graphs are in Figure 2.9, are computed in the following based on $\mathcal{A}^* \setminus \mathbb{E}$.*

First let us consider the BCQ $q_1 = \exists y_1, y_2[E(y_1) \wedge Q(y_1, y_2)]$. Since there exists a homomorphism from $G[q_1]$ to $G[\mathcal{A}^* \setminus \mathbb{E}]$, namely, $y_1 \mapsto a$, $y_2 \mapsto b$ and since $L[q_1](y_1) \subseteq L[\mathcal{A}^* \setminus \mathbb{E}](a)$, $L[q_1](y_1, y_2) \subseteq L[\mathcal{A}^* \setminus \mathbb{E}](a, b)$, $L[q_1](y_2) \subseteq L[\mathcal{A}^* \setminus \mathbb{E}](b)$, the answer to q_1 is "Yes". Actually, there are two labeled homomorphisms from $G[q_1]$ to $G[\mathcal{A}^* \setminus \mathbb{E}]$, the other one being, $y_1 \mapsto a$, $y_2 \mapsto c$.

Next, $q_2 = \exists y_1, y_2[A(y_1) \wedge Q(y_1, y_2)]$. *Since there is no labeled homomorphism and there exist non-clashy mappings from* $G[q_2]$ *to* $G[\mathcal{A}^* \setminus \mathbb{E}]$, *e.g.,* $y_1 \mapsto a,\ y_2 \mapsto b$, *answer to* q_2 *is "Unknown".*

Finally, consider the BCQ $q_3 = \exists y_1, y_2[Q(y_1, y_2) \wedge F(y_2)]$. *It is easy to see that all the mappings from* $G[q_3]$ *to* $G[\mathcal{A}^* \setminus \mathbb{E}]$ *are clashy. Hence, answer for the BCQ* q_3 *is "No".*

2.7 Complexities of computing $\mathcal{A}^*$, $\mathbb{E}$ and Query Answering

Recall that answering conjunctive queries in *DL-Lite*$_{\mathcal{R}}$ is in LogSpace with respect to *data complexity* (i.e., as a function of the size of the ABox, keeping the TBox and query fixed), and *NP-complete* with respect to *combined complexity*, (i.e., as a function of both the size of the KB and the query) see Calvanese et al. (2007); Ortiz and Šimkus (2012). Here we provide a brief discussion of the computational complexities of various algorithms given in this chapter.

a) Computation of the assertional closure $\mathcal{A}^*$: Starting with the input ABox $\mathcal{A}$, the algorithm expands it using the assertion expansion rules given in Figures 2.1, 2.2 and 2.3 in that order. Clearly, $\mathcal{A}^*$ can be computed in exponential time as a function of the size of the input KB.

b) Computation of the envelope $\mathbb{E}$: Initializing the set $\mathbb{E}$ as S_Σ, the algorithm first expands $\mathbb{E}$ using the secrecy closure rules given in Figures 2.5 and 2.6. Further the algorithm expands the resulting set $\mathbb{E}$ using the rules in Figure 8 for the BCQs in S_{CQ} until no more application of rules in Figures 2.5, 2.6 and 2.7 are possible. The most time-consuming step in the computation of $\mathbb{E}$ is the application of the BCQ$_h$-rule which may require an enumeration of all graph labeled homomorphisms $h : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$. This may incur substantial computational cost; in fact enumerating all graph homomorphisms $h : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ requires time exponential in the input size $|V[q]| + |V[\mathcal{A}^* \setminus \mathbb{E}]|$ as the corresponding counting problem is $\#P$-complete, see Dyer and Greenhill (2000).

It is important to note that since our main interest is in query-answering, the calculations (a) and (b) need to be performed just once before the query-answer phase begins.

c) Computation of answers to queries: It is obvious that the cost of computing an answer to a BCQ is higher than the cost of answering an instance query. By Hidden Name Assumption, the querying agent can ask instance queries about assertions in the set $\mathcal{A}_0^*$ whose size is of polynomial in the input KB, see Section 2.5. Hence, answering instance queries can be done in polynomial time as a function of the size of the input KB.

For answering a BCQ q, the reasoner $\mathfrak{R}$ computes a function which returns "Yes", "No" or "Unknown" in the following way: First $\mathfrak{R}$ checks if there is a labeled homomorphism $h : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ in which case $\mathfrak{R}$ answers "Yes"; if no such homomorphism exists, $\mathfrak{R}$ checks if there is a non-clashy mapping $k : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ in which case $\mathfrak{R}$ answers "Unknown"; otherwise, $\mathfrak{R}$ outputs "No".

It is known that the problem of existence of a labeled homomorphism $h : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ is $NP - complete$ and the problem of existence of a non-clashy mapping $k : V[q] \underset{c}{\rightarrow} V[\mathcal{A}^* \setminus \mathbb{E}]$ clearly belongs to NP. It is easy to see that these tests can be performed deterministically in $|V[\mathcal{A}^* \setminus \mathbb{E}]|^{O(|V[q]|)}$ time. Observe that this upper bound is polynomial in the KB size (when the query is fixed) and it is exponential in the query size (when KB is fixed).

2.8 Conclusions

In this chapter we have studied the problem of secrecy-preserving query answering over acyclic *DL-Lite*$_{\mathcal{R}}$ KBs. We have extended the conceptual logic-based framework for secrecy-preserving reasoning which was introduced by Tao et al., see Tao et al. (2014), so as to allow BCQs. As the OWA underlies the foundational aspects of KBs, to show that the reasoner is sound and complete we used the semantics based on Kleene's 3-valued logic, see Avron (1991); Tao et al. (2014). We provide syntactic characterizations for entailment and disentailment of BCQs in terms of properties of mappings (Section 2.4).

CHAPTER 3. SECRECY-PRESERVING QUERY ANSWERING IN $\mathcal{EL}\mathcal{H}$ KNOWLEDGE BASES

3.1 Introduction

In literature, most of the approaches dealing with "information protection" are based on access control mechanisms. For semantic web applications, the authors of Kagal et al. (2003) have proposed policy languages to represent obligation and delegation policies based on access control approach. Biskup et al. in Biskup and Weibert (2008); Biskup and Tadros (2012) studied secrecy in incomplete databases using controlled query evaluation (CQE). Since description logics (DLs) underlie web ontology languages (OWLs), recently researchers have shown an interest in studying secrecy-preserving reasoning in DL knowledge bases (KBs).

In Bao et al. (2007); Tao et al. (2010, 2014), the authors have developed a secrecy framework that attempts to satisfy the following competing goals: (a) it protects secret information and (b) queries are answered as informatively as possible (subject to satisfying property (a)). The notion of an *envelope* to hide secret information against logical inference was first defined and used in Tao et al. (2010). Further, in Tao et al. (2014), Tao et al., introduced a more elaborate conceptual framework for secrecy-preserving query answering (SPQA) under Open World Assumption (OWA) with multiple querying agents. This approach is based on OWA and (so far) it has been restricted to instance-checking queries. Specifically, in Bao et al. (2007); Tao et al. (2010, 2014) the main idea was to utilize the secret information within the reasoning process, but then answering "Unknown" whenever the answer is truly unknown or in case the true answer could compromise confidentiality.

The motivation for this work is that popular ontologies like GALEN, GO and SNOMED that can be viewed as KBs defined in languages belong to $\mathcal{EL}$ family. In addition, a number

of studies were reported in conjunctive query answering, reasoning and classifications in $\mathcal{ELH}$ and its extensions, see Bienvenu et al. (2013); Delaitre and Kazakov (2009).

In this chapter we extend the work of Tao et al., reported in Tao et al. (2010), to the $\mathcal{ELH}$ language. In addition to the extension, we make several new contributions. First, we study secrecy in the context of assertions as well as general concept inclusions (GCIs). To the best of our knowledge, secrecy-preserving reasoning for GCIs has not been studied before. As a first step in constructing SPQA system, we design two tableau algorithms to compute finite sets $\mathcal{T}^*$ and then $\mathcal{A}^*$, of consequences of the TBox $\mathcal{T} \cup \mathcal{R}^*$ and the KB $\langle \mathcal{A}, \mathcal{T}^*, \mathcal{R}^* \rangle$ respectively, restricted to individuals and concepts that actually occur in the given KB $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ and an extra "auxiliary" set of concepts defined over the signature of Σ. The approach to constructing SPQA system presented in this chapter is quite different from Tao et al. (2010). In Tao et al. (2010), the KB and envelope are expanded with new queries. This makes the subsequent query answering step more and more complicated. In general, the sets of all assertional consequences and GCI consequences of a given $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ may be infinite. By forcing the tableau algorithms to compute the consequences (both assertions and GCIs) of KB restricted to individuals and subconcepts that occur in a given prescribed set, we obtain finite $\mathcal{A}^*$ and $\mathcal{T}^*$ that in fact can be computed efficiently in polynomial time. These sets, once computed, remain fixed and are not modified. The two tableau algorithms are sound and complete under the restrictions stated above, see section 3.3. Since the sets $\mathcal{A}^*$ and $\mathcal{T}^*$ do not contain all the consequences of the KB, in order to answer user queries we have designed recursive algorithms which break the queries into smaller assertions or GCIs all the way until the information in the sets $\mathcal{A}^*$ and $\mathcal{T}^*$ can be used. In effect, we have split the task of query answering into two parts: in the first part we compute all the consequences of Σ restricted to concepts and individuals that occur in Σ, in the second part we use a recursive algorithm to evaluate more complex queries with the base case that has been computed in the first part.

In more detail, starting from the secrecy sets $\mathbb{S}_{\mathcal{A}}$ (of assertions) and $\mathbb{S}_{\mathcal{T}}$ (of GCIs), we compute finite sets of assertions and GCIs, viz., the *envelopes* $\mathbb{E}_{\mathcal{A}} \subseteq \mathcal{A}^*$ of $\mathbb{S}_{\mathcal{A}}$ and $\mathbb{E}_{\mathcal{T}} \subseteq \mathcal{T}^*$ of $\mathbb{S}_{\mathcal{T}}$ respectively. These envelopes are computed by two tableau algorithms based on the idea of inverting the expansion rules of two tableau algorithms listed in Figures 3.1 and 3.2. The

idea behind the envelope concept is that no expression in the envelope can be logically deduced from information outside the envelope. Once such envelopes are computed, the answers to the queries are censored whenever the queries belong to the envelopes. Since, generally, an envelope for a given secrecy set is not unique, the developer can force the algorithm to output a specific envelope from the available choices satisfying the needs of application domain, company policy, social obligations and user preferences.

Next, we discuss query answering procedures which allow us answer queries without revealing secrets. Usually in SPQA framework queries are answered by checking their membership (a) in $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ if the query is an assertion; and (b) in $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$ if the query is a GCI. Since $\mathcal{A}^*$ and $\mathcal{T}^*$ do not contain all the statements entailed by Σ, we need to extend the query answering procedure from just membership checking. Towards that end we designed two recursive algorithms to answer more complicated assertion and GCI queries. To answer an assertion query q, the algorithm first checks if $q \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ in which case the answer is "Yes"; otherwise, the given query is broken into subqueries based on the constructors, and the algorithm is applied recursively on the subqueries, see section 3.5. This query answering procedure runs in polynomial time in the size of the KB and the query q. Similar approach is used to answer GCI queries.

3.2 Syntax and Semantics

A vocabulary of $\mathcal{ELH}$ is a triple $< N_O, N_C, N_R >$ of countably infinite, pairwise disjoint sets. The elements of N_O are called *object* (or *individual) names*, the elements of N_C are called *concept names* and the elements of N_R are called *role names*. The set of $\mathcal{ELH}$ *concepts* is denoted by $\mathcal{C}$ and is defined by the following rules

$$C ::= A \ \mid \ \top \ \mid \ C \sqcap D \ \mid \ \exists r.C$$

where $A \in N_C$, $r \in N_R$, $\top$ denotes the "*top concept*", and $C, D \in \mathcal{C}$. *Assertions* are expressions of the form $C(a)$ or $r(a, b)$, *general concept inclusions (GCIs)* are expressions of the form $C \sqsubseteq D$ and *role inclusions* are expressions of the form $r \sqsubseteq s$ where $C, D \in \mathcal{C}$, $r, s \in N_R$ and $a, b \in N_O$. The semantics of $\mathcal{ELH}$ concepts is specified, as usual, by an *interpretation* $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ where Δ

is the *domain* of the interpretation, and $\cdot^{\mathcal{I}}$ is an *interpretation function* mapping each $a \in N_O$ to an element $a^{\mathcal{I}} \in \Delta$, each $A \in N_C$ to a subset $A^{\mathcal{I}} \subseteq \Delta$, and each $r \in N_R$ to a binary relation $r^{\mathcal{I}} \subseteq \Delta \times \Delta$. The interpretation function $\cdot^{\mathcal{I}}$ is extended inductively to all $\mathcal{ELH}$ concepts in the usual manner:

$$\top^{\mathcal{I}} = \Delta; \quad (C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}};$$
$$(\exists r.C)^{\mathcal{I}} = \{d \in \Delta \mid \exists e \in C^{\mathcal{I}} : (d, e) \in r^{\mathcal{I}}\}.$$

An *Abox* $\mathcal{A}$ is a finite, non-empty set of assertions. A *TBox* $\mathcal{T}$ is a finite set of GCIs and an *RBox* $\mathcal{R}$ is a finite set of role inclusions. An $\mathcal{ELH}$ KB is a triple $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ where $\mathcal{A}$ is an ABox, $\mathcal{T}$ is a TBox and $\mathcal{R}$ is an RBox. Let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be an interpretation, $C, D \in \mathcal{C}$, $r, s \in N_R$ and $a, b \in N_O$. We say that $\mathcal{I}$ *satisfies* $C(a)$, $r(a, b)$, $C \sqsubseteq D$ or $r \sqsubseteq s$, notation $\mathcal{I} \models C(a)$, $\mathcal{I} \models r(a, b)$, $\mathcal{I} \models C \sqsubseteq D$ or $\mathcal{I} \models r \sqsubseteq s$ if, respectively, $a^{\mathcal{I}} \in C^{\mathcal{I}}$, $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ or $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$. $\mathcal{I}$ is a *model* of Σ, notation $\mathcal{I} \models \Sigma$, if $\mathcal{I}$ satisfies all the assertions in $\mathcal{A}$, all the GCIs in $\mathcal{T}$ and all the role inclusions in $\mathcal{R}$. Let α be an assertion, a GCI or a role inclusion. We say that Σ *entails* α, notation $\Sigma \models \alpha$, if all models of Σ satisfy α.

3.3 Computation of $\mathcal{A}^*$ and $\mathcal{T}^*$

Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be an $\mathcal{ELH}$ KB. In this section, we give two tableau algorithms that compute $\mathcal{A}^*$, a set of assertional consequence of Σ, and $\mathcal{T}^*$ a set of GCI consequences of Σ, both restricted to concepts that occur in Σ. We assume that all RBoxes are acyclic. Before computing $\mathcal{T}^*$ and $\mathcal{A}^*$, we compute $\mathcal{R}^* = \mathcal{R}^+ \cup \mathcal{R}^\circ$, where $\mathcal{R}^+$ is the transitive closure of $\mathcal{R}$ with respect to role inclusion and $\mathcal{R}^\circ = \{r \sqsubseteq r \mid r \text{ occurs in } \Sigma\}$. As an example, consider a KB $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ where ABox $\mathcal{A} = \{A(a), \exists m.B(c)\}$, TBox $\mathcal{T} = \{A \sqsubseteq \exists n.D\}$ and RBox $\mathcal{R} = \{r \sqsubseteq s,\ p \sqsubseteq q,\ u \sqsubseteq v,\ s \sqsubseteq u\}$. Then, $\mathcal{R}^* = \mathcal{R} \cup \{s \sqsubseteq v,\ r \sqsubseteq u,\ r \sqsubseteq v\} \cup \{m \sqsubseteq m, n \sqsubseteq n, r \sqsubseteq r, s \sqsubseteq s, p \sqsubseteq p, q \sqsubseteq q, u \sqsubseteq u, v \sqsubseteq v\}$. $\mathcal{R}^*$ is easily computed in polynomial time and we omit the details.

Computation of $\mathcal{T}^*$: Denote by N_Σ the set of all concept names and role names occurring in Σ and let $\mathbb{S}$ be a finite set of concepts over the symbol set N_Σ. Let $\mathcal{C}_{\Sigma,\mathbb{S}}$ be the set of all subconcepts of concepts that occur in either $\mathbb{S}$ or Σ. Given Σ and $\mathcal{C}_{\Sigma,\mathbb{S}}$, we describe a

$$
\begin{array}{ll}
\mathrm{T}_{\sqsubseteq} - \text{rule}: & \text{if } C \sqsubseteq D \in \mathcal{T}^*,\ D \sqsubseteq E \in \mathcal{T} \text{ and } C \sqsubseteq E \ \notin \mathcal{T}^*, \\
 & \text{then } \mathcal{T}^* := \mathcal{T}^* \cup \{C \sqsubseteq E\}; \\
\mathrm{T}_{\sqcap}^{-} - \text{rule}: & \text{if } C \sqsubseteq D \sqcap E \in \mathcal{T}^*, \text{ and } C \sqsubseteq D \notin \mathcal{T}^* \text{ or } C \sqsubseteq E \ \notin \mathcal{T}^*, \\
 & \text{then } \mathcal{T}^* := \mathcal{T}^* \cup \{C \sqsubseteq D, C \sqsubseteq E\}; \\
\mathrm{T}_{\sqcap}^{+} - \text{rule}: & \text{if } C \sqsubseteq D,\ C \sqsubseteq E \in \mathcal{T}^*,\ D \sqcap E \in \mathcal{C}_{\Sigma,\mathbb{S}} \text{ and } C \sqsubseteq D \sqcap E \ \notin \mathcal{T}^*, \\
 & \text{then } \mathcal{T}^* := \mathcal{T}^* \cup \{C \sqsubseteq D \sqcap E\}; \\
\mathrm{T}_{H}^{+} - \text{rule}: & \text{if } C \sqsubseteq \exists r.D,\ D \sqsubseteq E \in \mathcal{T}^*,\ r \sqsubseteq s \in \mathcal{R}^*,\ \exists s.E \in \mathcal{C}_{\Sigma,\mathbb{S}} \text{ and } C \sqsubseteq \exists s.E \ \notin \mathcal{T}^*, \\
 & \text{then } \mathcal{T}^* := \mathcal{T}^* \cup \{C \sqsubseteq \exists s.E\}.
\end{array}
$$

Figure 3.1 TBox Tableau expansion rules

procedure that computes $\mathcal{T}^*$, a set of GCI consequences of the given KB Σ (restricted to concepts in $\mathcal{C}_{\Sigma,\mathbb{S}}$). That is, $\mathcal{T}^* = \{C \sqsubseteq D \mid C, D \in \mathcal{C}_{\Sigma,\mathbb{S}} \text{ and } \Sigma \models C \sqsubseteq D\}$. This procedure is similar to the calculus presented in Kazakov et al. (2014) (designed for $\mathcal{EL}^+$).

Let $AX_{\mathcal{T}} = \{C \sqsubseteq C, C \sqsubseteq \top, \top \sqsubseteq \top \mid C \in \mathcal{C}_{\Sigma,\mathbb{S}}\}$. $\mathcal{T}^*$ is initialized as $AX_{\mathcal{T}}$ and then expanded by exhaustively applying expansion rules listed in Figure 3.1. The $\mathrm{T}_{\sqsubseteq}$-rule derives a GCI based on transitivity of subsumption. $\mathrm{T}_{\sqcap}^{-}$-rule derives new GCIs by decomposing conjunction concepts into its two conjuncts. The $\mathrm{T}_{\sqcap}^{+}$-rule is just the "opposite" of the $\mathrm{T}_{\sqcap}^{-}$-rule. Finally, T_{H}^{+}-rule derives GCIs based on concept and role inclusions.

A TBox is *completed* if no expansion rule in Figure 3.1 is applicable to it. We denote by $\Lambda_{\mathcal{T}}$ the *algorithm* which, given Σ, $\mathcal{C}_{\Sigma,\mathbb{S}}$ and $\mathcal{R}^*$, non-deterministically applies expansion rules in Figure 3.1 until no further applications are possible. Since $\Lambda_{\mathcal{T}}$ has been restricted to derive GCIs whose left and right hand side concept expressions occur in $\mathcal{C}_{\Sigma,\mathbb{S}}$, the size of the $\mathcal{T}^*$ is at most a polynomial in the size of its input. Hence, the running time of $\Lambda_{\mathcal{T}}$ is polynomial in $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathbb{S}} \mid$. The correctness of $\Lambda_{\mathcal{T}}$ can be shown by proving soundness and completeness of $\Lambda_{\mathcal{T}}$. The soundness proof is obvious.

Example 3.3.1. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be a $\mathcal{ELH}$ KB, where $\mathcal{A} = \{C(a), r(b,a), \exists u.A(d)\}$, $\mathcal{T} = \{A \sqsubseteq B, C \sqsubseteq D \sqcap E, F \sqsubseteq \exists u.B\}$ and $\mathcal{R} = \{u \sqsubseteq v\}$. Then, $\mathcal{R}^* = \{r \sqsubseteq r, u \sqsubseteq u, v \sqsubseteq v, u \sqsubseteq v\}$. Thus, applying rules in Figure 3.1 to $\mathcal{T}$, we get $\{\top \sqsubseteq \top, A \sqsubseteq \top, C \sqsubseteq C, \exists u.A \sqsubseteq \exists u.A, \exists u.A \sqsubseteq \exists u.B,\ C \sqsubseteq D, C \sqsubseteq D \sqcap E\} \subseteq \mathcal{T}^*$.* $\square$

To prove the completeness of $\Lambda_{\mathcal{T}}$, we define the *canonical interpretation* $\mathcal{J} = \langle \Delta, \cdot^{\mathcal{J}} \rangle$ for a completed TBox $\mathcal{T}^*$ and an RBox $\mathcal{R}^*$ as follows:

$$\Delta = \{w_C \mid C \in \mathcal{C}_{\Sigma,\mathbb{S}}\};$$

$$\top^{\mathcal{J}} = \Delta;$$

$$\text{for } A \in N_C,\ A^{\mathcal{J}} = \{w_C \mid C \sqsubseteq A \in \mathcal{T}^*\};$$

$$\text{for } r \in N_R,\ r^{\mathcal{J}} = \{(w_C, w_D) \mid C \sqsubseteq \exists r.D \in \mathcal{T}^*\} \cup \bigcup_{u \sqsubseteq r \in \mathcal{R}^*} u^{\mathcal{J}}.$$

The interpretation function $\cdot^{\mathcal{J}}$ is extended to concept expressions as usual. To prove that $\mathcal{J}$ is a model of $\mathcal{T}^*$, we need the following definition and technical lemma.

Definition 3.3.1. *Let $\mathcal{J}$ be the canonical interpretation and u a role name that occurs in Σ. u is said to be minimal with respect to $(w_G, w_H) \in \Delta \times \Delta$ if*

1) $(w_G, w_H) \in u^{\mathcal{J}}$ and

2) there is no v that occurs in $\mathcal{R}$ such that $v \neq u$, $(w_G, w_H) \in v^{\mathcal{J}}$ and $v \sqsubseteq u \in \mathcal{R}^*$.

Lemma 3.3.1. *Let B, $C \in \mathcal{C}_{\Sigma,\mathbb{S}}$. Then,*

(a) $w_C \in C^{\mathcal{J}}$.

(b) $w_C \in B^{\mathcal{J}}$ if and only if $C \sqsubseteq B \in \mathcal{T}^*$.

Proof. *(a) By induction on the structure of C.*

- *$C = A \in N_C$ or $C = \top$, the claim follows from the definition of $\mathcal{J}$.*

- *$C = D \sqcap E$. Then, $D \sqcap E \sqsubseteq D \sqcap E \in \mathcal{T}^*$ and by the $T_{\sqcap}^{\sqsubset}$-rule, we have $D \sqcap E \sqsubseteq D, D \sqcap E \sqsubseteq E \in \mathcal{T}^*$, whence $w_{D \sqcap E} \in D^{\mathcal{J}}$ and $w_{D \sqcap E} \in E^{\mathcal{J}}$, by inductive hypothesis. By the semantics of $\sqcap$, $w_{D \sqcap E} \in D^{\mathcal{J}} \cap E^{\mathcal{J}} = (D \sqcap E)^{\mathcal{J}}$.*

- *$C = \exists r.D$. Then, $\exists r.D \sqsubseteq \exists r.D \in \mathcal{T}^*$ and by the definition of $\mathcal{J}$, $(w_{\exists r.D}, w_D) \in r^{\mathcal{J}}$; also, by the inductive hypothesis, $w_D \in D^{\mathcal{J}}$. By the semantics of $\exists$, $w_{\exists r.D} \in (\exists r.D)^{\mathcal{J}}$.*

(b) ($\Leftarrow$) By induction on the structure of B.

- $B \in N_C$. *Then,* $C \sqsubseteq B \in \mathcal{T}^*$ *whence* $w_C \in B^{\mathcal{J}}$, *by the definition of* $\mathcal{J}$.

- $B = \top$, *the claim follows from the definition of* $\mathcal{J}$.

- $B = D \sqcap E$. *Then,* $C \sqsubseteq D \sqcap E \in \mathcal{T}^*$. *By* $T^{-}_{\sqcap}$*-rule,* $C \sqsubseteq D$, $C \sqsubseteq E \in \mathcal{T}^*$ *implies* $w_C \in D^{\mathcal{J}}$ *and* $w_C \in E^{\mathcal{J}}$, *and by the inductive hypothesis whence* $w_C \in (D \sqcap E)^{\mathcal{J}} = B^{\mathcal{J}}$, *by the semantics of* $\sqcap$.

- $B = \exists r.D$. *We assume,* $C \sqsubseteq \exists r.D \in \mathcal{T}^*$. *Since* $C,\ D \in \mathcal{C}_{\Sigma,\mathbb{S}}$, *we have* w_C, $w_D \in \Delta$. *By the definition of* $\mathcal{J}$, $(w_C, w_D) \in r^{\mathcal{J}}$. *By part (a),* $w_D \in D^{\mathcal{J}}$ *hence* $w_C \in (\exists r.D)^{\mathcal{J}} = B^{\mathcal{J}}$, *by the semantics of* $\exists$.

($\Rightarrow$) *By induction on the structure of* B.

- *When* $B \in N_C$, *the claim follows from the definition of* $\mathcal{J}$.

- $B = \top$, *the claim follows from the definition of* $AX_{\mathcal{T}}$.

- $B = D \sqcap E$. *Then,* $w_C \in (D \sqcap E)^{\mathcal{J}} \Rightarrow w_C \in D^{\mathcal{J}}$ *and* $w_C \in E^{\mathcal{J}} \Rightarrow\ C \sqsubseteq D, C \sqsubseteq E \in \mathcal{T}^*$, *by inductive hypothesis. Since* $D \sqcap E$ *occurs in* $\mathcal{C}_{\Sigma,\mathbb{S}}$, *by the* $T^{+}_{\sqcap}$*-rule, we have* $C \sqsubseteq D \sqcap E = B \in \mathcal{T}^*$.

- $B = \exists r.D$. *Then,* $w_C \in (\exists r.D)^{\mathcal{J}} \Rightarrow$ *there is an element* $w_E \in \Delta$ *such that* $(w_C, w_E) \in r^{\mathcal{J}}$, $w_E \in D^{\mathcal{J}}$. *By inductive hypothesis,* $E \sqsubseteq D \in \mathcal{T}^*$. *Now, we have two subcases depending on a "manner" in which* (w_C, w_E) *entered* $r^{\mathcal{J}}$.

  - *If* r *is minimal with respect to* (w_C, w_E), *then, by the definition of* $\mathcal{J}$ *and Definition 3.3.1,* $C \sqsubseteq \exists r.E \in \mathcal{T}^*$. *Since* $r \sqsubseteq r \in \mathcal{R}^*$, *by the* T^{+}_{H}*-rule, we have* $C \sqsubseteq \exists r.D \in \mathcal{T}^*$. *Hence,* $C \sqsubseteq B \in \mathcal{T}^*$.

  - *If* r *is not minimal, then* $(w_C, w_E) \in u^{\mathcal{J}}$, $u \neq r$ *and* $u \sqsubseteq r \in \mathcal{R}^*$ *for some* u *that occurs in* $\mathcal{R}$. *If* u *is minimal with respect to* (w_C, w_E), *then by previous case* $C \sqsubseteq \exists u.E \in \mathcal{T}^*$ *and by the* T^{+}_{H}*-rule, we have* $C \sqsubseteq \exists r.D \in \mathcal{T}^*$. *Hence,* $C \sqsubseteq B \in \mathcal{T}^*$. *If* u *is not minimal with respect to* (w_C, w_E), *since RBox* $\mathcal{R}$ *is acyclic, there exists a chain* $v \sqsubseteq v_1 \sqsubseteq v_2 ..... \sqsubseteq v_k \sqsubseteq u$ *in* $\mathcal{R}$ *such that* v *is minimal with respect to* (w_C, w_E).

Since $\mathcal{R}^*$ is the transitive closure of $\mathcal{R}$, $v \sqsubseteq r \in \mathcal{R}^*$. Again by the previous case, $C \sqsubseteq \exists v.E \in \mathcal{T}^*$. By T_H^+-rule, we have $C \sqsubseteq \exists r.D \in \mathcal{T}^*$. Hence, $C \sqsubseteq B \in \mathcal{T}^*$. □

The following lemma claims that $\mathcal{J}$ satisfies $\mathcal{T}^*$ and $\mathcal{R}^*$. The proof is a consequence of Lemma 3.3.1

Lemma 3.3.2. $\mathcal{J} \models \mathcal{T}^* \cup \mathcal{R}^*$.

The completeness of $\Lambda_{\mathcal{T}}$ now follows by an easy argument.

Theorem 3.3.1. *Let Σ be a $\mathcal{ELH}$ KB and let $\mathcal{T}^*$ be the completed TBox. For any C, D $\in \mathcal{C}_{\Sigma,\mathbb{S}}$, if $\Sigma \models C \sqsubseteq D$ then $C \sqsubseteq D \in \mathcal{T}^*$.*

Proof. *Suppose $C \sqsubseteq D \notin \mathcal{T}^*$, i.e., by part (b) of Lemma 3.3.1, $w_C \notin D^{\mathcal{J}}$. On the other hand by part (a) of Lemma 3.3.1, $w_C \in C^{\mathcal{J}}$ and this implies that $\mathcal{J} \not\models C \sqsubseteq D$. Since by Lemma 3.3.2, $\mathcal{J} \models \mathcal{T}^*$, and since $\mathcal{T} \subseteq \mathcal{T}^*$, we obtain $\Sigma \not\models C \sqsubseteq D$.* □

Computation of $\mathcal{A}^*$: Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be an $\mathcal{ELH}$ KB, $\mathcal{R}^*$ be defined as at the beginning of this section and $\mathcal{T}^*$ be the completed TBox as computed previously. Also, let $\mathcal{O}_\Sigma$ be the set of individual names that occur in Σ and define $AX_{\mathcal{A}} = \{\top(a) \mid a \in \mathcal{O}_\Sigma\}$.

We outline the procedure that computes $\mathcal{A}^*$, the set of assertional consequences of Σ^* where $\Sigma^* = \langle \mathcal{A}, \mathcal{T}^*, \mathcal{R}^* \rangle$, restricted to the concepts and role names that occur in $\mathcal{C}_{\Sigma,\mathbb{S}}$ and Σ respectively. That is

$$\mathcal{A}^* = \{C(a) \mid C \in \mathcal{C}_{\Sigma,\mathbb{S}} \text{ and } \Sigma^* \models C(a)\} \cup \ \{r(a,b) \mid r \text{ occurs in } \Sigma \text{ and } \Sigma^* \models r(a,b)\}.$$

$\mathcal{A}^*$ is initialized as $\mathcal{A} \cup AX_{\mathcal{A}}$ and is expanded by exhaustively applying rules listed in Figure 3.2. $\mathrm{A}_{\sqcap}^{-}$-rule decomposes conjunctions, and the $\mathrm{A}_{\sqsubseteq}$-rule derives assertions based on the GCIs present in $\mathcal{T}^*$. To build new concept assertions whose concept expressions already occur in $\mathcal{C}_{\Sigma,\mathbb{S}}$, we use the $\mathrm{A}_{\sqcap}^{+}$ and $\mathrm{A}_{\exists}^{+}$-rules. Similarly, the $\mathrm{A}_{\exists H}^{+}$-rule derives concept assertions based on role inclusions. It is important to note that this procedure does not introduce any fresh individual names into $\mathcal{A}^*$. Thus some assertions of the form $\exists r.C(a)$ may not have "syntactic witnesses". Finally, the A_H-rule derives role assertions based on role inclusions.

$$
\begin{array}{ll}
\mathrm{A}_{\sqcap}^{-} - \text{rule}: & \text{if } C \sqcap D(a) \in \mathcal{A}^*, \text{ and } C(a) \notin \mathcal{A}^* \text{ or } D(a) \notin \mathcal{A}^*, \\
 & \text{then } \mathcal{A}^* := \mathcal{A}^* \cup \{C(a), D(a)\}; \\
\mathrm{A}_{\sqcap}^{+} - \text{rule}: & \text{if } C(a),\ D(a) \in \mathcal{A}^*,\ C \sqcap D \in \mathcal{C}_{\Sigma,\mathbb{S}} \text{ and } C \sqcap D(a) \notin \mathcal{A}^*, \\
 & \text{then } \mathcal{A}^* := \mathcal{A}^* \cup \{C \sqcap D(a)\}; \\
\mathrm{A}_{\exists}^{+} - \text{rule}: & \text{if } r(a,b),\ C(b) \in \mathcal{A}^*,\ \exists r.C \in \mathcal{C}_{\Sigma,\mathbb{S}} \text{ and } \exists r.C(a) \notin \mathcal{A}^*, \\
 & \text{then } \mathcal{A}^* := \mathcal{A}^* \cup \{\exists r.C(a)\}; \\
\mathrm{A}_{\sqsubseteq} - \text{rule}: & \text{if } C(a) \in \mathcal{A}^*,\ C \sqsubseteq D \in \mathcal{T}^*, \text{and } D(a) \notin \mathcal{A}^*, \\
 & \text{then } \mathcal{A}^* := \mathcal{A}^* \cup \{D(a)\}; \\
\mathrm{A}_{\exists H}^{+} - \text{rule}: & \text{if } \exists r.C(a) \in \mathcal{A}^*,\ r \sqsubseteq s \in \mathcal{R}^*,\ C \sqsubseteq D \in \mathcal{T}^*,\ \exists s.D \in \mathcal{C}_{\Sigma,\mathbb{S}} \text{ and } \exists s.D(a) \notin \mathcal{A}^*, \\
 & \text{then } \mathcal{A}^* := \mathcal{A}^* \cup \{\exists s.D(a)\}; \\
\mathrm{A}_{H} - \text{rule}: & \text{if } r(a,b) \in \mathcal{A}^*,\ r \sqsubseteq s \in \mathcal{R}^*, \text{ and } s(a,b) \notin \mathcal{A}^*, \\
 & \text{then } \mathcal{A}^* := \mathcal{A}^* \cup \{s(a,b)\}.
\end{array}
$$

Figure 3.2 ABox Tableau expansion rules.

An ABox is *completed* if no expansion rule in Figure 3.2 is applicable to it. We denote by $\Lambda_{\mathcal{A}}$ the *algorithm* which, given $\mathcal{A}$, $\mathcal{R}^*$, $\mathcal{T}^*$ and $\mathcal{C}_{\Sigma,\mathbb{S}}$, non-deterministic-ally applies expansion rules in Figure 3.2 until no further applications are possible. Since $\Lambda_{\mathcal{A}}$ derives only assertions involving concept expressions that occur in $\mathcal{C}_{\Sigma,\mathbb{S}}$, it is easy to see that the running time of $\Lambda_{\mathcal{A}}$ is polynomial in $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathbb{S}} \mid$.

Example 3.3.2. *(Example 3.3.1 cont.) Recall that $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be a $\mathcal{ELH}$ be the given KB, $\mathcal{R}^*$ the computed RBox and $\mathcal{T}^*$ the completed TBox. Then, by applying rules in Figure 3.2 to $\mathcal{A}$ and using $\mathcal{T}^*$ and $\mathcal{R}^*$ we get,*

$\mathcal{A}^* = \{\top(a), \top(b), \top(d), \exists u.A(d), \exists u.B(d), C(a),$
$r(b,a), D(a), E(a), D \sqcap E(a)\}.$ $\square$

The correctness of $\Lambda_{\mathcal{A}}$ can be shown by proving its soundness and completeness. The soundness is obvious. To prove the completeness of $\Lambda_{\mathcal{A}}$, we first define the canonical interpretation $\mathcal{K} = \langle \Delta, \cdot^{\mathcal{K}} \rangle$ for a completed ABox $\mathcal{A}^*$. The definition of $\mathcal{K}$ is similar to the definition of canon-

ical model $\mathcal{I}_{\mathcal{K}}$ presented in Lutz et al. (2008). Define the witness set, $\mathcal{W} = \{w_C \mid C \in \mathcal{C}_{\Sigma,\mathbb{S}}\}$.

$$\Delta = \mathcal{O}_\Sigma \cup \mathcal{W};$$

$$a^{\mathcal{K}} = a, \text{where } a \in \mathcal{O}_\Sigma;$$

$$\top^{\mathcal{K}} = \Delta;$$

for each $A \in N_C$,

$$A^{\mathcal{K}} = \{a \in \mathcal{O}_\Sigma \mid A(a) \in \mathcal{A}^*\} \cup \ \{w_C \in \mathcal{W} \mid C \sqsubseteq A \in \mathcal{T}^*\}; \text{ for each } r \in N_R,$$

$$r^{\mathcal{K}} = \{(a,b) \in \mathcal{O}_\Sigma \times \mathcal{O}_\Sigma \mid r(a,b) \in \mathcal{A}^*\} \cup \ \{(a, w_C) \in \mathcal{O}_\Sigma \times \mathcal{W} \mid \exists r.C(a) \in \mathcal{A}^*\}$$
$$\cup \{(w_C, w_D) \in \mathcal{W} \times \mathcal{W} \mid C \sqsubseteq \exists r.D \in \mathcal{T}^*\} \cup \ \bigcup\{u^{\mathcal{K}} \mid u \sqsubseteq r \in \mathcal{R}^*\}.$$

$\mathcal{K}$ is extended to compound concepts in the usual way. We argue that $\mathcal{K}$ is a model of $\mathcal{A}^*$, $\mathcal{T}^*$ and $\mathcal{R}^*$.

Lemma 3.3.3. *Let $a, b \in \mathcal{O}_\Sigma$ and suppose that the role name r occurs in Σ. If $(a,b) \in r^{\mathcal{K}}$, then $r(a,b) \in \mathcal{A}^*$.*

Proof. *Assume the hypotheses. We prove the claim by induction on how $r(a,b)$ has been generated by $\Lambda_{\mathcal{A}}$. The base case, when $r(a,b) \in \mathcal{A}$, is trivial. Let $(a,b) \in u^{\mathcal{K}}$ with $u \sqsubseteq r \in \mathcal{R}^*$. Then by induction hypothesis, $u(a,b) \in \mathcal{A}^*$ and by the A_H-rule, we have $r(a,b) \in \mathcal{A}^*$.* $\square$

We state the following lemma whose proof is similar to the proof of Lemma 3.3.1.

Lemma 3.3.4. *Let B, $C \in \mathcal{C}_{\Sigma,\mathbb{S}}$. Then,*

(a) $w_C \in C^{\mathcal{K}}$.

(b) $w_C \in B^{\mathcal{K}}$ *if and only if* $C \sqsubseteq B \in \mathcal{T}^*$.

The following definition is similar to Definition 3.3.1, but is based on the canonical interpretation of the ABox $\mathcal{A}^*$.

Definition 3.3.2. *Let $\mathcal{K}$ be the canonical interpretation, and u a role name that occurs in Σ. u is said to be minimal with respect to (a,b) if*

1) $(a,b) \in u^{\mathcal{K}}$ *and*

2) there is no role name, v that occurs in $\mathcal{R}$ such that $v \neq u$, $(a,b) \in v^{\mathcal{K}}$ and $v \sqsubseteq u \in \mathcal{R}^*$.

Lemma 3.3.5. *Let $a \in \mathcal{O}_\Sigma$ and $B \in \mathcal{C}_{\Sigma,\mathbb{S}}$. If $a \in B^{\mathcal{K}}$, then $B(a) \in \mathcal{A}^*$.*

Proof. *By induction on the structure of B.*

- *When $B \in N_C$, the claim follows directly from the definition of $\mathcal{K}$.*
- *When $B = \top$, the claim follows from the definition of $AX_{\mathcal{A}}$.*
- *$B = C \sqcap D$. Then, $a \in (C \sqcap D)^{\mathcal{K}} \Rightarrow a \in C^{\mathcal{K}}$ and $a \in D^{\mathcal{K}} \Rightarrow C(a), D(a) \in \mathcal{A}^*$, by inductive hypothesis. Since $C \sqcap D$ occurs in $\mathcal{C}_{\Sigma,\mathbb{S}}$, by the $A^+_\sqcap$-rule, we have $C \sqcap D(a) = B(a) \in \mathcal{A}^*$.*
- *$B = \exists r.C$. Then, $a \in (\exists r.C)^{\mathcal{K}}$ implies that there is an element $b \in \Delta$ such that $(a,b) \in r^{\mathcal{K}}$ and $b \in C^{\mathcal{K}}$. There are two cases.*
  - *$b \in \mathcal{O}_\Sigma$. Since r occurs in Σ and C occurs in $\mathcal{C}_{\Sigma,\mathbb{S}}$, by Lemma 3.3.3, we have $r(a,b) \in \mathcal{A}^*$ and by the inductive hypothesis, $C(b) \in \mathcal{A}^*$. Since $\exists r.C$ occurs in $\mathcal{C}_{\Sigma,\mathbb{S}}$, by the $A^+_\exists$-rule, we have $\exists r.C(a) = B(a) \in \mathcal{A}^*$.*
  - *$b = w_D \in \mathcal{W}$ for some $D \in \mathcal{C}_{\Sigma,\mathbb{S}}$. Then, we have $(a, w_D) \in r^{\mathcal{K}}$ and $w_D \in C^{\mathcal{K}}$. By part (b) of Lemma 3.3.1, $D \sqsubseteq C \in \mathcal{T}^*$. Now, we have two subcases depending on a manner in which (a, w_D) entered $r^{\mathcal{K}}$.*
    - *If r is minimal with respect to (a, w_D), then, by the definition of $\mathcal{K}$ and Definition 3.3.2, $\exists r.D(a) \in \mathcal{A}^*$. Since $r \sqsubseteq r \in \mathcal{R}^*$, by the $A^+_{\exists H}$-rule, we have $\exists r.C(a) \in \mathcal{A}^*$, i.e., $B(a) \in \mathcal{A}^*$.*
    - *If r is not minimal, then $(a, w_D) \in u^{\mathcal{K}}$, $u \neq r$ and $u \sqsubseteq r \in \mathcal{R}^*$ for some u that occurs in $\mathcal{R}$. If u is minimal with respect to (a, w_D), then by previous case $\exists u.D(a) \in \mathcal{A}^*$. By $A^+_{\exists H}$-rule, we have $\exists r.C(a) \in \mathcal{A}^*$. Hence, $B(a) \in \mathcal{A}^*$. If u is not minimal with respect to (a, w_D), since RBox $\mathcal{R}$ is acyclic, there exists a chain $v \sqsubseteq v_1 \sqsubseteq v_2 ..... \sqsubseteq v_k \sqsubseteq u$ in $\mathcal{R}$ such that v is minimal with respect to (a, w_E). Since $\mathcal{R}^*$ is the transitive closure of $\mathcal{R}$, $v \sqsubseteq r \in \mathcal{R}^*$. Again by the previous case, $\exists v.D(a) \in \mathcal{A}^*$. By $A^+_{\exists H}$-rule, we have $\exists r.C(a) \in \mathcal{A}^*$, i.e., $B(a) \in \mathcal{A}^*$.* $\square$

Lemma 3.3.6. *If $B(a) \in \mathcal{A}^*$, then $a \in B^{\mathcal{K}}$.*

Proof. *Again we use induction on the structure of B.*

- *$B \in N_C$. Then, $B(a) \in \mathcal{A}^* \Rightarrow a \in B^{\mathcal{K}}$, by the definition of $\mathcal{K}$.*
- *$B = \top$. The claim follows from the definition of $\mathcal{K}$.*
- *$B = C \sqcap D$. Then, $C \sqcap D(a) \in \mathcal{A}^*$. By $A_{\sqcap}^{-}$-rule, we have $C(a), D(a) \in \mathcal{A}^* \Rightarrow a \in C^{\mathcal{K}}$ and $a \in D^{\mathcal{K}} \Rightarrow a \in (C \sqcap D)^{\mathcal{K}}$, by inductive hypothesis.*
- *$B = \exists r.D$. Then, $\exists r.D(a) \in \mathcal{A}^*$. By the definition of $\mathcal{K}$, $(a, w_D) \in r^{\mathcal{K}}$. By Lemma 3.3.1, $w_D \in D^{\mathcal{K}}$. Hence, by the semantics of $\exists$, $a \in (\exists r.D)^{\mathcal{K}} = B^{\mathcal{K}}$.* $\square$

In the following we prove that $\mathcal{K}$ satisfies $\mathcal{A}^*$, $\mathcal{T}^*$ and $\mathcal{R}^*$.

Lemma 3.3.7. $\mathcal{K} \models \mathcal{A}^* \cup \mathcal{T}^* \cup \mathcal{R}^*$.

Proof. *It follows immediately from the definition of $\mathcal{K}$ that $\mathcal{K} \models \mathcal{R}^*$. Next, we show that $\mathcal{K}$ satisfies $\mathcal{A}^*$. $C(a) \in \mathcal{A}^*$; then, by Lemma 3.3.6, $a \in C^{\mathcal{K}}$, i.e., $\mathcal{K} \models C(a)$. For $r(a, b) \in \mathcal{A}^*$, $\mathcal{K} \models r(a, b)$, by the definition of $\mathcal{K}$. Hence $\mathcal{K} \models \mathcal{A}^*$.*

Now, we show that $\mathcal{K}$ satisfies $\mathcal{T}^*$. Let $F \sqsubseteq G \in \mathcal{T}^*$ and $a \in F^{\mathcal{K}}$. We have two cases.

- *$a \in \mathcal{O}_\Sigma$. Then, by Lemma 3.3.5, $F(a) \in \mathcal{A}^*$. Since $\mathcal{A}^*$ is completed, by the $A_{\sqsubseteq}$-rule, we get $G(a) \in \mathcal{A}^*$. By Lemma 3.3.6, $a \in G^{\mathcal{K}}$. Hence, $\mathcal{K} \models F \sqsubseteq G$.*
- *$a = w_C \in \mathcal{W}$ for some $C \in \mathcal{C}_{\Sigma,\mathbb{S}}$. This implies, by the definition of $\mathcal{K}$, that $C \sqsubseteq F \in \mathcal{T}^*$. Since $\mathcal{T}^*$ is completed, we have $C \sqsubseteq G \in \mathcal{T}^*$. Again by the definition of $\mathcal{K}$, $a \in G^{\mathcal{K}}$ which implies $\mathcal{K} \models F \sqsubseteq G$.* $\square$

We are ready to prove the completeness of $\Lambda_{\mathcal{A}}$.

Theorem 3.3.2. *Let $\Sigma^* = \langle \mathcal{A}, \mathcal{T}^*, \mathcal{R}^* \rangle$ be a $\mathcal{ELH}$ KB as defined in the beginning of this subsection and $\mathcal{A}^*$ the completed ABox. Suppose that $B \in \mathcal{C}_{\Sigma,\mathbb{S}}$ and r occurs in Σ. Then, for any $a, b \in \mathcal{O}_\Sigma$,*

- *$\Sigma^* \models B(a) \;\Rightarrow\; B(a) \in \mathcal{A}^*$.*

- $\Sigma^* \models r(a,b) \;\Rightarrow\; r(a,b) \in \mathcal{A}^*$.

Proof. *Since $\mathcal{A} \subseteq \mathcal{A}^*$, by Lemma 3.3.7, we have $\mathcal{K} \models \Sigma^*$. We show that $\mathcal{K} \not\models B(a)$ and $\mathcal{K} \not\models r(a,b)$. Assume that $B(a) \notin \mathcal{A}^*$. Then, $a \notin B^{\mathcal{K}}$ by Lemma 3.3.5 and hence $\mathcal{K} \not\models B(a)$. Now, assume that $r(a,b) \notin \mathcal{A}^*$. Then, $(a,b) \notin r^{\mathcal{K}}$ by Lemma 3.3.3 and hence $\mathcal{K} \not\models r(a,b)$.* $\square$

3.4 Secrecy-Preserving Reasoning

Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be an $\mathcal{ELH}$ KB. Also let $\mathbb{S}_{\mathcal{A}} \subseteq \mathcal{A}^* \setminus AX_{\mathcal{A}}$ and $\mathbb{S}_{\mathcal{T}} \subseteq \mathcal{T}^* \setminus AX_{\mathcal{T}}$ be the "secrecy sets". Given Σ, $\mathbb{S}_{\mathcal{A}}$ and $\mathbb{S}_{\mathcal{T}}$, the objective is to answer assertion or GCI queries while preserving secrecy. Our approach is to compute two sets $\mathbb{E}_{\mathcal{A}}$ and $\mathbb{E}_{\mathcal{T}}$, where $\mathbb{S}_{\mathcal{A}} \subseteq \mathbb{E}_{\mathcal{A}} \subseteq \mathcal{A}^* \setminus AX_{\mathcal{A}}$ and $\mathbb{S}_{\mathcal{T}} \subseteq \mathbb{E}_{\mathcal{T}} \subseteq \mathcal{T}^* \setminus AX_{\mathcal{T}}$, called the *secrecy envelopes* for $\mathbb{S}_{\mathcal{A}}$ and $\mathbb{S}_{\mathcal{T}}$ respectively, so that protecting $\mathbb{E}_{\mathcal{A}}$ and $\mathbb{E}_{\mathcal{T}}$, the querying agent cannot logically infer any assertion in $\mathbb{S}_{\mathcal{A}}$ and any GCI in $\mathbb{S}_{\mathcal{T}}$, see Tao et al. (2010) where the DL language is just $\mathcal{EL}$ and secrecy is restricted to membership assertions. Similarly, Tao et al. (2014) presents a general framework for secrecy preserving reasoning.

The role of OWA in answering the queries is the following: When answering a query with "Unknown", the querying agent should not be able to distinguish between the case that the answer to the query is truly unknown to the KB reasoner and the case that the answer is being protected for reasons of secrecy. We envision a situation in which once the ABox $\mathcal{A}^*$ and TBox $\mathcal{T}^*$ are computed, a reasoner $\mathfrak{R}$ is associated with it. $\mathfrak{R}$ is designed to answer queries as follows: If a query cannot be inferred from Σ, the answer is "Unknown". If it can be inferred and it is not in $\mathbb{E}_{\mathcal{A}} \cup \mathbb{E}_{\mathcal{T}}$, the answer is "Yes"; otherwise, the answer is "Unknown". Note that since the syntax of $\mathcal{ELH}$ does not include negation, an $\mathcal{ELH}$ KB cannot entail a negative query.

We make the following assumptions about the capabilities of the querying agent:

(a) does not have direct access to the KB Σ, but is aware of the underlying vocabulary,

(b) can ask queries in the form of assertions or GCIs, and

(c) cannot ask queries in the form of role inclusions.

$$
\begin{array}{rl}
\text{Inv-A}^{-}_{\sqcap} - \text{rule}: & \text{if } \{C(a), D(a)\} \cap \mathbb{E}_{\mathcal{A}} \neq \emptyset \text{ and } C \sqcap D(a) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}, \\
& \text{then } \mathbb{E}_{\mathcal{A}} := \mathbb{E}_{\mathcal{A}} \cup \{C \sqcap D(a)\}; \\
\text{Inv-A}^{+}_{\sqcap} - \text{rule}: & \text{if } C \sqcap D(a) \in \mathbb{E}_{\mathcal{A}},\ C \sqcap D \in \mathcal{C}_{\Sigma,\mathbb{S}} \text{ and } \{C(a), D(a)\} \subseteq \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}, \\
& \text{then } \mathbb{E}_{\mathcal{A}} := \mathbb{E}_{\mathcal{A}} \cup \{C(a)\} \text{ or } \mathbb{E}_{\mathcal{A}} := \mathbb{E}_{\mathcal{A}} \cup \{D(a)\}; \\
\text{Inv-A}^{+}_{\exists} - \text{rule}: & \text{if } \exists r.C(a) \in \mathbb{E}_{\mathcal{A}},\ \{r(a,b), C(b)\} \subseteq \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}} \text{ and } \exists r.C \in \mathcal{C}_{\Sigma,\mathbb{S}}, \\
& \text{then } \mathbb{E}_{\mathcal{A}} := \mathbb{E}_{\mathcal{A}} \cup \{r(a,b)\} \text{ or } \mathbb{E}_{\mathcal{A}} := \mathbb{E}_{\mathcal{A}} \cup \{C(b)\}; \\
\text{Inv-A}_{\sqsubseteq} - \text{rule}: & \text{if } D(a) \in \mathbb{E}_{\mathcal{A}},\ C \sqsubseteq D \in \mathcal{T}^*, \text{ and } C(a) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}, \\
& \text{then } \mathbb{E}_{\mathcal{A}} := \mathbb{E}_{\mathcal{A}} \cup \{C(a)\}; \\
\text{Inv-A}^{+}_{\exists H} - \text{rule}: & \text{if } \exists s.D(a) \in \mathbb{E}_{\mathcal{A}},\ C \sqsubseteq D \in \mathcal{T}^*,\ r \sqsubseteq s \in \mathcal{R}^*,\ \exists s.D \in \mathcal{C}_{\Sigma,\mathbb{S}} \text{ and} \\
& \exists r.C(a) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}, \text{ then } \mathbb{E}_{\mathcal{A}} := \mathbb{E}_{\mathcal{A}} \cup \{\exists r.C(a)\}; \\
\text{Inv-A}_{H} - \text{rule}: & \text{if } s(a,b) \in \mathbb{E}_{\mathcal{A}},\ r \sqsubseteq s \in \mathcal{R}^*, \text{ and } r(a,b) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}, \\
& \text{then } \mathbb{E}_{\mathcal{A}} := \mathbb{E}_{\mathcal{A}} \cup \{r(a,b)\}.
\end{array}
$$

Figure 3.3 Inverted ABox Tableau expansion rules

We formally define the notion of an envelope in the following.

Definition 3.4.1. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be a $\mathcal{ELH}$ KB, and let $\mathbb{S}_{\mathcal{A}}$ and $\mathbb{S}_{\mathcal{T}}$ be two finite secrecy sets. The secrecy envelopes $\mathbb{E}_{\mathcal{A}}$ and $\mathbb{E}_{\mathcal{T}}$ of $\mathbb{S}_{\mathcal{A}}$ and $\mathbb{S}_{\mathcal{T}}$ respectively, have the following properties:*

- $\mathbb{S}_{\mathcal{A}} \subseteq \mathbb{E}_{\mathcal{A}} \subseteq \mathcal{A}^* \setminus AX_{\mathcal{A}}$,

- $\mathbb{S}_{\mathcal{T}} \subseteq \mathbb{E}_{\mathcal{T}} \subseteq \mathcal{T}^* \setminus AX_{\mathcal{T}}$,

- *for every* $\alpha \in \mathbb{E}_{\mathcal{T}}$, $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}} \not\models \alpha$, *and*

- *for every* $\alpha \in \mathbb{E}_{\mathcal{A}}$, $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}} \not\models \alpha$.

The intuition for the above definition is that no information in $\mathbb{E}_{\mathcal{A}}$ and $\mathbb{E}_{\mathcal{T}}$ can be inferred from the corresponding sets $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ and $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$. To compute envelopes, we use the idea of inverting the rules of Figures 3.1 and 3.2 (see Tao et al. (2010), where this approach was first utilized for membership assertions). Induced by the TBox and ABox expansion rules in Figures 3.1 and 3.2, we define the corresponding "inverted" ABox and TBox expansion rules in Figures 3.3 and 3.4, respectively. These inverted expansion rules are denoted by prefixing Inv- to the name of the corresponding expansion rules.

From now on, we assume that $\mathcal{A}^*$, $\mathcal{T}^*$ and $\mathcal{R}^*$ have been computed and readily available for computing the envelopes. The computation of envelopes proceeds in two steps. In the first step, we compute $\mathbb{E}_{\mathcal{A}}$ by initializing it to $\mathbb{S}_{\mathcal{A}}$ and then expanding it using the inverted expansion rules listed in Figure 3.3 until no further applications are possible. We denote by $\Lambda^S_{\mathcal{A}}$ the algorithm which computes the set $\mathbb{E}_{\mathcal{A}}$. Due to non-determinism in applying the rules Inv-A$^+_{\sqcap}$ and Inv-A$^+_{\exists}$, different executions of $\Lambda^S_{\mathcal{A}}$ may result in different outputs. Since $\mathcal{A}^*$ is finite, the computation of $\Lambda^S_{\mathcal{A}}$ terminates. Let $\mathbb{E}_{\mathcal{A}}$ be an output of $\Lambda^S_{\mathcal{A}}$. Since the size of $\mathcal{A}^*$ is polynomial in $|\Sigma|+|\mathcal{C}_{\Sigma,\mathbb{S}}|$, and each application of inverted expansion rule moves some assertions from $\mathcal{A}^*$ into $\mathbb{E}_{\mathcal{A}}$, the size of $\mathbb{E}_{\mathcal{A}}$ is at most the size of $\mathcal{A}^*$. Therefore $\Lambda^S_{\mathcal{A}}$ takes polynomial time in $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathbb{S}} \mid$ to compute the envelope $\mathbb{E}_{\mathcal{A}}$.

In step two, we compute $\mathbb{E}_{\mathcal{T}}$ independent of $\mathbb{E}_{\mathcal{A}}$ by initializing it to $\mathbb{S}_{\mathcal{T}}$ and then expanding it using the inverted TBox expansion rules listed in Figure 3.4 until no further applications of rules are possible. We denote by $\Lambda^S_{\mathcal{T}}$ the algorithm which computes the set $\mathbb{E}_{\mathcal{T}}$. Similarly to $\Lambda^S_{\mathcal{A}}$, due to non-determinism in applying Inv-T$^+_{\sqcap}$ and Inv-T$^+_{H}$-rules, different executions of $\Lambda^S_{\mathcal{T}}$ may result in different outputs. Since $\mathcal{T}^*$ is finite, the computation of $\Lambda^S_{\mathcal{T}}$ terminates. Let $\mathbb{E}_{\mathcal{T}}$ be an output of $\Lambda^S_{\mathcal{T}}$. Since the size of $\mathcal{T}^*$ is polynomial in the size of Σ and $\mathcal{C}_{\Sigma,\mathbb{S}}$, and each application of inverted TBox expansion rule moves some GCIs from $\mathcal{T}^*$ into $\mathbb{E}_{\mathcal{T}}$, the size of $\mathbb{E}_{\mathcal{T}}$ is at most the size of $\mathcal{T}^*$. Therefore $\Lambda^S_{\mathcal{T}}$ takes polynomial time in $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathbb{S}} \mid$ to compute the envelope $\mathbb{E}_{\mathcal{T}}$.

Example 3.4.1. *(Example 3.3.2 cont.) Recall that $\mathcal{A}^*$ and $\mathcal{T}^*$ are the completed ABox and TBox respectively. Let $\mathbb{S}_{\mathcal{A}} = \{D \sqcap E(a)\}$ and $\mathbb{S}_{\mathcal{T}} = \{C \sqsubseteq D \sqcap E\}$ be the secrecy sets. Then, by using rules in Figure 3.3, we get the envelope for $\mathbb{S}_{\mathcal{A}}$,*

$$\mathbb{E}_{\mathcal{A}} = \mathbb{S}_{\mathcal{A}} \cup \{D(a)\}.$$

Similarly, using the rules in Figure 3.4, we get the envelope for $\mathbb{S}_{\mathcal{T}}$,

$$\mathbb{E}_{\mathcal{T}} = \mathbb{S}_{\mathcal{T}} \cup \{C \sqsubseteq D\}. \quad \square$$

Before proving the main results on envelopes, we prove the following auxiliary lemmas. First, we show that no assertions in $\mathbb{E}_{\mathcal{A}}$ is "logically reachable" from any assertion in $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$.

Inv-T$_{\sqsubseteq}$ − rule :	if $C \sqsubseteq E \in \mathbb{E}_{\mathcal{T}},\ D \sqsubseteq E \in \mathcal{T}$ and $C \sqsubseteq D\ \in \mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$,
	then $\mathbb{E}_{\mathcal{T}} := \mathbb{E}_{\mathcal{T}} \cup \{C \sqsubseteq D\}$;
Inv-T$^-_{\sqcap}$ − rule :	if $\{C \sqsubseteq D, C \sqsubseteq E\} \cap \mathbb{E}_{\mathcal{T}} \neq \emptyset$ and $C \sqsubseteq D \sqcap E \in \mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$,
	then $\mathbb{E}_{\mathcal{T}} := \mathbb{E}_{\mathcal{T}} \cup \{C \sqsubseteq D \sqcap E\}$;
Inv-T$^+_{\sqcap}$ − rule :	if $C \sqsubseteq D \sqcap E \in \mathbb{E}_{\mathcal{T}},\ D \sqcap E \in \mathcal{C}_{\Sigma,\mathbb{S}}$ and $\{C \sqsubseteq D, C \sqsubseteq E\} \subseteq \mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$,
	then $\mathbb{E}_{\mathcal{T}} := \mathbb{E}_{\mathcal{T}} \cup \{C \sqsubseteq D\}$ or $\mathbb{E}_{\mathcal{T}} := \mathbb{E}_{\mathcal{T}} \cup \{C \sqsubseteq E\}$;
Inv-T$^+_{H}$ − rule :	if $C \sqsubseteq \exists s.E \in \mathbb{E}_{\mathcal{T}},\ r \sqsubseteq s \in \mathcal{R}^*, \exists s.E \in \mathcal{C}_{\Sigma,\mathbb{S}}$ and
	$\{C \sqsubseteq \exists r.D, D \sqsubseteq E\} \subseteq \mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$,
	then $\mathbb{E}_{\mathcal{T}} := \mathbb{E}_{\mathcal{T}} \cup \{C \sqsubseteq \exists r.D\}$ or $\mathbb{E}_{\mathcal{T}} := \mathbb{E}_{\mathcal{T}} \cup \{D \sqsubseteq E\}$.

Figure 3.4 Inverted TBox Tableau expansion rules.

Lemma 3.4.1. *Let $\mathcal{A}^*$ be a completed ABox obtained from $\mathcal{A}$ by applying the tableau expansion rules in Figure 3.2. Also, let $\mathbb{E}_{\mathcal{A}}$ be a set of assertions which is completed by applying the tableau expansion rules in Figure 3.3 starting with the secrecy set $\mathbb{S}_{\mathcal{A}}$. Then, the ABox $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ is completed.*

Proof. *We have to show that no rule in Figure 3.2 is applicable to $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. The proof is by contradiction according to cases: assuming that a rule in Figure 3.2 is applicable and showing that a some inverse rule is applicable.*

- *If $A^-_{\sqcap}$-rule is applicable, then there is an assertion $C \sqcap D(a) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ such that $C(a) \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ or $D(a) \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. Since $\mathcal{A}^*$ is completed, $\{C(a), D(a)\} \subseteq \mathcal{A}^*$. Hence, $\{C(a), D(a)\} \cap \mathbb{E}_{\mathcal{A}} \neq \emptyset$. This makes the Inv-$A^-_{\sqcap}$-rule applicable.*

- *If $A^+_{\sqcap}$-rule is applicable, then there are assertions $C(a), D(a) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ such that $C \sqcap D \in \mathcal{C}_{\Sigma,\mathbb{S}}$ and $C \sqcap D(a) \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. Since $\mathcal{A}^*$ is completed, $C \sqcap D(a) \in \mathcal{A}^*$. Hence, $C \sqcap D(a) \in \mathbb{E}_{\mathcal{A}}$. This makes the Inv-$A^+_{\sqcap}$-rule applicable.*

- *If $A^+_{\exists}$-rule is applicable, then there are assertions $r(a,b), C(b) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ such that $\exists r.C \in \mathcal{C}_{\Sigma,\mathbb{S}}$ and $\exists r.C(a) \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. Since $\mathcal{A}^*$ is completed, $\exists r.C(a) \in \mathcal{A}^*$. Hence, $\exists r.C(a) \in \mathbb{E}_{\mathcal{A}}$. This makes the Inv-$A^+_{\exists}$-rule applicable.*

- *If $A_{\sqsubseteq}$-rule is applicable, then there is an assertion $C(a) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ and a GCI $C \sqsubseteq D \in \mathcal{T}^*$ such that $D(a) \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. Since $\mathcal{A}^*$ is completed, $D(a) \in \mathcal{A}^*$. Hence, $D(a) \in \mathbb{E}_{\mathcal{A}}$. This makes the Inv-$A_{\sqsubseteq}$-rule applicable.*

- *If $A^+_{\exists H}$-rule is applicable, then there is an assertion $\exists r.C(a) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$, a GCI $C \sqsubseteq D \in \mathcal{T}^*$, a role inclusion $r \sqsubseteq s \in \mathcal{R}^*$ such that $\exists s.D \in \mathcal{C}_{\Sigma,\mathbb{S}}$ and $\exists s.D(a) \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. Since $\mathcal{A}^*$ is completed, $\exists s.D(a) \in \mathcal{A}^*$. Hence, $\exists s.D(a) \in \mathbb{E}_{\mathcal{A}}$. This makes the Inv-$A^+_{\exists H}$-rule applicable.*

- *If A_H-rule is applicable, then there is an assertion $r(a,b) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ and a role inclusion $r \sqsubseteq s \in \mathcal{R}^*$ such that $s(a,b) \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. Since $\mathcal{A}^*$ is completed, $s(a,b) \in \mathcal{A}^*$. Hence, $s(a,b) \in \mathbb{E}_{\mathcal{A}}$. This makes the Inv-$A_H$-rule applicable.* $\square$

The next lemma is an analog of Lemma 4.4.1 for $\mathcal{T}^*$. Its proof is similar.

Lemma 3.4.2. *Let $\mathcal{T}^*$ be a completed TBox obtained from Σ and $\mathcal{C}_{\Sigma,\mathbb{S}}$ by applying the tableau expansion rules in Figure 3.1. Also, let $\mathbb{E}_{\mathcal{T}}$ be a set of GCIs which is completed by using tableau expansion rules in Figure 3.4 starting with the secrecy set $\mathbb{S}_{\mathcal{T}}$. Then, the TBox $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$ is completed.*

We now show that the completed sets $\mathbb{E}_{\mathcal{A}}$ and $\mathbb{E}_{\mathcal{T}}$ are in fact envelopes.

Theorem 3.4.1. *$\mathbb{E}_{\mathcal{A}}$ and $\mathbb{E}_{\mathcal{T}}$ are envelopes for $\mathbb{S}_{\mathcal{A}}$ and $\mathbb{S}_{\mathcal{T}}$ respectively .*

Proof. *We must show that the sets $\mathbb{E}_{\mathcal{A}}$ and $\mathbb{E}_{\mathcal{T}}$ satisfy the four properties of Definition 3.4.1. Properties 1 and 2 are obvious. To prove property 3, suppose $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}} \models \alpha$, for some $\alpha \in \mathbb{E}_{\mathcal{A}}$. This means, by Theorem 3.3.2, that $\alpha \in (\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}})^*$ and since, by Lemma 3.4.1, $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ is completed, $(\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}})^* = \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$, whence $\alpha \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. This is a contradiction. Proof of property 4 is similar, using Theorem 3.3.1 and Lemma 3.4.2 instead of Theorem 3.3.2 and Lemma 3.4.1, respectively.* $\square$

To answer queries as informatively as possible without revealing the secret information, we should aim to make the size of the envelope $\mathbb{E}$ as small as possible. From now on, we focus on computing a tight envelope $\mathbb{E}$ which is defined in Definition 2.5.2. We now show by an example, that the envelopes computed by using the rules in Figures 3.3 and 3.4 are not necessarily tight.

```
EvalA(q)
case q ∈ A* \ E_A
    return "Yes"
case q = C ⊓ D(a)
    if EvalA(C(a)) = "Yes" and EvalA(D(a)) = "Yes" then
        return "Yes"
    else
        return "Unknown"
case q = ∃r.C(a)
    if for some d ∈ O_Σ [ r(a,d) ∈ A* \ E_A and EvalA(C(d)) ="Yes"] then
        return "Yes"
    else
        if for some E ∈ C_{Σ,S} [E ⊑ C ∈ T* and EvalA(∃r.E(a)) = "Yes" ] then
            return "Yes"
        else
            if for some s ∈ R_R [s ⊑ r ∈ R* and EvalA(∃s.C(a)) = "Yes" ]
then
                return "Yes"
            else
                return "Unknown"
```

Figure 3.5 Query answering algorithm for assertional queries

Example 3.4.2. *Let* $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ *be a* $\mathcal{ELH}$ *KB, where* $\mathcal{A} = \{C(a), r(b,a)\}$, $\mathcal{T} = \{A \sqsubseteq B, C \sqsubseteq D \sqcap E, C \sqsubseteq D \sqcap F\}$ *and* $\mathcal{R} = \emptyset$. *Also let* $\mathbb{S}_{\mathcal{A}} = \{D \sqcap E(a), D \sqcap F(a)\}$ *and* $\mathbb{S}_{\mathcal{T}} = \{C \sqsubseteq D \sqcap E, C \sqsubseteq D \sqcap F\}$ *be the secrecy sets.*

Since $\Lambda^S_{\mathcal{A}}$ is non-deterministic, we may get different envelopes as an output. Some of the envelopes are

1 $\mathbb{E}_{\mathcal{A}} = \mathbb{S}_{\mathcal{A}} \cup \{D(a), F(a)\}$ – not tight,

2 $\mathbb{E}_{\mathcal{A}} = \mathbb{S}_{\mathcal{A}} \cup \{E(a), F(a)\}$ – tight and

3 $\mathbb{E}_{\mathcal{A}} = \mathbb{S}_{\mathcal{A}} \cup \{D(a)\}$ – minimum and tight.

Since $\Lambda^S_{\mathcal{T}}$ is non-deterministic, we may get different envelopes as an output depending on the choice made in the application of Inv-$\mathrm{T}^+_{\sqcap}$-rule when computing the envelopes. The envelopes are

1 $\mathbb{E}_{\mathcal{T}} = \mathbb{S}_{\mathcal{T}} \cup \{C \sqsubseteq D, C \sqsubseteq F\}$ – not tight,

2 $\mathbb{E}_{\mathcal{T}} = \mathbb{S}_{\mathcal{T}} \cup \{C \sqsubseteq E, C \sqsubseteq F\}$ – tight and

3 $\mathbb{E}_{\mathcal{T}} = \mathbb{S}_{\mathcal{T}} \cup \{C \sqsubseteq D\}$ – minimum and tight. $\square$

By using the procedure presented in Section 2.5, tight envelopes for the secrecy sets $\mathbb{S}_{\mathcal{A}}$ and $\mathbb{S}_{\mathcal{T}}$ can be computed in polynomial time.

3.5 Query Answering

The recursive procedures given in Figures 3.5 and 3.6 take an input q (as a query) and output "Yes" or "Unknown". In Section 3.4, we have described briefly how the reasoner $\mathfrak{R}$ responds to queries. In this section we provide a few more details. Here we assume that $\mathcal{A}^*$, $\mathbb{E}_{\mathcal{A}}$, $\mathcal{T}^*$, $\mathbb{E}_{\mathcal{T}}$ and $\mathcal{R}^*$ have all been precomputed and are considered to be globally accessible. Define the set $R_{\mathcal{R}} = \{r \mid r \text{ is a role name that occurs in } \mathcal{R}\}$. The recursive procedures for answering the assertional queries and the GCI queries are given in Figure 3.5 and Figure 3.6 respectively. In Lines 1 and 2 of Figure 3.5, we check the membership of q in $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ and answer "Yes" if $q \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. From line 3 onwards we consider several cases in which we break the query q into subqueries based on the constructors defined in the language $\mathcal{ELH}$ and apply the procedure recursively.

The following theorem proves the correctness of the algorithm.

Theorem 3.5.1. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be an $\mathcal{ELH}$ KB. Let $\mathcal{A}^*$ be an completed ABox, $\mathbb{E}_{\mathcal{A}}$ an envelope of the secrecy set $\mathbb{S}_{\mathcal{A}}$ and q an assertional query. Then,*

- *Soundness: EvalA(q) outputs "Yes"* $\Rightarrow$ $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}} \models q$

- *Completeness: EvalA(q) outputs "Unknown"* $\Rightarrow$ $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}} \not\models q$

Proof. *We first consider the soundness part. Let $\mathcal{I}$ be an arbitrary model of Σ and therefore $\mathcal{I}$ satisfies $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$, $\mathcal{T}^*$ and $\mathcal{R}^*$, and let q be a query. We argue inductively by cases:*

- *$q \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. Since $\Lambda_{\mathcal{A}}$ is sound, we have $\mathcal{I} \models q$.*

- *$q = C \sqcap D(a) \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. Then, EvalA(C(a)) = EvalA(D(a)) ="Yes" and by inductive hypothesis, $\mathcal{I} \models C(a)$ and $\mathcal{I} \models D(a)$. Hence, $a \in C^{\mathcal{I}} \cap D^{\mathcal{I}} = (C \sqcap D)^{\mathcal{I}}$, i.e., $\mathcal{I} \models C \sqcap D(a)$.*

- $q = \exists r.C(a) \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. *There are several subcases:*

  - *Let for some* $d \in \mathcal{O}_{\Sigma}$ $[\, r(a,d) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ *and EvalA*$(C(d))$ *=“Yes”]. Then by the first case* $\mathcal{I} \models r(a,d)$ *and by inductive hypothesis* $\mathcal{I} \models C(d)$. *This immediately proves* $\mathcal{I} \models \exists r.C(a)$.

  - *Let for some* $E \in \mathcal{C}_{\Sigma,\mathbb{S}}$ $[E \sqsubseteq C \in \mathcal{T}^*$ *and EvalA*$(\exists r.E(a))$ *= “Yes”]. Then* $\mathcal{I} \models E \sqsubseteq C$, *and by inductive hypothesis,* $\mathcal{I} \models \exists r.E(a)$, *whence* $\mathcal{I} \models \exists r.C(a)$.

  - *Let for some* $s \in R_{\mathcal{R}}$ $[s \sqsubseteq r \in \mathcal{R}^*$ *and EvalA*$(\exists s.C(a))$ *= “Yes”]. Then* $\mathcal{I} \models s \sqsubseteq r$, *and by inductive hypothesis,* $\mathcal{I} \models \exists s.C(a)$ *implying* $\mathcal{I} \models \exists r.C(a)$.

We prove the completeness part using a contrapositive argument. Assume that $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}} \models q$. *We have to show that EvalA*(q) *= “Yes”. Let* $\mathcal{K}$ *be the canonical interpretation as defined in section 3.3. By Lemma 3.3.7,* $\mathcal{K}$ *satisfies* $\mathcal{A}^*$, $\mathcal{T}^*$ *and* $\mathcal{R}^*$ *and hence* $\mathcal{K}$ *satisfies* $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ *and* q. *We argue that: if* $\mathcal{K} \models q$ *then EvalA*(q) *= “Yes”, by induction on the structure of* q. *There are two cases. If* $q \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$, *then the claim follows immediately. Next, consider the case* $q \notin \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$. *There are several cases:*

- $q = C \sqcap D(a)$. *To answer this query the algorithm computes EvalA*$(C(a))$ *and EvalA*$(D(a))$. *Now, the assumption* $\mathcal{K} \models C \sqcap D(a)$ *implies* $\mathcal{K} \models C(a)$ *and* $\mathcal{K} \models D(a)$ *which, by inductive hypothesis, implies that EvalA*$(C(a))$ *= EvalA*$(D(a))$ *= “Yes”. Hence, by Lines 4 and 5 in Figure 3.5, EvalA*$(C \sqcap D(a))$*=“Yes”.*

- $q = \exists r.C(a)$. *By the assumption,* $\mathcal{K} \models \exists r.C(a)$. *This implies, for some* $b \in \Delta$ $[(a,b) \in r^{\mathcal{K}}$ *and* $b \in C^{\mathcal{K}}]$. *There are two subcases:*

  - r *is minimal with respect to* (a,b). *Again there are two subcases:*

    - $b \in \mathcal{O}_{\Sigma}$. *Then,* $\mathcal{K} \models r(a,b)$ *and* $\mathcal{K} \models C(b)$. *By the first case* $r(a,b) \in \mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ *and by inductive hypothesis EvalA*$(C(b))$ *= “Yes”. Hence, by Lines 9 and 10 in Figure 3.5, EvalA* $(\exists r.C(a))$*=“Yes”.*

    - $b = w_D \in \mathcal{W}$ *for some* $D \in \mathcal{C}_{\Sigma,\mathbb{S}}$. *Then,* $\mathcal{K} \models \exists r.D(a)$ *and by part (b) of Lemma 3.3.4,* $D \sqsubseteq C \in \mathcal{T}^*$. *By inductive hypothesis EvalA*$(\exists r.D(a))$ *= “Yes”. Hence, by Lines 12 and 13 in Figure 3.5, EvalA*$(\exists r.C(a))$ *=“Yes”.*

- *r is not minimal with respect to (a, b). Since RBox $\mathcal{R}$ is acyclic, there exists a chain $s \sqsubseteq v_1 \sqsubseteq v_2 ..... \sqsubseteq v_k \sqsubseteq u$ in $\mathcal{R}$ such that s is minimal with respect to (a, b). Since $\mathcal{R}^*$ is the transitive closure of $\mathcal{R}$, $s \sqsubseteq r \in \mathcal{R}^*$. Again there are two cases:*
  - *$b \in \mathcal{O}_\Sigma$. Then, by Definition 3.3.2 and the definition of $\mathcal{K}$, $\mathcal{K} \models s(a, b)$. Also, $\mathcal{K} \models s \sqsubseteq r$ and $\mathcal{K} \models C(b)$. By the first subcase of the previous case EvalA($\exists s.C(a)$) = "Yes". Hence, by Lines 15 and 16 in Figure 3.5, EvalA ($\exists r.C(a)$)="Yes".*
  - *$b = w_D \in \mathcal{W}$ for some $D \in \mathcal{C}_{\Sigma,\mathbb{S}}$. Then, by Definition 3.3.2 and the definition of $\mathcal{K}$, $\mathcal{K} \models \exists s.D(a)$. Also, $\mathcal{K} \models s \sqsubseteq r$ and by part (b) of Lemma 3.3.4, $D \sqsubseteq C \in \mathcal{T}^*$. By the second subcase of the previous case EvalA($\exists s.C(a)$) = "Yes". Hence, by Lines 15 and 16 in Figure 3.5, EvalA($\exists r.C(a)$) ="Yes".* □

Since the algorithm given in Figure 3.5 runs in polynomial time in the size of $\mathcal{A}^* \setminus \mathbb{E}_\mathcal{A}$ and q, the assertional query answering can be done in polynomial time as a function of $\mid \mathcal{A}^* \mid + \mid q \mid$.

Next, suppose that the querying agent poses a GCI query q. In response, the reasoner $\mathfrak{R}$ invokes the query answering algorithm EvalT(q) given in Figure 3.6 and returns the answer as output. We prove in the following the correctness of the recursive algorithm given in Figure 3.6.

Theorem 3.5.2. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be an $\mathcal{ELH}$ KB. Let $\mathcal{T}^*$ be an completed TBox, $\mathbb{E}_\mathcal{T}$ an envelope of the secrecy set $\mathbb{S}_\mathcal{T}$ and q a GCI query. Then,*

- *Soundness: EvalT(q) outputs "Yes" $\Rightarrow$ $\mathcal{T}^* \setminus \mathbb{E}_\mathcal{T} \models q$*
- *Completeness: EvalT(q) outputs "Unknown" $\Rightarrow$ $\mathcal{T}^* \setminus \mathbb{E}_\mathcal{T} \not\models q$*

Proof. *We first consider the soundness part. Let $\mathcal{I}$ be an arbitrary model of Σ and therefore $\mathcal{I}$ satisfies $\mathcal{T}^* \setminus \mathbb{E}_\mathcal{T}$ and $\mathcal{R}^*$, and let q be a GCI query. We argue inductively by cases:*

- *$q \in \mathcal{T}^* \setminus \mathbb{E}_\mathcal{T}$. Since $\Lambda_\mathcal{T}$ is sound, we have $\mathcal{I} \models q$.*
- *$q = C \sqsubseteq D \sqcap E \notin \mathcal{T}^* \setminus \mathbb{E}_\mathcal{T}$. Then, EvalT($C \sqsubseteq D$) = EvalT($C \sqsubseteq E$) ="Yes" and by inductive hypothesis, $\mathcal{I} \models C \sqsubseteq D$ and $\mathcal{I} \models C \sqsubseteq E$. This implies, $C^\mathcal{I} \subseteq D^\mathcal{I}$ and $C^\mathcal{I} \subseteq E^\mathcal{I}$. Hence, $C^\mathcal{I} \subseteq (D^\mathcal{I} \cap E^\mathcal{I}) = (D \sqcap E)^\mathcal{I}$, i.e., $\mathcal{I} \models C \sqsubseteq D \sqcap E$.*

```
EvalT(q)
 1: case q ∈ T* \ E_T
 2:     return "Yes"
 3: case q = C ⊑ D ⊓ E
 4:     if EvalT(C ⊑ D) ="Yes" and EvalT(C ⊑ E) ="Yes" then
 5:         return "Yes"
 6:     else
 7:         return "Unknown"
 8: case q = C ⊑ ∃r.D
 9:     if  for some E ∈ C_{Σ,S} [E ⊑ D ∈ T* and EvalT(C ⊑ ∃r.E) ="Yes"] then
10:         return "Yes"
11:     else
12:         if  for some s ∈ R_R [s ⊑ r ∈ R* and EvalT(C ⊑ ∃s.D) ="Yes"] then
13:             return "Yes"
14:         else
15:             return "Unknown"
```

Figure 3.6 Query answering algorithm for GCI queries

- *$q = C \sqsubseteq \exists r.D \notin \mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$. There are two subcases:*

  - *Let for some $E \in \mathcal{C}_{\Sigma,\mathbb{S}}$ [$E \sqsubseteq D \in \mathcal{T}^*$ and EvalT($C \sqsubseteq \exists r.E$) ="Yes"]. Then by the first case $\mathcal{I} \models E \sqsubseteq D$, and by inductive hypothesis, $\mathcal{I} \models C \sqsubseteq \exists r.E$, whence $\mathcal{I} \models C \sqsubseteq \exists r.D$.*

  - *Let for some $s \in R_{\mathcal{R}}$ [$s \sqsubseteq r \in \mathcal{R}^*$ and EvalT($C \sqsubseteq \exists s.D$) = "Yes"]. Then $\mathcal{I} \models s \sqsubseteq r$, and by inductive hypothesis, $\mathcal{I} \models C \sqsubseteq \exists s.D$ implying $\mathcal{I} \models C \sqsubseteq \exists r.D$.*

We prove the completeness part using a contrapositive argument. Assume that $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}} \models q$. We have to show that EvalT(q) ="Yes". Let $\mathcal{J}$ be the canonical interpretation as defined in section 3.3. By Lemma 3.3.2, $\mathcal{J}$ satisfies $\mathcal{T}^*$ and $\mathcal{R}^*$. Hence $\mathcal{J}$ satisfies $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$ and q. We argue by induction on the structure of q that, if $\mathcal{J} \models q$ then EvalT(q) = "Yes". The basic case is, $q \in \mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$. Then, by Lines 1 and 2 in Figure 3.6, the claim is obvious. Next, consider the case $q \notin \mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$. There are several cases:

- *$q = C \sqsubseteq D \sqcap E$. The algorithm in Figure 3.6 computes EvalT($C \sqsubseteq D$) and EvalT($C \sqsubseteq E$). Now, the assumption $\mathcal{J} \models C \sqsubseteq D \sqcap E$ implies $\mathcal{J} \models C \sqsubseteq D$ and $\mathcal{J} \models C \sqsubseteq E$ which, by inductive hypothesis, implies that EvalT($C \sqsubseteq D$) = EvalT($C \sqsubseteq E$) = "Yes". Hence, by*

Lines 4 and 5 in Figure 3.6, EvalT($C \sqsubseteq D \sqcap E$) = "Yes".

- $q = C \sqsubseteq \exists r.D$. *By the assumption,* $\mathcal{J} \models C \sqsubseteq \exists r.D$. *This implies,* C, $D \in \mathcal{C}_{\Sigma,\mathbb{S}}$ *and* $\exists r.D \notin \mathcal{C}_{\Sigma,\mathbb{S}}$.

  - $\mathcal{J} \models C \sqsubseteq \exists r.E_1, E_1 \sqsubseteq E_2, ...E_{k-1} \sqsubseteq E_k, E_k \sqsubseteq D$ *where* $\exists r.E_1$, $E_2, ...$ $E_k \in \mathcal{C}_{\Sigma,\mathbb{S}}$ *and* $C \sqsubseteq \exists r.E_1, E_1 \sqsubseteq E_2, ...E_{k-1} \sqsubseteq E_k, E_k \sqsubseteq D \in \mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$. *Since, by Lemma 3.4.2,* $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$ *is completed,* $E_1 \sqsubseteq D \in \mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$. *Also, by the basic step, EvalT($C \sqsubseteq \exists r.E_1$) = "Yes". Hence, by Lines 9 and 10, EvalT($C \sqsubseteq \exists r.D$) = "Yes".*

  - $\mathcal{J} \models C \sqsubseteq \exists s.D$, $s \sqsubseteq v_1, v_1 \sqsubseteq v_2, ...v_k \sqsubseteq r$ *where* $\exists s.D \in \mathcal{C}_{\Sigma,\mathbb{S}}$, $s, v_1, v_2, ...v_k \in R_{\mathcal{R}}$, $C \sqsubseteq \exists s.D \in \mathcal{T}^*$ *and* $s \sqsubseteq v_1, v_1 \sqsubseteq v_2, ...v_k \sqsubseteq r \in \mathcal{R}^*$. *Then,* $s \sqsubseteq r \in \mathcal{R}^*$ *and by the basic step, EvalT($C \sqsubseteq \exists s.D$) = "Yes". Hence, by Lines 15 and 16, EvalT($C \sqsubseteq \exists r.D$) = "Yes".* $\square$

Since the algorithm runs in polynomial time in the size of $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$ and q, the GCI query answering can be done in polynomial time as a function of $\mid \mathcal{T}^* \mid + \mid q \mid$.

Example 3.5.1. *(Example 3.4.1 cont.) Recall that* $\mathcal{A}^*$ *and* $\mathcal{T}^*$ *are the completed ABox and TBox respectively. Also, recall that* $\mathbb{E}_{\mathcal{A}} = \mathbb{S}_{\mathcal{A}} \cup \{D(a)\}$ *and* $\mathbb{E}_{\mathcal{T}} = \mathbb{S}_{\mathcal{T}} \cup \{C \sqsubseteq D\}$ *are the the envelopes for* $\mathbb{S}_{\mathcal{A}}$ *and* $\mathbb{S}_{\mathcal{T}}$ *respectively.*

Suppose that the querying agent asks the assertional queries $C \sqcap E(a)$, $\exists r.C(b)$, $\exists r.E(b)$ *and* $D(a)$. *Using the algorithm in Figure 3.5, we get the following answers:*

q	*EvalA(q)*	*Remarks*
$C \sqcap E(a)$	*Yes*	*by Lines 4, 5*
$\exists r.E(b)$	*Yes*	*by Lines 12, 13*
$D(a)$	*Unknown*	*by Line 18*

Next, suppose that the querying agent asks the GCI queries $C \sqsubseteq C \sqcap E$, $\exists r.C \sqsubseteq \exists r.E$ *and* $C \sqsubseteq D$. *Using the algorithm in Figure 3.6, we get the following answers:*

q	*EvalT(q)*	*Remarks*
$C \sqsubseteq C \sqcap E$	*Yes*	*by Lines 4, 5*
$\exists r.C \sqsubseteq \exists r.E$	*Yes*	*by Lines 9, 10*
$C \sqsubseteq D$	*Unknown*	*by Line 15* $\square$

3.6 Conclusions

The main contribution of this chapter is that we allow secrets as well as queries to be of two types: (a) local type, assertions about specific individuals (e.g., $C(a)$ or $r(a, b)$), as well as (b) global type, GCIs (e.g., $C \sqsubseteq D$) which specify hierarchical inclusion relationships between concepts. Another contribution is in the way that we compute the consequences and preserve secrecy while answering queries. We break the process into two parts, first one precomputes all the consequences for concepts and individuals that occur in the given KB. For this we use four separate (but related) tableau procedures. As for the actual query answering, we parse the query all the way to constituents that occur in the previously precomputed set of consequences. Then, the queries are answered based on the membership of the constituents of the query in $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ and $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$. All the algorithms are efficient and can be implemented in polynomial time.

CHAPTER 4. KEEPING SECRETS IN MODALIZED DL KNOWLEDGE BASES

4.1 Introduction

Recently, Tao et al., in Tao et al. (2014) have developed a conceptual framework to study secrecy-preserving reasoning and query answering in DL knowledge bases (KBs) under Open World Assumptions (OWA). The approach uses the notion of an *envelope* to hide secret information against logical inference and it was first defined and used in Tao et al. (2010). This approach is based on the assumption that the information contained in a KB is incomplete (by OWA) and (so far) it has been restricted to very simple DLs and simple query languages. Specifically, in Tao et al. (2010, 2014); Krishnasamy Sivaprakasam and Slutzki (2016) the main idea was to utilize the secret information within the reasoning process, but then answering "Unknown" whenever the answer is truly unknown or in case the true answer could compromise confidentiality.

The modalized DLs are DLs with modal operators. Lutz et al., in Lutz et al. (2001) presented a tableau decision algorithm for modalized $\mathcal{ALC}$. In Tao et al. (2012), the authors studied satisfiability reasoning problem in acyclic modalized $\mathcal{ALC}$ KBs. Also in many reported research articles regarding privacy, modal logic is used to study privacy related reasoning tasks, see Barth and Mitchell (2005); Halpern and O'Neill (2005); Jafari et al. (2011). Specifically in Halpern and O'Neill (2005), the authors showed that the modal logic of knowledge for multiagent systems provides a fundamental framework for reasoning about anonymity. This framework was extended in Tsukada et al. (2009) to reasoning about privacy. In an attempt to reduce the complexity of reasoning in modal logic, Hemaspaandra in Hemaspaandra (2000) had considered several propositional modal logic languages with one modal operator. Motivated by

these works, in this chapter we study secrecy-preserving query answering problem for $\mathcal{ELH}^{\Diamond}$ KBs where $\mathcal{ELH}^{\Diamond}$ is the description logic $\mathcal{ELH}$ augmented with a modal operator $\Diamond$.

Given an $\mathcal{ELH}^{\Diamond}$ KB $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, a first step in constructing secrecy-preserving reasoning system, we use a tableau algorithm to compute a finite rooted labeled tree $\mathbb{T}$. The labeling set of the root node of this $\mathbb{T}$ is $\mathcal{A}^*$ which contains a set of consequences of the KB Σ, restricted to concepts that actually occur in Σ and an extra "auxiliary" set of concepts defined over the signature of Σ. Since the computed tree does not contain all the consequences of the KB, in order to answer user queries we have designed a recursive algorithm which breaks the queries into smaller assertions all the way until the information in $\mathbb{T}$ can be used. In effect, we have split the task of query answering into two parts: in the first part we compute all the consequences of Σ restricted to concepts and individuals that occur in Σ, in the second part we use a recursive algorithm to evaluate more complex queries with the base case that has been computed in the first part.

To protect the secret information in the secrecy set $\mathbb{S}$, we compute an envelope E which is a function that assigns a set of assertions to each node in $\mathbb{T}$. This envelope is computed by a another tableau algorithm based on the idea of inverting the local and global expansion rules given in the first tableau algorithm. The idea behind the envelope concept is that no expression in the envelope can be logically deduced from information outside the envelope. Once such envelope is computed, the answers to the queries are censored whenever the queries belong to the envelope. Since, generally, an envelope for a given secrecy set is not unique, the developer has some freedom to output a envelope (from the available choices) satisfying the needs of application domain, company policy, social obligations or user preferences.

Next, we discuss a query answering procedure which allows us to answer queries without revealing secrets. The queries are answered based on the information available in the secrecy-preserving tree obtained from the tree $\mathbb{T}$ and the envelope E, see Section 4.4. This tree, once computed, remains fixed. Usually in secrecy-preserving query answering framework queries are answered by checking their membership in a previously computed set, see Tao et al. (2010, 2014); Krishnasamy Sivaprakasam and Slutzki (2016). Since the secrecy-preserving tree does not contain all the statements entailed by Σ, we need to extend the query answering procedure

from just membership checking. Towards that end we designed a recursive algorithm to answer more complicated queries. To answer a query q, the algorithm first checks if q is a member of the labeling set of the root node of secrecy-preserving tree, in which case the answer is "Yes"; otherwise, the given query is broken into subqueries based on the logical constructors, and the algorithm is applied recursively on the subqueries, see Section 4.5.

4.2 Syntax and Semantics of $\mathcal{ELH}^{\Diamond}$

A vocabulary of $\mathcal{ELH}^{\Diamond}$ is a triple $< N_O, N_C, N_R >$ of countably infinite, pairwise disjoint sets. The elements of N_O are called *object* (or *individual) names*, the elements of N_C are called *concept names* and the elements of N_R are called *role names*. The set of $\mathcal{ELH}^{\Diamond}$ *concepts* is denoted by $\mathcal{C}$ and is defined by the following rules

$$C ::= A \mid \top \mid C \sqcap D \mid \exists r.C \mid \Diamond C$$

where $A \in N_C$, $r \in N_R$, $\top$ denotes the "*top concept*", $C, D \in \mathcal{C}$ and $\Diamond C$ denotes the *modal constructor*, read as *"diamond C"*. *Assertions* are expressions of the form $C(a)$ or $r(a, b)$, *general concept inclusions (GCIs)* are expressions of the form $C \sqsubseteq D$ and *role inclusions* are expressions of the form $r \sqsubseteq s$ where $C, D \in \mathcal{C}$, $r, s \in N_R$ and $a, b \in N_O$.

The semantics of $\mathcal{ELH}^{\Diamond}$ concepts is defined by using Kripke structures Blackburn et al. (2002); Kripke (1963). A *Kripke structure* is a tuple $\mathbb{M} = \langle S, \pi, \mathcal{E} \rangle$ where S is a set of *states*, $\mathcal{E} \subseteq S \times S$ is the *accessibility* relation, and π interprets the syntax of $\mathcal{ELH}^{\Diamond}$ at each state $s \in S$. The intuitive meaning of $(s, t) \in \mathcal{E}$ is that state t would be considered as a *possible state* from the state s in $\mathbb{M}$. Further, we denote by $\mathcal{E}(s)$ the set $\{t \mid (s, t) \in \mathcal{E}\}$ of the successors of the state s.

All the concepts and role names will be interpreted in a *common (i.e., state-independent) non-empty domain* which we denote by Δ, see Lutz et al. (2001); Tao et al. (2012). The interpretation of concepts and role names is defined inductively as follows: for all $a \in N_O$,

$A \in N_C$, $r \in N_R$, $C, D \in \mathcal{C}$ and for all $s \in S$,

$$\top^{\pi(s)} = \Delta; \quad a^{\pi(s)} \in \Delta;$$

$$A^{\pi(s)} \subseteq \Delta; \quad r^{\pi(s)} \subseteq \Delta \times \Delta;$$

$$(C \sqcap D)^{\pi(s)} = C^{\pi(s)} \cap D^{\pi(s)};$$

$$(\Diamond C)^{\pi(s)} = \bigcup_{t \in \mathcal{E}(s)} C^{\pi(t)};$$

$$(\exists r.C)^{\pi(s)} = \{d \in \Delta \mid \exists e \in C^{\pi(s)} : (d, e) \in r^{\pi(s)}\}.$$

An *ABox* $\mathcal{A}$ is a finite, non-empty set of assertions, a *TBox* $\mathcal{T}$ is a finite set of GCIs and an *RBox* $\mathcal{R}$ is a finite set of role inclusions. An $\mathcal{ELH}^{\Diamond}$ KB is a triple $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ where $\mathcal{A}$ is an ABox, $\mathcal{T}$ is a TBox and $\mathcal{R}$ is an RBox.

Let $\mathbb{M} = \langle S, \pi, \mathcal{E} \rangle$ be a Kripke structure, $s \in S$, $C, D \in \mathcal{C}$, $r, t \in N_R$ and $a, b \in N_O$. We say that $(\mathbb{M}, s)$ *satisfies* $C(a)$, $r(a, b)$, $C \sqsubseteq D$ or $r \sqsubseteq t$, notation $(\mathbb{M}, s) \models C(a)$, $(\mathbb{M}, s) \models r(a, b)$, $(\mathbb{M}, s) \models C \sqsubseteq D$ or $(\mathbb{M}, s) \models r \sqsubseteq t$ if, respectively, $a^{\pi(s)} \in C^{\pi(s)}$, $(a^{\pi(s)}, b^{\pi(s)}) \in r^{\pi(s)}$, $C^{\pi(s)} \subseteq D^{\pi(s)}$ or $r^{\pi(s)} \subseteq t^{\pi(s)}$. $(\mathbb{M}, s)$ *satisfies* Σ, notation $(\mathbb{M}, s) \models \Sigma$, if $(\mathbb{M}, s)$ satisfies all the assertions in $\mathcal{A}$, all the GCIs in $\mathcal{T}$ and all the role inclusions in $\mathcal{R}$. $\mathbb{M}$ *satisfies* Σ, or $\mathbb{M}$ is a *model* of Σ, if there exists a $s \in S$ such that $(\mathbb{M}, s) \models \Sigma$ and for all $t \in S$, $(\mathbb{M}, t) \models \mathcal{T} \cup \mathcal{R}$. Let α be an assertion, a GCI or a role inclusion. We say that Σ *entails* α, notation $\Sigma \models \alpha$, if for all Kripke structures $\mathbb{M}$ satisfying Σ and for all states s of $\mathbb{M}$, $(\mathbb{M}, s) \models \Sigma \Rightarrow (\mathbb{M}, s) \models \alpha$.

4.3 Computation of a Model for $\mathcal{ELH}^{\Diamond}$ KB Σ and $\mathcal{A}^*$

Before presenting an algorithm to compute a model for the given KB, we describe a preprocessing procedure to eliminate occurrences of $\top$ from the KB. In the first step we apply the following rules exhaustively to $\Sigma_3 = \langle \mathcal{A}_3, \mathcal{T}_3, \mathcal{R} \rangle$ until no further applications are possible to get a new KB $\Sigma_2 = \langle \mathcal{A}_2, \mathcal{T}_2, \mathcal{R} \rangle$. We initialize $\mathcal{A}_2$ as $\mathcal{A}_3$ and $\mathcal{T}_2$ as $\mathcal{T}_3$.

- If $\top \sqsubseteq C \in \mathcal{T}_2$, then remove it from $\mathcal{T}_2$ and replace C by $\top$ throughout Σ_2;

- If $\Diamond\top$ occurs in Σ_2, then replace $\Diamond\top$ by $\top$ throughout Σ_2 and

- If $C \sqcap \top$ or $\top \sqcap C$ occurs in Σ_2, then replace it by C throughout Σ_2.

Let $\mathcal{O}'$ be the set of individual names occurring in Σ_2. In the second step we apply the following rules exhaustively to Σ_2 until no further application is possible. Let $\Sigma_1 = \langle \mathcal{A}_1, \mathcal{T}_1, \mathcal{R} \rangle$ be the KB obtained from Σ_2. In this step, we initialize $\mathcal{A}_1$ as $\mathcal{A}_2$ and $\mathcal{T}_1$ as $\mathcal{T}_2$.

- If $\top(a) \in \mathcal{A}_1$, then remove $\top(a)$ from $\mathcal{A}_1$;
- If $\exists r.\top(a) \in \mathcal{A}_1$ and $\forall\ d \in \mathcal{O}'$, $r(a,d) \notin \mathcal{A}_1$, then add $r(a,b)$ to $\mathcal{A}_1$ where b is a fresh individual name and $\mathcal{O}' = \mathcal{O}' \cup \{b\}$; Remove $\exists r.\top(a)$ from $\mathcal{A}_1$;
- If $C \sqsubseteq \exists r.\top \in \mathcal{T}_1$, $C(a) \in \mathcal{A}_1$ and $\forall\ d \in \mathcal{O}'$, $r(a,d) \notin \mathcal{A}_1$, then add $r(a,b)$ to $\mathcal{A}_1$ where b is a fresh individual name and $\mathcal{O}' = \mathcal{O}' \cup \{b\}$;
- If $C \sqsubseteq \top \in \mathcal{T}_1$, then remove $C \sqsubseteq \top$ from $\mathcal{T}_1$, and
- If $\exists r.\top \sqsubseteq C \in \mathcal{T}_1$ and $r(a,b) \in \mathcal{A}_1$ and $C(a) \notin \mathcal{A}_1$, then add $C(a)$ to $\mathcal{A}_1$.

Finally, as a last step, we remove all the subsumptions of the form $C \sqsubseteq \exists r.\top$ and $\exists r.\top \sqsubseteq C$ from Σ_1. Here C, $D \in \mathcal{C}$, $r \in N_R$ and $a, b \in N_O$. Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be the output of this diminution procedure. It is easy to see that Σ can be computed using the rules from Σ_3 in the polynomial time in $\mid \Sigma_3 \mid$. From the rules, it is easy to see that $\Sigma_3 \models \alpha$ if and only if $\Sigma \models \alpha$ where α is an assertion, a GCI or a role inclusion. Hereafter in this chapter, we assume that all KBs are free from $\top$ symbol.

Denote by N_Σ the set of all concept names and role names occurring in Σ and let $\mathcal{S}$ be a finite set of concepts over the symbol set N_Σ[1]. Let $\mathcal{C}_{\Sigma,\mathcal{S}}$ be the set of all subconcepts of concepts that occur in $\mathcal{S}$ or Σ and define

$$\mathcal{A}^* = \{C(a) \mid C \in \mathcal{C}_{\Sigma,\mathcal{S}} \text{ and } \Sigma \models C(a)\} \cup \{r(a,b) \mid \Sigma \models r(a,b)\}.$$

We use $\mathcal{O}_\Sigma$ to denote the set of individual names that occur in Σ, and define the *witness set* $\mathcal{W} = \{w_C^r \mid r \text{ is a role name that occurs in } \Sigma \text{ and } C \in \mathcal{C}_{\Sigma,\mathcal{S}}\}$. Define $\mathcal{O}^* = \mathcal{O}_\Sigma \cup \mathcal{W}$. Given Σ and $\mathcal{C}_{\Sigma,\mathcal{S}}$, we outline a procedure that computes a tree called a *constraint tree over* Σ, see Lutz et al. (2001); Horrocks et al. (2006); Tao et al. (2012) for similar constructions. A constraint tree over Σ is a rooted tree $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ where $\mathbb{V}$ is a set of nodes, $k_0 \in \mathbb{V}$ is the root

[1] A technicality; $\mathcal{S}$ will be used in Section 4.4 in the context of secrecy-preserving reasoning.

$$
\begin{array}{ll}
\sqcap^{+} - \text{rule}: & \text{if } C(a),\ D(a) \in \mathbb{L}(i),\ C \sqcap D \in \mathcal{C}_{\Sigma,\mathcal{S}},\ \text{and} \\
& C \sqcap D(a)\ \notin \mathbb{L}(i),\ \text{then } \mathbb{L}(i) := \mathbb{L}(i) \cup \{C \sqcap D(a)\}; \\
\sqcap^{-} - \text{rule}: & \text{if } C \sqcap D(a) \in \mathbb{L}(i),\ \text{and } C(a) \notin \mathbb{L}(i) \text{ or } D(a)\ \notin \mathbb{L}(i), \\
& \text{then } \mathbb{L}(i) := \mathbb{L}(i) \cup \{C(a), D(a)\}; \\
\exists^{+} - \text{rule}: & \text{if } r(a,b),\ C(b) \in \mathbb{L}(i),\ \exists r.C \in \mathcal{C}_{\Sigma,\mathcal{S}} \text{ and} \\
& \exists r.C(a)\ \notin \mathbb{L}(i),\ \text{then } \mathbb{L}(i) := \mathbb{L}(i) \cup \{\exists r.C(a)\}; \\
\exists^{-} - \text{rule}: & \text{if } \exists r.C(a) \in \mathbb{L}(i),\ \text{and } \forall b \in \mathcal{O}^{*}, \{r(a,b), C(b)\} \not\subseteq \mathbb{L}(i), \\
& \text{then } \mathbb{L}(i) := \mathbb{L}(i) \cup \{r(a, w_C^r), C(w_C^r)\},\ \text{where } w_C^r \in \mathcal{W}; \\
\sqsubseteq\ -\text{rule}: & \text{if } C(a) \in \mathbb{L}(i),\ C \sqsubseteq D \in \mathcal{T},\ \text{and } D(a)\ \notin \mathbb{L}(i), \\
& \text{then } \mathbb{L}(i) := \mathbb{L}(i) \cup \{D(a)\}; \\
H - \text{rule}: & \text{if } r(a,b) \in \mathbb{L}(i),\ r \sqsubseteq s \in \mathcal{R},\ \text{and } s(a,b)\ \notin \mathbb{L}(i), \\
& \text{then } \mathbb{L}(i) := \mathbb{L}(i) \cup \{s(a,b)\};
\end{array}
$$

Figure 4.1 Local expansion rules

node of $\mathbb{T}$, $\mathbb{E}$ is a set of directed edges and $\mathbb{L}$ is a function that labels each node with a set of assertions which are obtained by applying the *expansion rules* specified below. The procedure builds $\mathbb{T}$ starting from the root node k_0 whose labeling set $\mathbb{L}(k_0)$ is initialized as the ABox $\mathcal{A}$. Further, $\mathbb{T}$ is grown by recursively applying the expansion rules in Figures 4.1 and 4.2. $\mathbb{T}$ is said to be *completed* if no expansion rule in Figures 4.1 or 4.2 is applicable to it. The procedure is designed to output a completed constraint tree $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ with $\mathbb{L}(k_0) = \mathcal{A}^{*}$. For the purpose of query answering, $\mathbb{T}$ is used as a "good approximation" of a (Kripke) model of the given KB, see Section 4.5.

In more detail, there are two kinds of expansion rules: (a) *local* expansion rules and (b) *global* expansion rules. Local expansion rules are given in Figure 4.1 and generate new assertions within a single labeling set. The $\sqcap^{-}$-rule decomposes conjunctions, and $\exists^{-}$-rule decomposes existential restriction assertions of the form $\exists r.C(a)$ by introducing a corresponding witness w_C^r from the set $\mathcal{W}$. The $\sqsubseteq$-rule derives new assertions based on the GCIs present in $\mathcal{T}$. To construct concept assertions whose associated concept expressions already belong to $\mathcal{C}_{\Sigma,\mathcal{S}}$, we use the $\sqcap^{+}$ and $\exists^{+}$-rules. Finally, the H-rule derives role assertions based on role inclusions in $\mathcal{R}$. The global expansion rules are given in Figure 4.2. The $\Diamond^{-}$-rule generates new nodes

$\Diamond^- -$ rule :	if there is a node i with $\Diamond C(a) \in \mathbb{L}(i)$ and
	i has no successor j with $C(a) \in \mathbb{L}(j)$,
	then add a new successor k of i with $\mathbb{L}(k) := \{C(a)\}$;
$\Diamond^+ -$ rule :	if there is a node i with $C(a) \in \mathbb{L}(i),\ \Diamond C \in \mathcal{C}_{\Sigma,\mathcal{S}}$
	and i has a parent j with $\Diamond C(a) \notin \mathbb{L}(j)$,
	then $\mathbb{L}(j) := \mathbb{L}(j) \cup \{\Diamond C(a)\}$.

Figure 4.2 Global expansion rules

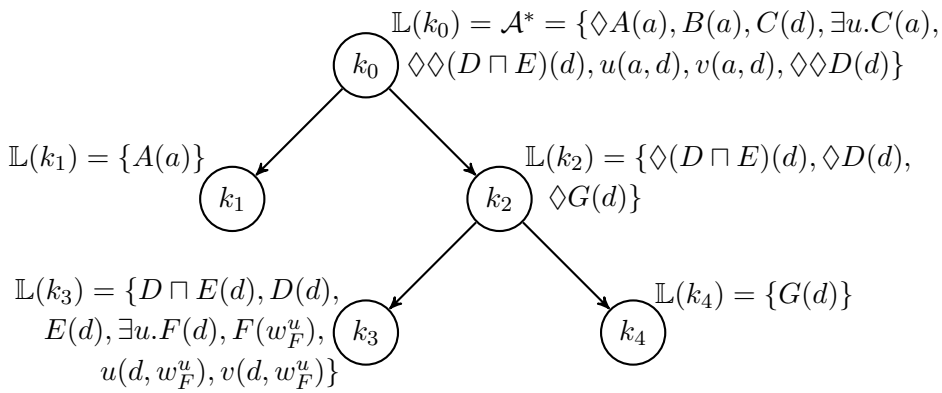

Figure 4.3 Completed constraint tree $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$

.

that are directly accessible from the current node. The $\Diamond^+$-rule adds a new $\Diamond$ assertion to the parent node from its current child node.

Example 4.3.1. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be a $\mathcal{ELH}^\Diamond$ KB, where $\mathcal{A} = \{\Diamond A(a), C(d), u(a,d)\}$, $\mathcal{T} = \{\Diamond A \sqsubseteq B, C \sqsubseteq \Diamond\Diamond(D \sqcap E), E \sqsubseteq \exists u.F, \Diamond D \sqsubseteq \Diamond G\}$, $\mathcal{R} = \{u \sqsubseteq v\}$ and $\mathcal{S} = \{\exists u.C,\ \Diamond\Diamond D\}$. Then, applying the rules in Figures 4.1 and 4.2 we compute the completed constraint tree $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ whose labeling sets are given in Figure 4.3.* $\square$

We will use the following notion of TBox acyclicity, called here $\Diamond$-*acyclicity*.

Definition 4.3.1. *A sequence $S_0, S_1 ...., S_n, ...$ of concept assertions over Σ, is called a $\Diamond$-sequence, if it satisfies the following conditions:*

- $S_0 = C_0(a_0),\ C_0 \in \mathcal{C}_{\Sigma,\mathbb{S}},\ a_0 \in \mathcal{O}^*$.

- *Given,* $S_n = C_n(a_n)$, *with* $C_n \in \mathcal{C}_{\Sigma,\mathbb{S}}$, $a_n \in \mathcal{O}^*$, *the next element in the sequence can be obtained as follows: Let* $\mathcal{B}_n$ *be the set of all assertions obtained by applying the local rules starting from* S_n. *Put* $\mathcal{D}_n = \mathcal{B}_n \cup \{S_n\}$.
  - *If* $\mathcal{D}_n$ *does not contain any* $\Diamond$*-assertions, then* S_n *is the last assertion of the sequence.*
  - *If* $\mathcal{D}_n$ *contains some* $\Diamond$*-assertions, then pick one, say* $\Diamond P(b)$, *and define* $S_{n+1} = C_{n+1}(a_{n+1}) = P(b)$.

The resulting $\Diamond$*-sequence is said to be non-repetitive, if for distinct* i, j, $C_i \neq C_j$.

Definition 4.3.2. *A TBox* $\mathcal{T}$ *is said to be* $\Diamond$*-acyclic (with respect to the rules given in Figures 4.1 and 4.2), if every* $\Diamond$*-sequence is non-repetitive.*

In this chapter, we assume that all TBoxes are $\Diamond$-acyclic as per Definition 4.3.2 (we shall omit the phrase "with respect to the rules"). We denote by Λ the *algorithm* which, given Σ and $\mathcal{C}_{\Sigma,\mathcal{S}}$, nondeterministically applies the expansion rules in Figures 4.1 and 4.2 until no further applications are possible. It is easy to see that for each node k in the constraint tree $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$, the size of $\mathbb{L}(k)$ is polynomial in $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathcal{S}} \mid$. An upper bound for the depth of $\mathbb{T}$ is given in the following claim which follows immediately from Definitions 4.3.1 and 4.3.2.

Claim 1. *The depth of* $\mathbb{T}$ *is* $O(\mid \mathcal{C}_{\Sigma,\mathcal{S}} \mid)$.

All executions of Λ terminate and by Claim 1, Λ builds a tree $\mathbb{T}$ whose depth is linear in $\mid \mathcal{C}_{\Sigma,\mathbb{S}} \mid$. Since the $\Diamond$-rule can in some cases be applied exponentially many times in $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathbb{S}} \mid$, $\mathbb{T}$ may have exponentially many nodes. For instance, consider a $\mathcal{ELH}^{\Diamond}$ KB $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, where $\mathcal{A} = \{A_1(a)\}$, $\mathcal{T} = \{A_i \sqsubseteq \Diamond \exists r.A_{i+1}, A_i \sqsubseteq \Diamond \exists s.A_{i+1}, 1 \leq i \leq n-1\}$ and $\mathcal{R} = \emptyset$. Clearly, TBox $\mathcal{T}$ is $\Diamond$-acyclic. To compute the constraint tree $\mathbb{T}$ for Σ, the global rules must be applied exponentially many times, implying that, the worst case the running time of Λ is exponential in $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathbb{S}} \mid$.

Before proving the correctness of Λ, we define the notion of interpretation of a constraint tree, see Lutz et al. (2001); Tao et al. (2012).

Definition 4.3.3. *Let $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ be a constraint tree over Σ, $\mathbb{M} = \langle S, \pi, \mathcal{E} \rangle$ a Kripke structure, and σ a mapping from $\mathbb{V}$ to S. We say that $\mathbb{M}$ satisfies $\mathbb{T}$ via σ if for all $k, k' \in \mathbb{V}$,*

- *$(k, k') \in \mathbb{E} \Rightarrow (\sigma(k), \sigma(k')) \in \mathcal{E}$, and*

- *$(\mathbb{M}, \sigma(k)) \models \mathbb{L}(k)$, i.e., $(\mathbb{M}, \sigma(k)) \models \alpha$ for every $\alpha \in \mathbb{L}(k)$*

We say that $\mathbb{M}$ satisfies $\mathbb{T}$, denoted as $\mathbb{M} \mid\models \mathbb{T}$, if there is a mapping σ such that $\mathbb{M}$ satisfies $\mathbb{T}$ via σ.

In the next lemma, we formulate the local soundness property of Λ. We say that f' is an extension of a function f if f' agrees with f on the domain of f.

Lemma 4.3.1. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be $\mathcal{ELH}^{\Diamond}$ KB with an $\Diamond$-acyclic TBox $\mathcal{T}$ and let $\mathbb{M} = \langle S, \pi, \mathcal{E} \rangle$ be a Kripke structure satisfying Σ. Also let $\mathbb{T}$ be a constraint tree over Σ, α a local or global expansion rule and $\mathbb{T}_\alpha$ a constraint tree obtained by applying α to $\mathbb{T}$. If $\mathbb{M}$ satisfies $\mathbb{T}$ via σ, then there exists a Kripke structure $\mathbb{M}' = \langle S, \pi', \mathcal{E} \rangle$ such that*

- *π' is an extension of π,*

- *$\mathbb{M}'$ satisfies $\mathbb{T}_\alpha$ via σ', where σ' is an extension of σ, and*

- *$\mathbb{M}'$ satisfies Σ.*

Proof. *(Outline). We present two cases to illustrate how $\mathbb{M}$ is transformed into $\mathbb{M}'$ by the applications of local and global extension rules; for more details see Tao et al. (2012). Assume the hypotheses and let α be the $\exists^-$-rule. Then, for some $k \in \mathbb{V}$, $\exists r.C(a) \in \mathbb{L}(k)$ in $\mathbb{T}$, and since $\mathbb{M}$ satisfies $\mathbb{T}$ via σ, we have $(\mathbb{M}, \sigma(k)) \models \exists r.C(a)$. By the semantics of existential restriction, there exists a $d \in \Delta$ such that $(a^{\pi(\sigma(k))}, d) \in r^{\pi(\sigma(k))}$ and $d \in C^{\pi(\sigma(k))}$. After applying the $\exists^-$-rule, $\mathbb{L}(k) := \mathbb{L}(k) \cup \{r(a, w_C^r), C(w_C^r)\}$. We have two cases: (1) If w_C^r occurs in $\mathbb{T}$ before the application of the $\exists^-$-rule to $\exists r.C(a)$, then $\mathbb{M}' = \mathbb{M}$; (2) If w_C^r does not occur in $\mathbb{T}$ before the application of the $\exists^-$-rule to $\exists r.C(a)$, then define the interpretation π' as π except for w_C^r: $(w_C^r)^{\pi'(\sigma(k))} = d$. The resulting constraint tree $\mathbb{T}_\alpha$ is satisfied by $\mathbb{M}' = \langle S, \pi', \mathcal{E} \rangle$ via σ. Since $\mathbb{M}$ satisfies Σ and π' is an extension of π, we conclude that $\mathbb{M}'$ satisfies Σ.*

Now let α be the $\Diamond^-$-rule. Then, for some $k \in \mathbb{V}$, $\Diamond C(a) \in \mathbb{L}(k)$ in $\mathbb{T}$ and k does not have a successor l with $C(a) \in \mathbb{L}(l)$. Since $\mathbb{M}$ satisfies $\mathbb{T}$ via σ, we have $(\mathbb{M}, \sigma(k)) \models \Diamond C(a)$. By the semantics of $\Diamond$, there exists a state $s \in S$ such that $(\sigma(k), s) \in \mathcal{E}$ and $a^{\pi(s)} \in C^{\pi(s)}$. After applying the $\Diamond^-$-rule, $\mathbb{L}(k') := \mathbb{L}(k') \cup \{C(a)\}$ where k' is a newly generated node, and (k, k') is the new edge. Now we extend σ to σ' by setting $\sigma'(k') = s$. The resulting constraint tree is $\mathbb{T}_\alpha = \langle \mathbb{V}_\alpha, k_0, \mathbb{E}_\alpha, \mathbb{L} \rangle$ where $\mathbb{V}_\alpha = \mathbb{V} \cup \{k'\}$ and $\mathbb{E}_\alpha = \mathbb{E} \cup \{(k, k')\}$. It follows that, $\mathbb{T}_\alpha$ is satisfied by $\mathbb{M}'$ via σ' where $\mathbb{M}' = \mathbb{M}$. Clearly, $\mathbb{M}'$ satisfies Σ. $\square$

Lemma 4.3.1 makes sure that each application of local and global rules preserves the model existence property. Next we define the *canonical Kripke structure* of a constraint tree.

Definition 4.3.4. *Let $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ be a completed constraint tree over Σ. The canonical Kripke structure $\mathbb{M}_\mathbb{T} = \langle S, \pi, \mathcal{E} \rangle$ for $\mathbb{T}$ is defined as follows:*

- $S = \mathbb{V}$,
- $\mathcal{E} = \mathbb{E}$,
- $\Delta = \mathcal{O}^* = \mathcal{O}_\Sigma \cup \mathcal{W}$,
- $a^{\pi(k)} = a$ *for all* $a \in \mathcal{O}^*$ *and each* $k \in \mathbb{V}$,
- $A^{\pi(k)} = \{a \in \mathcal{O}^* \mid A(a) \in \mathbb{L}(k)\}$, *for all* $A \in N_C \cap N_\Sigma$,
- $r^{\pi(k)} = \{(a, b) \in \mathcal{O}^* \times \mathcal{O}^* \mid r(a, b) \in \mathbb{L}(k)\}$, *for all* $r \in N_R \cap N_\Sigma$,

$\pi(k)$ is extended to compound concepts in the usual way (see Section 4.2).

The following lemma shows that $\mathbb{M}_\mathbb{T}$ satisfies the completed constraint tree $\mathbb{T}$.

Lemma 4.3.2. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be $\mathcal{ELH}^\Diamond$ KB with a $\Diamond$-acyclic TBox $\mathcal{T}$. Also let $\mathbb{T}$ be a completed constraint tree over Σ. Then $\mathbb{M}_\mathbb{T} \models\!\!\!\mid \mathbb{T}$.*

Proof. *Assume the hypotheses. To show that $\mathbb{M}_\mathbb{T} \models\!\!\!\mid \mathbb{T}$, let σ be the identity function. By Definition 4.3.4, for each k, $k' \in \mathbb{V}$, $(k, k') \in \mathbb{E} \Rightarrow (k, k') \in \mathcal{E}$. Now assume that $r(a, b) \in \mathbb{L}(k)$ where $r \in N_R$, a, $b \in \mathcal{O}^*$ and $k \in \mathbb{V}$. Again by Definition 4.3.4, $(a, b) \in r^{\pi(k)}$ implying*

$(\mathbb{M}_{\mathbb{T}}, k) \models r(a,b)$. *To show that* $C(a) \in \mathbb{L}(k)$ *implies* $(\mathbb{M}_{\mathbb{T}}, k) \models C(a)$, *we argue by induction on structure of* C. *When* $C \in N_C$, *the claim follows directly from Definition 4.3.4. Let* $k \in \mathbb{V}$ *and* $a \in \mathcal{O}^*$. *The case* $C = D \sqcap E$ *is easy and omitted.*

- $\exists r.D(a) \in \mathbb{L}(k)$. *There are two cases:*
  - $k = k_0$. *We have two subcases:*
    - *For some* $b \in \mathcal{O}_\Sigma$, $r(a,b)$, $D(b) \in \mathbb{L}(k)$. *By Definition 4.3.4,* $(\mathbb{M}_{\mathbb{T}}, k) \models r(a,b)$ *and by the induction hypothesis* $(\mathbb{M}_{\mathbb{T}}, k) \models D(b)$ *implying* $(\mathbb{M}_{\mathbb{T}}, k) \models \exists r.D(a) = C(a)$.
    - *For every* $b \in \mathcal{O}_\Sigma$, $\{r(a,b), D(b)\} \not\subseteq \mathbb{L}(k)$. *Since* $\mathbb{T}$ *is completed, by the* $\exists^-$*-rule,* $r(a, w_D^r)$, $D(w_D^r) \in \mathbb{L}(k)$. *By Definition 4.3.4,* $(\mathbb{M}_{\mathbb{T}}, k) \models r(a, w_D^r)$ *and by the induction hypothesis* $(\mathbb{M}_{\mathbb{T}}, k) \models D(w_D^r)$ *implying* $(\mathbb{M}_{\mathbb{T}}, k) \models \exists r.D(a) = C(a)$.
  - $k \neq k_0$. *Since* $\mathbb{T}$ *is completed, by the* $\exists^-$*-rule,* $r(a, w_D^r)$, $D(w_D^r) \in \mathbb{L}(k)$. *By Definition 4.3.4,* $(\mathbb{M}_{\mathbb{T}}, k) \models r(a, w_D^r)$ *and by the induction hypothesis* $(\mathbb{M}_{\mathbb{T}}, k) \models D(w_D^r)$ *implying* $(\mathbb{M}_{\mathbb{T}}, k) \models \exists r.D(a) = C(a)$.
- $\Diamond D(a) \in \mathbb{L}(k)$. *Since* $\mathbb{T}$ *is completed, by the* $\Diamond^-$*-rule, there exists* $k' \in \mathbb{V}$ *such that* $(k, k') \in \mathbb{E}$ *and* $D(a) \in \mathbb{L}(k')$. *By Definition 4.3.4,* $(k, k') \in \mathcal{E}$ *and by the induction hypothesis* $(\mathbb{M}_{\mathbb{T}}, k') \models D(a)$. *Hence* $(\mathbb{M}_{\mathbb{T}}, k) \models \Diamond D(a) = C(a)$. $\square$

Next we prove that $(\mathbb{M}_{\mathbb{T}}, k) \models \mathcal{T} \cup \mathcal{R}$, for each $k \in S$. We need the following auxiliary lemma.

Lemma 4.3.3. *For each* $C \in \mathcal{C}_{\Sigma,\mathcal{S}}$, *each* $a \in \mathcal{O}^*$ *and each* $k \in \mathbb{V}$, *if* $(\mathbb{M}_{\mathbb{T}}, k) \models C(a)$ *then* $C(a) \in \mathbb{L}(k)$.

Proof. *The proof is by induction on the structure of* C. *When* $C \in N_C$, *the claim follows directly from Definition 4.3.4. The case* $C = D \sqcap E$ *is easy and omitted.*

- $C = \exists r.D$. *By assumption,* $(\mathbb{M}_{\mathbb{T}}, k) \models \exists r.D(a)$. *This implies,* $(\mathbb{M}_{\mathbb{T}}, k) \models r(a,b)$ *and* $(\mathbb{M}_{\mathbb{T}}, k) \models D(b)$ *for some* $b \in \mathcal{O}^*$. *By Definition 4.3.4,* $r(a,b) \in \mathbb{L}(k)$ *and by induction hypothesis* $D(b) \in \mathbb{L}(k)$. *Since* $\mathbb{T}$ *is completed and* $\exists r.D \in \mathcal{C}_{\Sigma,\mathcal{S}}$, *by the* $\exists^+$*-rule,* $\exists r.D(a) = C(a) \in \mathbb{L}(k)$.

- $C = \Diamond D$. *By assumption,* $(\mathbb{M}_{\mathbb{T}}, k) \models \Diamond D(a)$. *Hence, for some* $k' \in \mathcal{E}(k)$, $(\mathbb{M}_{\mathbb{T}}, k') \models D(a)$. *By Definition 4.3.4,* $(k, k') \in \mathbb{E}$ *and by the induction hypothesis* $D(a) \in \mathbb{L}(k')$. *Since* $\mathbb{T}$ *is completed and* $\Diamond D \in \mathcal{C}_{\Sigma,\mathcal{S}}$, *by the* $\Diamond^+$*-rule,* $\Diamond D(a) = C(a) \in \mathbb{L}(k)$. □

Lemma 4.3.4. *For each* $k \in S$, $(\mathbb{M}_{\mathbb{T}}, k) \models \mathcal{T} \cup \mathcal{R}$.

Proof. *First we show that* $(\mathbb{M}_{\mathbb{T}}, k) \models \mathcal{T}$, *for each* $k \in S$. *Suppose that* $C \sqsubseteq D \in \mathcal{T}$ *and let* $a \in C^{\pi(k)}$. *This means that* $(\mathbb{M}_{\mathbb{T}}, k) \models C(a)$ *and by Lemma 4.3.3,* $C(a) \in \mathbb{L}(k)$. *Since* $\mathbb{T}$ *is completed, by the* $\sqsubseteq$*-rule,* $D(a) \in \mathbb{L}(k)$. *Since* $\mathbb{M}_{\mathbb{T}} \models \mathbb{T}$, *by Lemma 4.3.2,* $(\mathbb{M}_{\mathbb{T}}, k) \models D(a)$. *Therefore,* $(\mathbb{M}_{\mathbb{T}}, k) \models C \sqsubseteq D$. *Hence,* $(\mathbb{M}_{\mathbb{T}}, k) \models \mathcal{T}$.

Next, we show that $(\mathbb{M}_{\mathbb{T}}, k) \models \mathcal{R}$, *for each* $k \in S$. *Let* $r \sqsubseteq s \in \mathcal{R}$ *and assume that* $(\mathbb{M}_{\mathbb{T}}, k) \models r(a, b)$. *By Definition 4.3.4,* $r(a, b) \in \mathbb{L}(k)$. *Since* $\mathbb{T}$ *is completed, by the* H*-rule,* $s(a, b) \in \mathbb{L}(k)$. *Since* $\mathbb{M}_{\mathbb{T}} \models \mathbb{T}$, *by Lemma 4.3.2,* $(\mathbb{M}_{\mathbb{T}}, k) \models s(a, b)$. *Therefore,* $(\mathbb{M}_{\mathbb{T}}, k) \models r \sqsubseteq s$. *Hence* $(\mathbb{M}_{\mathbb{T}}, k) \models \mathcal{R}$. □

The following corollary is an immediate consequence of Lemmas 4.3.2 and 4.3.4.

Corollary 4.3.1. $\mathbb{M}_{\mathbb{T}}$ *satisfies* Σ.

Proof. *By Definitions 4.3.3 and 4.3.4 and Lemmas 4.3.2 and 4.3.4, we have that (1)* $(\mathbb{M}_{\mathbb{T}}, k_0) \models \Sigma$ *and (2) for each* $k \in \mathbb{V}$, $(\mathbb{M}_{\mathbb{T}}, k) \models \mathcal{T} \cup \mathcal{R}$. *Hence* $\mathbb{M}_{\mathbb{T}}$ *satisfies* Σ. □

The proof of the next theorem follows immediately from Definition 4.3.4 and Lemma 4.3.3. In a sense, this theorem captures the completeness property of the algorithm Λ.

Theorem 4.3.1. *Let* $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ *be a completed constraint tree over* Σ *and* $\mathbb{M}_{\mathbb{T}} = \langle S, \pi, \mathcal{E} \rangle$ *a canonical Kripke structure for* $\mathbb{T}$. *Then, for all* $k \in \mathbb{V}$, $C \in \mathcal{C}_{\Sigma,\mathcal{S}}$, $r \in N_\Sigma \cap N_R$, *and all* $a, b \in \mathcal{O}^*$

- $(\mathbb{M}_{\mathbb{T}}, k) \models r(a, b) \Rightarrow\ r(a, b) \in \mathbb{L}(k)$ *and*
- $(\mathbb{M}_{\mathbb{T}}, k) \models C(a) \Rightarrow\ C(a) \in \mathbb{L}(k)$.

Finally, the following is a consequence of Theorem 4.3.1 and Corollary 4.3.1.

Corollary 4.3.2. $\mathbb{L}(k_0) = \mathcal{A}^*$.

4.4 Secrecy-Preserving Reasoning in $\mathcal{ELH}^{\Diamond}$ KBs

Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be a $\mathcal{ELH}^{\Diamond}$ KB and $\mathbb{S} \subseteq \mathcal{A}^*$ the "secrecy set". Also let $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ be a completed constraint tree over Σ. Given Σ, $\mathbb{S}$ and $\mathbb{T}$, the objective is to answer assertion queries while preserving secrecy, i.e., answering queries so that assertions in $\mathbb{S}$ remain protected. Our approach is to compute a function E that assigns a finite set of assertions to each node in $\mathbb{T}$. E is called the *secrecy Envelope* for $\mathbb{S}$, so that protecting $E(i)$ for all $i \in \mathbb{V}$, the querying agent cannot logically infer any assertion in $\mathbb{S}$. The sets $E(i)$ for each $i \in \mathbb{V}$ are obtained by applying the *inverted expansion rules* given in Figures 4.4 and 4.5. The role of OWA in answering the queries is the following: When answering a query with "Unknown", the querying agent should not be able to distinguish between the case that the answer to the query is truly unknown to the KB reasoner and the case that the answer is being protected for reasons of secrecy. We envision a situation in which once the $\mathbb{T}$ is computed, a *reasoner* $\mathfrak{R}$ is associated with it, i.e., $\mathfrak{R}$ has unfettered access to $\mathbb{T}$. $\mathfrak{R}$ is designed to answer queries as follows: If a query cannot be inferred from Σ, the answer is "Unknown". If it can be inferred and it is not in $E(k_0)$, the answer is "Yes"; otherwise, the answer is "Unknown". We make the following assumptions about the capabilities of the querying agent:

(a) does not have direct access to ABox $\mathcal{A}$, but is aware of the underlying vocabulary of Σ,

(b) has full access to TBox $\mathcal{T}$ and RBox $\mathcal{R}$,

(c) can ask queries in the form of assertions, and

(d) is not aware of the witness set $\mathcal{W}$, by *hidden name assumptions* (HNA), for more details see Tao et al. (2010).

We formally define the notion of an envelope in the following:

Definition 4.4.1. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be a $\mathcal{ELH}^{\Diamond}$ KB, $\mathbb{S}$ a finite secrecy set and $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ a completed constraint tree. The secrecy envelope of $\mathbb{S}$ is a function E with domain $\mathbb{V}$ satisfying the following properties:*

- $\mathbb{S} \subseteq E(k_0)$,

Inv- $\sqcap^-$ −rule :	if $\{C(a),\ D(a)\} \cap E(i) \neq \emptyset$ and $C \sqcap D(a)\ \in \mathbb{L}(i) \setminus E(i)$, then $E(i) := E(i) \cup \{C \sqcap D(a)\}$;
Inv- $\sqcap^+$ −rule :	if $C \sqcap D(a) \in E(i),\ \{C(a),\ D(a)\} \subseteq \mathbb{L}(i) \setminus E(i)$ and $C \sqcap D \in \mathcal{C}_{\Sigma,\mathcal{S}}$, then $E(i) := E(i) \cup \{C(a)\}$ or $E(i) := E(i) \cup \{D(a)\}$;
Inv-$\exists^+$ − rule :	if $\exists r.C(a) \in E(i),\ \{r(a,b), C(b)\} \subseteq \mathbb{L}(i) \setminus E(i)$ with $b \in \mathcal{O}^*$ and $\exists r.C \in \mathcal{C}_{\Sigma,\mathcal{S}}$, then $E(i) := E(i) \cup \{r(a,b)\}$ or $E(i) := E(i) \cup \{C(b)\}$;
Inv- $\sqsubseteq$ −rule :	if $D(a) \in E(i),\ C \sqsubseteq D \in \mathcal{T}$, and $C(a)\ \in \mathbb{L}(i) \setminus E(i)$, then $E(i) := E(i) \cup \{C(a)\}$;
Inv-H − rule :	if $s(a,b) \in E(i),\ r \sqsubseteq s \in \mathcal{R}$, and $r(a,b)\ \in \mathbb{L}(i) \setminus E(i)$, then $E(i) := E(i) \cup \{r(a,b)\}$.

Figure 4.4 Inverted local expansion rules

- *for each* $i \in \mathbb{V}$, $E(i) \subseteq \mathbb{L}(i)$, *and*

- *for each* $i \in \mathbb{V}$, *each* $\alpha \in E(i)$, $\mathbb{L}(i) \setminus E(i) \not\models \alpha$.

The intuition for the above definition is that no information in $E(i)$ can be inferred from the set $\mathbb{L}(i) \setminus E(i)$ for each $i \in \mathbb{V}$. To compute an envelope, we use the idea of inverting the rules of Figures 1 and 2 (see Tao et al. (2010), where this approach was first utilized for membership assertions). Induced by the Local and Global expansion rules in Figures 4.1 and 4.2, we define the corresponding "inverted" Local and Global expansion rules in Figures 4.4 and 4.5, respectively. Note that the $\exists^-$-rule does not have its corresponding inverted rule. The reason for the omission is that an application of this rule results in adding assertions with individual names from the witness set. By HNA, the querying agent is barred from asking any queries that involve individual names from the witness set. Inverted expansion rules are denoted by prefixing Inv- to the name of the corresponding expansion rules.

From now on, we assume that $\mathbb{T}$ has been computed and is readily available for computing the envelope. The computation begins with the initialization step: The set $E(k_0)$ is initialized as $\mathbb{S}$, and $E(i)$ is initialized as $\emptyset$ for all $i \in \mathbb{V} \setminus \{k_0\}$. Next, the sets $E(k_0)$ and $E(i)$ for all $i \in \mathbb{V} \setminus \{k_0\}$ are expanded using the inverted expansion rules listed in Figures 4.4 and 4.5 until no further applications are possible. The resulting function E is said to be completed. We denote

Inv-$\Diamond^-$ – rule :	if there is a node j with $C(a) \in E(j)$ and $\Diamond C(a) \in \mathbb{L}(i) \setminus E(i)$ where i is the parent of j, then $E(i) := E(i) \cup \{\Diamond C(a)\}$;
Inv-$\Diamond^+$ – rule :	if there is a node i with $\Diamond C(a) \in E(i)$ and $C(a) \in \mathbb{L}(j) \setminus E(j)$ where j is a successor of i and $\Diamond C \in \mathcal{C}_{\Sigma,\mathcal{S}}$, then $E(j) := E(j) \cup \{C(a)\}$.

Figure 4.5 Inverted global expansion rule

by $\Lambda_{\mathbb{S}}$ the algorithm which computes the sets $E(i)$ for all $i \in \mathbb{V}$. Due to non-determinism in applying the rules Inv-$\sqcap^+$ and Inv-$\exists^+$, different executions of $\Lambda_{\mathbb{S}}$ may result different outputs. Since for each $i \in \mathbb{V}$, $\mathbb{L}(i)$ is finite, the computation of $\Lambda_{\mathbb{S}}$ terminates. Let the sets $E(i)$ for $i \in \mathbb{V}$ be an output of $\Lambda_{\mathbb{S}}$. Since the size of each $\mathbb{L}(i)$ is polynomial in $|\Sigma| + |\mathcal{C}_{\Sigma,\mathcal{S}}|$, and each application of inverted expansion rule moves an assertion from $\mathbb{L}(i)$ into $E(i)$, the size of $E(i)$ is at most the size of $\mathbb{L}(i)$. Since the size of $\mathbb{V}$ can be exponential, $\Lambda_{\mathbb{S}}$ may take exponential time to compute the sets $E(i)$. Define the *secrecy-preserving tree* (constraint) for the secrecy set $\mathbb{S}$ to be $\mathbb{T}_E = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L}_E \rangle$, where $\mathbb{L}_E(i) = \mathbb{L}(i) \setminus E(i)$ for all $i \in \mathbb{V}$.

Example 4.4.1. *(Example 4.3.1 cont.) Recall that* $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ *is the completed constraint tree. Let* $\mathbb{S} = \{B(a), \Diamond\Diamond D(d)\}$ *be the secrecy set. Then, by using rules in Figures 4.4 and 4.5 we compute the envelope for* $\mathbb{S}$*, and one of the corresponding secrecy-preserving trees is given in Figure 4.6:*

- $E(k_0) = \mathbb{S} \cup \{\Diamond A(a), C(d), \Diamond\Diamond(D \sqcap E)(d)\}$,
- $E(k_1) = \{A(a)\}$,
- $E(k_2) = \{\Diamond(D \sqcap E)(d), \Diamond D(d)\}$,
- $E(k_3) = \{D \sqcap E(d), D(d)\}$ and
- $E(k_4) = \emptyset$. $\square$

We use this secrecy-preserving tree for proving some properties of the envelopes and for answering queries. Before proving the main result on envelopes, we prove several auxiliary

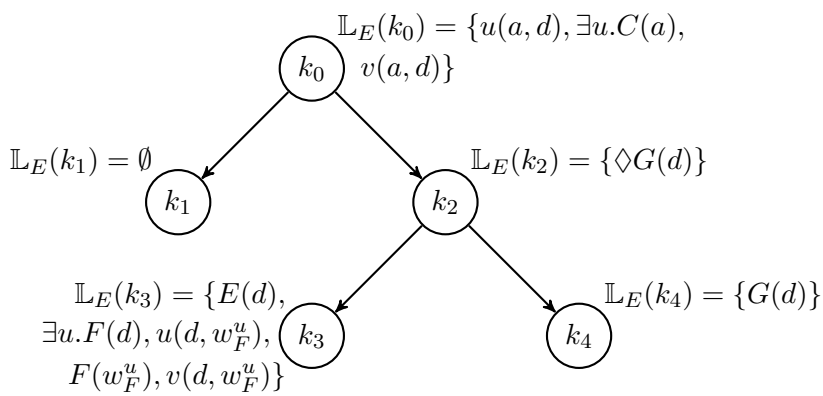

Figure 4.6 Secrecy-preserving tree $\mathbb{T}_E = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L}_E \rangle$

lemmas. First, we show that for each $i \in \mathbb{V}$, no assertions in $E(i)$ is "logically reachable" from the members of the set $\mathbb{L}_E(i)$.

Lemma 4.4.1. *Let the function E be completed by applying the inverted rules in Figures 4.4 and 4.5. Also, let $\mathbb{T}_E = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L}_E \rangle$ be a secrecy-preserving tree. Then, for each $i \in \mathbb{V}$, $\mathbb{L}_E(i)$ is completed.*

Proof. *Let i be any node in $\mathbb{V}$ and $j \in \mathbb{V}$ its successor. We have to show that no rule in Figures 4.1 or 4.2 is applicable to $\mathbb{L}_E(i) = \mathbb{L}(i) \setminus E(i)$. The proof is by contradiction according to cases: assuming that a rule in Figures 4.1 and 4.2 is applicable and showing that some inverse rule is applicable.*

- *If $\sqcap^-$-rule is applicable, then there is an assertion $C \sqcap D(a) \in \mathbb{L}_E(i)$ such that $C(a) \notin \mathbb{L}_E(i)$ or $D(a) \notin \mathbb{L}_E(i)$. Since $\mathbb{L}(i)$ is completed, $\{C(a), D(a)\} \subseteq \mathbb{L}(i)$. Hence, $\{C(a), D(a)\} \cap E(i) \neq \emptyset$. This makes the Inv-$\sqcap^-$-rule applicable.*

- *If $\sqcap^+$-rule is applicable, then there are assertions $C(a), D(a) \in \mathbb{L}_E(i)$ such that $C \sqcap D \in \mathcal{C}_{\Sigma,\mathcal{S}}$ and $C \sqcap D(a) \notin \mathbb{L}_E(i)$. Since $\mathbb{L}(i)$ is completed, $C \sqcap D(a) \in \mathbb{L}(i)$. Hence, $C \sqcap D(a) \in E(i)$. This makes the Inv-$\sqcap^+$-rule applicable.*

- *If $\exists^+$-rule is applicable, then there are assertions $r(a,b), C(b) \in \mathbb{L}_E(i)$ such that $\exists r.C \in \mathcal{C}_{\Sigma,\mathcal{S}}$ and $\exists r.C(a) \notin \mathbb{L}_E(i)$. Since $\mathbb{L}(i)$ is completed, $\exists r.C(a) \in \mathbb{L}(i)$. Hence, $\exists r.C(a) \in E(i)$. This makes the Inv-$\exists^+$-rule applicable.*

- *If $\sqsubseteq$-rule is applicable, then there is an assertion $C(a) \in \mathbb{L}_E(i)$ and a GCI $C \sqsubseteq D \in \mathcal{T}$ such that $D(a) \notin \mathbb{L}_E(i)$. Since $\mathbb{L}(i)$ is completed, $D(a) \in \mathbb{L}(i)$. Hence, $D(a) \in E(i)$. This makes the Inv-$\sqsubseteq$-rule applicable.*

- *If H-rule is applicable, then there is an assertion $r(a,b) \in \mathbb{L}_E(i)$ and a role inclusion $r \sqsubseteq s \in \mathcal{R}$ such that $s(a,b) \notin \mathbb{L}_E(i)$. Since $\mathbb{L}(i)$ is completed, $s(a,b) \in \mathbb{L}(i)$. Hence, $s(a,b) \in E(i)$. This makes the Inv-H-rule applicable.*

- *If $\Diamond^+$-rule is applicable, then there is an assertion $C(a) \in \mathbb{L}_E(j)$ such that $\Diamond C(a) \notin \mathbb{L}_E(i)$ where i is the predecessor of j. Since $\mathbb{L}(i)$ is completed, $\Diamond C(a) \in \mathbb{L}(i)$. Hence, $\Diamond C(a) \in E(i)$. This makes the Inv-$\Diamond^-$-rule applicable.*

- *If $\Diamond^-$-rule is applicable, then there is an assertion $\Diamond C(a) \in \mathbb{L}_E(i)$ such that i has no successor k with $C(a) \notin \mathbb{L}_E(k)$. Since $\mathbb{T}$ is a completed constraint tree, there is a node j which is a successor of i such that $C(a) \in \mathbb{L}(j)$. Hence, $C(a) \in E(j)$. This makes the Inv-$\Diamond^+$-rule applicable.* $\square$

Next we claim that the secrecy-preserving tree has similar properties as that of its completed constraint tree. The proof is similar to the proofs of the Lemmas 4.3.2, 4.3.3 and 4.3.4.

Lemma 4.4.2. *Let $\mathbb{T}_E = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L}_E \rangle$ be a secrecy-preserving tree obtained from the completed constraint tree $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ over Σ and the completed function E. Define the canonical Kripke structure $\mathbb{M}^E_{\mathbb{T}} = \langle S, \pi, \mathcal{E} \rangle$ for $\mathbb{T}_E$ as*

- $S = \mathbb{V}$,

- $\mathcal{E} = \mathbb{E}$,

- $\Delta = \mathcal{O}^* = \mathcal{O}_\Sigma \cup \mathcal{W}$,

- $a^{\pi(k)} = a$ *for all* $a \in \mathcal{O}^*$ *and each* $k \in \mathbb{V}$,

- $A^{\pi(k)} = \{a \in \mathcal{O}^* \mid A(a) \in \mathbb{L}_E(k)\}$, *for all* $A \in N_C$,

- $r^{\pi(k)} = \{(a,b) \in \mathcal{O}^* \times \mathcal{O}^* \mid r(a,b) \in \mathbb{L}_E(k)\}$, *for all* $r \in N_R$,

$\pi(k)$ is extended to compound concepts in the usual way (see Section 4.2). Then,

- $\mathbb{M}_{\mathbb{T}}^{E} \mid\models \mathbb{T}_E$,

- *For each* $C \in \mathcal{C}_{\Sigma,\mathcal{S}}$, *each* $a \in \mathcal{O}^*$ *and each* $k \in \mathbb{V}$, *if* $(\mathbb{M}_{\mathbb{T}}^{E}, k) \models C(a)$, *then* $C(a) \in \mathbb{L}_E(k)$ *and*

- *For each* $k \in S$, $(\mathbb{M}_{\mathbb{T}}^{E}, k) \models \mathcal{T} \cup \mathcal{R}$.

Finally, we show that a completed function E is in fact an envelope for the secrecy set $\mathbb{S}$, see Definition 4.4.1.

Theorem 4.4.1. *Let* $\mathbb{T} = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L} \rangle$ *be a completed constraint tree over* Σ. *Also, let* $\mathbb{T}_E = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L}_E \rangle$ *be a secrecy-preserving tree for the secrecy set* $\mathbb{S}$. *Then, the completed function* E *is an envelope for* $\mathbb{S}$.

Proof. *We have to show that the completed function* E *satisfies all three properties of Definition 4.4.1. Properties 1 and 2 are obvious. To prove property 3, suppose that for some* $i \in \mathbb{V}$, *some* $\alpha \in E(i)$, $\mathbb{L}_E(i) \models \alpha$.

Let $\mathbb{M}_{\mathbb{T}}^{E} = \langle S, \pi, \mathcal{E} \rangle$ *be the canonical Kripke structure for* $\mathbb{T}_E$. *By Lemma 4.4.2, for each* $i \in \mathbb{V}$, $(\mathbb{M}_{\mathbb{T}}^{E}, i) \models \mathbb{L}_E(i)$. *Again, by Lemma 4.4.2,* $\alpha \in \mathbb{L}_E(i)$. *This is a contradiction.*

4.5 Query Answering

Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be a $\mathcal{ELH}^{\Diamond}$ KB. We assume that the secrecy-preserving tree $\mathbb{T}_E = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L}_E \rangle$ has been precomputed and use $\mathbb{E}(k)$ to denote the set $\{k' \in \mathbb{V} \mid (k, k') \in \mathbb{E}\}$ of the successors of the node $k \in \mathbb{V}$. The reasoner $\mathfrak{R}$ answers queries based on the information in $\mathbb{T}_E$ and replies to a query q with "Yes" if $\Sigma \models q$ and $q \notin E(k_0)$; otherwise, the answer is "Unknown". Because of the syntactic restrictions of the language $\mathcal{ELH}^{\Diamond}$, $\mathfrak{R}$ does not answer "No" to any query.

Since the completed constraint tree $\mathbb{T}$ over Σ does not contain all the consequences of Σ, the completed secrecy-preserving tree $\mathbb{T}_E$ obtained from $\mathbb{T}$ does not contain all the information needed to answer queries. To address this problems we provide a procedure $\mathrm{Eval}(k, q)$ which works by recursively decomposing the compound queries all the way to the information available in $\mathbb{T}_E$. Initial call of this procedure is at the root node k_0 of $\mathbb{T}_E$. In lines 1 and 2 of Figure

```
Eval(k, q)
 1: case q ∈ L_E(k) = L(k) \ E(k)
 2:     return "Yes"
 3: case q = C ⊓ D(a)
 4:     if Eval(k, C(a)) = "Yes" and Eval(k, D(a)) = "Yes" then
 5:         return "Yes"
 6:     else
 7:         return "Unknown"
 8: case q = ∃r.C(a)
 9:     if for some d ∈ O* [ r(a, d) ∈ L_E(k) and
            Eval(k, C(d)) = "Yes"] then
10:         return "Yes"
11:     else
12:         return "Unknown"
13: case q = ◊C(a)
14:     if for some l ∈ E(k) [Eval(l, C(a)) = "Yes" ] then
15:         return "Yes"
16:     else
17:         return "Unknown"
```

Figure 4.7 Query answering algorithm for assertional queries

4.7, the reasoner checks the membership of q in $\mathbb{L}_E(k)$ and answers "Yes" if $q \in \mathbb{L}_E(k)$. From line 3 onwards we consider cases in which query q is broken into subqueries based on the constructors defined in $\mathcal{ELH}^{\Diamond}$ and apply the procedure recursively. The following theorem states the correctness claim of the algorithm.

Theorem 4.5.1. *Let $\Sigma = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be an $\mathcal{ELH}^{\Diamond}$ KB, $\mathbb{T}_E = \langle \mathbb{V}, k_0, \mathbb{E}, \mathbb{L}_E \rangle$ a completed secrecy-preserving tree and q a query. Then, for every $k \in \mathbb{V}$,*

- *Soundness: Eval(k, q) outputs "Yes" $\Rightarrow$ $\mathbb{L}_E(k) \models q$;*
- *Completeness: Eval(k, q) outputs "Unknown" $\Rightarrow$ $\mathbb{L}_E(k) \not\models q$.*

Proof. *We first consider the soundness part. Assume that Eval(k, q) = "Yes". Let $\mathbb{M} = \langle S, \pi, \mathcal{E} \rangle$ be a Kripke structure satisfying $\mathbb{L}_E(k)$. Since $\mathbb{T}_E$ is a completed constraint tree, by Lemma 4.3.1, $\mathbb{M} \models\!\models \mathbb{T}_E$, i.e., $\mathbb{M}$ satisfies $\mathbb{T}_E$ via some mapping $\sigma : \mathbb{V} \to S$. By Definition 4.3.3, $(\mathbb{M}, \sigma(k)) \models \mathbb{L}_E(k)$. Now, we prove the claim by induction on the structure of q. The inductive hypothesis is, for each $k \in \mathbb{V}$ and each assertion α, if Eval(k, α) = "Yes", then $(\mathbb{M}, \sigma(k)) \models \alpha$.*

The base case: If $q \in \mathbb{L}_E(k)$, then by Definition 4.3.3, $(\mathbb{M}, \sigma(k)) \models q$. Next, consider the case $q \notin \mathbb{L}_E(k)$.

- *$q = C \sqcap D(a)$. Then, Eval$(k, C(a))$ = Eval$(k, D(a))$ = "Yes" (by Lines 3 and 5 in Figure 4.7) and by inductive hypothesis, $(\mathbb{M}, \sigma(k)) \models C(a)$ and $(\mathbb{M}, \sigma(k)) \models D(a)$. Hence, $(\mathbb{M}, \sigma(k)) \models C \sqcap D(a)$.*

- *$q = \exists r.C(a)$. Then, for some $d \in \mathcal{O}^*$, $r(a, d) \in \mathbb{L}_E(k)$ and Eval$(k,\ C(d))$ = "Yes" (by Lines 8 and 9 in Figure 4.7). By Definition 4.3.3, $(\mathbb{M}, \sigma(k)) \models r(a, d)$ and by the inductive hypothesis $(\mathbb{M}, \sigma(k)) \models C(d)$. Hence, $(\mathbb{M}, \sigma(k)) \models \exists r.C(a)$.*

- *$q = \Diamond C(a)$. Then, for some $l \in \mathbb{E}(k)$, Eval$(l, C(a))$ = "Yes" (by Lines 13 and 14 in Figure 4.7). By Definition 4.3.3, $(\sigma(k), \sigma(l)) \in \mathcal{E}$ and by the inductive hypothesis $(\mathbb{M}, \sigma(l)) \models C(a)$. By the semantics of $\Diamond$, $(\mathbb{M}, \sigma(k)) \models \Diamond C(a)$.*

To prove the completeness part assume that $\mathbb{L}_E(k) \models q$. We have to show that Eval(k, q) = "Yes". Let $\mathbb{M}^E_{\mathbb{T}}$ be the canonical Kripke structure for $\mathbb{T}_E$ as defined in Section 4.4. By Lamma 4.4.2, $\mathbb{M}^E_{\mathbb{T}} \models \mathbb{T}_E$ and for all $k \in \mathbb{V}$, $(\mathbb{M}^E_{\mathbb{T}}, k) \models \mathcal{T} \cup \mathcal{R}$. Therefore $(\mathbb{M}^E_{\mathbb{T}}, k) \models \mathbb{L}_E(k)$ and hence, by the assumption, for every k, $(\mathbb{M}^E_{\mathbb{T}}, k) \models q$. We prove the claim by induction on the structure of q. The inductive hypothesis is, for each $k \in \mathbb{V}$ and each assertion α if $(\mathbb{M}^E_{\mathbb{T}}, k) \models \alpha$, then Eval$(k, \alpha)$ = "Yes". The base case: Let $q = C(a)$ where $C \in \mathcal{C}_{\Sigma,\mathcal{S}}$. Then, by Lemma 4.4.2, $C(a) \in \mathbb{L}_E(k)$. By Lines 1 and 2 in Figure 4.7, the claim follows immediately. Next, let $q = C(a)$ where $C \notin \mathcal{C}_{\Sigma,\mathcal{S}}$.

- *$q = C \sqcap D(a)$. To answer this query the algorithm computes Eval$(k, C(a))$ and Eval$(k, D(a))$. Now, the assumption $(\mathbb{M}^E_{\mathbb{T}}, k) \models C \sqcap D(a)$ implies $(\mathbb{M}^E_{\mathbb{T}},\ k) \models C(a)$ and $(\mathbb{M}^E_{\mathbb{T}}, k) \models D(a)$ which, by inductive hypothesis, implies that Eval$(k, C(a))$ = Eval$(k, D(a))$ = "Yes". Hence, by Lines 4 and 5 in Figure 4.7, Eval$(k, C \sqcap D(a))$ = "Yes".*

- *$q = \exists r.C(a)$. By the assumption, $(\mathbb{M}^E_{\mathbb{T}}, k) \models \exists r.C(a)$. This implies that, for some $d \in \mathcal{O}^*, (\mathbb{M}^E_{\mathbb{T}}, k) \models r(a, d)$ and $(\mathbb{M}^E_{\mathbb{T}}, k) \models C(d)$. By Theorem 4.3.1, $r(a, d) \in \mathbb{L}_E(k)$ and by the inductive hypothesis Eval$(k, C(d))$="Yes". Hence, by the Lines 9 and 10 in Figure 4.7, Eval$(k, \exists r.C(a))$= "Yes".*

- $q = \Diamond C(a)$. *Then,* $(\mathbb{M}^E_{\mathbb{T}}, k) \models \Diamond C(a)$. *This implies that, for some* $l \in \mathcal{E}(k)$, $(\mathbb{M}^E_{\mathbb{T}}, l) \models C(a)$. *By Definition 4.3.4,* $(k, l) \in \mathbb{E}$ *and therefore* $l \in \mathbb{E}(k)$. *By the inductive hypothesis Eval*$(l, C(a))$ = *"Yes". Hence, by the Lines 14 and 15 in Figure 4.7, Eval*$(k, \Diamond C(a))$= *"Yes".* □

Given an assertional query q, the algorithm given in Figure 4.7 checks for some assertions related to query q in the labeling sets of nodes along a particular path in $\mathbb{T}_E$. Since the size of each labeling set is bounded by $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathcal{S}} \mid$, by the Claim 1, this algorithm runs in time polynomial in $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathcal{S}} \mid$. Hence the assertional query answering can be done in polynomial time in the size of $\mid \Sigma \mid + \mid \mathcal{C}_{\Sigma,\mathcal{S}} \mid$.

Example 4.5.1. *(example 4.4.1 cont.) Recall that* $\mathbb{T}_E$ *is a secrecy-preserving tree. Suppose that the querying agent asks the assertional queries* $\exists u.C(a)$, $\Diamond\Diamond\exists v.F(d)$ *and* $\Diamond A(a)$. *Using the algorithm in Figure 4.7, we get the following answers:*

q	Eval(k, q)	Remarks
$\exists u.C(a)$	Yes	by Lines 1 and 2
$\Diamond\Diamond\exists v.F(d)$	Yes	by Lines 14, 15, 9 and 1
$\Diamond A(a)$	Unknown	by Lines 14 and 17

□

4.6 Conclusions

In this chapter we have studied the problem of secrecy-preserving query answering over $\mathcal{ELH}^{\Diamond}$ KBs. We have used the conceptual logic-based framework for secrecy-preserving reasoning which was introduced by Tao et al., see Tao et al. (2014), to a description logic $\mathcal{ELH}$ augmented with a modal operator $\Diamond$. The main contribution is in the way that we compute the consequences and preserve secrecy while answering queries. We break the process into two parts, the first one using the $\Diamond$-assertions in the KB, precomputes the rooted labeled tree $\mathbb{T}$ and the envelope E for the given secrecy set $\mathbb{S}$. For this we use two separate (but related) tableau procedures. In query answering step, given $\mathbb{T}$ and E, we define the secrecy-preserving tree $\mathbb{T}_E$. Once $\mathbb{T}_E$ has been computed, the query answering procedure is efficient and can be implemented in polynomial time.

CHAPTER 5. SUMMARY AND DISCUSSION

The main focus of this dissertation is to study the problem of SPQA with single querying agent in lightweight DLs $DL\text{-}Lite_{\mathcal{R}}$, and $\mathcal{ELH}$ and its extension $\mathcal{ELH}^{\Diamond}$. The main steps involved in the construction of SPQA system are (1) computing all or some of the consequences of the given KB by designing suitable tableau algorithms, (2) computing the envelopes for the given secrecy sets by designing tableau style procedures based on the idea of inverting the rules in tableau algorithms and (3) answering queries as informatively as possible without revealing the secrets in the secrecy sets. For each language, we used a different strategy to design procedures to compute the consequences of the KB, envelopes and query answering steps. We also considered different types of statements like assertional, GCI and BCQ in the secrecy sets. Important contributions in this work are listed in the following:

- In Chapter 2, we studied the problem of SPQA over acyclic $DL\text{-}Lite_{\mathcal{R}}$ KBs with BCQs in the secrecy set. A tableau algorithm was designed to compute consequences of a KB. Some rules in the tableau algorithm are able to compute negative assertions entailed by the KB. To show the tableau algorithm sound and complete, we used the semantics based on Kleene's 3-valued logic. We provided syntactic characterizations for entailment and disentailment of BCQs in terms of properties of mappings. In the envelope computation step, since BCQs are not the part of $DL\text{-}Lite_{\mathcal{R}}$ syntax, we designed two special rules to protect BCQs in the secrecy set. Graph matching technique was used to answer the BCQs.

- In Chapter 3, we considered the DL $\mathcal{ELH}$ to study the SPQA problem with assertions and GCIs as secrets. The important contribution in this work was the way in which we compute the consequences and preserve secrecy while answering queries. The idea is to

break the process into two parts, first one precomputes all the consequences for concepts and individuals that occur in the given KB. Two tableau algorithms were designed to compute the consequences. Based on the rules in these two algorithms, we designed two more tableau style algorithms to compute the corresponding envelopes. In the query answering step, the queries were parsed all the way to constituents that occur in the previously precomputed set of consequences. Then, the queries were answered based on the membership of the constituents of the query in $\mathcal{A}^* \setminus \mathbb{E}_{\mathcal{A}}$ and $\mathcal{T}^* \setminus \mathbb{E}_{\mathcal{T}}$. All the algorithms are efficient and can be implemented in polynomial time.

- In chapter 4, we studied the problem of secrecy-preserving query answering over $\mathcal{ELH}^{\Diamond}$ KBs. We used the conceptual logic-based framework for secrecy-preserving reasoning to a description logic $\mathcal{ELH}$ augmented with a modal operator $\Diamond$. In the first step, to compute a set of consequences of $\mathcal{ELH}^{\Diamond}$ KB, we designed a tableau algorithm to construct a rooted labeled tree $\mathbb{T}$ whose root node have the labeling set which contains the consequences of the KB. Given the computed tree $\mathbb{T}$ and the secrecy set $\mathbb{S}$, we next computed an envelope E for the secrecy set. In query answering step, given $\mathbb{T}$ and E, we defined the secrecy-preserving tree $\mathbb{T}_E$. Since $\mathbb{T}_E$ does not contain all the consequences of the given KB, the information available in $\mathbb{T}_E$ is not sufficient to answer queries correctly. To fix this issue, we designed an inductive algorithm to answer queries by breaking the queries all the way to constituents that are available in $\mathbb{T}_E$. Given $\mathbb{T}_E$, the query answering procedure is efficient and can be implemented in polynomial time.

We conclude this dissertation by mentioning some of the future work that we intend to pursue. We would like to study this SPQA problem in probabilistic DLs Lutz and Schroder (2010); Gutierrez-Basulto et al. (2012) and temporal DLs Gutierrez-Basulto et al. (2012). We also would like to extend SPQA problem that we studied in this dissertation to multiagents settings similar to Tao et al. (2014).

APPENDIX A. ADDITIONAL MATERIAL

A.1 Additional Material for Chapter 2

A.1.1 Proof of Lemma 2.3.1

Lemma 2.3.1 *(Soundness of Λ, Part A) Let $\mathcal{A}^*_{12}$ be a completed ABox obtained from Σ by first applying the rules listed in Figure 1 and then the rules of Figure 2. Then for every OW-model $\mathcal{I}$ of Σ, there is a OW-model $\mathcal{I}^*_{12}$ of Σ such that $\mathcal{I}^*_{12} \models \mathcal{A}^*_{12}$, where the domain of $\mathcal{I}^*_{12}$ is same as the domain of $\mathcal{I}$ and $\mathcal{I}^*_{12}$ remains same as $\mathcal{I}$ except for the interpretation of fresh individuals.*

Proof. Let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be an arbitrary OW-model of Σ. It suffices to show that after applying each expansion rule, there is an OW-model of Σ that satisfies the new assertions being added to $\mathcal{A}^*$ by that rule and that differs from (possibly) original interpretation only on how it interprets the new fresh individuals. The proof is by induction on the construction of $\mathcal{A}^*$. For the induction step, we use $\mathcal{A}'$ ($\mathcal{A}''$), $\mathcal{O}'$ ($\mathcal{O}''$) and $\mathcal{I}' = \left\langle \Delta, \cdot^{\mathcal{I}'} \right\rangle$ ($\mathcal{I}'' = \left\langle \Delta, \cdot^{\mathcal{I}''} \right\rangle$) to denote the ABox, the set of individual names appearing in the ABox, and the model of the ABox (and the TBox), respectively, before (after) the application of an expansion rule. The OW-model at the termination of this stage (when no more applications of rules in Figures 1 and 2 are possible) is denoted by $\mathcal{I}^*_{12}$. The base case is before any expansion rules have been applied in which case, $\mathcal{A}^* = \mathcal{A}$ and we can take $\mathcal{I}' = \mathcal{I}'' = \mathcal{I}$. There are seven cases (the number of rules in Figures 1 and 2, all rather simple).

- If $\sqsubseteq_{NL}$-rule is applicable, then $A(a) \in \mathcal{A}'$, $A \sqsubseteq L \in \mathcal{T}$ and $L(a) \in \mathcal{A}''$. By induction hypothesis, $\mathcal{I}' \models A(a)$, hence $a^{\mathcal{I}'} \in A^{\mathcal{I}'}_Y \subseteq L^{\mathcal{I}'}_Y$. Let $\mathcal{I}'' = \mathcal{I}'$. Then, $a^{\mathcal{I}''} = a^{\mathcal{I}'} \in L^{\mathcal{I}'}_Y = L^{\mathcal{I}''}_Y$. Hence, $\mathcal{I}'' \models L(a)$.

- If $\sqsubseteq_{N\exists}$-rule is applicable, we have two cases (i) $R = P$, and (ii) $R = P^-$. We consider

the latter case, the first one being easier. By assumption, $A(a) \in \mathcal{A}'$, $A \sqsubseteq \exists P^- \in \mathcal{T}$, $P(c,a) \in \mathcal{A}''$ where c is fresh, and $\mathcal{O}'' = \mathcal{O}' \cup \{c\}$. By induction hypothesis $\mathcal{I}' \models A(a)$, hence $a^{\mathcal{I}'} \in A_Y^{\mathcal{I}'} \subseteq (\exists P^-)_Y^{\mathcal{I}'}$. This implies $(d, a^{\mathcal{I}'}) \in P_Y^{\mathcal{I}'}$ for some $d \in \Delta$. Let $\mathcal{I}''$ be same as $\mathcal{I}'$ except redefine $c^{\mathcal{I}''} = d$. Then, $(c^{\mathcal{I}''}, a^{\mathcal{I}''}) = (d, a^{\mathcal{I}'}) \in P_Y^{\mathcal{I}'} = P_Y^{\mathcal{I}''}$. Hence, $\mathcal{I}'' \models P(c,a)$.

- If $\sqsubseteq_{\exists L}$-rule is applicable, we have two cases (i) $R = P$, and (ii) $R = P^-$. We argue the latter case. By assumption, $P(b,a) \in \mathcal{A}'$, $\exists P^- \sqsubseteq L \in \mathcal{T}$ and $L(a) \in \mathcal{A}''$. By induction hypothesis, $\mathcal{I}' \models P(b,a) \Rightarrow (b^{\mathcal{I}'}, a^{\mathcal{I}'}) \in P_Y^{\mathcal{I}'} \Rightarrow (a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in (P^-)_Y^{\mathcal{I}'} \Rightarrow a^{\mathcal{I}'} \in (\exists P^-)_Y^{\mathcal{I}'} \Rightarrow a^{\mathcal{I}'} \in L_Y^{\mathcal{I}'}$. Let $\mathcal{I}'' = \mathcal{I}'$. Then, $a^{\mathcal{I}''} = a^{\mathcal{I}'} \in L_Y^{\mathcal{I}'} = L_Y^{\mathcal{I}''}$. Hence, $\mathcal{I}'' \models L(a)$.

- If $\sqsubseteq_{\exists\exists}$-rule is applicable, we have four cases (i) $R = P, S = Q$, (ii) $R = P, S = Q^-$, (iii) $R = P^-, S = Q$ and (iv) $R = P^-, S = Q^-$. We consider the case (iv), the other cases can be argued similarly. By assumption, $P(b,a) \in \mathcal{A}'$, $\exists P^- \sqsubseteq \exists Q^- \in \mathcal{T}$ and $Q(c,a) \in \mathcal{A}''$, where c is fresh, and $\mathcal{O}'' = \mathcal{O}' \cup \{c\}$. By induction hypothesis, $\mathcal{I}' \models P(b,a)$, hence $(b^{\mathcal{I}'}, a^{\mathcal{I}'}) \in P_Y^{\mathcal{I}'}$. That is, $(a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in (P^-)_Y^{\mathcal{I}'} \Rightarrow a^{\mathcal{I}'} \in (\exists P^-)_Y^{\mathcal{I}'} \Rightarrow a^{\mathcal{I}'} \in (\exists Q^-)_Y^{\mathcal{I}'}$. This implies $(d, a^{\mathcal{I}'}) \in Q_Y^{\mathcal{I}'}$ for some $d \in \Delta$. Let $\mathcal{I}''$ be same as $\mathcal{I}'$ except redefine $c^{\mathcal{I}''} = d$. Then, $(c^{\mathcal{I}''}, a^{\mathcal{I}''}) = (d, a^{\mathcal{I}'}) \in Q_Y^{\mathcal{I}'} = Q_Y^{\mathcal{I}''}$. Hence, $\mathcal{I}'' \models Q(c,a)$.

- If $\sqsubseteq_{RE}$-rule is applicable, we have eight cases (i) $R = P, E = Q$, (ii) $R = P, E = Q^-$, (iii) $R = P^-, E = Q$, (iv) $R = P^-, E = Q^-$, (v) $R = P, E = \neg Q$, (vi) $R = P, E = \neg Q^-$, (vii) $R = P^-, E = \neg Q$ and (viii) $R = P^-, E = \neg Q^-$. We consider the case (viii), the other cases being easier. By assumption, $P(b,a) \in \mathcal{A}'$, $P^- \sqsubseteq \neg Q^- \in \mathcal{T}$ and $\neg Q(b,a) \in \mathcal{A}''$. By induction hypothesis, $\mathcal{I}' \models P(b,a)$, hence $(b^{\mathcal{I}'}, a^{\mathcal{I}'}) \in P_Y^{\mathcal{I}'}$. That is, $(a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in (P^-)_Y^{\mathcal{I}'} \Rightarrow (a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in (\neg Q^-)_Y^{\mathcal{I}'} \Rightarrow (a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in (Q^-)_N^{\mathcal{I}'}$. This implies $(b^{\mathcal{I}'}, a^{\mathcal{I}'}) \in Q_N^{\mathcal{I}'}$. Let $\mathcal{I}'' = \mathcal{I}'$. Then, $(b^{\mathcal{I}''}, a^{\mathcal{I}''}) = (b^{\mathcal{I}'}, a^{\mathcal{I}'}) \in Q_N^{\mathcal{I}'} = Q_N^{\mathcal{I}''}$. Hence, $\mathcal{I}'' \models \neg Q(b,a)$.

- If $\sqsubseteq_{N\nexists}$-rule is applicable, we have two cases (i) $R = P$, and (ii) $R = P^-$. We consider the latter case, the first one being easier. By assumption, $A(a) \in \mathcal{A}'$, $A \sqsubseteq \neg\exists P^- \in \mathcal{T}$, and $\neg P(c,a) \in \mathcal{A}''$ for every $c \in \mathcal{O}' = \mathcal{O}''$. By induction hypothesis, $\mathcal{I}' \models A(a)$, hence $a^{\mathcal{I}'} \in A_Y^{\mathcal{I}'} \subseteq (\neg\exists P^-)_Y^{\mathcal{I}'} = (\exists P^-)_N^{\mathcal{I}'}$. This implies $(d, a^{\mathcal{I}'}) \in P_N^{\mathcal{I}'}$ for every $d \in \Delta$. Let $\mathcal{I}''$ be

same as $\mathcal{I}'$. Then, for every $c \in \mathcal{O}''$, $(c^{\mathcal{I}''}, a^{\mathcal{I}''}) = (c^{\mathcal{I}'}, a^{\mathcal{I}'}) \in (\neg P)^{\mathcal{I}'}_Y = (\neg P)^{\mathcal{I}''}_Y$. Hence, for every $c \in \mathcal{O}''$, $\mathcal{I}'' \models \neg P(c, a)$.

- If $\sqsubseteq_{\exists\nexists}$-rule is applicable, we have four cases (i) $R = P, S = Q$, (ii) $R = P, S = Q^-$, (iii) $R = P^-, S = Q$ and (iv) $R = P^-, S = Q^-$. We consider the case (iv), the other cases can be argued similarly. By assumption, $P(b, a) \in \mathcal{A}'$, $\exists P^- \sqsubseteq \neg\exists Q^- \in \mathcal{T}$ and $\neg Q(c, a) \in \mathcal{A}''$, for every $c \in \mathcal{O}' = \mathcal{O}''$. By induction hypothesis, $\mathcal{I}' \models P(b, a)$, hence $(b^{\mathcal{I}'}, a^{\mathcal{I}'}) \in P^{\mathcal{I}'}_Y$. That is, $(a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in (P^-)^{\mathcal{I}'}_Y \Rightarrow a^{\mathcal{I}'} \in (\exists P^-)^{\mathcal{I}'}_Y \Rightarrow a^{\mathcal{I}'} \in (\neg\exists Q^-)^{\mathcal{I}'}_Y = (\exists Q^-)^{\mathcal{I}'}_N$. This implies $(d, a^{\mathcal{I}'}) \in Q^{\mathcal{I}'}_N$ for every $d \in \Delta$. Let $\mathcal{I}''$ be same as $\mathcal{I}'$. Then, for every $c \in \mathcal{O}''$, $(c^{\mathcal{I}''}, a^{\mathcal{I}''}) = (c^{\mathcal{I}'}, a^{\mathcal{I}'}) \in Q^{\mathcal{I}'}_N = (\neg Q)^{\mathcal{I}'}_Y = (\neg Q)^{\mathcal{I}''}_Y$. Hence, for every $c \in \mathcal{O}''$, $\mathcal{I}'' \models \neg Q(c, a)$.

□

A.1.2 Proof of Lemma 2.3.2

Lemma 2.3.2 *(Soundness of Λ, Part B) Let $\mathcal{A}^*$ be a completed ABox obtained from $\mathcal{A}^*_{12}$ by applying the rules listed in Figure 3. For any OW-model $\mathcal{I}$ of Σ, let $\mathcal{I}^* = \langle \Delta^*, \cdot^{\mathcal{I}^*} \rangle$ be an OW-interpretation as defined above. Then, $\mathcal{I}^*$ is an OW-model of Σ and $\mathcal{I}^* \models \mathcal{A}^*$.*

Proof. Let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be a OW-model of Σ. Then, by Lemma 2.3.1, $\mathcal{I}^*_{12}$ is an OW-model of $\langle \mathcal{A}^*_{12}, \mathcal{T} \rangle$ and by Corollary 2.3.1, $\mathcal{I}^* = \langle \Delta^*, \cdot^{\mathcal{I}^*} \rangle$ is also a OW-model of $\langle \mathcal{A}^*_{12}, \mathcal{T} \rangle$ and hence a model of Σ. To prove the entailment it suffices to show that after applying each expansion rule in Figure 3, $\mathcal{I}^*$ satisfies the new assertions being added to $\mathcal{A}^*$. The proof is by induction on the construction of $\mathcal{A}^*$. The base case is before any expansion rules in Figure 3 have been applied. In this case, $\mathcal{A}^* = \mathcal{A}^*_{12}$ and, by Corollary 2.3.1, $\mathcal{I}^*$ satisfies $\mathcal{A}^*$. For the induction step, we use $\mathcal{A}'$ ($\mathcal{A}''$) to denote the ABox, before (after) the application of an expansion rule.

- If $\sqsubseteq_{NL\neg}$-rule is applicable, then $\neg L(a) \in \mathcal{A}'$, $A \sqsubseteq L \in \mathcal{T}$ and $\neg A(a) \in \mathcal{A}''$. By induction hypothesis, $\mathcal{I}^* \models \neg L(a)$, hence $a^{\mathcal{I}^*} \in (\neg L)^{\mathcal{I}^*}_Y = L^{\mathcal{I}^*}_N \subseteq A^{\mathcal{I}^*}_N$. Hence, $\mathcal{I}^* \models \neg A(a)$.

- If $\sqsubseteq_{N\exists\neg}$-rule is applicable, we have two cases (i) $R = P$, and (ii) $R = P^-$. We consider the latter case, the first one being easier. By assumption, $\forall b \in \mathcal{O}^*$ $[\neg P(b, a) \in \mathcal{A}']$, $A \sqsubseteq \exists P \in \mathcal{T}$ and $\neg A(a) \in \mathcal{A}''$. By induction hypothesis, $\mathcal{I}^* \models \neg P(a, b)$, for every $b \in \mathcal{O}^*$

$\Rightarrow \forall d \in \Delta^* \ [(a^{\mathcal{I}^*}, d) \in (\neg P)^{\mathcal{I}^*}_Y = P^{\mathcal{I}^*}_N] \Rightarrow a^{\mathcal{I}^*} \in (\exists P)^{\mathcal{I}^*}_N$. Hence $a^{\mathcal{I}^*} \in (\exists P)^{\mathcal{I}^*}_N \subseteq A^{\mathcal{I}^*}_N$. Therefore, $\mathcal{I}^* \models \neg A(a)$.

- If $\sqsubseteq_{RE\neg}$-rule is applicable, we have eight cases (i) $R = P, E = Q$, (ii) $R = P, E = Q^-$, (iii) $R = P^-, E = Q$, (iv) $R = P^-, E = Q^-$, (v) $R = P, E = \neg Q$, (vi) $R = P, E = \neg Q^-$, (vii) $R = P^-, E = \neg Q$ and (viii) $R = P^-, E = \neg Q^-$. We consider the case (viii), the other cases being easier. By assumption, $Q(b, a) \in \mathcal{A}'$, $P^- \sqsubseteq \neg Q^- \in \mathcal{T}$ and $\neg P(b, a) \in \mathcal{A}''$. By induction hypothesis, $\mathcal{I}^* \models Q(b, a)$, hence $(b^{\mathcal{I}^*}, a^{\mathcal{I}^*}) \in Q^{\mathcal{I}^*}_Y$. That is, $(a^{\mathcal{I}^*}, b^{\mathcal{I}^*}) \in (Q^-)^{\mathcal{I}^*}_Y = (\neg Q^-)^{\mathcal{I}^*}_N \subseteq (P^-)^{\mathcal{I}^*}_N$ This implies $(b^{\mathcal{I}^*}, a^{\mathcal{I}^*}) \in P^{\mathcal{I}^*}_N$. Hence, $\mathcal{I}^* \models \neg P(b, a)$.

- If $\sqsubseteq_{N\sharp\neg}$-rule is applicable, we have two cases (i) $R = P$, and (ii) $R = P^-$. We argue the latter case. By assumption, $P(b, a) \in \mathcal{A}'$, $A \sqsubseteq \neg\exists P^- \in \mathcal{T}$ and $\neg A(a) \in \mathcal{A}''$. By induction hypothesis, $\mathcal{I}^* \models P(b, a) \Rightarrow (b^{\mathcal{I}^*}, a^{\mathcal{I}^*}) \in P^{\mathcal{I}^*}_Y \Rightarrow a^{\mathcal{I}^*} \in (\exists P^-)^{\mathcal{I}^*}_Y \Rightarrow a^{\mathcal{I}^*} \in A^{\mathcal{I}^*}_N$. Hence, $\mathcal{I}^* \models \neg A(a)$.

- If $\sqsubseteq_{\exists L\neg}$-rule is applicable, we have two cases (i) $R = P$, and (ii) $R = P^-$. We consider the latter case. By assumption, $\neg L(a) \in \mathcal{A}'$, $\exists P^- \sqsubseteq L \in \mathcal{T}$ and $\neg P(b, a) \in \mathcal{A}''$ for every $b \in \mathcal{O}^*$. By induction hypothesis, $\mathcal{I}^* \models \neg L(a)$, hence $a^{\mathcal{I}^*} \in (\neg L)^{\mathcal{I}^*}_Y = L^{\mathcal{I}^*}_N \subseteq (\exists P^-)^{\mathcal{I}^*}_N$. This implies, $(d, a^{\mathcal{I}^*}) \in P^{\mathcal{I}^*}_N$ for every $d \in \Delta^*$. Hence, for every $b \in \mathcal{O}^*$, $(b^{\mathcal{I}^*}, a^{\mathcal{I}^*}) \in (\neg P)^{\mathcal{I}^*}_Y$ and so for every $b \in \mathcal{O}^*$, $\mathcal{I}^* \models \neg P(b, a)$.

- If $\sqsubseteq_{\exists\exists\neg}$-rule is applicable, we have four cases (i) $R = P, S = Q$, (ii) $R = P, S = Q^-$, (iii) $R = P^-, S = Q$ and (iv) $R = P^-, S = Q^-$. We consider the case (iv), the other cases can be argued similarly. By assumption, $\forall b \in \mathcal{O}^* \ [\neg Q(b, a) \in \mathcal{A}']$, $\exists P^- \sqsubseteq \exists Q^- \in \mathcal{T}$ and $\neg P(c, a) \in \mathcal{A}''$, for some $c \in \mathcal{O}^*$. By induction hypothesis, $\mathcal{I}^* \models \neg Q(b, a)$, for every $b \in \mathcal{O}^* \Rightarrow \forall d \in \Delta^*[(d, a^{\mathcal{I}^*}) \in (\neg Q)^{\mathcal{I}^*}_Y = (Q)^{\mathcal{I}^*}_N] \Rightarrow a^{\mathcal{I}^*} \in (\exists Q^-)^{\mathcal{I}^*}_N$. Hence, $a^{\mathcal{I}^*} \in (\exists Q^-)^{\mathcal{I}^*}_N \subseteq (\exists P^-)^{\mathcal{I}^*}_N$. This implies, $(d, a^{\mathcal{I}^*}) \in (P)^{\mathcal{I}^*}_N$ for all $d \in \Delta^*$. Then, for every $c \in \mathcal{O}^*$, $(c^{\mathcal{I}^*}, a^{\mathcal{I}^*}) \in (\neg P)^{\mathcal{I}^*}_Y$. Hence, for every $c \in \mathcal{O}^*$, $\mathcal{I}^* \models \neg P(c, a)$.

- If $\sqsubseteq_{\exists\sharp\neg}$-rule is applicable, we have four cases (i) $R = P, S = Q$, (ii) $R = P, S = Q^-$, (iii) $R = P^-, S = Q$ and (iv) $R = P^-, S = Q^-$. We consider the case (iv), the other cases can be argued similarly. By assumption, $Q(b, a) \in \mathcal{A}'$, $\exists P^- \sqsubseteq \neg\exists Q^- \in \mathcal{T}$ and $\neg P(c, a) \in$

$\mathcal{A}''$, for every $c \in \mathcal{O}^*$. By induction hypothesis, $\mathcal{I}^* = Q(b,a) \Rightarrow (b^{\mathcal{I}^*}, a^{\mathcal{I}^*}) \in Q_Y^{\mathcal{I}^*} \Rightarrow (a^{\mathcal{I}^*}, b^{\mathcal{I}^*}) \in (Q^-)_Y^{\mathcal{I}^*} \Rightarrow a^{\mathcal{I}^*} \in (\exists Q^-)_Y^{\mathcal{I}^*}$. Hence, $a^{\mathcal{I}^*} \in (\exists Q^-)_Y^{\mathcal{I}^*} \subseteq (\exists P^-)_N^{\mathcal{I}^*}$. This implies $(d, a^{\mathcal{I}^*}) \in P_N^{\mathcal{I}^*}$ for every $d \in \Delta^*$. Then, for every $c \in \mathcal{O}^*$, $(c^{\mathcal{I}^*}, a^{\mathcal{I}^*}) \in P_N^{\mathcal{I}^*} = (\neg P)_Y^{\mathcal{I}^*}$. Hence, for every $c \in \mathcal{O}^*$, $\mathcal{I}^* \models \neg P(c,a)$.

□

A.1.3 Proof of Lemma 2.3.3

Lemma 2.3.3 *Let $\Sigma = \langle \mathcal{A}, \mathcal{T} \rangle$ be a DL-Lite$_\mathcal{R}$ KB. Then $\forall \alpha \in \mathcal{A} \cup \mathcal{T},\ \mathcal{J} \models \alpha$.*

Proof. First we consider an assertion $\alpha \in \mathcal{A}$. There are four cases:

- $\alpha = A(a) \in \mathcal{A}$. Then, $A(a) \in \mathcal{A}^* \Rightarrow a^{\mathcal{J}} = a \in A_Y^{\mathcal{J}} \Rightarrow \mathcal{J} \models A(a)$,
- $\alpha = \neg A(a) \in \mathcal{A}$. Then, $\neg A(a) \in \mathcal{A}^* \Rightarrow a^{\mathcal{J}} = a \in A_N^{\mathcal{J}} \Rightarrow \mathcal{J} \models \neg A(a)$,
- $\alpha = P(a,b) \in \mathcal{A}$. Then, $P(a,b) \in \mathcal{A}^* \Rightarrow (a^{\mathcal{J}}, b^{\mathcal{J}}) = (a,b) \in P_Y^{\mathcal{J}} \Rightarrow \mathcal{J} \models P(a,b)$ and
- $\alpha = \neg P(a,b) \in \mathcal{A}$. Then, $\neg P(a,b) \in \mathcal{A}^* \Rightarrow (a^{\mathcal{J}}, b^{\mathcal{J}}) = (a,b) \in P_N^{\mathcal{J}} \Rightarrow \mathcal{J} \models \neg P(a,b)$.

Next we consider subsumptions in $\mathcal{T}$. We recall that for each subsumption $E \sqsubseteq F \in \mathcal{T}$, we must show that $E_Y^{\mathcal{J}} \subseteq F_Y^{\mathcal{J}}$ and $F_N^{\mathcal{J}} \subseteq E_N^{\mathcal{J}}$. In the following we shall use, without mention, the fact that $\mathcal{A}^*$ is completed. We also recall that for any $B \in N_C$, $(\neg B)_Y^{\mathcal{J}} = B_N^{\mathcal{J}}$ and $(\neg B)_N^{\mathcal{J}} = B_Y^{\mathcal{J}}$. There are several cases.

- $\alpha = A \sqsubseteq L$ and let $D \in N_C$. There are two sub-cases.
  - case 1: $L = D$. Then, $a \in A_Y^{\mathcal{J}} \Rightarrow A(a) \in \mathcal{A}^* \Rightarrow D(a) \in \mathcal{A}^*(\sqsubseteq_{NL} - \text{rule}) \Rightarrow a \in D_Y^{\mathcal{J}}$. Similarly, $a \in D_N^{\mathcal{J}} \Rightarrow \neg D(a) \in \mathcal{A}^* \Rightarrow \neg A(a) \in \mathcal{A}^*(\sqsubseteq_{NL\neg} - \text{rule})$ $\Rightarrow a \in A_N^{\mathcal{J}}$.
  - case 2: $L = \neg D$. Then, $a \in A_Y^{\mathcal{J}} \Rightarrow A(a) \in \mathcal{A}^* \Rightarrow \neg D(a) \in \mathcal{A}^*(\sqsubseteq_{NL} - \text{rule}) \Rightarrow a \in D_N^{\mathcal{J}} \Rightarrow a \in (\neg D)_Y^{\mathcal{J}}$. Similarly, $a \in (\neg D)_N^{\mathcal{J}} \Rightarrow a \in D_Y^{\mathcal{J}} \Rightarrow D(a) \in \mathcal{A}^* \Rightarrow \neg A(a) \in \mathcal{A}^*(\sqsubseteq_{NL\neg} - \text{rule}) \Rightarrow a \in A_N^{\mathcal{J}}$.
- $\alpha = A \sqsubseteq \exists R$ and let $P \in N_R$. There are two sub-cases.

- case 1: $R = P$. Then, $a \in A_Y^{\mathcal{J}} \Rightarrow A(a) \in \mathcal{A}^* \Rightarrow P(a,b) \in \mathcal{A}^*(\sqsubseteq_{N\exists} - \text{rule})$, where b is fresh $\Rightarrow (a,b) \in P_Y^{\mathcal{J}} \Rightarrow a \in (\exists P)_Y^{\mathcal{J}}$. Similarly, $a \in (\exists P)_N^{\mathcal{J}}$
  $\Rightarrow \forall b \in \mathcal{O}^*, (a,b) \in P_N^{\mathcal{J}} \Rightarrow \forall b \in \mathcal{O}^*, \neg P(a,b) \in \mathcal{A}^* \Rightarrow \neg A(a) \in \mathcal{A}^*(\sqsubseteq_{N\exists\neg} - \text{rule}) \Rightarrow a \in A_N^{\mathcal{J}}$.
- case 2 : $R = P^-$. Then, $a \in A_Y^{\mathcal{J}} \Rightarrow A(a) \in \mathcal{A}^* \Rightarrow P(b,a) \in \mathcal{A}^*(\sqsubseteq_{N\exists} - \text{rule})$, where b is fresh $\Rightarrow (b,a) \in P_Y^{\mathcal{J}} \Rightarrow a \in (\exists P^-)_Y^{\mathcal{J}}$. Similarly, $a \in (\exists P^-)_N^{\mathcal{J}}$
  $\Rightarrow \forall b \in \mathcal{O}^*, (b,a) \in P_N^{\mathcal{J}} \Rightarrow \forall b \in \mathcal{O}^*, \neg P(b,a) \in \mathcal{A}^* \Rightarrow \neg A(a) \in \mathcal{A}^*(\sqsubseteq_{N\exists\neg} - \text{rule}) \Rightarrow a \in A_N^{\mathcal{J}}$.

- $\alpha = \exists R \sqsubseteq L$. There are four sub-cases $(R = P, L = A), (R = P^-, L = A), (R = P, L = \neg A)$ and $(R = P^-, L = \neg A)$, where $A \in N_C, P \in N_R$. We shall prove only two sub-cases, The other two sub-cases can be proved similarly.
  - case 1: $R = P$ and $L = A$. Then, $a \in (\exists P)_Y^{\mathcal{J}} \Rightarrow \exists b \in \mathcal{O}^*, (a,b) \in P_Y^{\mathcal{J}} \Rightarrow P(a,b) \in \mathcal{A}^* \Rightarrow A(a) \in \mathcal{A}^*(\sqsubseteq_{\exists L} - \text{rule}) \Rightarrow a \in A_Y^{\mathcal{J}}$. Similarly, $a \in A_N^{\mathcal{J}} \Rightarrow \neg A(a) \in \mathcal{A}^* \Rightarrow \forall b \in \mathcal{O}^*, \neg P(a,b) \in \mathcal{A}^*(\sqsubseteq_{\exists L\neg} - \text{rule}) \Rightarrow \forall b \in \mathcal{O}^*, (a,b) \in P_N^{\mathcal{J}} \Rightarrow a \in (\exists P)_N^{\mathcal{J}}$.
  - case 2: $R = P^-$ and $L = \neg A$. Then, $a \in (\exists P^-)_Y^{\mathcal{J}} \Rightarrow \exists b \in \mathcal{O}^*, (b,a) \in P_Y^{\mathcal{J}} \Rightarrow P(b,a) \in \mathcal{A}^* \Rightarrow \neg A(a) \in \mathcal{A}^*(\sqsubseteq_{\exists L} - \text{rule}) \Rightarrow a \in (\neg A)_Y^{\mathcal{J}}$. Similarly, $a \in (\neg A)_N^{\mathcal{J}} \Rightarrow a \in A_Y^{\mathcal{J}} \Rightarrow A(a) \in \mathcal{A}^* \Rightarrow \forall b \in \mathcal{O}^*, \neg P(b,a) \in \mathcal{A}^*(\sqsubseteq_{\exists L\neg} - \text{rule}) \Rightarrow \forall b \in \mathcal{O}^*, (b,a) \in P_N^{\mathcal{J}} \Rightarrow a \in (\exists P^-)_N^{\mathcal{J}}$.

- $\alpha = \exists R \sqsubseteq \exists S$. There are four sub-cases $(R = P, S = Q), (R = P^-, S = Q), (R = P, S = Q^-)$ and $(R = P^-, S = Q^-)$, where $P, Q \in N_R$. Here we shall prove only two sub-cases. The others can be proved similarly.
  - case 1: $R = P$ and $S = Q^-$. Then, $a \in (\exists P)_Y^{\mathcal{J}} \Rightarrow \exists b \in \mathcal{O}^*, (a,b) \in P_Y^{\mathcal{J}} \Rightarrow P(a,b) \in \mathcal{A}^* \Rightarrow Q(c,a) \in \mathcal{A}^*(\sqsubseteq_{\exists\exists} - \text{rule})$, c is fresh $\Rightarrow (c,a) \in Q_Y^{\mathcal{J}} \Rightarrow (a,c) \in (Q^-)_Y^{\mathcal{J}} \Rightarrow a \in (\exists Q^-)_Y^{\mathcal{J}}$. Similarly, $a \in (\exists Q^-)_N^{\mathcal{J}} \Rightarrow \forall b \in \mathcal{O}^*, (b,a) \in Q_N^{\mathcal{J}} \Rightarrow \forall b \in \mathcal{O}^*, \neg Q(b,a) \in \mathcal{A}^* \Rightarrow \forall c \in \mathcal{O}^*, \neg P(a,c) \in \mathcal{A}^*(\sqsubseteq_{\exists\exists\neg} - \text{rule}) \Rightarrow \forall c \in \mathcal{O}^*, (a,c) \in P_N^{\mathcal{J}} \Rightarrow a \in (\exists P)_N^{\mathcal{J}}$.
  - case 2: $R = P^-$ and $S = Q^-$. Then, $a \in (\exists P^-)_Y^{\mathcal{J}} \Rightarrow \exists b \in \mathcal{O}^*, (b,a) \in P_Y^{\mathcal{J}} \Rightarrow P(b,a) \in \mathcal{A}^* \Rightarrow Q(c,a) \in \mathcal{A}^*(\sqsubseteq_{\exists\exists} - \text{rule})$, c is fresh $\Rightarrow (c,a) \in Q_Y^{\mathcal{J}} \Rightarrow (a,c) \in$

$(Q^-)^{\mathcal{J}}_Y \Rightarrow a \in (\exists Q^-)^{\mathcal{J}}_Y$. Similarly, $a \in (\exists Q^-)^{\mathcal{J}}_N \Rightarrow \forall b \in \mathcal{O}^*, (b,a) \in Q^{\mathcal{J}}_N \Rightarrow \forall b \in \mathcal{O}^*, \neg Q(b,a) \in \mathcal{A}^* \Rightarrow \forall c \in \mathcal{O}^*, \neg P(c,a) \in \mathcal{A}^*$
$(\sqsubseteq_{\exists\exists\neg} - \text{rule}) \Rightarrow \forall c \in \mathcal{O}^*, (c,a) \in P^{\mathcal{J}}_N \Rightarrow a \in (\exists P^-)^{\mathcal{J}}_N$.

- $\alpha = R \sqsubseteq E$. There are eight sub-cases $(R = P, E = Q), (R = P^-, E = Q), (R = P, E = Q^-), (R = P^-, E = Q^-), (R = P, E = \neg Q), (R = P^-, E = \neg Q), (R = P, E = \neg Q^-)$ and $(R = P^-, E = \neg Q^-)$, where $P, Q \in N_R$. We argue only three of these.

  - case 1: $R = P$ and $E = Q^-$. Then, $(a,b) \in P^{\mathcal{J}}_Y \Rightarrow P(a,b) \in \mathcal{A}^* \Rightarrow Q(b,a) \in \mathcal{A}^*(\sqsubseteq_{RE} - \text{rule}) \Rightarrow (b,a) \in Q^{\mathcal{J}}_Y \Rightarrow (a,b) \in (Q^-)^{\mathcal{J}}_Y$. Similarly, $(a,b) \in (Q^-)^{\mathcal{J}}_N \Rightarrow (b,a) \in Q^{\mathcal{J}}_N \Rightarrow \neg Q(b,a) \in \mathcal{A}^* \Rightarrow \neg P(a,b) \in \mathcal{A}^*(\sqsubseteq_{RE\neg} - \text{rule}) \Rightarrow (a,b) \in P^{\mathcal{J}}_N$.

  - case 2: $R = P^-$ and $E = \neg Q$. Then, $(a,b) \in (P^-)^{\mathcal{J}}_Y \Rightarrow (b,a) \in P^{\mathcal{J}}_Y \Rightarrow P(b,a) \in \mathcal{A}^* \Rightarrow \neg Q(a,b) \in \mathcal{A}^*(\sqsubseteq_{RE} - \text{rule}) \Rightarrow (a,b) \in Q^{\mathcal{J}}_N \Rightarrow (a,b) \in$
    $(\neg Q)^{\mathcal{J}}_Y$. Similarly, $(a,b) \in (\neg Q)^{\mathcal{J}}_N \Rightarrow (a,b) \in Q^{\mathcal{J}}_Y \Rightarrow Q(a,b) \in \mathcal{A}^* \Rightarrow \neg P(b,a) \in \mathcal{A}^*(\sqsubseteq_{RE\neg} - \text{rule}) \Rightarrow (b,a) \in P^{\mathcal{J}}_N \Rightarrow (a,b) \in (P^-)^{\mathcal{J}}_N$.

  - case 3: $R = \neg P^-$ and $E = \neg Q^-$. Then, $(a,b) \in (\neg P^-)^{\mathcal{J}}_Y \Rightarrow (a,b) \in (P^-)^{\mathcal{J}}_N \Rightarrow (b,a) \in P^{\mathcal{J}}_N \Rightarrow \neg P(b,a) \in \mathcal{A}^* \Rightarrow \neg Q(b,a) \in \mathcal{A}^*(\sqsubseteq_{RE} - \text{rule}) \Rightarrow (b,a) \in Q^{\mathcal{J}}_N \Rightarrow (a,b) \in (Q^-)^{\mathcal{J}}_N \Rightarrow (a,b) \in (\neg Q^-)^{\mathcal{J}}_Y$. Similarly, $(a,b) \in (\neg Q^-)^{\mathcal{J}}_N \Rightarrow (a,b) \in (Q^-)^{\mathcal{J}}_Y \Rightarrow (b,a) \in Q^{\mathcal{J}}_Y \Rightarrow Q(b,a) \in \mathcal{A}^* \Rightarrow P(b,a) \in \mathcal{A}^*(\sqsubseteq_{RE\neg} - \text{rule}) \Rightarrow (b,a) \in P^{\mathcal{J}}_Y \Rightarrow (a,b) \in (P^-)^{\mathcal{J}}_Y \Rightarrow (a,b) \in (\neg P^-)^{\mathcal{J}}_N$.

- $\alpha = A \sqsubseteq \neg\exists R$. There are two sub-cases.

  - case 1: $R = P$, where $P \in N_R$. Then, $a \in A^{\mathcal{J}}_Y \Rightarrow A(a) \in \mathcal{A}^* \Rightarrow \forall b \in \mathcal{O}^*, \neg P(a,b) \in \mathcal{A}^*(\sqsubseteq_{N\nexists} - \text{rule}) \Rightarrow \forall b \in \mathcal{O}^*, (a,b) \in P^{\mathcal{J}}_N \Rightarrow a \in (\exists P)^{\mathcal{J}}_N \Rightarrow a \in (\neg\exists P)^{\mathcal{J}}_Y$. Similarly, $a \in (\neg\exists P)^{\mathcal{J}}_N \Rightarrow a \in (\exists P)^{\mathcal{J}}_Y \Rightarrow \exists b \in \mathcal{O}^*, (a,b) \in P^{\mathcal{J}}_Y \Rightarrow P(a,b) \in \mathcal{A}^* \Rightarrow \neg A(a) \in \mathcal{A}^*(\sqsubseteq_{N\nexists\neg} - \text{rule}) \Rightarrow a \in A^{\mathcal{J}}_N$.

  - case 2 : $R = P^-$. Then, $a \in A^{\mathcal{J}}_Y \Rightarrow A(a) \in \mathcal{A}^* \Rightarrow \forall b \in \mathcal{O}^*, \neg P(b,a) \in \mathcal{A}^*(\sqsubseteq_{N\nexists} - \text{rule}) \Rightarrow \forall b \in \mathcal{O}^*, (b,a) \in P^{\mathcal{J}}_N \Rightarrow a \in (\exists P^-)^{\mathcal{J}}_N \Rightarrow a \in (\neg\exists P^-)^{\mathcal{J}}_Y$. Similarly, $a \in (\neg\exists P^-)^{\mathcal{J}}_N \Rightarrow a \in (\exists P^-)^{\mathcal{J}}_Y \Rightarrow \exists b \in \mathcal{O}^*, (b,a) \in P^{\mathcal{J}}_Y \Rightarrow P(b,a)$
    $\in \mathcal{A}^* \Rightarrow \neg A(a) \in \mathcal{A}^*(\sqsubseteq_{N\nexists\neg} - \text{rule}) \Rightarrow a \in A^{\mathcal{J}}_N$.

- $\alpha = \exists R \sqsubseteq \neg\exists S$. There are four sub-cases $(R = P, S = Q), (R = P^-, S = Q), (R = P, S = Q^-)$ and $(R = P^-, S = Q^-)$, where $P, Q \in N_R$. We shall prove only two sub-cases; other two sub-cases can be proved similarly.

  - case 1: $R = P$ and $S = Q^-$. Then, $a \in (\exists P)^{\mathcal{J}}_Y \Rightarrow \exists b \in \mathcal{O}^*, (a, b) \in P^{\mathcal{J}}_Y \Rightarrow P(a, b) \in \mathcal{A}^* \Rightarrow \forall c \in \mathcal{O}^*, \neg Q(c, a) \in \mathcal{A}^*(\sqsubseteq_{\exists\nexists} - \text{rule}) \Rightarrow \forall c \in \mathcal{O}^*, (c, a) \in Q^{\mathcal{J}}_N \Rightarrow a \in (\exists Q^-)^{\mathcal{J}}_N \Rightarrow a \in (\neg\exists Q^-)^{\mathcal{J}}_Y$. Similarly, $a \in (\neg\exists Q^-)^{\mathcal{J}}_N \Rightarrow a \in (\exists Q^-)^{\mathcal{J}}_Y \Rightarrow \exists b \in \mathcal{O}^*, (b, a) \in Q^{\mathcal{J}}_Y \Rightarrow Q(b, a) \in \mathcal{A}^* \Rightarrow \forall c \in \mathcal{O}^*, \neg P(a, c) \in \mathcal{A}^*(\sqsubseteq_{\exists\nexists\neg} - \text{rule}) \Rightarrow \forall c \in \mathcal{O}^*, (a, c) \in P^{\mathcal{J}}_N \Rightarrow a \in (\exists P)^{\mathcal{J}}_N$.

  - case 2: $R = P^-$ and $S = Q^-$. Then, $a \in (\exists P^-)^{\mathcal{J}}_Y \Rightarrow \exists b \in \mathcal{O}^*, (b, a) \in P^{\mathcal{J}}_Y \Rightarrow P(b, a) \in \mathcal{A}^* \Rightarrow \forall c \in \mathcal{O}^*, \neg Q(c, a) \in \mathcal{A}^*(\sqsubseteq_{\exists\nexists} - \text{rule}) \Rightarrow \forall c \in \mathcal{O}^*,$ $(c, a) \in Q^{\mathcal{J}}_N \Rightarrow a \in (\exists Q^-)^{\mathcal{J}}_N \Rightarrow a \in (\neg\exists Q^-)^{\mathcal{J}}_Y$. Similarly, $a \in (\neg\exists Q^-)^{\mathcal{J}}_N \Rightarrow a \in (\exists Q^-)^{\mathcal{J}}_Y \Rightarrow \exists b \in \mathcal{O}^*, (b, a) \in Q^{\mathcal{J}}_Y \Rightarrow Q(b, a) \in \mathcal{A}^* \Rightarrow \forall c \in \mathcal{O}^*, \neg P(c, a)$ $\in \mathcal{A}^*(\sqsubseteq_{\exists\nexists\neg} - \text{rule}) \Rightarrow \forall c \in \mathcal{O}^*, (c, a) \in P^{\mathcal{J}}_N \Rightarrow a \in (\exists P^-)^{\mathcal{J}}_N$.

$\square$

BIBLIOGRAPHY

Acciarri, A., Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Palmieri, M., and Rosati, R. (2005). Quonto: querying ontologies. In *AAAI*, volume 5, pages 1670–1671.

Artale, A., Calvanese, D., Kontchakov, R., and Zakharyaschev, M. (2009). The dl-lite family and relations. *Journal of artificial intelligence research*, 36(1):1–69.

Avron, A. (1991). Natural 3-valued logicscharacterization and proof theory. *The Journal of Symbolic Logic*, 56(01):276–294.

Baader, F. (2003). *The description logic handbook: Theory, implementation and applications.* Cambridge university press.

Baader, F., Lutz, C., and Suntisrivaraporn, B. (2006). Efficient reasoning in $\mathcal{EL}^+$. In *Proccedings 2006 International Workshop on Description Logics*, volume 12, pages 15–26.

Bao, J., Slutzki, G., and Honavar, V. (2007). Privacy-preserving reasoning on the semantic web. In *Web Intelligence, IEEE/WIC/ACM Conference,791–797.*

Barth, A. and Mitchell, J. C. (2005). Enterprise privacy promises and enforcement. In *Proceedings of the 2005 workshop on Issues in the theory of security*, pages 58–66. ACM.

Bell, D. E. and LaPadula, L. J. (1973). Secure computer systems: Mathematical foundations. Technical report, DTIC Document.

Bienvenu, M., Ortiz, M., Šimkus, M., and Xiao, G. (2013). Tractable queries for lightweight description logics. In *Proceedings of the Twenty-Third UJCAI,768–774.*

Biskup, J. and Tadros, C. (2012). Revising belief without revealing secrets. In *Foundations of Information and Knowledge Systems*, pages 51–70. Springer.

Biskup, J., Tadros, C., and Wiese, L. (2010). Towards controlled query evaluation for incomplete first-order databases. In *Foundations of Information and Knowledge Systems*, pages 230–247. Springer.

Biskup, J. and Weibert, T. (2008). Keeping secrets in incomplete databases. *International Journal of Information Security*, 7(3):199–217.

Blackburn, P., De Rijke, M., and Venema, Y. (2002). *Modal Logic*, volume 53. Cambridge University Press.

Brachman, R. J., Levesque, H. J., and Reiter, R. (1992). *Knowledge representation*. MIT press.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., and Rosati, R. (2007). Tractable reasoning and efficient query answering in description logics: The dl-lite family. *Journal of Automated reasoning*, 39(3):385–429.

Delaitre, V. and Kazakov, Y. (2009). Classifying $\mathcal{ELH}$ ontologies in **SQL** databases. In *Proceedings of OWL: Experiences and Directions (OWLED 2009)*.

Dyer, M. and Greenhill, C. (2000). The complexity of counting graph homomorphisms. *Random Structures and Algorithms*, 17(3-4):260–289.

Fagin, R., Moses, Y., Vardi, M. Y., and Halpern, J. Y. (2003). *Reasoning about knowledge*. MIT press.

Fitting, M. (1985). A Kripke-Kleene semantics for logic programs*. *The Journal of Logic Programming*, 2(4):295–312.

Grau, B. C., Kharlamov, E., Kostylev, E. V., and Zheleznyakov, D. (2013). Controlled query evaluation over OWL 2 RL ontologies. In *The Semantic Web–ISWC 2013*, pages 49–65. Springer.

Gutierrez-Basulto, V., Jung, J. C., and Lutz, C. (2012). Complexity of branching temporal description logics. In *ECAI*, pages 390–395.

Halpern, J. Y. and O'Neill, K. R. (2005). Anonymity and information hiding in multiagent systems. *Journal of Computer Security*, 13(3):483–514.

Hemaspaandra, E. (2000). The complexity of poor mans logic. In *STACS 2000*, pages 230–241. Springer.

Hitzler, P., Krotzsch, M., and Rudolph, S. (2009). *Foundations of semantic web technologies*. CRC Press.

Horrocks, I., Hustadt, U., Sattler, U., and Schmidt, R. (2006). Computational modal logic. *Handbook of modal logic*, 3:181–245.

Jafari, M., Fong, P. W., Safavi-Naini, R., Barker, K., and Sheppard, N. P. (2011). Towards defining semantic foundations for purpose-based privacy policies. In *Proceedings of the first ACM conference on Data and application security and privacy*, pages 213–224. ACM.

Kagal, L., Finin, T., and Joshi, A. (2003). A policy based approach to security for the semantic web. In *International Semantic Web Conference*, volume 2870, pages 402–418. Springer.

Kazakov, Y., Krötzsch, M., and Simančík, F. (2014). The incredible elk. *JAR*, 53(1):1–61.

Kripke, S. A. (1963). Semantical analysis of modal logic 1 normal modal propositional calculi. *Mathematical Logic Quarterly*, 9(5-6):67–96.

Krishnasamy Sivaprakasam, G. and Slutzki, G. (2016). Secrecy-preserving query answering in $\mathcal{ELH}$ knowledge bases. In *Proceedings of 8th International Conference on Agents and Artificial Intelligence*.

Krötzsch, M. (2012). Owl 2 profiles: An introduction to lightweight ontology languages. In *Proceedings of the 8th Reasoning web summer school*, pages 112–183. Springer.

Lutz, C. and Schroder, L. (2010). Probabilistic description logics for subjective uncertainty. In *Proceedings of the 12th International Conference on Principles of Knowledge Representation and Reasoning*, pages 393–403.

Lutz, C., Sturm, H., Wolter, F., and Zakharyaschev, M. (2001). Tableaux for temporal description logic with constant domains. In *Automated Reasoning*, pages 121–136. Springer.

Lutz, C., Toman, D., and Wolter, F. (2008). Conjunctive query answering in $\mathcal{EL}$ using a database system. In *Proceedings of the 5th International Workshop on OWL: Experiences and Directions (OWLED 2008)*.

Mei, J., Liu, S., Xie, G., Kalyanpur, A., Fokoue, A., Ni, Y., Li, H., and Pan, Y. (2009). A practical approach for scalable conjunctive query answering on acyclic $\mathcal{EL}^+$ knowledge base. In *The Semantic Web-ISWC 2009*, pages 408–423. Springer.

Mendez, J. and Suntisrivaraporn, B. (2009). Reintroducing CEL as an OWL 2 $\mathcal{EL}$ reasoner. In *Proceedings of 22nd International Workshop on Description Logics*.

Ortiz, M. and Šimkus, M. (2012). Reasoning and query answering in description logics. In *Proceedings of the 8th Reasoning web summer school*, pages 1–53. Springer.

Sicherman, G. L., De Jonge, W., and Van de Riet, R. P. (1983). Answering queries without revealing secrets. *ACM Transactions on Database Systems (TODS)*, 8(1):41–59.

Tao, J., Slutzki, G., and Honavar, V. (2010). Secrecy-preserving query answering for instance checking in $\mathcal{EL}$. In *Proceedings of Web Reasoning and Rule Systems, 195–203*.

Tao, J., Slutzki, G., and Honavar, V. (2012). Pspace tableau algorithms for acyclic modalized $\mathcal{ALC}$. *Journal of Automated Reasoning*, 49(4):551–582.

Tao, J., Slutzki, G., and Honavar, V. (2014). A conceptual framework for secrecy-preserving reasoning in knowledge bases. *TOCL*, 16(1):3:1–3:32.

Tsukada, Y., Mano, K., Sakurada, H., and Kawabe, Y. (2009). Anonymity, privacy, onymity, and identity: A modal logic approach. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 3, pages 42–51. IEEE.

Weitzner, D. J., Abelson, H., Berners-Lee, T., Feigenbaum, J., Hendler, J., and Sussman, G. J. (2008). Information accountability. *Communications of the ACM*, 51(6):82–87.