# Deciding Data Privacy for $\mathcal{ALC}$ Knowledge Bases

Inauguraldissertation
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

**Phiniki Stouppa**

von Zypern

Leiter der Arbeit:

Dr. T. Studer

Institut für Informatik und angewandte Mathematik

# Deciding Data Privacy for $\mathcal{ALC}$ Knowledge Bases

Inauguraldissertation
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

**Phiniki Stouppa**

von Zypern

Leiter der Arbeit:

Dr. T. Studer

Institut für Informatik und angewandte Mathematik

Von der Philosophisch-naturwissenschaftlichen Fakultät
angenommen.

Bern, 7. Mai 2009

Der Dekan:
Prof. U. Feller

# Contents

# Introduction

In information systems, data privacy refers to the confidentiality of certain information that might be stored in the system. systems are often required to share part of their data with third-parties. This can be realized, for instance, through direct access to the systems or through reports that are provided at a future time. Privacy concerns arise when, at the same time, a system is also required to keep certain sensitive information confidential. For this purpose, confidentiality verifications ought to be provided. Such verifications would assure that all shared data preserve the privacy of the confidential information and so, there is no leakage of it.

The problem of providing such verifications is called the *data privacy problem*. This is an active topic that has appeared recently in the literature and is of wide interest. In fact, there is not just one data privacy problem but rather a family of problems each of which serves certain privacy concerns. For instance, perfect privacy [MS04] is concerned with the problem of verifying that the possibility of guessing the confidential information is not influenced at all by the shared data. Other privacy related issues are discussed in a separated section.

In this thesis, we examine the privacy problem of inferring accurately the confidential information. That is to say, given the shared data, decide whether one can be certain about the validity of the confidential information. This notion of privacy, the so-called *provable data privacy* is defined on the notion of certain answers, a notion that stem out from the study of incomplete databases [vdM98] and is now widely used in the context of data integration [CCGL02, Hal01] and data exchange [AL05, FKMP05]. Provable data privacy has initially been introduced in [SS05] from the perspective of relational database systems. There, it was shown that, when conjunctive queries are considered, provable data privacy can be decided in PTime. In this thesis, we present a general definition of this problem that applies to arbitrary systems. In order to agree on terminology, the minimum requirements of a system are first described.

The data of a system is stored in the *repository* (e.g. a database) and there is an interface for accessing the repository via queries. A set of queries is called a *view definition* $D_V$. When the queries are issued (i.e. they are all evaluated on the same repository) an answer to each of them is obtained. A *definite answer* can be boolean (true or false) or a non-empty set of tuples of constants. The queries together with their answers comprise a *view* $V$, which is an *instance of* $D_V$, as they both contain the same queries.

We now turn to the problem of provable privacy. Shared data (i.e. the data that are provided to a user) consists of a view and some general knowledge about the repository, the so-called *background knowledge.* Such a knowledge may include common-sense information or data of the repository that are not expressible in a view and is always in consistence with the repository. In general, it is assumed that the background knowledge remains stable over time while the view is changeable. The way a query is evaluated is also assumed to be shared. *confidential information* (or the *secret*) takes the form of a query, the definite answers of which are intended to be unreachable. For instance, if one wishes to hide all bunkers, the confidential query would be the retrieval $\mathsf{Bunker}(x)$. Confidential queries can be only queries that are supported by the system.

Now, when a user with limited access to the system is provided with a view and a background knowledge (also referred to as the user's knowledge), s/he can try to imagine the whole repository. Every repository which is in accordance with the user's knowledge is considered *possible.* The problem is then described as follows: given a confidential query, a view and a background knowledge, is there an answer that appears in the evaluation of the query on every possible repository? If the answer is positive then we say that data privacy is not preserved for that query with respect to the view and the background knowledge. Otherwise, data privacy is preserved.

**Example** *Consider a description logic-based repository containing topographical data of a region. It includes, for instance, information about the location of buildings and fields in several districts. Let* Butterflies *be one such district and the background knowledge* $\mathcal{R}_{bg}$ *that is provided to the user be:*

$$\begin{aligned}\mathcal{R}_{bg} = \{&(\mathsf{butterflies}, \mathsf{field}_{210}) : \mathsf{Includes}, (\mathsf{field}_{210}, \mathsf{cellar}_1) : \mathsf{Has},\\ &(\exists \mathsf{Has}.\mathsf{Cellar}) \sqsubseteq \mathsf{Building}\}\end{aligned}$$

*The also public view* $V$ *contains the following information:*

$$\begin{aligned}V = \{&\langle \mathsf{Cellar}, \{\mathsf{cellar}_1\}\rangle,\\ &\langle \forall \mathsf{Includes}.(\neg \mathsf{Building} \vee \mathsf{Bunker}), \{\mathsf{butterflies}\}\rangle\}\end{aligned}$$

*Then, data privacy is not preserved for the secret query* $\mathsf{Bunker}$ *because* $\mathsf{field}_{210} : \mathsf{Bunker}$ *is true in every possible repository that respects* $\mathcal{R}_{bg}$ *and* $V$*. If, however, the answer to the second query of* $V$ *were the empty set, then privacy would be preserved.*

Once having this problem, we can define a more general problem as follows: instead of considering privacy with respect to a shared view, we can consider privacy when only a shared view definition is given (i.e. only a set of queries). Then, we can decide privacy for any view that might be an instance of the given definition. That is to say, we decide privacy for some views that might appear in the future. This is a continuation of the first problem and was listed in the further work section of [SS05].

**Example** *Consider the repository of the example above. Since data privacy is not preserved for* $\mathsf{Bunker}$ *wrt.* $\mathcal{R}_{bg}$ *and* $V$*, then data privacy is not preserved for* $\mathsf{Bunker}$ *wrt.* $\mathcal{R}_{bg}$ *and* $D_V$*,either, with*

$$D_V = \{\mathsf{Cellar}, \forall \mathsf{Includes}.(\neg \mathsf{Building} \vee \mathsf{Bunker})\}.$$

*On the other hand, privacy is preserved for the query* $\mathsf{Bunker}$ *wrt.* $\mathcal{R}_{bg}$ *and* $D'_V = \{\mathsf{Cellar}\}$*, as no instance of the latter reveals any information about which can be a* $\mathsf{Bunker}$*.*

This thesis is about deciding the two privacy problems discussed above for $\mathcal{ALC}$ knowledge bases, the basic knowledge bases built on description logics. Description logics are a family of decidable fragments of first order logic that can be used in representing knowledge. The $\mathcal{ALC}$ language is the simplest description logic language in which full negation on concepts is expressible. A repository (also called a knowledge base) contains $\mathcal{ALC}$-concept axioms and $\mathcal{ALC}$-concept and role assertions. Allowed queries are axioms (boolean), concept assertions (boolean), and concepts (retrieval). Query evaluation is based on the usual entailment of first order logic and might differ from system to system. The expressivity of confidential information depends on the way queries are evaluated. For this reason, we have chosen an evaluation that allows for the privacy preservation of more information. The following confidential information is expressible:

- $C \sqsubseteq D$: "it is confidential that all $C$ objects are also $D$ objects"
- $a : C$: "it is confidential that $a$ is a $C$ object that appears in the repository"

- $C$: "for every object $a$ that appears in the repository, it is confidential that $a$ is a $C$ object"

Note that the confidential information does not need to be valid. When it is valid, however, the information must be hidden. Saying, for instance, that "$a : C$ is confidential" means "if $a$ is a $C$ object in the repository, then one cannot infer this".

Also, queries are evaluated under the open world assumption. We therefore assume that the knowledge stored in the system is incomplete. For instance, it might be unknown whether $a$ is a $C$ object or a $\neg C$ object. Furthermore, the set of individuals that appear in the repository is not shared. One is aware only of the existence of those individuals that appear in the view and the background knowledge. Because of this, the confidential query $a : \top$ is meaningful and expresses that "it is confidential that $a$ is an object in the repository". It is also possible to preserve the privacy of an axiom $\top \sqsubseteq C$ while sharing the retrieval query $C$.

Another remark is that, the view (resp. view definition) and the background knowledge may exhibit some information that is not expressible in the repository. For instance, one might infer that a concept $C$ is not empty. This information is not expressible in an $\mathcal{ALC}$ knowledge base and therefore, it cannot be hidden either. That an object $a$ is not in the repository can also be inferred. This might happen when the query $a : \top$ is shared through the view (resp. view definition) and the evaluation of it returns "false". Since one is aware of how the entailment is computed and $a : \top$ is an allowed query, it is assumed that the validity of $a : \top$ is known. Therefore, the evaluation returns false only when $a$ is not in the repository.

## Results

In this thesis it is shown that, for the $\mathcal{ALC}$ knowledge bases considered above, the data privacy problems are not harder to decide than the entailment problem. We give complete decision procedures for both problems that are ExpTime-computable. These procedures are of optimal complexity as it is also shown that the problems are ExpTime-complete. A partial decision procedure that is PTime-computable is also presented in the present work. It is based on the syntactic representation of the information and provides sufficient (but not necessary) conditions for preserving both data privacy problems. This solution is the continuation of an idea listed in the further work section of [SS05], the idea of identifying patterns for which privacy is preserved. An application to $\mathcal{ALC}$ modular ontologies (i.e. $\mathcal{ALC}$ knowledge bases that consist of sub-knowledge bases) demonstrates the usefulness and limitations of this solution.

Apart from some simple model-theoretic properties that have been applied, all results obtained are essentially proof-theoretical. In particular, we have proven and applied properties of a deductive system that is suitable for showing the inconsistency of an $\mathcal{ALC}$ knowledge base. These properties were particularly useful in the proof of the partial solution. A second side-result we obtained concerns an expected result on the complexity of the $\mathcal{ALC}$-concept unsatisfiability. Concept unsatisfiability was shown in [BCM$^+$03] to be ExpTime-hard for an arbitrary knowledge base. Here, we show that this is also the case when the knowledge base is consistent.

We have already published earlier versions of the results presented here. In [SS07], the problem of provable privacy on views was defined on an information system that is general enough and applies to both data and knowledge bases. The results of data privacy on views for $\mathcal{ALC}$ knowledge bases were also presented there. In [SS09] the problem of privacy on view definitions was addressed, demonstrated and decided for $\mathcal{ALC}$ knowledge bases. Both the complete and the partial solution were presented there.

# Outline

After this introduction, we continue in Chapter 1 with the problems of provable data privacy, that are defined formally for an arbitrary system. Brief overviews on information systems and other privacy issues are also included. We continue in Chapter 2 with an introduction to the $\mathcal{ALC}$ knowledge bases. Then, a generalized version of the deductive system is presented. The properties of the system that are going to be applied to the main results are also provided there. The complete and partial solutions to the data privacy problems are shown in Chapter 3. Their complexity is also analyzed there. Chapter 4 includes a couple of direct applications of the privacy solutions, the main one of which is the application to modular ontologies. Finally, we conclude with an overview of the results and further work.

Although the thesis is supposed to be read in the given order, some sections can be omitted or read in a different order. Section 1.4 is independent from the rest. For those who are familiar with $\mathcal{ALC}$-knowledge bases, Sections 2.1, 2.2 and 2.3 can be omitted. All technical results are presented in Section 2.5 and Chapters 3 and 4, the main ones of which are in Chapter 3. Sections 2.5 and 3.1 are independent and both apply to Sections 3.2 and 3.3. Section 4.1 requires all previous results and 4.2 requires 4.1 and 3.3.

## Preliminaries

This work is addressed to everybody with interest in logical and proof-theoretical applications to computer science. It requires only some elementary knowledge in logic and complexity. All preliminaries are briefly described and demonstrated, except for the complexity issues that are assumed to be known.

## Acknowledgements

This thesis has been mainly supported by the Swiss National Science Foundation to whom I am very grateful for this. I would also like to thank Prof. Jäger and Thomas Studer for finding ways to enhance my financial situation and, of course, for all the academic assistance and supervision they provided me with during my thesis. This thesis also owes a lot to the group seminars as well as to the joint meetings of ABM and Münchenwiler, in where preliminary versions of it have been presented.

I am also thankful to the (former and actual) members of the TIL group for the many inspired discussions and environment. Finally, I would like to thank some friends for supporting me their way: Joël for the open accommodation, Alessio for the music, Leonard for the exotic cooking, and Maja and Natassa for all the assistance, encouragement, and daily support.

# Chapter 1

# Data Privacy in Information Systems

## 1.1 Information Systems

An information system is a system designed for storing and manipulating information. In general, it consists of a repository (or a number of repositories), an interface for accessing the repository, and a number of services that can be obtained through the interface. A repository $R$ might be, for instance, a data or knowledge base, together with its constraints. It can be also seen as a set of sentences of first order logic. Through out the paper we assume that only consistent repositories are accessible.

**Example 1.1.1** *We continue on the example from the introduction but now in a system of first order sentences. Consider a repository $R$ with the following data:*

$$\mathsf{District}(\mathsf{butterflies}) \tag{1.1}$$

$$\mathsf{District}(\mathsf{redroses}) \tag{1.2}$$

$$\mathsf{Includes}(\mathsf{butterflies}, \mathsf{field}_{210}) \tag{1.3}$$

$$\mathsf{Includes}(\mathsf{butterflies}, \mathsf{field}_{211}) \tag{1.4}$$

$$\mathsf{Includes}(\mathsf{butterflies}, \mathsf{field}_{212}) \tag{1.5}$$

$$\mathsf{Includes}(\mathsf{redroses}, \mathsf{field}_{315}) \tag{1.6}$$

*A set of constraints shapes these data: every district includes at least one field,*

$$\forall x\ (\mathsf{District}(x) \Rightarrow (\exists y\ \mathsf{Includes}(x, y) \wedge \mathsf{Field}(y))) \tag{1.7}$$

*every building is also a field*

$$\forall x\ (\mathsf{Building}(x) \Rightarrow \mathsf{Field}(x)) \tag{1.8}$$

*and districts and fields are disjoint.*

$$\forall x\ ((\mathsf{District}(x) \wedge \mathsf{Field}(x)) \Rightarrow \perp) \tag{1.9}$$

*In every district that includes buildings there is at least one bunker.*

$$\begin{aligned}\forall x\ ((\mathsf{District}(x) \wedge \exists y\ \mathsf{Includes}(x,y) \wedge \mathsf{Building}(y))\\ \Rightarrow (\exists z\ \mathsf{Includes}(x,z) \wedge \mathsf{Bunker}(z)))\end{aligned} \tag{1.10}$$

*A bunker has always a cellar*

$$\forall x\ (\mathsf{Bunker}(x) \Rightarrow (\exists y\ \mathsf{Has}(x,y) \wedge \mathsf{Cellar}(y))) \tag{1.11}$$

*and every field with a cellar is a building.*

$$\forall x\ ((\exists y\ \mathsf{Has}(x,y) \wedge \mathsf{Cellar}(y)) \Rightarrow \mathsf{Building}(x)) \tag{1.12}$$

The services obtained from $R$ are all the queries $q$ the user is allowed to issue. When issued, the queries are evaluated on the current data of the repository. We write $\mathsf{ans}(q, R)$ for the evaluation of the query on $R$. This is a well-defined, non-random procedure. The answers to a query and how these are actually evaluated differ from system to system. In general possible answers are "yes","no" and "don't know", when the query is boolean. When it is retrieval the answers are tuples of constants from the domain of $R$.

**Example 1.1.2** *Consider the repository $R$ from Example 1.1.1. Boolean queries are first order sentences and retrieval queries are first order formulae with $n > 0$ free variables. Their evaluation* $\mathsf{ans}()$ *can be defined on the entailment as follows: for $q$ a sentence,* $\mathsf{ans}(q, R)$ *returns "yes" when $R \models q$, "no" when $R \models \neg q$ and "don't know" when neither $R \models q$ nor $R \models \neg q$ holds. For $q$ a formula,* $\mathsf{ans}(q, R)$ *returns the set of $n$-ary tuples $\vec{t}$ of constants from the domain of $R$ for which $R \models q(\vec{t})$ holds.*

The set of queries a user can issue is called a view definition $D_V$. When a view definition is issued (i.e. all queries of the view definition are issued) we get a view $V$. A view is a set of tuples $\langle q, r\rangle$ with $r$ being an evaluation of $q$. A view is an instance of a view definition if it contains precisely the queries of the view definition. Note that every view is the instance of exactly one view definition. We write $R \Vdash V$ to say that the view $V$ results from issuing $D_V$ on the consistent $R$, i.e. for every $\langle q, r\rangle \in V$, $r = \mathsf{ans}(q, R)$.

**Example 1.1.3** *Consider the repository R from Example 1.1.1 and the view definitions*

$$\begin{aligned} D_{V_1} &= \{\mathsf{Field}(x),\ \mathsf{Cellar}(x),\ \forall x\ (\mathsf{Bunker}(x) \Rightarrow \mathsf{Building}(x))\} \quad \textit{and} \\ D_{V_2} &= \{\neg\mathsf{Building}(x),\ \forall y\ (\mathsf{Includes}(x,y) \Rightarrow (\neg\mathsf{Building}(y) \vee \mathsf{Bunker}(y)))\}. \end{aligned}$$

*Then,* $R \Vdash V_1$ *and* $R \Vdash V_2$ *for*

$$\begin{aligned} V_1 = \{&\langle \mathsf{Field}(x), \emptyset\rangle, \\ &\langle \mathsf{Cellar}(x), \emptyset\rangle, \\ &\langle \forall x\ (\mathsf{Bunker}(x) \Rightarrow \mathsf{Building}(x)),\ \textit{"yes"}\rangle\} \quad \textit{and} \end{aligned}$$

$$\begin{aligned} V_2 = \{&\langle \neg\mathsf{Building}(x), \{\mathsf{butterflies}, \mathsf{redroses}\}\rangle, \\ &\langle \forall y\ (\mathsf{Includes}(x,y) \Rightarrow (\neg\mathsf{Building}(y) \vee \mathsf{Bunker}(y))), \emptyset\rangle\} \end{aligned}$$

*instances of* $D_{V_1}$ *and* $D_{V_2}$*, respectively.*

## 1.2 Data Privacy on Views

The notion of provable data privacy has initially been introduced in [SS05] from the perspective of relational database systems. Here, we give a general definition of this notion that applies to arbitrary systems. The problem of data privacy is to decide whether some sensitive information is kept secret or not from an unauthorized user. In general, it is assumed that the user is granted access to a specific view $V$ and to some general (background) knowledge of the system. We also assume that the user is aware of how $\mathsf{ans}()$ is implemented. The background knowledge can be seen as a part of the repository, called $\mathcal{R}_{bg}$.

A secret is formalized as a query the answers of which should not be provided to the user. That is to say, a query is secret when the answer to it is "don't know" or $\emptyset$, depending on whether it is boolean or retrieval. For the sake of simplicity, an empty set is returned when the answer is "don't know", too. Secret queries are restricted to those queries that are allowed to be issued (i.e. $\mathsf{ans}(q, R)$ is given).

Given the user's knowledge and a query that is intended to be unaccessible to the user, the data privacy problem is to decide whether answers to the secret query are provided or not to the user.

**Example 1.2.1** *Consider the repository R from Example 1.1.1 and the view definitions and views* $D_{V_1}$*,* $D_{V_2}$ *and* $V_2$ *from Example 1.1.3. If*

$$\mathcal{R}_{bg} = R \cup \{Has(\mathsf{field}_{210}, \mathsf{cellar}_1)\} \quad \textit{and} \quad V = V_1' \cup V_2'$$

*with*

$$\begin{aligned} V_1' = \{ & \langle \mathsf{Field}(x), \{\mathsf{field}_{210}, \mathsf{field}_{212}\}\rangle, \\ & \langle \mathsf{Cellar}(x), \{\mathsf{cellar}_1\}\rangle, \\ & \langle \forall x\ (\mathsf{Bunker}(x) \Rightarrow \mathsf{Building}(x)),\ \text{“}yes\text{”}\rangle\} \quad \textit{and} \end{aligned}$$

$$V_2' = \ V_2$$

*is the user's knowledge, then the query* $\mathsf{Bunker}(x)$ *remains secret as we cannot conclude from* $\mathcal{R}_{bg}$ *and* $V$ *whether a specific constant is a Bunker. If we instead consider*

$$\begin{aligned} V_2' = \{ & \langle \neg\mathsf{Building}(x), \{\mathsf{butterflies}, \mathsf{redroses}, \mathsf{field}_{212}\}\rangle, \\ & \langle \forall y\ (\mathsf{Includes}(x,y) \Rightarrow (\neg\mathsf{Building}(y) \vee \mathsf{Bunker}(y))), \{\mathsf{butterflies}\}\rangle\} \end{aligned}$$

*we can conclude that the constant* $\mathsf{field}_{210}$ *must be a Bunker. This is because* $\mathsf{field}_{210}$ *has a Cellar (see* $\mathcal{R}_{bg}$ *and* $V_1'$*) and therefore it is a Building (see (1.12)). Since it is also included in* butterflies*, then* $V_2'$ *implies that it is a Bunker.*

We say that *data privacy is preserved* for a query $q$ with respect to a tuple $\langle \mathcal{R}_{bg}, V\rangle$ if there are no answers (other than the empty set) to $q$ that follow with certainty from the information of $V$ and $\mathcal{R}_{bg}$. This can be made precise by the notion of *certain answers* [vdM98]. The function $\mathsf{certain}(q, \langle \mathcal{R}_{bg}, V\rangle)$ returns the answers to $q$ that hold in every repository that - according to the user's knowledge - could be the current one (a so-called *possible* repository).

**Definition 1.2.2** *A repository* $R$ *is* possible *with respect to a tuple* $\langle \mathcal{R}_{bg}, V\rangle$ *if* [1] $\mathcal{R}_{bg} \subseteq R$*, and* $R \Vdash V$*. By* $\mathsf{Poss}_{\langle \mathcal{R}_{bg}, V\rangle}$*, we denote the set of all possible repositories with respect to the tuple* $\langle \mathcal{R}_{bg}, V\rangle$*.*

**Definition 1.2.3** *The* certain answers *to a query* $q$ *with respect to a tuple* $\langle \mathcal{R}_{bg}, V\rangle$ *are defined by*

$$\mathsf{certain}(q, \langle \mathcal{R}_{bg}, V\rangle) := \bigcap_{R \in \mathsf{Poss}_{\langle \mathcal{R}_{bg}, V\rangle}} \mathsf{ans}(q, R).$$

Since the view granted to the user is the one obtained from the current repository, we can restrict ourselves to tuples $\langle \mathcal{R}_{bg}, V\rangle$ for which there is at least one possible repository (namely, the current one). In other words, we consider only views $V$ with $\mathsf{Poss}_{\langle \mathcal{R}_{bg}, V\rangle} \neq \emptyset$. Tuples that satisfy this property are called *valid.*

[1] if $\mathcal{R}_{bg}$ is not a part of $R$ then the premise changes to $R \cup \mathcal{R}_{bg}$ is consistent and $R \cup \mathcal{R}_{bg} \Vdash V$.

**Definition 1.2.4** Data privacy is preserved for $q$ with respect to a valid tuple $\langle \mathcal{R}_{bg}, V \rangle$ *if*

$$\mathsf{certain}(q, \langle \mathcal{R}_{bg}, V \rangle) = \emptyset.$$

The above definition does not provide us, however, with a decision procedure for data privacy, as the computation of the function $\mathsf{certain}()$ might require an infinite number of possible repositories. In [SS05] a solution to the problem is provided for relational databases with conjunctive queries. In this thesis a solution for $\mathcal{ALC}$ knowledge bases with boolean and (concept) retrieval queries is presented.

## 1.3 Data Privacy on View Definitions

The problem of data privacy on views can be easily extended to cover a family of views. One such generalization can be made by considering all the views that are instances of a specific view definition. In this way one can preordain data privacy on views that will be given to the user at a future time.

**Example 1.3.1** *Consider Example 1.2.1. Data privacy is preserved for the query* $\mathsf{Bunker}$ *when an instance of* $D_{V_1}$ *and the background knowledge* $\mathcal{R}_{bg}$ *are provided. This is because, the Cellars and Fields that might be exhibited in any possible instance of it, do not say anything about which is a Bunker. On the other hand, data privacy is not preserved for* $\mathsf{Bunker}$ *when an instance of* $D_{V_1} \cup D_{V_2}$ *and* $\mathcal{R}_{bg}$ *are provided. As we have seen, there are instances of it (take* $V = V_1' \cup V_2'$ *from Example 1.2.1) that might exhibit some Bunkers.*

The problem of data privacy on view definitions can be formally stated as follows:

**Definition 1.3.2** *A view* $V$ is relevant *wrt. a tuple* $\langle \mathcal{R}_{bg}, D_V \rangle$ *if it satisfies the following: (i)* $V$ *is an instance of* $D_V$ *and (ii)* $\langle \mathcal{R}_{bg}, V \rangle$ *is valid.*

**Definition 1.3.3** Data privacy is preserved for $q$ wrt. a tuple $\langle \mathcal{R}_{bg}, D_V \rangle$ *if for every relevant view* $V$ *wrt.* $\langle \mathcal{R}_{bg}, D_V \rangle$*, data privacy is preserved for* $q$ *wrt.* $\langle \mathcal{R}_{bg}, V \rangle$*. The data privacy problem on view definitions is to decide whether data privacy is preserved for* $q$ *wrt.* $\langle \mathcal{R}_{bg}, D_V \rangle$*.*

## 1.4 Other Work On Privacy

An increased interest on data privacy issues has appeared in the literature over the last decade. Names like confidentiality, information disclosure and data security have all been used to describe the same phenomenon: the secrecy of data from unauthorized users at a logical level.

Research on data privacy is concerned with the following problems:

1. what does data privacy mean?

2. how to detect whether data privacy is violated in a system?

3. how to enforce privacy when this is violated?

Complete and partial answers to these questions are listed below in the order addressed. Note, however, that this presentation is not meant to be complete.

### Notions of privacy

As it has already been mentioned in the introduction, data privacy might apply to different levels, depending on the kind of privacy one wishes to obtain. A variety of privacy guarantees has been developed for this purpose.

The problem of *inferring precisely* the secret information from a given view has first been addressed in statistical databases [HM70, DD79] and other complete systems [SDJVdR83]. As we will see later on, provable privacy in $\mathcal{ALC}$ knowledge bases is equivalent to this problem. Provable data privacy has also been considered in [BB04, BW08] for the study of a variety of enforcement methods on relational databases over boolean queries.

The notion of *$k$-anonymity* has been introduced in [SS98] and provides a flexible privacy guarantee that allows for both certain and probable inference: given a number $k > 1$, the secrecy of a query $q$ satisfies $k$-anonymity if the user cannot distinguish among at least $k$-many tuples that might belong to the evaluation of $q$. In other words, if we take one answer (= tuple) out of each evaluation of $q$ on a possible repository, then we find at least $k$ different answers. The case when $k = 2$ captures precisely the notion of provable privacy: when 2-anonymity is satisfied then there is not a single answer that appears in every evaluation. And when it is not satisfied then there is at least one answer that appears in every evaluation of $q$.

The guarantee that an answer to a query cannot be guessed with a *certain amount of probability* (given as a threshold) is the notion described, for instance, in [AA01, ESAG02]. This notion is defined in terms of probability measures. In [MKGV07] the notion of *$l$-diversity* is introduced that takes into consideration the knowledge the user can obtain from external sources, the so-called a-priori knowledge. External sources comprise what the user might know about the secret even when no system data (e.g. background knowledge or views) are provided. $L$-diversity computes the probabilities based on the random worlds - a restricted form of possible repositories - and it is claimed to extend $k$-anonymity. An instantiation of $l$-diversity that is also equipped with a more expressive a-priori knowledge has been presented in [MKMG07].

The strictest form of privacy is a relative privacy guarantee, the so-called *perfect privacy*. Perfect privacy has been introduced in [MS04] and guarantees that the only information provided through the system about the secret is what the user could anyway know a-priori. This notion is also defined in terms of probability measures. Whenever the probability of guessing the secret is modified, while, in addition to the external sources, data from the system are provided to the user, then perfect privacy is violated. Therefore, this notion is concerned not only with hiding the answers to the secret query, but also with hiding the number of answers to it or, even, the existence of an answer.

A relaxation on perfect privacy that is more appropriate for practical applications has been presented in [DMS05]. There, slight (limit) diversions on the measures are considered safe. This relaxation has been obtained by applying an alternative probabilistic model that uses asymptotic conditional probabilities. A number of other privacy guarantees including perfect privacy and provable privacy (i.e. certain answers) are shown to be expressible by these means.

Two relative privacy guarantees that concern *safe updates* of the background knowledge are provided in [ND07]. The first one assumes that the user has no information about the possible repositories from external sources. When this is the case, an update is considered safe (i.e. it does not further expose the secret) when it does not change the set of evaluations of the secret on the possible repositories (although the set of possible repositories might be affected by the update). Otherwise, it is required that the set of possible repositories remains exactly as it was. The latter guarantee is claimed to correspond to perfect privacy.

A *minimal privacy* that guarantees the secrecy of the complete answer to the secret query has also been introduced in [ND07]. A secret query is considered safe when there are at least two different evaluations of it on the possible repositories. Therefore, the user cannot know whether a specific answer is actually the complete answer to the secret query (although s/he might infer the answer). This notion of privacy is similar to the notion of provable privacy but weaker: consider the case when the secret is a retrieval query $q$ with a single variable. If every evaluation of $q$ with respect to the possible repositories contains the constant $a$ but there are two different evaluations of it, say, $\{a\}$ and $\{a, b\}$, then the minimal privacy is satisfied.

In [RMSR04] the leakage of any additional information with respect to what is already given to the user is considered to violate privacy. It is therefore assumed that public data describe precisely what the user is allowed to know and no mistakes have occurred. So, the question here is whether, in addition to the public data, the user is allowed to issue a query. Given that the user has access only to an authorization view, any query that is evaluated differently on the whole database than on the authorization view, under any database instance, leaks information. Therefore, issuing such queries may reveal some answers that could not be inferred solely from the authorization view and so, these are not allowed as they may increase the user's knowledge. When the user knows that all allowed queries satisfy this property (called *conditional validity*), then this notion of privacy is similar to the minimal one: when a query is allowed then the user knows that the answer he gets is the complete answer to the query and so the minimal privacy is not satisfied. When a query is not allowed then s/he knows that this happens precisely because the view might not include the complete answer to the query and so, minimal privacy is satisfied[2]. This privacy guarantee can also be described as a relative one: the privacy of *any* additional information is preserved when updating the user's knowledge with the answers to a query, if those answers were already known to the user.

Note that other privacy guarantees might be also desirable, although they might still lack formalization. Consider, for instance, a notion of probable guarantee that takes into account the number of possible repositories that entail the secret.

## Detecting Privacy

The privacy guarantees presented above have mainly been applied to the frame of relational databases, though applications to data mining and XML-documents have also been considered. Here, we present some of the most recent results. A survey on privacy issues that also includes older studies can be found in [FJ02].

As we already mentioned in the introduction, *provable privacy* has been studied in [SS05] on relational database systems over conjunctive queries.

[2] The relation between those guarantees is only a claim and is not based on results found in the literature.

There, it was shown that the problem of provable privacy on views can be decided in PTime.

The *precise inference* problem has been studied on relational databases, for instance, in [BFJ00]. There, the problem is shown to be decidable on views and view definitions for a restricted form of selection-projection queries.

In [YWJ05] the notion of *k-anonymity* has been studied on relational databases on which the views are restricted to include only projections and selections on a single private table. In the absence of constraints, $k$-anonymity has been shown to be decidable in PTime. The addition of constraints (in the form of functional dependencies) leads to $\Sigma_2^p$-hardness. Certain subcases of the latter that can be decided in PTime are also considered. The notion of $k$-anonymity has also been studied in XML-documents [KMMZ06] for the case the secret is a relationship and the view definition is given in terms of a query. The 2-anonymity problem is shown to be decidable on a view (i.e. when the view definition is evaluated on a specific document), as there are finite many documents that need to be checked. When queries are restricted syntactically (for details refer to [KMMZ05]), then the problem becomes PTime on both views and view definitions (i.e. when only a document schema is given).

Privacy guarantees defined on probabilities have been studied in the field of relational databases and data mining. In [MKGV07, MKMG07] algorithms for constructing views that preserve *l-diversity* are provided. Those algorithms apply to relational databases and allow for a-priori knowledge that is expressible in the form of logical implications. In *data mining*, the privacy problem is whether there is a leakage of confidential information when the user can access the results of a data mining. Data mining is a technique that applies to large data sets and reveals implicit, non-obvious information. A *probabilistic* guarantee that takes into consideration the a-priori probabilistic distribution can be found in [EGS03]. There, a partial decision to the problem is provided by showing that, when the randomization operator satisfies a condition then privacy is preserved. Randomization is a privacy enforcement method in data mining that does not provide complete probabilistic based privacy. This condition is called amplification and does not require knowing the prior distribution.

*Perfect privacy* has been studied on exchanged database views over conjunctive queries with inequalities. In [MS04] it is shown that the problem on view definitions is $\Pi_2^p$-complete when the answers to the queries are independent events. Although perfect privacy assumes a given domain and probability distribution, the results obtained are mostly independent of both. In [DP05] it is shown that it is actually enough to compare the possible repositories before and after the exhibition of system data. There, the results on perfect privacy have been extended, covering also the case when there are correlations between answers. As it is shown, the generalized problem on views is $\Pi_2^p$-complete and it is undecidable on view definitions. A relaxation on the user's knowledge about the domain allows for decidability in PSpace on a specific view, while the problem remains undecidable when considering view definitions. In [MG06] subclasses of conjunctive queries for which perfect privacy (as it is defined in [MS04]) can be decided in PTime are identifying. These results have been established through a connection of perfect privacy to the problem of checking query containment.

Both *relative* and *minimal* guarantees proposed in [ND07] have been defined and studied (in the same paper) on database integrations over unions of conjunctive queries with equalities. In such a scenario a user has access only to a public schema (i.e. view definition) and a set of constraints that relate data from a (private) source to data that are allowed to appear in the public view. Query answering is performed using certain answers, which is the equivalent of reasoning under the open world assumption. Algorithms for deciding all guarantees are provided that give upper bounds to the problems: the minimal guarantee can be decided in NP while the relative guarantees can be decided in $\Pi_2^p$. These results assume a given (non-public) source and correspond to the privacy problem on views. A practical integration setting on which the same problems are all decidable in PTime is also provided there. Once considering arbitrary sources the problems become undecidable. Arbitrary sources imply that the problem must be decided on every view of the given view definition and therefore, these problems correspond to the privacy problem on view definitions.

In the frame of fine-grained access control using authorization views, *conditional validity* (which is similar to minimal privacy, see discussion above) has been decided partially in [RMSR04] for relational databases. There, inference rules that are sufficient for constructing validity preserving queries are provided, as well as algorithms for deciding validity.

Note that apart from the privacy application in data integration and the results on provable privacy, all other studies are concerned with the privacy implications on complete databases.

## Privacy Enforcement

Another research direction is that of enforcing privacy of the sensitive information via modifications of the public data. Whenever privacy is violated, the view is "reduced" to ensure privacy. An obvious reduction of the view is to provide answers only to a proper subset of its queries. This idea has been realized, for instance, in [SDJVdR83, BB04, BW08, RMSR04] by refusing to return the answer to some queries. In [BB04, BW08] alternative modifications on the answers returned to the user - the so-called controlled query evaluations - have also been discussed. Such modifications include returning incorrect answers to some queries or returning a combination of refused and incorrect answers. A uniform formulation for studying controlled query evaluation that takes into consideration a number of privacy policies, as well as brief summaries of other approaches on the same field are provided in [BB04]. There, the evaluation policies have been studied and compared for complete databases over boolean queries. A result obtained is that, when the user is not aware of the reasons an answer is not provided to him, then refusal is preferable over other methods. It has also been argued that a combination of refusal and lying overcomes some deficiencies of the other methods. These results have recently been extended in incomplete information systems [BW08]. Studies in providing incorrect answers from the perspective of relational databases can be found, for instance, in [WSQ94, BKS95].

Other enforcement methods include application-driven controls. For example, in data mining a privacy enforcement is obtained by randomizing the values in individual records [AA01, ESAG02]. This approach restricts privacy only in a limited way. An improvement to this method is described in [EGS03] where the randomization operator is also required to satisfy the amplification condition (mentioned earlier). Additional application-driven controls can be found in [FJ02].

# Chapter 2

# The $\mathcal{ALC}$ Knowledge Base

This chapter consists implicitly of two parts. In the first part, the $\mathcal{ALC}$ knowledge bases [SSS91, BCM$^+$03] are defined. Knowledge bases are repositories suitable for modeling knowledge. One way to formalize knowledge bases is to use description logics, a family of decidable fragments of first order logic. In such repositories the information is formalized in logical sentences. A presentation of the $\mathcal{ALC}$ language (attributive language with complements), the simplest language with full negation on concepts used in description logics is first given. The syntax and semantics of $\mathcal{ALC}$ knowledge bases follows and finally, in the system services, the allowed queries and their evaluation are defined. Basic properties of $\mathcal{ALC}$ knowledge bases are also shown.

In the second part, a deductive system is presented, which serves as a consistency prover for $\mathcal{ALC}$ knowledge bases. This system is more general than the one usually applied in the sense that, now, the entities of its sequents are coloured. Colouring is a technique that helps classifying the entities according to their origins. Some properties of the proofs of this system conclude the chapter.

## 2.1 The Language $\mathcal{ALC}$

The $\mathcal{ALC}$ language consists of a countable set of *individuals* $\mathsf{Ind}$, a countable set of *atomic concepts* $\mathsf{AConc}$, a countable set of *roles* $\mathsf{Rol}$ and the *concepts* built on $\mathsf{AConc}$ and $\mathsf{Rol}$ as follows:

$$C, D := A \mid \neg A \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

where $A \in \mathsf{AConc}$, $R \in \mathsf{Rol}$, the operators $\sqcap$ and $\sqcup$ stand for conjunction and disjunction, respectively, and $\forall R$ and $\exists R$ are called $R$-universal and $R$-existential restrictions, respectively. We use $a, b, c, \ldots$ to denote individuals,

$A, A_1, A_2, \ldots$ to denote atomic concepts, $R, R_1, R_2, \ldots$ to denote roles and $C, C_1, C_2, \ldots, D, D_1, D_2, \ldots$ to denote arbitrary concepts.

Note that the language includes only concepts in negation normal form. The complement of a concept $\neg(C)$ is inductively defined, as usual, by using the law of double negation, de Morgan's laws and the dualities for quantifiers:

$$\begin{array}{rclrcl}
\neg(A) & := & \neg A & \neg(\neg A) & := & A \\
\neg(C \sqcap D) & := & (\neg(C) \sqcup \neg(D)) & \neg(C \sqcup D) & := & (\neg(C) \sqcap \neg(D)) \\
\neg(\forall R.C) & := & \exists R.\neg(C) & \neg(\exists R.C) & := & \forall R.\neg(C)
\end{array}$$

When the scope of the negation is unambiguous, we also write $\neg C$ instead of $\neg(C)$. Moreover, the constants $\top$ and $\bot$ abbreviate $A \sqcup \neg A$ and $A \sqcap \neg A$, respectively, for some $A \in \mathsf{AConc}$. Individuals, concepts and roles are interpreted in the following way:

**Definition 2.1.1** *An interpretation $\mathcal{I}$ consists of a non-empty domain $\Delta^{\mathcal{I}}$ and a mapping $()^{\mathcal{I}}$ that assigns*

- *to each individual $a \in \mathsf{Ind}$ an element $(a)^{\mathcal{I}} \in \Delta^{\mathcal{I}}$*
- *to each atomic concept $A \in \mathsf{AConc}$ a set $(A)^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$*
- *to each role $R \in \mathsf{Rol}$ a relation $(R)^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$*

The elements of a domain are denoted by $d, d_1, d_2, \ldots$. The interpretation $\mathcal{I}$ extends then on concepts as follows:

$$\begin{array}{rcl}
(\neg A)^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \setminus (A)^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} & = & (C)^{\mathcal{I}} \cap (D)^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} & = & (C)^{\mathcal{I}} \cup (D)^{\mathcal{I}} \\
(\forall R.C)^{\mathcal{I}} & = & \{d_1 \in \Delta^{\mathcal{I}} \mid \forall d_2\ ((d_1, d_2) \in (R)^{\mathcal{I}} \Rightarrow d_2 \in (C)^{\mathcal{I}})\} \\
(\exists R.C)^{\mathcal{I}} & = & \{d_1 \in \Delta^{\mathcal{I}} \mid \exists d_2\ ((d_1, d_2) \in (R)^{\mathcal{I}}\ \&\ d_2 \in (C)^{\mathcal{I}})\}
\end{array}$$

A concept $C$ is *satisfiable* if there is an interpretation $\mathcal{I}$ for which $C^{\mathcal{I}} \neq \emptyset$. The problem of deciding whether a concept is satisfiable or not is PSpace-complete [SSS91, Sch91].

**Example 2.1.2** *Let $\mathcal{I}$ be the interpretation with domain $\Delta^{\mathcal{I}} = \{d_1, d_2, d_3\}$ and the following mapping $()^{\mathcal{I}}$:*

- *$(d)^{\mathcal{I}} = d_1$, for all individuals $d$.*
- *$(A_1)^{\mathcal{I}} = \{d_1, d_3\}$ and $(A_2)^{\mathcal{I}} = \{d_2, d_3\}$, for the atomic concepts $A_1$ and $A_2$. $(A)^{\mathcal{I}} = \emptyset$, for all other concepts $A$.*

- *$(R)^{\mathcal{I}} = \{(d_1, d_2)\}$, for all roles $R$.*

*Then, $(A_1 \sqcap \neg A_2)^{\mathcal{I}} = (A_1)^{\mathcal{I}} \cap (\Delta^{\mathcal{I}} \setminus (A_2)^{\mathcal{I}}) = \{d_1, d_3\} \cap \{d_1\} = \{d_1\}$ and therefore, the concept $A_1 \sqcap \neg A_2$ is satisfiable. The concept $\exists R.(A_1 \sqcap \neg A_1)$ is unsatisfiable since, by definition, $(A_1 \sqcap \neg A_1)^{\mathcal{I}'} = (A_1)^{\mathcal{I}'} \cap (\Delta^{\mathcal{I}'} \setminus (A_1)^{\mathcal{I}'}) = \emptyset$ for an arbitrary interpretation $\mathcal{I}'$.*

## 2.2 The Knowledge Base

We are now ready to define the $\mathcal{ALC}$ knowledge bases with general concept inclusion axioms (GCIs). An $\mathcal{ALC}$-knowledge base $\mathcal{O}$ is the union of

1. a finite *terminological* set (TBox) of *inclusion axioms* $\top \sqsubseteq C$[1], where $C$ is an $\mathcal{ALC}$ concept called *inclusion concept*, and

2. a finite *assertional* set (ABox) of assertions $a : C$ (*concept assertion*) or $(a, b) : R$ (*role assertion*) where, $R \in \mathsf{Rol}$ is an *assertional role* and $C$ is an $\mathcal{ALC}$ concept called *assertional concept*.

The set of individuals that appear in $\mathcal{O}$ are denoted by $\mathsf{Ind}(\mathcal{O})$. An interpretation $\mathcal{I}$ is a *model* of

- an inclusion axiom $\top \sqsubseteq C$ $(\mathcal{I} \models \top \sqsubseteq C)$ if $(C)^{\mathcal{I}} = \Delta^{\mathcal{I}}$,
- a concept assertion $a : C$ $(\mathcal{I} \models a : C)$ if $(a)^{\mathcal{I}} \in (C)^{\mathcal{I}}$,
- a role assertion $(a, b) : R$ $(\mathcal{I} \models (a, b) : R)$ if $((a)^{\mathcal{I}}, (b)^{\mathcal{I}}) \in (R)^{\mathcal{I}}$.

An interpretation $\mathcal{I}$ is a *model* of an $\mathcal{ALC}$-knowledge base $\mathcal{O}$ if $\mathcal{I} \models \phi$, for every $\phi \in \mathcal{O}$. A knowledge base $\mathcal{O}$ that has a model is a *consistent* knowledge base. Moreover, a concept is satisfiable wrt. a knowledge base $\mathcal{O}$ if there is a model $\mathcal{I}$ of $\mathcal{O}$ for which $(C)^{\mathcal{I}} \neq \emptyset$.

**Example 2.2.1** *Let $\mathcal{O}$ be the knowledge base with the following* TBox $\mathcal{T}$ *and* ABox $\mathcal{A}$*:*

$$\begin{array}{lllll} \mathcal{T} = \{ & \top \sqsubseteq A_1 \sqcup \forall R.A_2 \,, & & \mathcal{A} = \{ & a : A_1 \,, \\ & \top \sqsubseteq \exists R.\neg A_1 & \} & & b : A_2 \,, \\ & & & & (b, c) : R \quad \} \end{array}$$

*Consider the interpretation $\mathcal{I}$ with domain $\Delta^{\mathcal{I}} = \{d_1, d_2, d_3\}$ and the following mapping $()^{\mathcal{I}}$:*

[1] In general, inclusion axioms are given in the form $C_1 \sqsubseteq C_2$. This, however, can be linearly transformed to its equivalent $\top \sqsubseteq \neg C_1 \sqcup C_2$ which is more appropriate for our purpose.

- *$(a)^{\mathcal{I}} = d_1$ , $(b)^{\mathcal{I}} = d_2$ and $(c)^{\mathcal{I}} = d_3$, for the individuals $a$,$b$ and $c$. $(d)^{\mathcal{I}} = d_3$, for all other individuals $d$.*
- *$(A_1)^{\mathcal{I}} = \{d_1, d_3\}$ and $(A_2)^{\mathcal{I}} = \{d_2, d_3\}$, for the atomic concepts $A_1$ and $A_2$. $(A)^{\mathcal{I}} = \emptyset$, for all other concepts $A$.*
- *$(R)^{\mathcal{I}} = \{(d_1, d_2), (d_2, d_2), (d_2, d_3), (d_3, d_2)\}$, for the role $R$.*

  *$(R')^{\mathcal{I}} = \emptyset$, for all other roles $R'$.*

*Then, $\mathcal{I}$ is a model of $\mathcal{O}$. Therefore $\mathcal{O}$ is a consistent knowledge base. Moreover, since the domain and the mappings of $\mathcal{I}$ on atomic concepts match those of the interpretation of Example 2.1.2, we also conclude that the concept $(A_1 \sqcap \neg A_2)$ is satisfiable wrt. $\mathcal{O}$. The concept $\forall R.A_1$ is, however, unsatisfiable wrt. $\mathcal{O}$ since every model $\mathcal{I}'$ of $\mathcal{O}$ models the inclusion axiom $\top \sqsubseteq \exists R.\neg A_1$ and therefore, $(\neg(\forall R.A_1))^{\mathcal{I}'} = (\exists R.\neg A_1)^{\mathcal{I}'} = \Delta^{\mathcal{I}'}$.*

Deciding the consistency of a knowledge base is an ExpTime-complete problem [Don03, Sch91][2]. The problem of concept satisfiability wrt. a knowledge base is also ExpTime-complete, as the consistency problem reduces to it and vice-versa. Recall that the satisfiability problem becomes PSpace-complete when the knowledge base is empty.

**Theorem 2.2.2** *Given a knowledge base $\mathcal{O}$, an $\mathcal{ALC}$ concept $C$ and an individual $n_{ew} \notin \mathsf{Ind}(\mathcal{O})$ the following hold:*

1. *$\mathcal{O}$ is consistent iff $\top$ is satisfiable wrt. $\mathcal{O}$.*
2. *$C$ is satisfiable wrt. $\mathcal{O}$ iff $\mathcal{O} \cup \{n_{ew} : C\}$ is consistent.*

*Proof*. 1. It is enough to observe that for an arbitrary interpretation $\mathcal{I}$, $(\top)^{\mathcal{I}} = (A \sqcup \neg A)^{\mathcal{I}} = (A)^{\mathcal{I}} \cup (\Delta^{\mathcal{I}} \backslash (A)^{\mathcal{I}}) = \Delta^{\mathcal{I}} \neq \emptyset$.

2. ($\Rightarrow$) Let $\mathcal{I}$ be the model of $\mathcal{O}$ for which $(C)^{\mathcal{I}} \neq \emptyset$ and let $d \in (C)^{\mathcal{I}}$. Take $\mathcal{I}'$ be the interpretation which is identical to $\mathcal{I}$ except in that $(n_{ew})^{\mathcal{I}'} = d$. In the case $(n_{ew})^{\mathcal{I}} = d$ then take $\mathcal{I}' = \mathcal{I}$. The individual $n_{ew}$ does not appear in $\mathcal{O}$ and so its mapping in $\mathcal{I}'$ does not affect any of the inclusion axioms and assertions of $\mathcal{O}$. Therefore, $\mathcal{I}'$ is also a model of $\mathcal{O}$ and $\mathcal{I}' \models n_{ew} : C$.
($\Leftarrow$) Trivial. □

The logical consequences of a knowledge base are defined in the usual way: for $\psi$ an inclusion axiom or an assertion, we say that $\mathcal{O} \models \psi$ (in words, $\mathcal{O}$ *entails* $\psi$) if for every model $\mathcal{I}$ of $\mathcal{O}$, $\mathcal{I} \models \psi$.

[2] More details are available from the DL complexity navigator at `http://www.cs.man.ac.uk/~ezolin/dl/`.

**Example 2.2.3** *Let $\mathcal{O}$ be the knowledge base from Example 2.2.1. Since $\top \sqsubseteq \exists R.\neg A_1 \in \mathcal{O}$, every individual is forced to be R-related to some other individual and therefore, $\mathcal{O} \models b : \exists R.\top$ holds. A less obvious consequence is $\mathcal{O} \models \top \sqsubseteq \exists R.\forall R.A_2$. However, $\mathcal{O} \not\models c : A_2$. To see this, consider the interpretation $\mathcal{I}'$ with domain $\Delta^{\mathcal{I}'} = \{d_1, d_2, d_3, d_4\}$ and the following mapping $()^{\mathcal{I}'}$:*

- *$(a)^{\mathcal{I}'} = d_1$ , $(b)^{\mathcal{I}'} = d_2$ and $(c)^{\mathcal{I}'} = d_3$, for the individuals $a, b$ and $c$. $(d)^{\mathcal{I}'} = d_4$, for all other individuals $d$.*
- *$(A_1)^{\mathcal{I}} = \{d_1, d_2, d_3\}$ and $(A_2)^{\mathcal{I}} = \{d_2, d_4\}$, for the atomic concepts $A_1$ and $A_2$. $(A)^{\mathcal{I}} = \emptyset$, for all other concepts $A$.*
- *$(R)^{\mathcal{I}} = \{(d_1, d_4), (d_2, d_3), (d_2, d_4), (d_3, d_4), (d_4, d_4)\}$, for the role $R$. $(R')^{\mathcal{I}} = \emptyset$, for all other roles $R'$.*

*Then, $\mathcal{I}'$ is a model of $\mathcal{O}$ but $\mathcal{I}' \not\models c : A_2$. Note that $\mathcal{O} \not\models c : \neg A_2$, either. Take, for instance, the model $\mathcal{I}$ from Example 2.2.1.*

The last remark on the above example demonstrates the (admitted) incompleteness of $\mathcal{O}$, i.e. we make use of the open world assumption. Another characteristic of description logics is their monotonicity: it is easy to check that a knowledge base entails all consequences of its sub-knowledge bases.

Finally, knowledge bases can be compared to each other with respect to their consequences: two knowledge bases $\mathcal{O}_1$ and $\mathcal{O}_2$ are *logically equivalent* when they have the same models or, equivalently, the same logical consequences. A knowledge base $\mathcal{O}_1$ is *at least as strong as* a second knowledge base $\mathcal{O}_2$ when every consequence of $\mathcal{O}_2$ is also a consequence of $\mathcal{O}_1$. To show this, it suffices to consider only the exact inclusion axioms and assertions of $\mathcal{O}_2$ instead of every single consequence of it.

**Definition 2.2.4** *Given two knowledge bases $\mathcal{O}_1$ and $\mathcal{O}_2$, $\mathcal{O}_1$ is at least as strong as $\mathcal{O}_2$ iff for all inclusion axioms and assertions $\psi$ of $\mathcal{O}_2$, $\mathcal{O}_1 \models \psi$.*

Finally, $\mathcal{O}_1$ is *stronger than* $\mathcal{O}_2$ if it is at least as strong as $\mathcal{O}_2$ and there is a consequence of $\mathcal{O}_1$ that is not entailed by $\mathcal{O}_2$.

## 2.3 System Services

In knowledge bases that are based on description logics the following system services are generally available [BCM+03, Are00]:

- *concept satisfiability*: is $C$ satisfiable wrt. the (current) knowledge base?

- *subsumption*: is $\top \sqsubseteq C$ entailed by the knowledge base?
- *consistency check*: is the knowledge base consistent?
- *concept assertion*: is $a : C$ entailed by the knowledge base?
- *individual/concept retrieval queries*: given a concept $C$, for which individuals $a$, $a : C$ is entailed by the knowledge base? And given an individual $a$, for which atomic concepts $A$, $a : A$ is entailed by the knowledge base?

As a consequence of Theorem 2.2.2, both concept satisfiability and consistency check can be reduced to subsumption: a concept $C$ is satisfiable wrt. $\mathcal{O}$ iff $\mathcal{O} \not\models \top \sqsubseteq \neg C$. And $\mathcal{O}$ is consistent iff $\mathcal{O} \not\models \top \sqsubseteq \bot$.

The reasoning tasks on an $\mathcal{ALC}$-knowledge base are formulated below as *queries*. For the time being we do not consider concept retrieval queries. The evaluation of the queries ranges over $Ans = \{\mathsf{tt}\} \cup \mathcal{P}(\mathsf{Ind})$ where $\mathsf{tt}$ is a special constant denoting "true" (or "yes") and $\mathcal{P}(\mathsf{Ind})$ is the powerset of $\mathsf{Ind}$.

**Definition 2.3.1** *An $\mathcal{ALC}$* query *$q$ is an inclusion axiom or a concept assertion (called boolean query) or an $\mathcal{ALC}$ concept (called retrieval query). When a query $q$ is issued on a knowledge base $\mathcal{O}$ we obtained the* evaluation of $q$ with respect to $\mathcal{O}$ *($\mathsf{ans}(q, \mathcal{O})$) which ranges over $Ans$ and is determined as follows:*

$$\begin{array}{rcl}
\mathsf{ans}(\top \sqsubseteq C, \mathcal{O}) & := & \{tt\}\ ,\ if\ \mathcal{O} \models \top \sqsubseteq C, \\
\mathsf{ans}(\top \sqsubseteq C, \mathcal{O}) & := & \emptyset\ ,\ if\ \mathcal{O} \not\models \top \sqsubseteq C, \\
\mathsf{ans}(a : C, \mathcal{O}) & := & \{tt\}\ ,\ if\ \mathcal{O} \models a : C\ and\ a \in \mathsf{Ind}(\mathcal{O}), \\
\mathsf{ans}(a : C, \mathcal{O}) & := & \emptyset\ ,\ if\ \mathcal{O} \not\models a : C\ or\ a \notin \mathsf{Ind}(\mathcal{O}), \\
\mathsf{ans}(C, \mathcal{O}) & := & \{a \in \mathsf{Ind}(\mathcal{O}) \mid \mathcal{O} \models a : C\}\ .
\end{array}$$

Query answering is therefore obtained by a number of entailments. These entailments are reducible to the consistency problem as follows:

**Theorem 2.3.2** *Let $\mathcal{O}$ be a knowledge base and $n_{ew} \in \mathsf{Ind} \setminus \mathsf{Ind}(\mathcal{O})$. Then,*

1. *$\mathcal{O} \models \top \sqsubseteq C$ iff $\mathcal{O} \cup \{n_{ew} : \neg C\}$ is inconsistent and*
2. *$\mathcal{O} \models a : C$ iff $\mathcal{O} \cup \{a : \neg C\}$ is inconsistent.*

*Proof*. 1. $\mathcal{O} \models \top \sqsubseteq C$ iff for every model $\mathcal{I}$ of $\mathcal{O}$, $(C)^{\mathcal{I}} = \Delta^{\mathcal{I}}$ iff $(\neg C)^{\mathcal{I}} = \emptyset$ iff (Theorem 2.2.2) $\mathcal{O} \cup \{n_{ew} : \neg C\}$ is inconsistent.

2. We show $\mathcal{O} \not\models a : C$ iff $\mathcal{O} \cup \{a : \neg C\}$ is consistent.
($\Rightarrow$) By the assumption there is a model $\mathcal{I}$ of $\mathcal{O}$ such that $\mathcal{I} \not\models a : C$ and therefore $a^{\mathcal{I}} \in (\neg C)^{\mathcal{I}}$. But then $\mathcal{I}$ is a model of $\mathcal{O} \cup \{a : \neg C\}$.
($\Leftarrow$) By the assumption there exists a model $\mathcal{I}$ of $\mathcal{O}$ such that $\mathcal{I} \models a : \neg C$ and so $a^{\mathcal{I}} \notin (C)^{\mathcal{I}}$. Therefore $\mathcal{I}$ is a model of $\mathcal{O}$ and $\mathcal{I} \not\models a : C$. □

As a consequence of Theorem 2.3.2 we get that a query can be answered in ExpTime. Moreover, the inconsistency problem is reducible to query answering and so, query answering is an ExpTime-complete problem, too. As described already in Chapter 1, queries and their answers are presented to the users through views. Formally this is defined as follows:

**Definition 2.3.3** *An $\mathcal{ALC}$* view definition *$D_V$ is a finite set of $\mathcal{ALC}$ queries. An $\mathcal{ALC}$* view *$V$ is a set of tuples $\langle q_i, r_i \rangle$ that satisfies the following conditions:*

1. *$r_i \subseteq \mathsf{Ind}$ and finite if $q_i$ is a retrieval query,*

2. *$r_i \subseteq \{\mathit{tt}\}$ if $q_i$ is a boolean query, and*

3. *$\{\langle q, r \rangle, \langle q, r' \rangle\} \subseteq V$ implies $r = r'$.*

*A view $V$ is an instance of a view definition $D_V$ when*

$$D_V = \{q \mid \text{ there exists some } r \text{ such that } \langle q, r \rangle \in V\}$$

*Issuing the queries in $D_V$ on a consistent knowledge base $\mathcal{O}$ results to the instance $V$ (of $D_V$) for which, for every $\langle q, r \rangle \in V$, $r = \mathsf{ans}(q, \mathcal{O})$ (in symbols $\mathcal{O} \Vdash V$).*

**Example 2.3.4** *Given the knowledge base $\mathcal{O}$ from Example 2.2.1 and the view definition*

$$\begin{aligned} D_V = \{ \ & \top \sqsubseteq \exists R.\forall R.A_2, \ \top \sqsubseteq A_1 \sqcup A_2, \ \top \sqsubseteq \exists R.(\neg A_1 \sqcap A_2), \\ & b : \neg A_2, \ \exists R.\top, \ A_1 \sqcap \neg A_2, \ A_1 \sqcup A_2 \ \} \ , \end{aligned}$$

*the instance $V$ of $D_V$ for which $\mathcal{O} \Vdash V$ holds is:*

$$\begin{aligned} V = \{ & \langle\ \top \sqsubseteq \exists R.\forall R.A_2\ ,\ \{tt\}\ \rangle, \\ & \langle\ \top \sqsubseteq A_1 \sqcup A_2\ ,\ \emptyset\ \rangle, \\ & \langle\ \top \sqsubseteq \exists R.(\neg A_1 \sqcap A_2)\ ,\ \emptyset\ \rangle, \\ & \langle\ b : \neg A_2\ ,\ \emptyset\ \rangle, \\ & \langle\ \exists R.\top\ ,\ \{a, b, c\}\ \rangle, \\ & \langle\ A_1 \sqcap \neg A_2\ ,\ \emptyset\ \rangle, \\ & \langle\ A_1 \sqcup A_2\ ,\ \{a, b\}\ \rangle\ \} \end{aligned}$$

## 2.4 The Deductive System $S_{\mathcal{ALC}}$

The consistency of an $\mathcal{ALC}$-knowledge base can be decided with the help of tableaux systems [BCM$^+$03, BS01, DM00]. The labelled deductive system $S_{\mathcal{ALC}}$ presented below corresponds to the usual labelled tableaux system for $\mathcal{ALC}$-knowledge bases. It derives sequents of the form $\Gamma\ \ ;\ \ \hat{T}$ where $\Gamma$ is a multiset of assertions and $\hat{T}$ is an optional concept. Generally speaking, $\Gamma$ corresponds to the information of an ABox while $\hat{T}$ represents the information of a TBox. If such a sequent is provable in $S_{\mathcal{ALC}}$, then the corresponding knowledge base is inconsistent.

The system $S_{\mathcal{ALC}}$ consists of the following left-hand sided rules where the schematic letters $x, y$ stand for individuals, $A$ for an atomic concept, $C$ and $D$ for arbitrary concepts, and $R$ for a role.

$$\frac{}{x : A,\ x : \neg A,\ \Gamma\ \ ;\ \ \hat{T}}\ (ax),$$

$$\frac{x : \underline{\hat{T}},\ \Gamma\ \ ;\ \ \underline{\hat{T}}}{\Gamma\ \ ;\ \ \underline{\hat{T}}}\ (GCI) \quad \text{where } x \text{ appears in } \Gamma \text{ and } x : \hat{T} \notin \Gamma,$$

$$\frac{x : \underline{C},\ x : \underline{D},\ x : \underline{C \sqcap D},\ \Gamma\ \ ;\ \ \hat{T}}{x : \underline{C \sqcap D},\ \Gamma\ \ ;\ \ \hat{T}}\ (\sqcap) \quad \text{where } \{x : C,\ x : D\} \nsubseteq \Gamma,$$

$$\frac{x : \underline{C},\ x : \underline{C \sqcup D},\ \Gamma\ \ ;\ \ \hat{T} \qquad x : \underline{D},\ x : \underline{C \sqcup D},\ \Gamma\ \ ;\ \ \hat{T}}{x : \underline{C \sqcup D},\ \Gamma\ \ ;\ \ \hat{T}}\ (\sqcup)$$

where $\{x : C,\ x : D\} \cap \Gamma = \emptyset$,

$$\frac{y : \underline{C},\ (x,y) : \underline{R},\ x : \underline{\exists R.C},\ \Gamma\ \ ;\ \ \hat{T}}{x : \underline{\exists R.C},\ \Gamma\ \ ;\ \ \hat{T}}\ (\exists)$$

where $\{(x,z) : R,\ z : C\} \not\subseteq \Gamma$ for any $z$ and $y$ is fresh,

$$\frac{y : \underline{C},\ x : \underline{\forall R.C},\ (x,y) : R,\ \Gamma\ \ ;\ \ \hat{T}}{x : \underline{\forall R.C},\ (x,y) : R,\ \Gamma\ \ ;\ \ \hat{T}}\ (\forall) \quad \text{where } y : C \notin \Gamma.$$

If $a : C$ (or $(a,b) : R$) is an assertion of a sequent $S$ then $C$ (or $R$) is called *entity* of $S$ and $a$ (or $(a,b)$) is its *label*. The single concept $\hat{T}$ is also an entity of $S$. The entities that are explicitly stated in a rule are called *active entities*. The entity $\hat{T}$ is active only in $(GCI)$.

We colour every entity of a sequent by exactly one colour.[3] This is an information that is useful in view of the privacy setting and will be used later on to distinguish public information from private one. If all entities of a sequent are coloured the same, then the colour is omitted. Also, a coloured $\Gamma$ denotes that all entities of $\Gamma$ are coloured the same.

It is convenient to colour also rule applications according to the colours of their active concepts. Rule applications can be then single-coloured or mixed. A rule application is *well-coloured* if every entity that appears in the conclusion has the same colour as its duplication in the premise, and the entity that is underlined in the conclusion (as shown in the rules above) has the same colour as all underlined entities in the premise. Also, note that the side conditions of the rules apply independently of the colour of their entities.

A *coloured derivation* $\Delta$ is a tree of well-coloured rule applications. The sequent that appears at the root of $\Delta$ is its *conclusion* whereas the sequents on its leaves are its *premises*. Finally, a coloured $S_{\mathcal{ALC}}$ *proof* of a sequent $S$ is a coloured derivation in $S_{\mathcal{ALC}}$ with conclusion $S$ and all of its premises being empty.

**Definition 2.4.1** *Let $\mathcal{O}$ be a knowledge base with a non-empty* ABox $\mathcal{A}$ *and a* TBox $\mathcal{T} = \{\top \sqsubseteq C_i \mid 0 \leq i \leq n\}$*. Then, $\mathcal{O}$ is $S_{\mathcal{ALC}}$-*provable *if there is an $S_{\mathcal{ALC}}$ proof of the sequent $\mathcal{A}\ \ ;\ \ \hat{T}$ where*

$$\hat{T} = \prod_{0 \leq i \leq n} C_i \quad .$$

[3] For the printed version, instead of colouring, entities are prefixed with a symbol, e.g. $?C$ or $!C$.

The following theorem restates the well-known decision procedure result for the consistency of an $\mathcal{ALC}$-ABox with respect to an $\mathcal{ALC}$-TBox.

**Theorem 2.4.2 (see for instance [BCM⁺03])** *An $\mathcal{ALC}$-knowledge base with a non-empty* ABox *is inconsistent iff it is $S_{\mathcal{ALC}}$-provable.*

**Example 2.4.3** *Let $\mathcal{O}$ be the knowledge base from Example 2.2.1. Since $\mathcal{O} \models a : \exists R.\forall R.A_2$, the knowledge base $\mathcal{O} \cup \{a : \forall R.\exists R.\neg A_2\}$ is inconsistent and $S_{\mathcal{ALC}}$-provable. In its bi-coloured proof presented below,*

$$!\hat{T}\ (= (A_1 \sqcup \forall R.A_2) \sqcap \exists R.\neg A_1)$$

*is omitted. Assertions that are not relevant to the rule applications are also omitted. Mixed rule applications are presented uncoloured.*

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\ldots, d : !A_1,\ d : !\neg A_1,\ \ldots}\,(!ax) \qquad \dfrac{\dfrac{}{\ldots, e : !A_2,\ e : ?\neg A_2,\ (d,e) : ?R,\ \ldots}\,(ax)}{\ldots, d : !\forall R.A_2,\ e : ?\neg A_2,\ (d,e) : ?R,\ \ldots}\,(\forall)}{\ldots,\ d : !A_1 \sqcup \forall R.A_2,\ d : !\neg A_1,\ e : ?\neg A_2,\ (d,e) : ?R,\ \ldots}\,(!\sqcup)}{\ldots,\ d : !(A_1 \sqcup \forall R.A_2) \sqcap \exists R.\neg A_1,\ d : !\neg A_1,\ e : ?\neg A_2,\ (d,e) : ?R,\ \ldots}\,(!\sqcap)}{\ldots,\ d : !\neg A_1,\ e : ?\neg A_2,\ (d,e) : ?R,\ \ldots}\,(!GCI)}{\ldots,\ d : !\neg A_1,\ d : ?\exists R.\neg A_2,\ \ldots}\,(?\exists)}{\ldots,\ (a,d) : !R,\ d : !\neg A_1,\ a : ?\forall R.\exists R.\neg A_2,\ \ldots}\,(\forall)}{\ldots,\ a : !A_1 \sqcup \forall R.A_2,\ a : !\exists R.\neg A_1,\ a : ?\forall R.\exists R.\neg A_2,\ \ldots}\,(!\exists)}{a : !(A_1 \sqcup \forall R.A_2) \sqcap \exists R.\neg A_1,\ a : ?\forall R.\exists R.\neg A_2,\ \ldots}\,(!\sqcap)}{a : ?\forall R.\exists R.\neg A_2,\ a : !A_1,\ b : !A_2,\ (b,c) : !R}\,(!GCI)$$

## 2.5 Some Properties of $S_{\mathcal{ALC}}$

We now present some properties of $S_{\mathcal{ALC}}$ proofs that will be used in the next chapter.

**Lemma 2.5.1** *Let $S$ be a sequent of the form $\Gamma_x,\ \Gamma\ \ ;\ \ \hat{T}$, where $\Gamma_x$ is the least multiset of assertions in $S$ satisfying the following conditions:*

- *If $x$ is a label in $S$, then $x$ appears in $\Gamma_x$.*
- *$\Gamma_x$ has no labels in common with $\Gamma$.*

*If $S$ is $S_{\mathcal{ALC}}$-provable then the sequent $\Gamma_x \;\;;\;\; \hat{T}$ or $\Gamma \;\;;\;\; \hat{T}$ is also $S_{\mathcal{ALC}}$-provable.*

*Proof.* Let $\Pi$ be a proof of $S$. We prove the theorem by induction on the length $l$ of $\Pi$.

Base case: $l = 1$. Then $S = y : A,\ y : \neg A,\ \Delta \;\;;\;\; \hat{T}$. If $y : A \in \Gamma_x$ then also $y : \neg A \in \Gamma_x$ and so, $\Gamma_x \;\;;\;\; \hat{T}$ is provable. Otherwise, if $y : A \in \Gamma$, then also $y : \neg A \in \Gamma$. But then, $\Gamma \;\;;\;\; \hat{T}$ is provable.

Induction step. We assume that the theorem holds for proofs of length $n$. By a case analysis on the rule application $r$ of $\Pi$ that concludes $S$, we show that the theorem also holds for proofs of length $n + 1$:

- $r = GCI$. Then the premise of $r$ is $S' = y : \hat{T},\ \Gamma_x,\ \Gamma \;\;;\;\; \hat{T}$, where $y$ appears in $S$ and $y : \hat{T} \notin S$. By the definition of $S$, $y$ appears in exactly one of $\Gamma_x$ and $\Gamma$. Adding the new assertion to the multiset $y$ appears in, gives a sequent that matches the preconditions of the theorem, and $S'$ takes precisely that form. Therefore, the induction hypothesis applies to $S'$. Again, we distinguish between the possible locations of $y$:
  - If $y$ appears in $\Gamma_x$, then the induction hypothesis on $S'$ results a proof of $y : \hat{T},\ \Gamma_x \;\;;\;\; \hat{T}$ or a proof of $\Gamma \;\;;\;\; \hat{T}$. Applying $GCI$ to the first sequent results a proof of $\Gamma_x \;\;;\;\; \hat{T}$. Therefore, in both cases the theorem has been shown.
  - If $y$ appears in $\Gamma$, then the induction hypothesis on $S'$ results a proof of $\Gamma_x \;\;;\;\; \hat{T}$ or a proof of $y : \hat{T},\ \Gamma \;\;;\;\; \hat{T}$. Applying $GCI$ to the latter results a proof of $\Gamma_x \;\;;\;\; \hat{T}$ or a proof of $\Gamma \;\;;\;\; \hat{T}$, as required.

- $r = \sqcap$. Then $S = y : C_1 \sqcap C_2,\ \Delta \;\;;\;\; \hat{T}$ and $\{y : C_1, y : C_2\} \nsubseteq \Delta$. The premise of $r$ is $S' = y : C_1,\ y : C_2,\ y : C_1 \sqcap C_2,\ \Delta \;\;;\;\; \hat{T}$. As in the previous case, the induction hypothesis applies to $S'$ when the new assertions are added to the multiset that contains $y : C_1 \sqcap C_2$. We distinguish between the possible locations of these assertions:
  - If $y : C_1 \sqcap C_2 \in \Gamma_x$, then the induction hypothesis on $S'$ yields a proof of $y : C_1,\ y : C_2,\ \Gamma_x \;\;;\;\; \hat{T}$ or a proof of $\Gamma \;\;;\;\; \hat{T}$. Applying the ($\sqcap$)-rule to the first sequent yields a proof of $\Gamma_x \;\;;\;\; \hat{T}$ and completes the required results.
  - If $y : C_1 \sqcap C_2 \in \Gamma$, then the induction hypothesis on $S'$ yields a proof of $y : C_1,\ y : C_2,\ \Gamma \;\;;\;\; \hat{T}$ or a proof of $\Gamma_x \;\;;\;\; \hat{T}$. Applying the ($\sqcap$)-rule to the first sequent yields a proof of $\Gamma \;\;;\;\; \hat{T}$, as required.

- $r = \sqcup$. Then $S = y : C_1 \sqcup C_2,\ \Delta \ ;\ \hat{T}$ and $\{y : C_1, y : C_2\} \cap \Delta = \emptyset$. The premises of $r$ are the $S_1 = y : C_1,\ y : C_1 \sqcup C_2,\ \Delta \ ;\ \hat{T}$ and the $S_2 = y : C_2,\ y : C_1 \sqcup C_2,\ \Delta \ ;\ \hat{T}$. As in the previous case, the induction hypothesis applies to both $S_1$ and $S_2$ when, in each of the cases, the new assertion is added to the multiset that contains $y : C_1 \sqcup C_2$. We distinguish between the possible locations of these assertions:
  - If $y : C_1 \sqcup C_2 \in \Gamma_x$, then the induction hypothesis on $S_1$ yields a proof of $y : C_1,\ \Gamma_x \ ;\ \hat{T}$ or a proof of $\Gamma \ ;\ \hat{T}$. And the induction hypothesis on $S_2$ yields a proof of $y : C_2,\ \Gamma_x \ ;\ \hat{T}$ or a proof of $\Gamma \ ;\ \hat{T}$. Thus, there is either a proof of $\Gamma \ ;\ \hat{T}$ or there are the proofs of $y : C_1,\ \Gamma_x \ ;\ \hat{T}$ and $y : C_2,\ \Gamma_x \ ;\ \hat{T}$. Applying the ($\sqcup$)-rule to these sequents yields a proof of $\Gamma_x \ ;\ \hat{T}$ which completes the required results.
  - If $y : C_1 \sqcup C_2 \in \Gamma$, then the proof is similar to the previous case.

- $r = \exists$. Then $S = y : \exists R.C,\ \Delta \ ;\ \hat{T}$ and the premise of $r$ is $S' = z : C,\ (y,z) : R,\ y : \exists R.C,\ \Delta \ ;\ \hat{T}$, where $z$ is fresh. Since $z$ does not appear in $\Delta$, adding the new assertions to the multiset that contains $y : \exists R.C$ yields a sequent that satisfies the preconditions of the theorem, and therefore the induction hypothesis applies to $S'$. The case distinction is similar to that of the previous rules.

- $r = \forall$. Then $S = y : \forall R.C,\ (y,z) : R,\ \Delta \ ;\ \hat{T}$ and the premise of $r$ is $S' = z : C,\ y : \forall R.C,\ (y,z) : R,\ \Delta \ ;\ \hat{T}$. By the definition of $S$, $(y,z) : R$ is in the same multiset $y : \forall R.C$ is in, and $z$ does not appear in the other multiset. This implies that $S'$ satisfies the preconditions and so the induction hypothesis applies to it. The case distinction is similar to that of the previous rules. □

**Lemma 2.5.2** *Let $\Gamma$ and $\Gamma'$ be two sequents that contain the same assertions, but have different number of occurrences. If $\Gamma \ ;\ \hat{T}$ is $S_{\mathcal{ALC}}$-provable then $\Gamma' \ ;\ \hat{T}$ is also $S_{\mathcal{ALC}}$-provable.*

*Proof.* Observing the rules of $S_{\mathcal{ALC}}$, it is easy to see that the number of occurrences of each of the assertions in a sequent does not influence their applicability. Therefore, given a proof of $\Gamma \ ;\ \hat{T}$ we can construct, bottom-up, a proof of $\Gamma' \ ;\ \hat{T}$ by applying the same rules in the same order and with the same active entities. □

**Lemma 2.5.3** *If* $\Gamma \; ; \; \hat{T}$ *is* $S_{\mathcal{ALC}}$*-provable then so is* $\Gamma[x/y] \; ; \; \hat{T}$ *, where* $\Gamma[x/y]$ *is the multiset obtained by replacing all occurrences of the label* $x$ *in* $\Gamma$ *with the label* $y$*.*

*Proof.* By induction on the length $l$ of $\Pi$. Base step: $l = 1$. Then $\Pi$ consists of an (ax)-rule application and $\{z : A, z : \neg A\} \subseteq \Gamma$, for some $z$. Thus, either $\{z : A, z : \neg A\} \subseteq \Gamma[x/y]$ or $\{y : A, y : \neg A\} \subseteq \Gamma[x/y]$. Therefore, $\Gamma[x/y] \; ; \; \hat{T}$ is also provable.

Induction step: let $\Pi$ be a proof of length $l + 1$ and $(r)$ be the last rule application that concludes $\Gamma \; ; \; \hat{T}$. If $(r)$ has premise $\Gamma' \; ; \; \hat{T}$ (resp. premises $\Gamma_1 \; ; \; \hat{T}$ and $\Gamma_2 \; ; \; \hat{T}$), then by induction hypothesis we have that there is a proof of $\Gamma'[x/y] \; ; \; \hat{T}$ (resp. $\Gamma_1[x/y] \; ; \; \hat{T}$ and $\Gamma_2[x/y] \; ; \; \hat{T}$). If $r = \exists$ and the fresh label in $\Gamma'$ is $y$, we apply the induction hypothesis twice so that $y$ is at first replaced with a fresh label $z \neq y$ and only then $x$ is replaced with $y$. This will give a proof of $\Gamma'[y/z][x/y]$ and the fresh label in $\Gamma'$ will remain fresh after the replacement in $\Gamma'[y/z][x/y]$. Since all occurrences of $x$ are replaced by $y$, there are two cases:

- $(r)$ still applies to $\Gamma'[x/y] \; ; \; \hat{T}$ (resp. $\Gamma_1[x/y] \; ; \; \hat{T}$ and $\Gamma_2[x/y] \; ; \; \hat{T}$) resulting $\Gamma[x/y] \; ; \; \hat{T}$. In the case $r = \exists$ and the premise is $\Gamma'[y/z][x/y]$, $y$ does not occur in $\Gamma$ and so the conclusion is $\Gamma[y/z][x/y] = \Gamma[x/y]$, as required.

- $(r)$ is not applicable because one of its side conditions is not fulfilled. This means that when $r \neq \exists$, the entities (resp. some of the entities) that would be eliminated in case the rule were applicable, are already in $\Gamma'[x/y]$ (resp. $\Gamma_1[x/y]$ or $\Gamma_2[x/y]$) and this is possible only if they are in $\Gamma[x/y]$, too. Thus, the two sequents $\Gamma'[x/y]$ (resp. $\Gamma_1[x/y]$ or $\Gamma_2[x/y]$) and $\Gamma[x/y]$ contain the same assertions but have different number of occurrences and so, as stated in 2.5.2, $\Gamma[x/y] \; ; \; \hat{T}$ is also provable. For the case $r = \exists$, the entities posed by the side condition might not be identical to the entities that would be eliminated. These, however, become identical once we apply the induction hypothesis on $\Gamma'[x/y]$ or $\Gamma'[y/z][x/y]$ with an appropriate replacement.

□

**Lemma 2.5.4** *Let* $\Gamma \neq \emptyset$ *and* $\Pi$ *be a bi-coloured* $S_{\mathcal{ALC}}$*-proof of the sequent*

$$!\Gamma, \; ?\Delta \; ; \; !\hat{T}$$

*that has only single-coloured rule applications* $!r$ *(resp.* $?r$*). Then,* $!\Gamma \; ; \; !\hat{T}$ *(resp.* $?\Delta \; ; \;$ *) is* $S_{\mathcal{ALC}}$*-provable.*

*Proof.* By induction on the length $l$ of $\Pi$.

Base step: $l = 1$. Then $\Pi$ consists of a $(!ax)$-rule application (resp. $(?ax)$-rule application) and thus, $\{x : A, x : \neg A\} \subseteq !\Gamma$ (resp. $\{x : A, x : \neg A\} \subseteq ?\Delta$) which implies that $!\Gamma \;\; ; \;\; !\hat{T}$ (resp. $?\Delta \;\; ; \;\;$ ) is provable.

Induction step: Let $\Pi$ be a proof of length $l + 1$ and $(r)$ be the last rule application that concludes $!\Gamma, \; ?\Delta \;\; ; \;\; !\hat{T}$:

1. $r \in \{\sqcap, \exists, \forall\}$. We show how to obtain the proofs when $r = !r$. The last rule application is then of the form: $\dfrac{!\Gamma', \; ?\Delta \;\; ; \;\; !\hat{T}}{!\Gamma, \; ?\Delta \;\; ; \;\; !\hat{T}} \; (!r)$ and all active entities are in $\Gamma$ and $\Gamma'$.

   The proof of $!\Gamma', \; ?\Delta \;\; ; \;\; !\hat{T}$ has length $l$ and therefore, by induction hypothesis there is a proof of $!\Gamma' \;\; ; \;\; !\hat{T}$. The rule $!r$ is still applicable to the first sequent and this results a proof of $!\Gamma \;\; ; \;\; !\hat{T}$. Therefore, the claim holds. The case when $r = ?r$ is shown similarly.

2. $r = \sqcup$. Again, we show only the case where $r = !\sqcup$:

$$\frac{!\Gamma_1, \; ?\Delta \;\; ; \;\; !\hat{T} \qquad !\Gamma_2, \; ?\Delta \;\; ; \;\; !\hat{T}}{!\Gamma, \; ?\Delta \;\; ; \;\; !\hat{T}} \; (!\sqcup)$$

   Applying the induction hypothesis to the premises results proofs of the sequents $!\Gamma_1 \;\; ; \;\; !\hat{T}$ and $!\Gamma_2 \;\; ; \;\; !\hat{T}$. A $!\sqcup$ rule is still applicable on those sequents resulting to a proof of $!\Gamma \;\; ; \;\; !\hat{T}$, as required. The case when $r = ?\sqcup$ is shown similarly.

3. $r = GCI$. Then the rule application is of the form:

$$\frac{a : !\hat{T}, \; !\Gamma, \; ?\Delta \;\; ; \;\; !\hat{T}}{!\Gamma, \; ?\Delta \;\; ; \;\; !\hat{T}} \; (!GCI) \; ,$$

   where $a$ appears in $\Gamma \cup \Delta$. By induction hypothesis there is a proof of the sequent $a : !\hat{T}, \; !\Gamma \;\; ; \;\; !\hat{T}$. We distinguish between the possible locations of $a$:

   - $a$ appears in $\Gamma$. We apply $!GCI$ to the first sequent which results the required $!\Gamma \;\; ; \;\; !\hat{T}$.
   - $a$ does not appear in $\Gamma$. Since $\Gamma \neq \emptyset$, there is a label $b$ occurring in $\Gamma$. Applying Lemma 2.5.3 to $a : !\hat{T}, \; !\Gamma \;\; ; \;\; !\hat{T}$ with the substitution [a/b] we get a proof of $b : !\hat{T}, \; !\Gamma \;\; ; \;\; !\hat{T}$. An application of the $!GCI$ to the latter results to a proof of $!\Gamma \;\; ; \;\; !\hat{T}$, as required. $\square$

**Lemma 2.5.5** *Let $\mathcal{O}$ be a consistent knowledge base and $\mathcal{O} \cup \{a : C\}$ (resp. $\mathcal{O} \cup \{(a, b) : R\}$) be an inconsistent one. If $\Pi$ is an $S_{\mathcal{ALC}}$ proof of the sequent that corresponds to the latter, then $a : C$ (resp. $(a, b) : R$) is an active entity in the conclusion of a rule application of $\Pi$.*

*Proof.* Assume that $a : C$ (resp. $(a, b) : R$) is not an active entity in the conclusion of a rule of $\Pi$. Then, every rule applied to a sequent $S$ of $\Pi$ applies also to the sequent obtained by removing $a : C$ (resp. $(a, b) : R$) from $S$. Therefore, we can construct a proof of a sequent that corresponds to $\mathcal{O}$ by simply removing $a : C$ (resp. $(a, b) : R$) from the sequents of $\Pi$ and therefore, $\mathcal{O}$ is inconsistent - a contradiction. □

# Chapter 3

# Data Privacy in $\mathcal{ALC}$ Knowledge Bases

This chapter includes the main results of the thesis, namely, the ExpTime decidability of the privacy problems in $\mathcal{ALC}$ knowledge bases, as well as the PTime procedure that identifies only some of the data preserving cases. The definitions of data privacy presented in Chapter 1 apply here unchanged. A repository is now a knowledge base and, therefore, $R$ is denoted instead by $\mathcal{O}$ and $\mathcal{R}_{bg}$ by $\mathcal{O}_{bg}$. Note that, in this setting $\mathcal{O}_{bg}$ is not reducible to $V$ since role assertions are not expressible in views.

We first show that data privacy on views reduces to a finite number of entailments. More precisely, it reduces to the evaluation of the confidential query on a consistent knowledge base. The ExpTime completeness of this problem is then demonstrated. After that, we show that data privacy on view definitions can be decided by considering only a finite number of views. In particular, these views are all the valid instances that have at most one fresh individual. The ExpTime completeness of this problem is then easily shown. We conclude with the syntactic criterion on concepts and roles that is sufficient (but not necessary) for data privacy preservation. The results on view definitions are obtained with the help of system $S_{\mathcal{ALC}}$.

## 3.1 Deciding Privacy on Views

When applied to $\mathcal{ALC}$ knowledge bases, the problem of data privacy on views reduces to query answering on the *canonical* knowledge base. This knowledge base expresses precisely the information contained in $\mathcal{O}_{bg}$ and $V$.

**Definition 3.1.1** *Given a knowledge base $\mathcal{O}_{bg}$ and a view $V$, the canonical knowledge base $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$ is defined as*

$$\begin{aligned}\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} := \mathcal{O}_{bg} \cup &\\ &\{\top \sqsubseteq C \mid \langle \top \sqsubseteq C, \{\mathit{tt}\}\rangle \in V\} \cup\\ &\{a : C \mid \langle a : C, \{\mathit{tt}\}\rangle \in V\} \cup\\ &\{a : C \mid \textit{there is a set } \mathsf{In} \textit{ with } \langle C, \mathsf{In}\rangle \in V \textit{ and } a \in \mathsf{In}\}.\end{aligned}$$

Note that whenever $\langle\mathcal{O}_{bg}, V\rangle$ is valid, $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$ is a consistent knowledge base.

**Theorem 3.1.2** *Given a valid tuple $\langle\mathcal{O}_{bg}, V\rangle$, data privacy is preserved for a query $q$ wrt. $\langle\mathcal{O}_{bg}, V\rangle$ if and only if $\mathsf{ans}(q, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}) = \emptyset$.*

We first prove a proposition that we will use in the proof of Theorem 3.1.2.

**Proposition 3.1.3** *For a valid tuple $\langle\mathcal{O}_{bg}, V\rangle$ the following hold:*

1. *every $\mathcal{O} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$ is at least as strong as $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$.*
2. *for every $\mathcal{O} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$, $\mathsf{Ind}(\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}) \subseteq \mathsf{Ind}(\mathcal{O})$.*
3. *$\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$.*

*Proof*. 1. and 2. Since $\mathcal{O} \Vdash V$ and $\mathcal{O}_{bg} \subseteq \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$, every $\phi \in \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$ is entailed by an arbitrary $\mathcal{O} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$ and every $a \in \mathsf{Ind}(\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$ is also in $\mathcal{O}$. 3. The consistency of $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$ follows from that $\langle\mathcal{O}_{bg}, V\rangle$ is a valid tuple. It is also easy to see that $\mathcal{O}_{bg} \subseteq \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$. It remains to show that $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \Vdash V$. We distinguish between the possible tuples in $V$:

- For every $\langle \top \sqsubseteq C, \{tt\}\rangle \in V$, $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \models \top \sqsubseteq C$ by the definition of $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$.
- For every $\langle \top \sqsubseteq C, \emptyset\rangle \in V$ we have $\mathcal{O} \not\models \top \sqsubseteq C$, for $\mathcal{O} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$. Because of (1), this implies $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \not\models \top \sqsubseteq C$.
- For every $\langle a : C, \{tt\}\rangle \in V$, $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \models a : C$ by the definition of $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$.
- For every $\langle a : C, \emptyset\rangle \in V$ we have $\mathcal{O} \not\models a : C$ or $a \notin \mathsf{Ind}(\mathcal{O})$, for $\mathcal{O} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$. By (1) and (2) this implies $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \not\models a : C$ or $a \notin \mathsf{Ind}(\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$.

- For every $\langle C, \mathsf{In}\rangle \in V$ we have $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \models a : C$, for every $a \in \mathsf{In}$. For every $b \notin \mathsf{In}$ there are two cases: $b \in \mathsf{Ind}(\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$ or $b \notin \mathsf{Ind}(\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$. We show that in both cases $b \notin \mathsf{ans}(C, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$.

  (i) If $b \in \mathsf{Ind}(\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$ then because of (2) $b \in \mathsf{Ind}(\mathcal{O})$ too, for every $\mathcal{O} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$. Since in addition $b \notin \mathsf{In}$, we have $\mathcal{O} \not\models b : C$. By (1) this implies $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \not\models b : C$, too. Therefore, $b \notin \mathsf{ans}(C, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$.

  (ii) If $b \notin \mathsf{Ind}(\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$, then by the definition of $\mathsf{ans}()$ we also have $b \notin \mathsf{ans}(C, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$.

  Therefore, $\mathsf{In} = \mathsf{ans}(C, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$. □

*Proof of theorem.* We show that

$$\mathsf{certain}(q, \langle\mathcal{O}_{bg}, V\rangle) = \emptyset \quad \text{iff} \quad \mathsf{ans}(q, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}) = \emptyset$$

($\Rightarrow$) We distinguish between the possible queries.

(i) If $q = \top \sqsubseteq C$ then $\mathsf{certain}(q, \langle\mathcal{O}_{bg}, V\rangle) = \emptyset$ implies that there is an $\mathcal{O} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$ with $\mathcal{O} \not\models \top \sqsubseteq C$. By Proposition 3.1.3 (1) this implies that $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \not\models \top \sqsubseteq C$. Therefore, $\mathsf{ans}(q, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}) = \emptyset$.

(ii) If $q = a : C$ then $\mathsf{certain}(q, \langle\mathcal{O}_{bg}, V\rangle) = \emptyset$ implies that there is an $\mathcal{O} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$ with $\mathcal{O} \not\models a : C$ or $a \notin \mathsf{Ind}(\mathcal{O})$. By Proposition 3.1.3 (1) and (2) we have $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \not\models \top \sqsubseteq C$ or $a \notin \mathsf{Ind}(\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$. Therefore, $\mathsf{ans}(q, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}) = \emptyset$.

(iii) If $q = C$ then $\mathsf{certain}(q, \langle\mathcal{O}_{bg}, V\rangle) = \emptyset$ implies that for every $a \in \mathsf{Ind}$ there is an $\mathcal{O} \in \mathsf{Poss}_{\langle\mathcal{O}_{bg},V\rangle}$ with $\mathcal{O} \not\models a : C$. By Proposition 3.1.3 (1), $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle} \not\models a : C$, for every $a \in \mathsf{Ind}$. Thus, $\mathsf{ans}(q, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}) = \emptyset$.

($\Leftarrow$) Since by Proposition 3.1.3 (3) $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$ is a possible knowledge base, $\mathsf{ans}(q, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}) = \emptyset$ implies that $\mathsf{certain}(q, \langle\mathcal{O}_{bg}, V\rangle) = \emptyset$. □

### The complexity of data privacy on views

According to Definition 2.3.1, $\mathsf{ans}(q, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle})$ can be computed by a number of entailments which is polynomial to the size of $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$. As it has already been stated, the entailment problem is reducible to the consistency problem which is solvable in ExpTime. Moreover, $\mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}$ grows polynomially wrt. $\mathcal{O}_{bg}$ and $V$ which implies that deciding $\mathsf{ans}(q, \mathcal{C}_{\langle\mathcal{O}_{bg},V\rangle}) = \emptyset$ is not harder than deciding the entailment problem on a consistent knowledge base. Therefore, Theorem 3.1.2 provides an ExpTime decision procedure to the problem of data privacy on views.

The problem is also ExpTime-hard as the problem of concept unsatisfiability wrt. a consistent TBox (see below) is polynomially reducible to the problem of data privacy.

**Proposition 3.1.4** *A concept $C$ is unsatisfiable wrt. a consistent* TBox $\mathcal{T}$ *iff data privacy for $\top \sqsubseteq \neg C$ wrt. $\mathcal{T}$ and the empty view is not preserved.*

*Proof.* A concept $C$ is unsatisfiable wrt. $\mathcal{T}$ iff for every model $\mathcal{I}$ of $\mathcal{T}$, $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}}$ iff $\mathcal{T} \models \top \sqsubseteq \neg C$ iff $\mathsf{ans}(\top \sqsubseteq \neg C, \mathcal{C}_{\langle \mathcal{T}, \emptyset \rangle}) = \{tt\}$ iff data privacy for $\top \sqsubseteq \neg C$ wrt. $\mathcal{T}$ and the empty view is not preserved. □

Concept unsatisfiability wrt. an arbitrary TBox is proved to be ExpTime-hard for instance in [BCM$^+$03] and [Hof05]. In [BCM$^+$03] this has been shown by reducing a known ExpTime-hard problem from the area of graph theory to the unsatisfiability problem. The problem used can be stated as follows: given an AND-OR graph in the form of a circuit and some source nodes, decide whether a target node (of the graph) is reachable (from some sources). The reduction in question is obtained via an appropriate TBox construction. However, the TBox is not explicitly restricted to be consistent.

In [Hof05] the subsumption problem wrt. a TBox in a logic weaker than $\mathcal{ALC}$ is shown to be ExpTime-hard. This has been obtained via a reduction to a game theoretical problem. The TBox constructed there is consistent and a model of it is explicitly given.

In the following we show that a slight variation of the TBox presented in [BCM$^+$03] is also consistent. We begin with a presentation of the original TBox which represents the circuit. If the graph has $n$ nodes and each of them has at most $d$ predecessors, then the circuit has $\log n$ inputs and $1 + d \log n$ outputs. The inputs of the circuit are denoted by the concepts $A_1, \ldots, A_{\log n}$ and are the binary encoding of a node. Therefore, each of the $A_i$ s stands for those nodes that have their $i$-th digit 1 and so, every node is represented uniquely with a conjunction over positive and negative $A_i$ s. For example, the node $n_2$ corresponds to the concept $\neg A_1 \sqcap \ldots \sqcap \neg A_{(\log n)-2} \sqcap A_{(\log n)-1} \sqcap \neg A_{\log n}$. We denote this by $Conc(n_2)$. The outputs of the circuit are denoted by the concepts $B^1_1, \ldots, B^1_{\log n}, \ldots, B^d_1, \ldots, B^d_{\log n}$ and are the binary encodings of the (at most) $d$ predecessors of the input node. If a node has $d' < d$ predecessors, then the first $d'$ sequences of $B^i_j$ s represent its predecessors and the remaining are set to zero.

There is one additional output denoted by the concept $A_{nd}$ which returns 1 when the input node is an AND node and 0 when it is an OR node. Note that, since only nodes with a predecessor belong to a type, the value of this output is omitted when it is not relevant. We additionally use concepts $W_1, \ldots, W_k$ to code the $k$ internal gates of the circuit and the concepts $AND$ and $OR$ for the type of the input node.

Let $X_j$ and $X_k$ be two schematic concepts that stand for input concepts or gate concepts. The TBox $\mathcal{T}_C$ is then constructed as follows[1]:

$$\begin{array}{rcl}
Conc(n_i) \sqsubseteq \bot & & \text{for every source node } n_i \ , \\
W_i &=& X_j \sqcap X_k \quad \text{for every } \wedge\text{- gate } i \text{ with inputs } X_j \text{ and } X_k \ , \\
W_i &=& X_j \sqcup X_k \quad \text{for every } \vee\text{- gate } i \text{ with inputs } X_j \text{ and } X_k \ , \\
W_i &=& \neg X_j \qquad \text{for every } \neg \text{ - gate } i \text{ with input } X_j \ , \\
B^i_j &=& X_k \qquad \text{for every output } B^i_j \ , \\
AND &=& A_{nd} \sqcap (B^1_1 \sqcup \ldots \sqcup B^1_{\log n}) \ , \\
OR &=& \neg A_{nd} \sqcap (B^1_1 \sqcup \ldots \sqcup B^1_{\log n}) \ , \\
AND &\sqsubseteq& \exists R^1.\top \sqcup \ldots \sqcup \exists R^d.\top \ , \\
OR &\sqsubseteq& \exists R^1.\top \sqcap \ldots \sqcap \exists R^d.\top \quad \text{and,} \\
B^i_j &\sqsubseteq& \forall R^i.A_j \qquad \text{and} \qquad \neg B^i_j \ \sqsubseteq \ \forall R^i.\neg A_j \quad \text{for every output } B^i_j \ .
\end{array}$$

The TBox $\mathcal{T}_C$ describes correctly the behaviour of the unreachable nodes of the graph. Instead of showing the consistency of $\mathcal{T}_C$, we show the consistency of a slight variation of it in which the zero node is explicitly excluded from the $AND$ and $OR$ concepts. The inclusion axioms for $AND$ and $OR$ are modified as follows:

$$AND = A_{nd} \sqcap (B^1_1 \sqcup \ldots \sqcup B^1_{\log n}) \sqcap (A_1 \sqcup \ldots \sqcup A_{\log n})$$

$$OR = \neg A_{nd} \sqcap (B^1_1 \sqcup \ldots \sqcup B^1_{\log n}) \sqcap (A_1 \sqcup \ldots \sqcup A_{\log n})$$

This modification does not affect the proof and the resulted TBox, say $\mathcal{T}'_C$, represents correctly the circuit. A simple model $\mathcal{I}$ of $\mathcal{T}'_C$ can be then constructed as follows:

- $\Delta^I = \{n_0\}$,
- all atomic concepts and roles are mapped to the empty set, except for
- the concepts $W_1, \ldots, W_k$ and all output concepts $B^i_j$ which are mapped either to the empty set or to $\Delta^I$ - according to the output (0 or 1) of their corresponding gates and outputs, when all inputs are set to 0.

[1] As part of the TBox in [BCM+03] is only informally described, some of the axioms might not be the original ones. This is, however, the TBox that best fits to the description.

It is easy to check that this interpretation is indeed a model of $\mathcal{T}'_C$.

**Corollary 3.1.5** *The problem of $\mathcal{ALC}$ data privacy for a query wrt. a view and a knowledge base is ExpTime-complete.*

## 3.2 Deciding Privacy on View Definitions

The problem of data privacy on view definitions is decidable when applied to $\mathcal{ALC}$ knowledge bases, since it is enough to consider only the views entailed by a finite set of knowledge bases $\mathbb{P}$.

**Definition 3.2.1** *Given a tuple $\langle \mathcal{O}_{bg}, D_V \rangle$, $\mathsf{Ind}_g$ is the set of given individuals (i.e. $\mathsf{Ind}_g = \mathsf{Ind}(\mathcal{O}_{bg} \cup \{a : C \in D_V\})$). Let an individual $n_{ew} \notin \mathsf{Ind}_g$. Then, a knowledge base $P \in \mathbb{P}$ if*

1. *$P \supseteq \mathcal{O}_{bg}$ and consistent,*
2. *if $\top \sqsubseteq C \in P$ then $\top \sqsubseteq C \in (\mathcal{O}_{bg} \cup D_V)$, and*
3. *if $a : C \in P$ then $a : C \in \mathcal{O}_{bg} \cup D_V$ or ($a \in \mathsf{Ind}_g \cup \{n_{ew}\}$ and $C \in D_V$).*

In Theorem 3.2.3 we show that data privacy is preserved on a view definition if it is preserved on every view entailed by some $P \in \mathbb{P}$. We first prove the following lemma:

**Lemma 3.2.2** *Let $\mathbb{P}$ be the set of knowledge bases constructed wrt. a tuple $\langle \mathcal{O}_{bg}, D_V \rangle$ and an individual $n_{ew}$. If $P \in \mathbb{P}$ and $V$ is the instance of $D_V$ entailed by $P$, then $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ and $P$ are logically equivalent.*

*Proof*. First, we show that every element of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ is entailed by $P$ and therefore $P$ is at least as strong as $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$. Since $P \supseteq \mathcal{O}_{bg}$, the elements of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ that come from $\mathcal{O}_{bg}$ are entailed by $P$. The rest of the elements come from $V$ which, by definition, is a view entailed by $P$ and so each of these elements is also entailed by $P$.

Second, we show that $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \supseteq P$ and therefore $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ is at least as strong as $P$. Since $\mathcal{O}_{bg} \subseteq \mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$, the elements of $P$ that come from $\mathcal{O}_{bg}$ are also in $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$. The rest of the elements come from $D_V$. Now, since $V$ is the instance of $D_V$ entailed by $P$, we have that for every inclusion axiom $\top \sqsubseteq C \in P \setminus \mathcal{O}_{bg}$ there is a tuple $\langle \top \sqsubseteq C, \{\mathsf{tt}\} \rangle \in V$. Similarly, for every assertion $a : C \in P \setminus \mathcal{O}_{bg}$ there is either a tuple $\langle C, \mathsf{In} \rangle \in V$ with $a \in \mathsf{In}$, or a tuple $\langle a : C, \{\mathsf{tt}\} \rangle \in V$. Therefore, these elements are also in $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$. □

**Theorem 3.2.3** *Data privacy is preserved for $q$ wrt. a tuple $\langle \mathcal{O}_{bg}, D_V \rangle$ if and only if, for every instance $V$ of $D_V$ that is entailed by some $P \in \mathbb{P}$, data privacy is preserved for $q$ wrt. $\langle \mathcal{O}_{bg}, V \rangle$.*

*Proof.* ($\Rightarrow$) Trivial, as every $V$ entailed by $P$ is a relevant view.
($\Leftarrow$) We prove the contrapositive. Assume that $V$ is a view based on $\langle \mathcal{O}_{bg}, D_V \rangle$ on which $q$ is not preserved. As a consequence of Theorem 3.1.2, $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \models q$, if $q$ is boolean or $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \models d : q$, for some $d \in \mathsf{Ind}(\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle})$, if $q$ is retrieval. Let $\mathcal{T}$ be the TBox of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ and $\mathcal{A}$ its ABox. We show that $q$ (resp. $d : q$) is entailed by a subset of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ that contains at most one additional individual (i.e. an individual that appears neither in $\mathcal{O}_{bg}$ nor in $D_V$). Assume that there are more than one such individuals appearing in $V$. We distinguish between the possible forms of $q$:

- $q = \top \sqsubseteq C$. We show that $\mathcal{T} \models \top \sqsubseteq C$. Since $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \models \top \sqsubseteq C$, $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \cup \{a : \neg C\}$ is inconsistent, for $a \notin \mathsf{Ind}(\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle})$. Therefore, there is a proof of $a : \neg C,\ \mathcal{A}\ ;\ \hat{T}$ in $S_{\mathcal{ALC}}$ where, $\hat{T}$ is the concept that represents all inclusion axioms of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$. Since $a$ does not appear in $\mathcal{A}$, by Lemma 2.5.1 we get a proof of $a : \neg C\ ;\ \hat{T}$ or a proof of $\mathcal{A}\ ;\ \hat{T}$. While the latter is not possible because it implies that $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ is inconsistent, the first proof implies that $\mathcal{T} \cup \{a : \neg C\}$ is inconsistent and so, by Theorem 2.3.2, $\mathcal{T} \models \top \sqsubseteq C$.

- $q = C$. Adding $d : \neg C$ to $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ would cause inconsistency and so, there is a proof $\Pi$ of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \cup \{d : \neg C\}$ in $S_{\mathcal{ALC}}$. Let $\Gamma_x$ be the set of assertions of one of the additional individuals $x \neq d$. Note that $x$ does not appear in any role assertion in $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$. Therefore, $\Gamma_x$ contains only concept assertions and Lemma 2.5.1 applies to $\Pi$ with such a $\Gamma_x$. This gives either a proof of $\Gamma_x\ ;\ \hat{T}$ or a proof of $\Gamma,\ d : \neg C\ ;\ \hat{T}$, where $\hat{T}$ is the concept that represents $\mathcal{T}$ and $\Gamma = \mathcal{A} \backslash \Gamma_x$. While the first proof is not possible since it would imply that $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ is inconsistent, the second proof implies that there is a subset of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ with one additional individual less, that also entails $d : q$ (by Theorem 2.3.2). Applying the lemma iteratively to the above proof results a knowledge base that contains at most one additional individual.

- $q = a : C$. This case is similar to the case $q = C$.

Now, let $C' \subseteq \mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ be the obtained knowledge base that has at most one additional individual $x$. Renaming every occurrence of $x$ in $C'$ by $n_{ew}$ results in a knowledge base, say $C^r$, which is equivalent to $C'$ modulo individual renaming. Therefore $C^r$ also entails some private data, and so does $C^r \cup \mathcal{O}_{bg}$, too. The latter is a knowledge base in $\mathbb{P}$. Let $V^r$ be the instance of $D_V$ that is entailed by $C^r \cup \mathcal{O}_{bg}$. Then, Lemma 3.2.2 results that $\mathcal{C}_{\langle \mathcal{O}_{bg}, V^r \rangle}$ is equivalent to $C^r \cup \mathcal{O}_{bg}$ and so, data privacy for $q$ is not preserved on $V^r$ either. □

### The complexity of data privacy on view definitions

A naive ExpTime decision procedure for this problem can be constructed directly from the above theorem: first compute $\mathbb{P}$ and all views entailed by its elements, and then decide data privacy on each of these views. Let $P^+$ be the knowledge base constructed from $\mathcal{O}_{bg}$ and $D_V$ as follows:

$$\begin{aligned} P^+ = \ & \{\top \sqsubseteq C \in D_V\} \cup \\ & \{a : C \in D_V\} \cup \\ & \{a : C \mid (a \in \mathsf{Ind}_g \text{ or } a = n_{ew}) \text{ and } C \in D_V\}. \end{aligned}$$

Then, $\mathbb{P}$ can be constructed by first computing all subsets of $P^+$ and then checking their consistency wrt. $\mathcal{O}_{bg}$. Since $P^+$ can be constructed polynomially wrt. the size of $\mathcal{O}_{bg}$ and $D_V$, there are at most $2^{p(n)}$ subsets of $P^+$ of maximal cardinality $p(n)$, where $n$ is the total size of $\mathcal{O}_{bg}, D_V$ and $q$. Since consistency is decidable in ExpTime, computing $\mathbb{P}$ stays in ExpTime. Now, in order to compute the views entailed by some $P \in \mathbb{P}$, a polynomial number of entailments on every $P \in \mathbb{P}$ is required. Therefore the computation of all views stays also in ExpTime. Finally, Corollary 3.1.5 together with the fact that $V$ grows polynomially wrt. the size of $D_V$ and $P$, imply that the total time required for checking privacy on all of the (at most) exponentially many views is again exponential wrt. $n$.

The problem of data privacy on view definitions is also ExpTime-hard as the corresponding problem on views is polynomially reducible to this problem: data privacy for $q$ is preserved wrt. $\mathcal{O}_{bg}$ and $V$ iff it is preserved wrt. $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ and the empty view definition.

**Corollary 3.2.4** *The problem of $\mathcal{ALC}$ data privacy on view definitions is ExpTime-complete.*

## 3.3 An Efficient Condition for Privacy

In the sequel we present a condition on $\mathcal{O}_{bg}, D_V$ and $q$ which can be decided in PTime and implies data privacy for $q$ wrt. $\langle \mathcal{O}_{bg}, D_V \rangle$. Thus, we have a sufficient condition for data privacy that can be checked efficiently. It is based on the syntactic structure of the concepts that constitute the background knowledge and the view definition.

Because of the syntactic nature of this method, we first need to exclude some "common sense" queries from being secrets. These queries have trivial answers that hold on any knowledge base (including the empty knowledge base) and can be therefore, guessed by the user, whatever the syntactic appearance of the query is. An inclusion axiom is dangerous when it is tautological, that is, it can be answered positively on the empty knowledge base. A tautological concept is, however, dangerous only when at least one individual is (potentially) exhibited. In the definition below, this is controlled by imposing a condition on retrieval and assertional queries. In this way, the privacy of the query that expresses the secrecy of all individuals of the system is not excluded.

**Definition 3.3.1** *A* query $q$ is trivial wrt. a tuple $\langle \mathcal{O}_{bg}, D_V \rangle$ *when*

- $ans(\top \sqsubseteq C, \emptyset) = \{\mathit{tt}\}$*, if* $q = \top \sqsubseteq C$*.*
- $ans(\top \sqsubseteq C, \emptyset) = \{\mathit{tt}\}$*, if* $q \in \{C, a : C\}$ *and the following holds:*

$$\begin{aligned}(\mathsf{Ind}(\mathcal{O}_{bg}) = \emptyset) \Rightarrow ((\exists_{C' \in D_V}\ \mathcal{O}_{bg} \not\models \top \sqsubseteq \neg C') \vee \\ (\exists_{a:C' \in D_V}\ \mathcal{O}_{bg} \not\models \top \sqsubseteq \neg C'))\ .\end{aligned}$$

*An* $\mathcal{ALC}$ *query qualifies as a* privacy condition on a tuple $\langle \mathcal{O}_{bg}, D_V \rangle$ *if it is not trivial wrt.* $\langle \mathcal{O}_{bg}, D_V \rangle$*.*

Next, we define the boolean function $s_{afe}()$ that decides whether a concept $D$ or a role $R$ exhibits some information about $q$. Given a knowledge base $\mathcal{O}_{bg}$, a view definition $D_V$ and a privacy condition $q$ on $\langle \mathcal{O}_{bg}, D_V \rangle$, the information about a concept $D$ is *safe* if $s_{afe}(D, q)$ returns 1; and the information of a role $R$ is safe if $s_{afe}(R, \langle \mathcal{O}_{bg}, D_V, q \rangle)$ returns 1.

The following conventions apply to the definition of $s_{afe}()$. *Concepts and roles of a tuple* $\langle \mathcal{O}_{bg}, D_V \rangle$ are all inclusion and assertional concepts, assertional roles and retrieval queries that appear in $\mathcal{O}_{bg}$ or $D_V$. The set of *subterms* $s(C)$ of a concept $C$ is inductively defined by:

$$\begin{aligned} s(A) :=& \{A\} & s(\neg A) :=& \{\neg A\} \\ s(C \star D) :=& \{C \star D\} \cup s(C) \cup s(D) & s(QR.C) :=& \{QR.C\} \cup s(C) \end{aligned}$$

where $\star$ is either $\sqcup$ or $\sqcap$ and $Q$ is either $\forall$ or $\exists$. Note that negated atomic concepts are not decomposable. For instance, the subterms of $A_1 \sqcup \exists R.\neg A_2$ are $A_1, \neg A_2, \exists R.\neg A_2$ and $A_1 \sqcup \exists R.\neg A_2$.

If a concept $C_2$ has a subterm $C_1$ then $C_2$ is also written as $C_2[C_1]$. If, in addition, there is an occurrence of $C_1$ in $C_2$ that is not prefixed by a quantifier, then $C_2$ may also be written as $C_2[C_1]^0$. Similarly, if we want to emphasize that $C_1$ is not prefixed in $C_2$ by an existential quantifier, then $C_2$ may also be written as $C_2[C_1]^{0^\exists}$. For example, the concept $A_1 \sqcup \forall R_2.(\neg A_2)$ can be also written as $A_1 \sqcup \forall R_2.(\neg A_2)[\neg A_2]$ or $A_1 \sqcup \forall R_2.(\neg A_2)[\neg A_2]^{0^\exists}$ but not as $A_1 \sqcup \forall R_2(\neg A_2)[\neg A_2]^0$ .

We also introduce the notion of similarity on concepts: two concepts $C_1$ and $D_1$ are *similar* when either (i) $C_1 = D_1$ or (ii) $C_1 = \forall R.C_2$ and $D_1 = \forall R.D_2$ or (iii) $C_1 = \exists R.C_2$ and $D_1 = \exists R.D_2$.

Now, assume we are given a query $q^{\mathrm{C}} \in \{\top \sqsubseteq C, C, a : C\}$. The function $s_{afe}()$ is defined on concepts and roles as follows:

For a concept $D$, $s_{afe}(D, q^{\mathrm{C}}) = 1$ iff there are no similar $D_1$ and $C_1$ subterms of $D$ and $C$, respectively, such that either

1. $D[D_1]^0$ and $C[C_1]^{0^\exists}$ hold, or
2. $D[D_1]^0$, $C[\exists R.C'[C_1]]^{0^\exists}$ and $C[\forall R.C'']$ hold.

For a role $R$ and a tuple $\langle \mathcal{O}_{bg}, D_V \rangle$, $s_{afe}(R, \langle \mathcal{O}_{bg}, D_V, q^{\mathrm{C}} \rangle) = 1$ iff:

1. $C$ is not of the form $C[\exists R.C']^0$ and
2. for every concept $D_2$ for which $D_1[\forall R.D_2]^{0^\exists}$ is a concept of $\langle \mathcal{O}_{bg}, D_V \rangle$, $s_{afe}(D_2, q^{\mathrm{C}}) = 1$.

Note that for the needs of the function $s_{afe}()$ the first condition on similarity can be restricted to atomic and negated atomic concepts only.

Using the above function, the privacy of a query is guaranteed when all concepts and roles of the background knowledge and the view definition are safe. As we already mentioned this solution is partial.

**Example 3.3.2** *We can correctly detect that data privacy is preserved for $A$ wrt. $\langle \{R_1(a,b), R_2(b,c)\}, \{\forall R_1 \exists R_2 A\} \rangle$. However, we cannot detect that data privacy is preserved for $A$ wrt. $\langle \{R_1(a,b), R_2(c,d)\}, \{\forall R_1 \forall R_2 A\} \rangle$ or even for $A \sqcap B$ wrt. $\langle \emptyset, \{A\} \rangle$. In the first case this is because we do not take care of the individuals at all. As a consequence, the data privacy of $a : C$ and $C$ is indistinguishable. In the second case (when $q = A \sqcap B$) this is because we do not check whether one of the conjuncts forms a trivial query.*

**Example 3.3.3** *Both cases presented in Example 1.3.1 illustrate the correctness of the solution. That is, we can detect that data privacy is preserved for the privacy condition* Bunker *wrt. $\langle \mathcal{R}_{bg}, D_{V_1} \rangle$. And we cannot detect data* *privacy for* Bunker *wrt.* $\langle \mathcal{R}_{bg}, D_{V_1} \cup D_{V_2} \rangle$*, as the second query of* $D_{V_2}$ *cancels the safeness of the assertional role* Includes*. The limitations of the solution emerges when one modifies the latter example by setting* $\mathcal{R}_{bg} = R$*. Now, privacy is preserved for* Bunker *wrt.* $\langle R, D_{V_1} \cup D_{V_2} \rangle$ *but the functions still return the same values and so,* Includes *cannot be shown to be safe.*

In the next chapter we will see an application of this solution to modular ontologies. We now prove the correctness of this solution:

**Lemma 3.3.4** *Assume that we are given a query* $q^c$ *and a bi-coloured* $S_{\mathcal{ALC}}$ *proof* $\Pi$ *of a sequent* $S_1 = d : ?\neg C,\ !\Gamma \ \ ; \ \ !\hat{T}$*. Furthermore, assume that*

*(i)* $s_{afe}(!R, \langle \Gamma \cup \{\top \sqsubseteq \hat{T}\}, \emptyset, q^c \rangle) = s_{afe}(!D, q^c) = 1$*, for all entities* $!R$ *and* $!D$*, respectively, in* $S_1$*.*

*Let* $S_2$ *be a sequent in* $\Pi$ *of the form*

*(ii)* $x : ?rd,\ x : !gr,\ \Delta \ \ ; \ \ !\hat{T}$ *with*

*(iii)* $?rd = C_1[\neg C_2]^0$ *,* $!gr = D_1[D_2]^0$ *and,* $C_2$ *and* $D_2$ *are similar.*

*Then, there is a mixed-rule application in the path between* $S_1$ *and* $S_2$*.*

*Proof*. By induction on the length $n$ of the path between $S_1$ and $S_2$. Base case: $n = 0$. Then $S_1 = S_2$ and so $C = \neg rd$. Therefore, $C$ is of the form $C[C_2]^{0^{\exists}}$. Since $C_2$ and $D_2$ are similar, by the definition of $s_{afe}()$ on concepts (first condition) and the form of $gr$, $s_{afe}(gr, q^{\neg rd}) = 0$ and so, (i) is contradicted. Therefore, this is not possible.

Induction step: assume that there are $n+1$ rule applications between $S_1$ and $S_2$ and that all of them are single-coloured. Let $r$ be the rule application with premise $S_2$ and conclusion $S_2'$. By a case analysis on $r$ we show that in all possible cases, $S_2'$ satisfies (ii) and (iii). Thus by the induction hypothesis there is a mixed-rule application between $S_1$ and $S_2$.

If both $?rd$ and $!gr$ are in $S_2'$ then $S_2'$ satisfies (ii) and (iii). Otherwise, one of the two is an active entity in $S_2$ that does not appear in $S_2'$. There are the following cases on $r$:

- $r = (?GCI)$. This case is not possible.

- $r = (!GCI)$. Then $S_2' = x : ?rd,\ \Delta \ \ ; \ \ !gr$. Since $?rd$ appears in $S_2'$, by the form of $S_1$ we have that $\neg rd$ is a subterm of $C$. There are two cases on $C$:

- $C = C[\neg rd]^{0^{\exists}}$. Then $C$ is also of the form $C[C_2]^0$ and so by (iii) and the definition of $s_{afe}()$, $s_{afe}(!gr, q^{\mathrm{C}}) = 0$, which contradicts (i) ($!gr$ is an entity in $S_1$).
- $C$ is of the form $C[\exists R.C'[\neg rd]]^{0^{\exists}}$ and not of the form $C[\neg rd]^{0^{\exists}}$. This implies that $z : ?\forall R.\neg C'$ is an active entity on a rule below $r$ and so, since all rules below $r$ are single-coloured, there is an $?R$ entity in $\Pi$. By the form of $S_1$, this is possible only if there is an entity $?\exists R.C''$ in $\Pi$, which means that $C$ is also of the form $C[\forall R.\neg C'']$. Since $C$ is also of the form $C[\exists R.C'[C_2]]^{0^{\exists}}$, by the definition of $s_{afe}()$ on concepts (second condition), $s_{afe}(!gr, q^{\mathrm{C}}) = 0$. This, however, contradicts (i) as $gr$ is an entity in $S_1$.

- $r \in \{(\sqcap), (\sqcup)\}$. Then $S_2' = x : ?C'[rd]^0,\ x : !D'[gr]^0,\ \Delta'\ \ ;\ \ !\hat{T}$. Both $C'$ and $D'$ qualify as $?rd$ and $!gr$, respectively, and so $S_2'$ satisfies (ii) and (iii).

- $r = (\exists)$. This cannot be the case, since the active concept that does not appear in the conclusion has to have a fresh label. Therefore, not both $!gr$ and $?rd$ can have the same label.

- $r = (?\forall)$. Then $S_2' = y : ?\forall R.rd,\ (y, x) : ?R,\ x : !gr,\ \Delta'\ \ ;\ \ !\hat{T}$. Since $(y, x) : ?R$ cannot occur in $S_1$, this assertion was created by an $(?\exists)$-rule below $r$, and therefore $y : ?\exists R.C'$ is an active entity in $S_2'$ and $x$ is fresh. Since all rules below $r$ are single-coloured and $x$ is fresh, $x : !gr$ can appear in $S_2$ only in the case $!\hat{T}$ is of the form $!\hat{T}[gr]^0$. Reasoning is then continued similarly to the $!GCI$ case.

- $r = (!\forall)$. Then, $S_2' = y : !\forall R.gr,\ (y, x) : !R,\ x : ?rd,\ \Delta'\ \ ;\ \ !\hat{T}$. If $(y, x) : !R$ were created by an $(!\exists)$-rule, then $x$ would be a new label and, because of the single-coloured rules below $S_2'$, $x : ?rd$ would not be possible. Therefore,

$$(y, x) : !R \text{ appears in } S_1. \tag{3.1}$$

Furthermore, the presence of $!\forall R.gr$ implies that there is an entity of the form $!D'[\forall R.gr]$ in $S_1$.

- If $!D'$ is of the form $!D'[\forall R.gr]^{0^{\exists}}$ then, by (3.1) and (i), we have $s_{afe}(!R, \langle \Gamma \cup \{\top \sqsubseteq \hat{T}\}, \emptyset, q^{\mathrm{C}}\rangle) = 1$ and $!D'[\forall R.gr]^{0^{\exists}}$ is an entity in $\Gamma$. So, by the definition of $s_{afe}()$ on roles $s_{afe}(gr, q^{\mathrm{C}}) = 1$. Therefore, by the definition of $s_{afe}()$ on concepts and (iii), we have that $C$ cannot be of the form $C[\neg rd]^{0^{\exists}}$ (for details see the first case of

$!GCI$). However, by $S_1$ we have that $C[\neg rd]$ and thus, there is an active entity $?\forall R'.C'[rd]^0$ below $r$. Therefore, if $w$ is the label of this entity, we have that $(w, x) : ?R'$ appears in a sequent below $S_2'$ (since $x : ?rd$ is in $S_2'$). Again, this assertion must have been created by an $(?\exists)$-rule and thus $x$ is fresh which contradicts (3.1).

- Otherwise, we find that for every $z_i : !D''[\forall R.gr]^{0^\exists}$ in $\Pi$, $z_i$ is a fresh variable. Since $y$ is a label of such a $D''$ ($y : !\forall R.gr$ occurs in $S_2'$), $y$ is also fresh and so (3.1) is contradicted. □

**Theorem 3.3.5** *Given a consistent $\mathcal{ALC}$-knowledge base $\mathcal{O}_{bg}$, a view definition $D_V$ and a privacy condition $q$ on $\langle \mathcal{O}_{bg}, D_V \rangle$, data privacy is preserved for $q$ wrt. $\langle \mathcal{O}_{bg}, D_V \rangle$ if $s_{afe}(D, q) = s_{afe}(R, \langle \mathcal{O}_{bg}, D_V, q \rangle) = 1$, for every concept $D$ and role $R$ of $\langle \mathcal{O}_{bg}, D_V \rangle$.*

*Proof*. By contradiction. Let $q \in \{\top \sqsubseteq C, C, a : C\}$. Assume that (a) there is a $V$ on $\langle \mathcal{O}_{bg}, D_V \rangle$ such that data privacy is not preserved for $q$ with respect to $V$ while (b) $s_{afe}(D, q) = s_{afe}(R, \langle \mathcal{O}_{bg}, D_V, q \rangle) = 1$, for all concepts and roles $D$ and $R$, respectively, of $\langle \mathcal{O}_{bg}, D_V \rangle$ .

Applying Theorem 3.1.2 to assumption (a) yields $ans(q, \mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}) \neq \emptyset$. That is, $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \models \top \sqsubseteq C$ or $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \models d : C$ for some $d \in \mathsf{Ind}(\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle})$. Using Theorem 2.3.2 we can construct now the inconsistent knowledge base $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \cup \overline{q}$, where $\overline{q}$ is given as follows:

$$\begin{aligned} \overline{\top \sqsubseteq C} &:= \{d' : \neg C\}, \text{ for } d' \notin \mathsf{Ind}(\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}), \\ \overline{a : C} &:= \{a : \neg C\}, \\ \overline{C} &:= \{d : \neg C\} . \end{aligned}$$

Theorem 2.4.2 implies that the knowledge base $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle} \cup \overline{q}$ has a proof in $S_{\mathcal{ALC}}$ and thus, the sequent $\Gamma,\ \overline{q}\ \ ;\ \ \hat{T}$ is $S_{\mathcal{ALC}}$-provable, where $\Gamma$ and $\hat{T}$ are the ABox and the TBox transformation of the canonical knowledge base $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$. We distinguish between public and private information in the sequent by colouring the entities derived from $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ green (resp. !) and the entity of $\overline{q}$ red (resp. ?). Let $\Pi$ be a bi-coloured proof of

$$!\Gamma,\ ?\overline{q}\ \ ;\ \ !\hat{T}\ . \tag{3.2}$$

According to the colours of its rule applications (green, red or mixed), $\Pi$ has either at least one mixed rule or it has no mixed rule at all. We distinguish between these two cases:

1. $\Pi$ has no mixed rule application. Since $\mathcal{C}_{\langle \mathcal{O}_{bg}, V \rangle}$ is consistent, by Lemma 2.5.5 we have that $?\overline{q}$ is an active entity in $\Pi$ and therefore, $\Pi$ has

only $?r$-rule applications. Without loss of generality, we can also assume that there is at least one assertion in $\mathcal{C}_{\langle \mathcal{O}_{bg}, V\rangle}$. Then, Lemma 2.5.4 applies and yields a proof of $?\overline{q}\ \ ;\ $. This means that $\overline{q}$ is an inconsistent knowledge base and so, Theorem 2.3.2 applied to $\overline{q}$ results $\emptyset \models \top \sqsubseteq C$. Furthermore, in the case $q \in \{C, a : C\}$, $d$ (or $a$) is an individual either in $\mathcal{O}_{bg}$ or in $V$, which means that either $\mathsf{Ind}(\mathcal{O}_{bg}) \neq \emptyset$ or that there exists a query $D'$ or $d : D'$ in $D_V$ such that $\mathcal{O}_{bg} \not\models \top \sqsubseteq \neg D'$. Therefore, $q$ is a trivial query and the assumption of the theorem is contradicted.

2. $\Pi$ has at least one mixed rule. Let $r$ be a mixed-rule for which all rules below $r$ are single-coloured. If we can show that Lemma 3.3.4 applies with $S_1 = (3.2)$ and $S_2$ the conclusion of $r$, then there is a mixed rule application below $r$ which contradicts the definition of $r$ and thus the theorem is shown.

Therefore, it remains to prove that the assumptions of Lemma 3.3.4 hold. First we show that for all entities $!R$ and $!D$ of the sequent (3.2)

$$s_{afe}(!R, \langle \Gamma \cup \{\top \sqsubseteq \hat{T}\}, \emptyset, q\rangle) = s_{afe}(!D, q) = 1. \tag{3.3}$$

We first show the case of the concepts. Since the value of $s_{afe}(D, q)$ depends only on $q$, by (b) we have that $s_{afe}(D, q) = 1$ for all $D \in \Gamma$. The same is also the case when $C = \hat{T}$ since, $\hat{T}$ is a conjunction of safe concepts wrt. $q$ and the conjunction of two safe concepts is again safe wrt. the same query.

We now turn to the case of the roles. According to the definition of $s_{afe}()$ on roles, the degree of $R$ might change only if there is a concept $D_1[\forall R.D_2]^{0^{\exists}}$ in $\langle \Gamma \cup \{\top \sqsubseteq \hat{T}\}, \emptyset\rangle$ and there is no concept $D_1'[\forall R.D_2]^{0^{\exists}}$ in $\langle \mathcal{O}_{bg}, D_V\rangle$. However, this is not possible since, on the one hand, by the construction of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V\rangle}$ all concepts of $\langle \mathcal{C}_{\langle \mathcal{O}_{bg}, V\rangle}, \emptyset\rangle$ are also concepts of $\langle \mathcal{O}_{bg}, D_V\rangle$. Therefore, $\Gamma$ does not introduce any new concepts. On the other hand, $\hat{T}$ is a conjunction of concepts of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V\rangle}$ and so, for every concept $D_1[\forall R.D_2]^{0^{\exists}}$ of $\langle \Gamma \cup \{\top \sqsubseteq \hat{T}\}, \emptyset\rangle$ there is a concept $D_1'[\forall R.D_2]^{0^{\exists}}$ in $\langle \mathcal{O}_{bg}, D_V\rangle$. Therefore, together with (b), we conclude that $s_{afe}(R, \langle \Gamma \cup \{\top \sqsubseteq \hat{T}\}, \emptyset, q\rangle) = 1$ and thus (3.3) holds.

Next, we show that the conclusion of $r$ has the form required for the sequent $S_2$ in Lemma 3.3.4. The mixed rule application $r$ must be of the form:

$$\frac{}{x : !A,\ x : ?\neg A,\ \Gamma'\ \ ;\ \ !\hat{T}}\ (ax), \qquad \frac{}{x : ?A,\ x : !\neg A,\ \Gamma'\ \ ;\ \ !\hat{T}}\ (ax),$$

$$\frac{y : ?C',\ x : ?\forall R'.C',\ (x, y) : !R',\ \Gamma'\ \ ;\ \ !\hat{T}}{x : ?\forall R'.C',\ (x, y) : !R',\ \Gamma'\ \ ;\ \ !\hat{T}}\ (\forall),$$

or

$$\frac{y : !C',\ x : !\forall R'.C',\ (x,y) : ?R',\ \Gamma'\ (!\hat{T})}{x : !\forall R'.C',\ (x,y) : ?R',\ \Gamma' \ \ ; \ \ !\hat{T})}\ (\forall)$$

where $y : C'$ does not appear in $\Gamma'$. If $r = (ax)$ then its conclusion is trivially of that form. Otherwise, if $r = (\forall)$ we observe that in both applications of $(\forall)$, the role assertion $(x,y) : R'$ was created by an $(\exists)$-rule. This can be seen as follows:

- For the first rule. Since $?\forall R'.C'$ occurs in the conclusion of $r$, we have either

  (i) $\neg C$ is of the form $\neg(C[\forall R'.C']^0)$ or

  (ii) $QR''.C''[\forall R'.C']^0$ for some quantifier $Q$, is an active entity in a rule $?r'$ that appears below $r$.

  We show that in both cases, $(x,y) : !R'$ cannot appear in (3.2). If (i) holds we have $C[\exists R'.\neg C']^0$ and, by the definition of $s_{afe}()$ on roles, we have $s_{afe}(R', \langle \mathcal{O}_{bg}, D_V, q\rangle) = 0$. Therefore, by (b), $!R$ does not appear in (3.2). If (ii) holds, then there is an entity $(z,x) : ?R''$ in the premise of $r'$ which, by definition of (3.2), cannot appear in (3.2). This implies that $(z,x) : ?R''$ was created by an $(\exists)$-rule and $x$ is fresh. Therefore, $(x,y) : R'$ does not appear in (3.2).

- For the second rule. By the definition of $q$, the set $\overline{q}$ does not contain any role assertions.

Consequently, in both cases the role assertion was created in the course of the proof and this can happen only by means of an $(\exists)$-rule application. Thus, $x : !\exists R'.D'$ or $x : ?\exists R'.D'$ appears in the proof before the first or the second $(\forall)$-rule, respectively. Since nothing is thrown away while applying rules, the existential concepts occur in $\Gamma'$ in their respective rule application above. Therefore, the conclusion of $r$ has the required form and Lemma 3.3.4 applies to $\Pi$ with $S_1 = (3.2)$ and $S_2$ the conclusion of $r$. □

## The complexity of the procedure

The following algorithm computes the function $s_{afe}()$ based directly on its definition. Given a concept $D$ and a query $q^{\mathrm{C}}$, $s_{afe}(D, q^{\mathrm{C}})$ can be computed as follows: find all occurrences of positive atoms $A$, negated atoms $\neg A$, universal and existential role restrictions $\forall R$ and $\exists R$, respectively, that appear in $D$ and are not prefixed by a quantifier, and check whether any of them appear also in $C$. If there are such occurrences and are not prefixed by an existential quantifier in $C$ then $s_{afe}(D, q^{\mathrm{C}}) = 0$. Otherwise, let $R'$ be any of the outmost existentially restricted roles that prefix some of the above occurrences in $C$. If $R'$ is also a universal restriction in $C$ then, again, $s_{afe}(D, q^{\mathrm{C}}) = 0$. In all other cases $s_{afe}(D, q^{\mathrm{C}}) = 1$. Finding all the above occurrences takes linear time wrt. the size of $D$ since, at worst all subterms of $D$ will be checked. Checking $C$ for a specific occurrence takes again linear time and thus, the total computation stays in PTime wrt. the size of $C$ and $D$.

Given a role $R$ and a tuple $\langle \mathcal{O}_{bg}, D_V, q^{\mathrm{C}} \rangle$, $s_{afe}(R, \langle \mathcal{O}_{bg}, D_V, q^{\mathrm{C}} \rangle)$ can be computed by a number of $s_{afe}()$ computations on concepts, which are as many as there are concepts of the form $D_1[\forall R.D_2]^{0^{\exists}}$ occurring in $\langle \mathcal{O}_{bg}, D_V \rangle$. Finding these concepts takes linear time wrt. the size of $\langle \mathcal{O}_{bg}, D_V \rangle$. Thus, the $s_{afe}()$ function on a role can be computed in PTime, too.

To conclude, deciding data privacy for a privacy condition $q$ wrt. $\langle \mathcal{O}_{bg}, D_V \rangle$ using the above functions takes polynomial time wrt. the size of $q$ and $\langle \mathcal{O}_{bg}, D_V \rangle$.

**Theorem 3.3.6** *Given a knowledge base $\mathcal{O}_{bg}$, a view definition $D_V$ and a privacy condition $q$, it can be decided in PTime whether for every concept $D$ and role $R$ of $\langle \mathcal{O}_{bg}, D_V \rangle$, $s_{afe}(D, q) = s_{afe}(R, \langle \mathcal{O}_{bg}, D_V, q \rangle) = 1$.*

The above theorem assumes that the privacy condition is given a priori. An ExpTime algorithm that checks whether a query $q$ qualifies as a privacy condition can be obtained directly from the definition of triviality. When $q$ is an inclusion axiom, a single entailment on the empty knowledge base suffices and therefore, the decision is obtained in PSpace. When $q$ is retrieval or an assertion, a number of entailments on the background knowledge is required and therefore, the decision is (at worst case) obtained in ExpTime.

The algorithm can be further simplified if one weakens slightly the results by ruling out of the privacy condition tautological retrieval queries and assertions altogether - although they are not dangerous. In that case, the condition on the definition of a trivial assertion or retrieval query is removed and the triviality of a query can be decided with a single entailment on the empty knowledge base, too.

# Chapter 4

# Some Immediate Results

## 4.1 Securing Actual Data

The purpose of data privacy is to assure that certain potential *data* are not provided to unauthorized users. The privacy of the query $\top \sqsubseteq C$ assures that in the case $\top \sqsubseteq C$ is true in the current knowledge base, the user will not be able to infer it. This is indeed the case as the user cannot distinguish between the current knowledge base and any of the other possible knowledge bases that respect $\mathcal{O}_{bg}$ and $V$. Since privacy is preserved for this query, it follows that there is a possible knowledge base in which this query is not true. From the perspective of the user, that knowledge base could be the current one and therefore, the data $\top \sqsubseteq C$ is secured. Following the same reasoning, the privacy of the query $a : C$ (resp. $C$) assures that in the case the (some) assertion(s) of the form $a : C$ is (are) true in the current knowledge base, the user will not be able to infer it (any of them).

In addition to the privacy of inclusion axioms and concept assertions, we can also assure the privacy of certain role assertions that might follow from the current knowledge base. In particular, we can show that no role assertions are provided to the user other than the ones that are explicitly stated in $\mathcal{O}_{bg}$.

We first show the following proposition:

**Proposition 4.1.1** *Given a knowledge base $\mathcal{O}$, $\mathcal{O} \models (a, b) : R$ if and only if $(a, b) : R \in \mathcal{O}$.*

*Proof.* ($\Rightarrow$) Assume $\mathcal{O} \models (a, b) : R$. Then by monotonicity we have that $\mathcal{O} \cup \{a : \forall R.B, b : \neg B\} \models (a, b) : R$, too, where $B \in \mathsf{AConc}$ and does not occur in $\mathcal{O}$. Following the model definition, however, this is possible only if the augmented knowledge base is inconsistent and therefore, it is $S_{\mathcal{ALC}}$-provable. Let $\Pi$ be a proof of the sequent that corresponds to this knowledge base. Since $B$ does not occur in $\mathcal{O}$, we have that $\mathcal{O} \cup \{a : \forall R.B\}$ is a consistent knowledge base. Therefore, by Lemma 2.5.5 there is a rule application $r$ in $\Pi$ in which $b : \neg B$ is an active entity. The only rule in which $b : \neg B$ can be active is an axiom rule. Thus, $b : B$ is also active in $r$. Since $B$ does not occur anywhere else in $\mathcal{O}$, this entity came from a ($\forall$)-rule application in which $a : \forall R.B$ was active too. However, this is possible only if $(a, b) : R$ is an entity in the same application rule, which is in turn possible only in the case $(a, b) : R$ was already in $\mathcal{O}$.

($\Leftarrow$) Trivial. □

Next, we need to allow role assertions as queries. The evaluation of a role assertion is similar to that of a concept assertion:

$$\begin{array}{lcl} \mathsf{ans}((a,b) : R, \mathcal{O}) & := & \{\mathsf{tt}\} \text{ , if } \mathcal{O} \models (a,b) : R, \\ \mathsf{ans}((a,b) : R, \mathcal{O}) & := & \emptyset \text{ , if } \mathcal{O} \not\models (a,b) : R. \end{array}$$

Proposition 4.1.1 implies that a role assertion is entailed by a knowledge base only when its individuals appear in that knowledge base. Therefore, the expected condition $\{a, b\} \subseteq \mathsf{Ind}(\mathcal{O})$ is omitted. Also, note that this extension is required only for defining data privacy on role assertions and does not propagate to the definitions of $V$ and $D_V$ given in Section 2.3. As so, the scheme of the knowledge base we consider remains the same. Now we can define the secured data of a knowledge base:

**Definition 4.1.2** *Let SECR be the subset of a knowledge base that contains some secrecies. The data in SECR is secured wrt. a valid tuple $\langle \mathcal{O}_{bg}, V\rangle$ (resp. $\langle \mathcal{O}_{bg}, D_V\rangle$) if and only if data privacy is preserved for each element in SECR wrt. $\mathcal{O}_{bg}$ and $V$ (resp. $D_V$).*

The results on data privacy preservation shown in the previous chapters can be easily extended to cover the data privacy preservation of role assertions. More specifically, the constructions of $\mathcal{C}_{\langle \mathcal{O}_{bg}, V\rangle}$ and $\mathbb{P}$ remain the same and the case $q = (a, b) : R$ in Theorem 3.1.2 is similar to the case $q = \top \sqsubseteq C$. The non-trivial direction of Theorem 3.2.3 can be shown for $q = (a, b) : R$ as follows: Since $\mathcal{C}_{\langle \mathcal{O}_{bg}, V\rangle} \models (a, b) : R$ and there are no role assertions in $V$, Proposition 4.1.1 implies that $(a, b) : R \in \mathcal{O}_{bg}$. Every $P$ includes $\mathcal{O}_{bg}$ and therefore, $P \models (a, b) : R$, too.

**Corollary 4.1.3** *Data privacy is preserved for $(a, b) : R$ wrt. $\langle \mathcal{O}_{bg}, V\rangle$ or $\langle \mathcal{O}_{bg}, D_V\rangle$ if and only if $(a, b) : R \notin \mathcal{O}_{bg}$.*

The above corollary allows us to easily extend the partial condition presented in Section 3.3 to cover role assertion queries. Theorem 3.3.5 now holds for every query other than a role assertion. For the role assertion just check whether it is an element of $\mathcal{O}_{bg}$.

Note that the notion of secured data refers to the exact data of SECR and not to their consequences. Any data, however, that imply some secured data is also secured. Furthermore, the security of some data can be implied by the privacy of a retrieval query. These two observations give alternative ways in deciding privacy, depending on what is known a priori.

**Proposition 4.1.4** *1. Let $p$ and $q$ be elements of a knowledge base. If $q$ is secured wrt. $\langle \mathcal{O}_{bg}, V\rangle$ and $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \cup \{p\} \models q$, then $p$ is also secured wrt. $\langle \mathcal{O}_{bg}, V\rangle$.*

*2. If data privacy is preserved for $C$ wrt. $\langle \mathcal{O}_{bg}, V\rangle$ and $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \models D \sqsubseteq C$ $(= \top \sqsubseteq \neg D \sqcup C)$, then data privacy is also preserved for $D$ wrt. $\langle \mathcal{O}_{bg}, V\rangle$.*

*3. Let $\langle \mathcal{O}_{bg}, V\rangle$ be a tuple for which $\mathsf{Ind}(\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle}) \neq \emptyset$. If data privacy is preserved for $C$ wrt. $\langle \mathcal{O}_{bg}, V\rangle$ then data privacy is also preserved for (i) $a : C$ and (ii) $\top \sqsubseteq C$ wrt. $\langle \mathcal{O}_{bg}, V\rangle$.*

*Proof.* 1. By contradiction, assume that $p$ is not secured. Then, $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \models p$ and so $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle}$ and $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \cup \{p\}$ are logically equivalent. Thus, $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \models q$ and $q$ is not secured either.
2. By contradiction, assume that $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \models a : D$. But then $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \models a : C$ by the model definition on inclusions.
3. By contradiction, assume that (i) $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \models a : C$ and $a \in \mathsf{Ind}(\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle})$, or that (ii) $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \models \top \sqsubseteq C$. Since there is at least one individual in $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle}$, both cases imply that $\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle} \models b : C$, for some $b \in \mathsf{Ind}(\mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle})$ and so, $\mathsf{ans}(C, \mathcal{C}_{\langle \mathcal{O}_{bg},V\rangle}) \neq \emptyset$. □

## 4.2 Data Privacy in Modular Ontologies

Above we have seen how data stored in a knowledge base can be shown to be secured. Based on those results and the partial solution presented in Section 3.3, we will see how a whole sub-knowledge base can be shown to be secured. Now, not only the exact data stored in the knowledge base is secured, but also their consequences.

We first present a general application to modular ontologies (for a survey see [WHB07]). After that, we present a more specific instance of it to $\mathcal{E}$-Connections. In both solutions, the ontology language imposes, by definition, certain syntactic restrictions on the knowledge base. Therefore, these solutions can be identified simply by the construction of the knowledge base and so, their validity can be checked quickly.

We begin with the general application. Assume that a knowledge base consists of several $\mathcal{ALC}$ sub-knowledge bases (called modules). Depending on the method applied, the elements of a module are structurally restricted. We do not wish to choose a specific method at this point, instead we only require that these restrictions are expressible in terms of inclusion axioms and/or assertions of an $\mathcal{ALC}$ knowledge base. In other words, each module $M_i$ consists of (i) inclusion axioms and concept assertions built on certain, restricted $\mathcal{ALC}$ concepts $C_i$, as well as of (ii) role assertions built on certain assertional roles $R_i$. For example, an inclusion axiom of the module $M_1$ has the form $\top_1 \sqsubseteq C_1$ and $\top_1 = A_1 \sqcup \neg A_1$ and $C_1$ are concepts allowed in $M_1$. The set of individuals $\mathsf{Ind}$ is shared by all modules.

In this general type of modular ontologies reasoning is applied as in a normal knowledge base. If $n$ is the number of modules in the ontology then the knowledge base that corresponds to it is $\mathcal{O} = M_1 \cup \ldots \cup M_n$ and the consequences of the ontology are defined on $\mathcal{O}$. A query $q_i$ obeys to the same restrictions the entities of $M_i$ do. The query evaluation $\mathsf{ans}(q_i, \mathcal{O})$ remains as before (i.e. the difference now is on the allowed queries, not the way they are evaluated).

**Definition 4.2.1** *Let $M_1$ be a module of an ontology $\mathcal{O}$. Then, $M_1$ is* data-free *secured wrt. a valid tuple $\langle \mathcal{O}_{bg}, V\rangle$ (resp. $\langle \mathcal{O}_{bg}, D_V\rangle$) if and only if every element allowed in $M_1$ is secured wrt. $\langle \mathcal{O}_{bg}, V\rangle$ (resp. $\langle \mathcal{O}_{bg}, D_V\rangle$).*

Note: this definition covers the privacy of every possible $M_1$ and its consequences.

**Proposition 4.2.2** *A module $M_1$ is data-free secured wrt. a public module $M_2$ and a view $V$ (resp. a view definition $D_V$) when the following conditions are satisfied:*

- *if $R$ or $A$ is allowed to occur in $M_1$, then it may also occur in $M_2$ or $V$ (resp. $D_V$) but only in concepts $C$ of $\langle M_2, V\rangle$ (resp. $\langle M_2, D_V\rangle$) that are of the form $C[QR'.C'[A]^0]$ or of the form $C[QR'.C'[Q'R.C'']^0]$, where $\{Q, Q'\} \subseteq \{\exists, \forall\}$ (i.e. behind an $R'$-restriction). In addition to this, when $Q$ is a universal quantifier, $R'$ is not allowed to be an assertional role in $M_2$.*

- *inclusion axioms and concept assertions allowed in $M_1$ qualify as privacy conditions on the tuple $\langle M_2, D_V\rangle$.*

*Proof.* We must show that data privacy is preserved for every inclusion axiom and assertion that is allowed in $M_1$. Since the semantics of the modular ontology are identical to those of a usual $\mathcal{ALC}$ knowledge base, all results shown by now are also valid here. Therefore, it is enough to show that (i) the premise of Theorem 3.3.5 and (ii) Corollary 4.1.3 are valid. The latter holds trivially as no assertional role $R_1$ may appear in $M_2$. For (i), we must show that every inclusion axiom or concept assertion $q^{C_1}$ that is allowed in $M_1$ qualifies as a privacy condition and that $s_{afe}(D, q^{C_1}) = s_{afe}(R_2, \langle M_2, D_V, q^{C_1}\rangle) = 1$, for every concept $D$ and role $R_2$ of $\langle M_2, D_V\rangle$. That every $q^{C_1}$ qualifies as a privacy condition is given by the construction of the ontology. The equations are shown as follows:

- $s_{afe}(D, q^{C_1})$. By the construction of the ontology, if $C_1$ is of the form $C_1[C']$ then every $A$ and $R$ that occur in $C'$ can appear in $D$ only behind an $R'$-restriction. Since any concept $D'$ that is similar to $C'$ shares at least a role or atomic concept with $C'$, it is not possible to have $D[D']^0$. Therefore, $s_{afe}(D, q^{C_1}) = 1$.

- $s_{afe}(R_2, \langle M_2, D_V, q^{C_1}\rangle)$. By the construction of the ontology, $R_2$ is not allowed to occur in $M_1$ and so, $C_1$ cannot be of the form $C_1[\exists R_2.C']^0$. Thus, the first condition of the function is satisfied. For the second condition: since $R_2$ is an assertional role in $M_2$, by the construction of the ontology it follows that, for every concept $D[\forall R_2.D']$, $D'$ cannot be of the form $D'[A]^0$ or $D'[QR.D']^0$, where $A$ and $R$ are allowed to occur in $M_1$. This means that $D'$ could be a concept of $\langle M_2, D_V\rangle$ and so - as it is shown above - $s_{afe}(D', q^{C_1}) = 1$.

The case of $V$ follows from the above results since, for $V$ an instance of $D_V$, $s_{afe}(R', \langle M_2, D_V, q^{C_1}\rangle) = 1$ and therefore $s_{afe}(R', \langle \mathcal{C}_{\langle M_2, V\rangle}, \emptyset, q^{C_1}\rangle) = 1$, too. Also, a privacy condition $q^{C_1}$ on $\langle M_2, D_V\rangle$ is a privacy condition on $\langle \mathcal{C}_{\langle M_2, V\rangle}, \emptyset\rangle$, too. □

We demonstrate the above solution on a certain kind of modular ontologies, the $\mathcal{E}$-Connections [CPSK05, KLWZ04] and, in particular, in a certain instance of two $\mathcal{ALC}$ knowledge bases ($\mathcal{C}^{\mathcal{E}}(\mathcal{ALC}, \mathcal{ALC})$). In such an ontology, each of the constituted knowledge bases, $\mathcal{E}_i$, has its own distinct language $L_i$ that consists of the countable sets of individuals $\mathsf{Ind}_i$, atomic concepts $\mathsf{AConc}_i$ and local roles $\mathsf{Rol}_i$. In addition to these, a set of special roles $\mathcal{E}$ that connect individuals of $\mathcal{E}_1$ to individuals of $\mathcal{E}_2$ is provided. The $\mathcal{ALC}$-Concepts of $L_1$

and $L_2$ are built as follows:

$$C_1 := A_1 \mid \neg A_1 \mid C_1 \sqcup D_1 \mid C_1 \sqcap D_1 \mid \exists R_1.C_1 \mid \forall R_1.C_1 \mid \exists E.C_2 \mid \forall E.C_2$$

$$C_2 := A_2 \mid \neg A_2 \mid C_2 \sqcup D_2 \mid C_2 \sqcap D_2 \mid \exists R_2.C_2 \mid \forall R_2.C_2$$

where all concepts and roles indexed by $i \in \{1, 2\}$ belong to $L_i$ and $E \in \mathcal{E}$. The knowledge base $\mathcal{E}_i$ is then a set of inclusion axioms $\top_i \sqsubseteq C_i$, concept assertions $a_i : C_i$ and role assertions $(a_i, b_i) : R_i$ with $\{a_i, b_i\} \subset \mathsf{Ind}_i$. In addition to these, $\mathcal{E}_1$ may also include $E$ role assertions $(a_1, a_2) : E$. As usual, the individuals of $\mathsf{Ind}_j$ that appear in $\mathcal{E}_i$ are denoted by $\mathsf{Ind}_j(\mathcal{E}_i)$.

The semantics of $\mathcal{E}$-Connections generalize those of the usual $\mathcal{ALC}$ knowledge bases. We refer to $\mathcal{E}$-Connections that are built as described above as $\mathcal{O}_{\mathcal{E}}$.

**Definition 4.2.3** *An interpretation $\mathcal{I}^e$ of an $\mathcal{E}$-Connection $\mathcal{O}_{\mathcal{E}}$ is a tuple of the form $\langle \mathcal{I}_1, \mathcal{I}_2, (\mathcal{E})^{\mathcal{I}^e} \rangle$, where $\mathcal{I}_1$ and $\mathcal{I}_2$ are normal $\mathcal{ALC}$ interpretations that assign values to the sets of $L_1$ and $L_2$, respectively, and $(\mathcal{E})^{\mathcal{I}^e}$ assigns to every $E \in \mathcal{E}$ a relation $(E)^{\mathcal{I}^e} \subseteq \Delta^{\mathcal{I}_1} \times \Delta^{\mathcal{I}_2}$.*

The interpretation $\mathcal{I}^e$ extends then as follows:

$$\begin{array}{rcl}
(a_i)^{\mathcal{I}^e} & = & (a_i)^{\mathcal{I}_i} \\
(A_i)^{\mathcal{I}^e} & = & (A_i)^{\mathcal{I}_i} \\
(R_i)^{\mathcal{I}^e} & = & (R_i)^{\mathcal{I}_i} \\
(\neg A_i)^{\mathcal{I}^e} & = & \Delta^{\mathcal{I}_i} \setminus (A_i)^{\mathcal{I}^e} \\
(C_i \sqcap D_i)^{\mathcal{I}^e} & = & (C_i)^{\mathcal{I}^e} \cap (D_i)^{\mathcal{I}^e} \\
(C_i \sqcup D_i)^{\mathcal{I}^e} & = & (C_i)^{\mathcal{I}^e} \cup (D_i)^{\mathcal{I}^e} \\
(\forall R_i.C_i)^{\mathcal{I}^e} & = & \{d_i \in \Delta^{\mathcal{I}_i} \mid \forall d_2 \; ((d_1, d_2) \in (R_i)^{\mathcal{I}^e} \Rightarrow d_2 \in (C_i)^{\mathcal{I}^e})\} \\
(\exists R_i.C_i)^{\mathcal{I}^e} & = & \{d_i \in \Delta^{\mathcal{I}_i} \mid \exists d_2 \; ((d_1, d_2) \in (R_i)^{\mathcal{I}^e} \; \& \; d_2 \in (C_i)^{\mathcal{I}^e})\} \\
(\forall E.C_2)^{\mathcal{I}^e} & = & \{d_1 \in \Delta^{\mathcal{I}_1} \mid \forall d_2 \; ((d_1, d_2) \in (E)^{\mathcal{I}^e} \Rightarrow d_2 \in (C_2)^{\mathcal{I}^e})\} \\
(\exists E.C_2)^{\mathcal{I}^e} & = & \{d_1 \in \Delta^{\mathcal{I}_1} \mid \exists d_2 \; ((d_1, d_2) \in (E)^{\mathcal{I}^e} \; \& \; d_2 \in (C_2)^{\mathcal{I}^e})\}
\end{array}$$

An interpretation models inclusion axioms and assertions in the usual way. For the sake of uniformity, we define an $\mathcal{E}$-Connection as the union of the two knowledge bases $\mathcal{E}_1$ and $\mathcal{E}_2$. In this way, the definition of privacy preservation applies also here.

The evaluation of a query $q^{C_i}$ on an $\mathcal{E}$-Connection differs from the usual evaluation in that now, the evaluation is restricted to individuals of $\mathsf{Ind}_i$ and not of $\mathsf{Ind}$.

$$\begin{array}{rcl}
\mathsf{ans}(\top_i \sqsubseteq C_i, \mathcal{O}_\mathcal{E}) & := & \{\mathsf{tt}\} \text{ , if } \mathcal{O}_\mathcal{E} \models \top_i \sqsubseteq C_i, \\
\mathsf{ans}(\top_i \sqsubseteq C_i, \mathcal{O}_\mathcal{E}) & := & \emptyset \text{ , if } \mathcal{O}_\mathcal{E} \not\models \top_i \sqsubseteq C_i, \\
\mathsf{ans}(a : C_i, \mathcal{O}_\mathcal{E}) & := & \{\mathsf{tt}\} \text{ , if } \mathcal{O}_\mathcal{E} \models a : C_i \text{ and } a \in \mathsf{Ind}_i(\mathcal{O}_\mathcal{E}), \\
\mathsf{ans}(a : C_i, \mathcal{O}_\mathcal{E}) & := & \emptyset \text{ , if } \mathcal{O}_\mathcal{E} \not\models a : C_i \text{ or } a \notin \mathsf{Ind}_i(\mathcal{O}_\mathcal{E}), \\
\mathsf{ans}(C_i, \mathcal{O}_\mathcal{E}) & := & \{a \in \mathsf{Ind}_i(\mathcal{O}_\mathcal{E}) \mid \mathcal{O}_\mathcal{E} \models a : C_i\} \text{ .}
\end{array}$$

Theorem 2.3.2 is valid in this setting too, after replacing $\mathsf{Ind}$ with $\mathsf{Ind}_i$.

**Proposition 4.2.4** *If $\mathcal{O}_\mathcal{E}$ is inconsistent then it is inconsistent under the normal $\mathcal{ALC}$ semantics, too.*

*Proof.* We prove the contrapositive. Assume that there is an $\mathcal{ALC}$ interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{E}_1 \cup \mathcal{E}_2$. Then, $\mathcal{I}'$ is the interpretation with $\Delta^{\mathcal{I}'} = \{a' \mid a \in \Delta^\mathcal{I}\}$ and all its assignments are like in $\mathcal{I}$, except that now every occurrence of $a \in \Delta^\mathcal{I}$ that occur in the assignments of $\mathcal{I}$ is replaced by $a'$. A model $\mathcal{I}^e = \langle \mathcal{I}_1, \mathcal{I}_2, (\mathcal{E})^{\mathcal{I}^e} \rangle$ of $\mathcal{O}_\mathcal{E}$ is constructed as follows:

- $\Delta^{\mathcal{I}_1} = \Delta^\mathcal{I}$ and all assignments of $\mathcal{I}_1$ are identical to those of $\mathcal{I}$.
- $\Delta^{\mathcal{I}_2} = \Delta^{\mathcal{I}'}$ and all assignments of $\mathcal{I}_2$ are identical to those of $\mathcal{I}'$.
- $E^{\mathcal{I}^e} = \{(a, b') \mid (a, b) \in E^\mathcal{I}\}$.

By an easy induction on the structure of $C_i$, we can show that $C_1^{\mathcal{I}^e} = C_1^\mathcal{I}$ and $C_2^{\mathcal{I}^e} = C_2^{\mathcal{I}'}$. Therefore, we conclude that $\mathcal{I}^e$ is a model of $\mathcal{O}_\mathcal{E}$ and the proposition holds. □

Consider now two $\mathcal{ALC}$ $\mathcal{E}$-Connected knowledge bases $\mathcal{E}_1$ and $\mathcal{E}_2$ built as described above.

**Proposition 4.2.5** *If every allowed inclusion axiom and concept assertion in $\mathcal{E}_2$ qualifies as a privacy condition and there is no role assertion $(a_1, b_2) : E$ in $\mathcal{E}_1$ for any $E$-universal restriction that occurs in $\mathcal{E}_1$, then $\mathcal{E}_2$ is data-free secured wrt. $\langle \mathcal{E}_1, \emptyset \rangle$.*

*Proof.* As in the previous proposition, we show that both (i) Theorem 3.3.5 and (ii) Corollary 4.1.3 apply to this setting. The latter follows by the construction of $\mathcal{O}_\mathcal{E}$ as no assertion of $R_2$ is allowed in $\mathcal{E}_1$. For (i): it is easy to check that $\mathcal{E}_2$ and $\mathcal{E}_1$ satisfy the restrictions of the modules $M_1$

and $M_2$, respectively, of Proposition 4.2.2. Therefore, the equations on $s_{afe}()$ are satisfied. Furthermore, the elements of $\mathcal{E}_2$ qualify as privacy conditions, by definition, and so, the assumptions of the theorem are satisfied. Now we need to show that the theorem holds. By contradiction, we assume that data privacy is not preserved and so $\mathcal{E}_1 \cup \{d : \neg C_2\}$, with $d \in \mathsf{Ind}_i$ is inconsistent. By Proposition 4.2.5 this implies that the normal $\mathcal{ALC}$ knowledge base of the same data is inconsistent and therefore, reasoning is continued as in the existing proof of Theorem 3.3.5. □

# Conclusions

This thesis comprises an initial study on the problem of provable privacy for $\mathcal{ALC}$-knowledge bases. In particular, we have applied privacy to the following setting: background knowledge is an $\mathcal{ALC}$-knowledge base whereas views might provide positive or neutral answers to inclusion axioms (boolean), concepts (retrieval) and concept assertions (boolean) that are built on $\mathcal{ALC}$-concepts. In addition to the queries that are allowed in a view, a secret can also be a role assertion (see Section 4.1). Provable privacy guarantees that the secret information cannot be inferred accurately from a publicly available background knowledge and a view.

We have presented complete procedures that decide provable privacy for $\mathcal{ALC}$-knowledge bases. More specifically, we have shown that privacy wrt. a view reduces to a polynomial number of entailments and is an ExpTime-complete problem. Privacy wrt. a view definition (i.e. a family of views) is also ExpTime-complete and can be decided by considering only exponentially many views.

A privacy preserving condition that can be decided in PTime and applies to both views and view definitions is also provided. This condition poses restrictions on the structure of the secret query that depends on the structure of the publicly available data (i.e. the background knowledge and the view or view definition). Because of its syntactic nature, the secret query has the a priori requirement to be non-trivial. This means that its validity is syntax-sensitive and, therefore, it depends on the current knowledge base. This a-priori assumption can be computed in PSpace when the secret is an inclusion axiom and in ExpTime when it is a concept or a concept assertion. Adding a further restriction to the non-valid secrets leads to a (worst-case) PSpace computation. Finally, the applicability of this procedure is demonstrated on modular ontologies. There, it is shown that when an ontology is connected to a second ontology, the information of the second ontology is not exhibited to users that have access only to the first ontology.

Current results on provable privacy can be strengthen in a variety of ways as the system requirements we have considered are minimal. A first extension could deal with the formulation of stronger secrecies. We believe that the formulation of a secret in terms of a query and the notion of provable privacy can express stronger notions of privacy in a very simple and intuitive way. Note, however, that the current definition might not be able to capture correctly certain queries. This happens because we have defined provable privacy in systems that do not take into consideration implicit knowledge that might come from the view definition. If, for instance, $a : C$ is a query in the view definition, then we know that $a$ is a valid individual, independently of its evaluation. However, this information is not expressible in a repository. This means that the current definition of provable privacy will not work as intended once, for instance, the secret query "$a$ is not in the repository" or "$a$ is a valid individual" are considered. In order to capture correctly such privacy issues, stronger forms of background knowledge and appropriate adjustments of the privacy definition should be considered.

Another important direction of further work is the study of privacy issues on stronger ontologies. The $\mathcal{ALC}$ is a simple language that is nevertheless sensible to study. There are, however, very powerful languages that are used in today's ontological systems. For instance, web applications make use of the description logic $\mathcal{SHOIN}(D)$. Apart from the direct benefits, such a study will also allow the identification of stronger privacy preserving patterns.

# Bibliography

[AA01] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *PODS '01*, pp. 247–255. ACM, 2001. ISBN 1-58113-361-8.

[AL05] M. Arenas and L. Libkin. XML Data Exchange: Consistency and Query Answering. In *PODS*, pp. 13–24. 2005.

[Are00] C. E. Areces. *Logic Engineering: the case of description and hybrid logics*. Ph.D. thesis, ILLC, University of Amsterdam, 2000.

[BB04] J. Biskup and P. A. Bonatti. Controlled query evaluation for enforcing confidentiality in complete information systems. *International Journal of Information Security*, 3(1):14–27, 2004.

[BCM$^+$03] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, eds. *The Description Logic Handbook*. Cambridge University Press, 2003.

[BFJ00] A. Brodsky, C. Farkas, and S. Jajodia. Secure Databases: Constraints, Inference Channels, and Monitoring Disclosures. *IEEE Trans. on Knowl. and Data Eng.*, 12(6):900–919, 2000. ISSN 1041-4347.

[BKS95] P. A. Bonatti, S. Kraus, and V. S. Subrahmanian. Foundations of Secure Deductive Databases. *Transactions on Knowledge and Data Engineering*, 7(3):406–422, 1995. ISSN 1041-4347.

[BS01] F. Baader and U. Sattler. An Overview of Tableau Algorithms for Description Logics. *Studia Logica*, 69:5–40, 2001.

[BW08] J. Biskup and T. Weibert. Keeping secrets in incomplete databases. *Journal of Information Security*, 7(3):199–217, 2008.

[CCGL02] A. Calì, D. Calvanese, G. D. Giacomo, and M. Lenzerini. Data Integration under Integrity Constraints. In *Proc. of CAiSE 2002*, volume 2348 of *LNCS*, pp. 262–279. Springer, 2002.

[CPSK05] B. Cuenca Grau, B. Parsia, E. Sirin, and A. Kalyanpur. Automated Partitioning of OWL Ontologies using E-Connections. In *Proceedings of Int. Workshop on Description Logics*. 2005.

[DD79] D. E. Denning and P. J. Denning. Data Security. *ACM Comput. Surv.*, 11(3):227–249, 1979. ISSN 0360-0300.

[DM00] F. M. Donini and F. Massacci. EXPTIME Tableaux for ALC. *Artificial Intelligence*, 124(1):87–138, 2000.

[DMS05] N. Dalvi, G. Miklau, and D. Suciu. Asymptotic Conditional Probabilities for Conjunctive Queries. In *ICDT*, volume 3363 of *LNCS*, pp. 289–305. Springer, 2005.

[Don03] F. M. Donini. *Complexity of Reasoning*, chapter 3, pp. 96–136. The Description Logic Handbook. Cambridge University Press, 2003.

[DP05] A. Deutsch and Y. Papakonstantinou. Privacy in database publishing. In *ICDT*. 2005.

[EGS03] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *PODS '03: Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 211–222. ACM, New York, NY, USA, 2003. ISBN 1-58113-670-6.

[ESAG02] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *8th ACM SIGKDD International Conference on Knowledge Discovery in Databases and Data Mining*. 2002.

[FJ02] C. Farkas and S. Jajodia. The inference problem: a survey. *SIGKDD Explor. Newsl.*, 4(2):6–11, 2002. ISSN 1931-0145.

[FKMP05] R. Fagin, P. G. Kolaitis, R. Miller, and L. Popa. Data Exchange: Semantics and Query Answering. *Theoretical Computer Science*, 336:89–124, 2005.

[Hal01] A. Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, 2001. ISSN 1066-8888.

[HM70] L. Hoffman and W. Miller. Getting a personal dossier from a statistical data bank. *Datamation*, 16:74–75, 1970.

[Hof05] M. Hofmann. Proof-Theoretic Approach to Description-Logic. In *20th IEEE Symposium on Logic in Computer Science (LICS 2005)*, pp. 229–237. IEEE Computer Society, 2005.

[KLWZ04] O. Kutz, C. Lutz, F. Wolter, and M. Zakharyaschev. E-connections of abstract description systems. *Artifical Intelligence*, 156(1):1–73, 2004.

[KMMZ05] Y. Kanza, A. O. Mendelzon, R. J. Miller, and Z. Zhang. Authorization-Based Access Control for XML, 2005. Technical report CSRG-527, University of Toronto, Department of Computer Science. Available from ftp://ftp.cs.toronto.edu/csrg-technical-reports.

[KMMZ06] Y. Kanza, A. O. Mendelzon, R. J. Miller, and Z. Zhang. Authorization-Transparent Access Control for XML under the Non-Truman Model. In *EDBT*, volume 3896 of *LNCS*, pp. 222–239. Springer, 2006.

[MG06] A. Machanavajjhala and J. Gehrke. On the efficiency of checking perfect privacy. In *PODS '06: Proceedings of Principles of database systems*, pp. 163–172. ACM Press, 2006.

[MKGV07] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), 2007. ISSN 1556-4681.

[MKMG07] D. J. Martin, D. Kifer, A. Machanavajjhala, and J. Gehrke. Worst-Case Background Knowledge for Privacy-Preserving Data Publishing. In *Proceedings ICDE*. 2007.

[MS04] G. Miklau and D. Suciu. A formal analysis of information disclosure in data exchange. In *SIGMOD*, pp. 575–586. ACM, 2004.

[ND07] A. Nash and A. Deutsch. Privacy in GLAV Information Integration. In T.Schwentick and D.Suciu, eds., *ICDT 2007*, volume 4353 of *LNCS*, pp. 89–103. Springer, 2007.

[RMSR04] S. Rizvi, A. Mendelzon, S. Sudarshan, and P. Roy. Extending query rewriting techniques for fine-grained access control. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pp. 551–562. ACM, New York, NY, USA, 2004. ISBN 1-58113-859-8.

[Sch91] K. Schild. A Correspondence Theory for Terminological Logics: Preliminary Report. In *Twelfth International Conference on Artificial Intelligence IJCAI'91*, pp. 466–471. 1991.

[SDJVdR83] G. L. Sicherman, W. De Jonge, and R. P. Van de Riet. Answering queries without revealing secrets. *ACM Trans. Database Syst.*, 8(1):41–59, 1983. ISSN 0362-5915.

[SS98] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *PODS*, p. 188. ACM Press, 1998. ISBN 0-89791-996-3.

[SS05] K. Stoffel and T. Studer. Provable Data Privacy. In K. Viborg, J. Debenham, and R. Wagner, eds., *Database and Expert Systems Applications DEXA 2005*, volume 3588 of *LNCS*, pp. 324–332. Springer, 2005.

[SS07] P. Stouppa and T. Studer. A Formal Model of Data Privacy. In I. Virbitskaite and A. Voronkov, eds., *Perspectives of System Informatics PSI'06*, volume 4378 of *LNCS*, pp. 401–411. Springer, 2007.

[SS09] P. Stouppa and T. Studer. Data Privacy for $\mathcal{ALC}$ Knowledge Bases. In S. Artemov and A. Nerode, eds., *Proceedings of Logical Foundations of Computer Science LFCS'09*, volume 5407 of *LNCS*, pp. 409–421. Springer, 2009.

[SSS91] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[vdM98] R. van der Meyden. Logical approaches to incomplete information: a survey. In *Logics for databases and information systems*, pp. 307–356. Kluwer Academic Publishers, 1998. ISBN 0-7923-8129-7.

[WHB07] Y. Wang, P. Haase, and J. Bao. A Survey of formalisms for modular ontologies. In *Proceedings from IJCAI'07*. 2007.

[WSQ94] M. Winslett, K. Smith, and X. Qian. Formal query languages for secure relational databases. *ACM Trans. Database Syst.*, 19(4):626–662, 1994. ISSN 0362-5915.

[YWJ05] C. Yao, X. S. Wang, and S. Jajodia. Checking for $k$-anonymity violation by views. In *Proceedings of the 31st VLDB*, pp. 910–921. 2005.