Foundations of a Logic based approach to Multimedia Document Retrieval

Dissertation

zur Erlangung des Grades eines
Doktors der Naturwissenschaften
der Universität Dortmund
am Fachbereich Informatik

von

Umberto Straccia

Dortmund
1999

Tag der mündliche Prüfung:	23.6.99
Dekan:	Prof. Dr. Bernd Reusch
Gutachter:	Prof. Dr. Norbert Fuhr Prof. Dr. Katharina Morik

Abstract

Multimedia Information Retrieval (MIR), *i.e.* the retrieval of those multimedia objects of a collection which are relevant to a user information need, is an intensively investigated research area. It involves research from several fields of computer science, notably, Information Retrieval, Image Retrieval, Audio Retrieval, Video Retrieval, the Database community and Artificial Intelligence. This variety reveals that many different aspects are involved in MIR, each requiring a specific background and methodology, and that there may be different approaches not only within the same discipline, but also across different ones.

A principled approach to the description of a MIR model requires the formal specification of three basic entities of retrieval: (*i*) the representation of multimedia objects; (*ii*) the representation (called query) of a user information need; and (*iii*) the retrieval function, returning a ranked list of objects for each information need.

We believe that any MIR model should address the bidimensional aspect of multimedia objects: that is, their *form* and their *semantics* (or *meaning*). The *form* of an object is a collective name for all its *media dependent* features, whereas the *semantics* of an object is a collective name for those features that pertain to the slice of the real world being *represented*, which exists independently of the existence of a object referring to it. Unlike form, the semantics of an object is thus *media independent*.

Corresponding to these two dimensions, there are three categories of retrieval: one for each dimension (*form-based retrieval* and *semantics-based retrieval*) and one concerning the combination of both of them. Form-based retrieval methods automatically create the object representations to be used in retrieval by extracting features from multimedia objects, such as the number of occurrences of words in text, colour distributions in images, and video frame sequences in videos. Semantics-based retrieval methods rely on a symbolic representation of the aboutness of multimedia objects, *e.g.* "this image is about a girl". That is, descriptions formulated in some suitable formal language. User queries may thus address both dimensions, *e.g.* "find images about girls wearing clothes with a texture like this". In it, the texture addresses an image feature (form), whereas the aboutness addresses the meaning of an image (semantics).

Despite the fact that several MIR models have been proposed, there has been little work done in proposing MIR models in which all three categories of retrieval are tackled in a principled way. Not surprisingly, promising models involve the so-called logic-based approach to information retrieval. This thesis is a contribution in this direction. Indeed, we will propose an *object-oriented data model* for representing medium dependent features of multimedia objects (form properties) and a *four-valued fuzzy horn description logic* for representing multimedia object's semantics and domain knowledge (medium independent features -semantic properties). In particular, the logic is characterised by (*i*) a description logic component which allows the representation of the structured objects (of interest) in the real world; (*ii*) a

horn rule component which allows us to reason about structured objects; (*iii*) a non-classical, four-valued semantics which allows us to deal with possible inconsistencies arising from the representation of document semantics; (*iv*) a fuzzy component which allows for the treatment of the inherent imprecision in multimedia document representation and retrieval. Retrieval is then defined in terms of logical entailment, where the object-oriented data model hase been integrated within the logic.

The rational of the above choices relies on the fact that the principles of object-oriented design, namely *aggregation*, *classification* and *generalisation*, have been widely used in the context of multimedia object representation, revealing its appropriateness for representing medium depended features. In contrast, the components of the proposed logic have been thoroughly investigated and a wide range of results, automated reasoning techniques, and systems are available which makes the model a viable tool for practical use in the context of MIR.

The main feature of the model is that all three above-mentioned categories of retrieval are addressed in a formal, flexible and extensible framework. The model allows us to represent both form properties and semantic properties of multimedia data, combining in a neat way different techniques –notably database techniques and semantic information processing (knowledge representation and reasoning), with the aim of developing *intelligent* multimedia retrieval systems.

Acknowledgements

Writing a thesis is never done alone and lots of people have contributed to make this happen. So I would like to thank all of you who have been there for me. While this part of my thesis may be much easier to read and may be much more understandable than many other parts, it is definitely the most difficult to write. Anyhow, I will give a try and explicitly thank a number of people. However, if, by mistake, I happen to have forgotten to acknowledge You explicitly, Dear Reader, please feel free to add Yourself to the list.

First of all, I would like to thank Norbert Fuhr. Without his valuable support and encouragement this work would have been simply impossible.

Thanks go to Constantino Thanos, my boss, and his research group which provided a comfortable research environment for me. Especially, I would like to thank Costantino for giving me the necessary free time within the FERMI project[1].

I would also to thank all the colleagues of the FERMI project and in particular Carlo Meghini and Fabrizio Sebastiani which gave me useful ideas and with which I had many interesting discussions (ranging from multimedia information retrieval to soccer).

Last but not least, I would like to thank my wife, Maria Pia, encouraging and tolerating me during this thesis.

[1]ESPRIT project FERMI 8134 - "Formalisation and Experimentation in the Retrieval of Multimedia Information", funded by the European Community under the ESPRIT Basic Research scheme.

External Publications

Work included in the thesis

- Carlo Meghini and Umberto Straccia. A relevance terminological logic for information retrieval. In *Proceedings of SIGIR-96, 19th International Conference on Research and Development in Information Retrieval*, pages 197–205, Zurich, Switzerland, 1996.

- Carlo Meghini and Umberto Straccia. Extending a description logic to cope with the completeness of multimedia documents. In *Proc. of the 12th European Conf. on Artificial Intelligence (ECAI-96): Workshop on Knowledge Representation for Interactive Multimedia Systems*, pages 42–50, Budapest, Hungary, 1996.

- Carlo Meghini, Fabrizio Sebastiani, and Umberto Straccia. Reasoning about form and content for multimedia information retrieval. In *Proceedings of WIRUL-96, 2nd Workshop on information Retrieval, Uncertainty and Logic*, pages 57–58, Glasgow, Scotland, 1996.

- Carlo Meghini, Fabrizio Sebastiani, and Umberto Straccia. Reasoning about the form and content for multimedia objects (extended abstract). In *Proceedings of AAAI 1997 Spring Symposium on Intelligent Integration and Use of Text, Image, Video and Audio*, pages 89–94, Stanford University, California, 1997.

- Carlo Meghini, Fabrizio Sebastiani, and Umberto Straccia. The terminological image retrieval model. In Alberto Del Bimbo, editor, *Proceedings of ICIAP'97, 9th International Conference On Image Analysis And Processing*, volume II of *Lecture Notes in Computer Science*, pages 156–163, Florence, I, September 1997. Springer-Verlag.

- Carlo Meghini, Fabrizio Sebastiani, and Umberto Straccia. Modelling the retrieval of structured documents containing texts and images. In Lecture Notes in Computer Science, editor, *Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries*, number 1324, pages 325–344, Pisa, September 1997.

- Carlo Meghini, Fabrizio Sebastiani, and Umberto Straccia. The terminological image retrieval model. In *Proceedings of 4th DELOS Workshop on Image Indexing and Retrieval*, pages 35–43, San Miniato, Italy, 1997.

- Carlo Meghini, Fabrizio Sebastiani, and Umberto Straccia. On the role of logic in image retrieval. In *Proceedings of IR-98, 1st Workshop on Image Retrieval*, Milano, Italy, 1998.

- Carlo Meghini, Fabrizio Sebastiani, and Umberto Straccia. MIRLOG: a logic for multimedia information retrieval. In Fabio Crestani, Mounia Lalmas, and Cornelis J. van Rijsbergen, editors, *Logic and Uncertainty in Information Retrieval: Advanced models for the representation and retrieval of information.* Kluwer Academic Publishing, Dordrecht, NL, 1998.

- Carlo Meghini, Fabrizio Sebastiani, and Umberto Straccia. A system for the fast prototyping of multidimensional image retrieval. In *Proceedings of ICMCS'99, IEEE International Conference on Multimedia Computing and Systems*, Firenze, IT, 1999.

- Umberto Straccia. Document retrieval by relevance terminological logics. In *Proceedings of MIRO Workshop*, Glasgow, 1995.
- Umberto Straccia. A sequent calculus for reasoning in four-valued description logics. In *Proc. of the Int. Conf. on Analytic Tableaux and Related Methods (TABLEAUX-97)*, number 1227 in Lecture Notes In Artificial Intelligence, pages 343–357, Pont-à-Mousson, France, 1997.
- Umberto Straccia. A four-valued fuzzy propositional logic. In *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI-97)*, pages 128–133, Nagoya, Japan, 1997.
- Umberto Straccia. A fuzzy description logic. In *Proc. of the 15th Nat. Conf. on Artificial Intelligence (AAAI-98)*, pages 594–599, 1998.
- Umberto Straccia. A framework for the retrieval of multimedia objects based on four-valued fuzzy description logics. In F. Crestani and Gabriella Pasi, editors, *Soft Computing in Information Retrieval: Techniques and Applications*. Physica Verlag (Springer Verlag), Heidelberg, Germany, 1999.

Related work

- Paolo Buongarzoni, Carlo Meghini, Rossella Salis, Fabrizio Sebastiani, and Umberto Straccia. Logical and computational properties of the description logic MIRTL. In Borgida A., Lenzerini M., Nardi D., and Nebel B., editors, *International Workshop on Description Logics*, pages 80–84, Rome, Italy, 1995.
- Carlo Meghini, Fabrizio Sebastiani, Umberto Straccia, and Costantino Thanos. A model of information retrieval based on a terminological logic. In *Proceedings of SIGIR-93, 16th International Conference on Research and Development in Information Retrieval*, pages 298–307, Pittsburgh, PA, 1993.
- Fabrizio Sebastiani and Umberto Straccia. Default reasoning in a terminological logic. *Computers and Artificial Intelligence*, 14(3):225–251, 1995.
- Umberto Straccia. Default inheritance reasoning in hybrid KL-ONE-style logics. In *Proceedings of IJCAI-93, 13th International Joint Conference on Artificial Intelligence*, pages 676–681, Chambery, France, 1993.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The great improvements in hardware, software and communication technologies achieved in the last years have radically changed the way in which information is produced and made available to computer users. The appearance of multimedia document repositories, (*e.g.* the Web) is amongst the most notable change in the information system scenario. Nowadays a large and increasing number of people attempt to satisfy their information needs by accessing such repositories, a task to which they devote a non-negligible share of their time and which has a critical impact on the quality of their work. So, it is not surprising that *Multimedia Information Retrieval* (MIR), *i.e.* the retrieval of those multimedia documents of a collection which are relevant to a user information need, has become an intensively investigated research area.

To this end, research has been carried out leading to the definition of numerous multimedia retrieval models, prototype systems and, to date, some commercial systems. A primary feature which characterises MIR is that it involves research from several fields of computer science. Just to mention the main streams, work on MIR has been carried out within Information Retrieval, Image Retrieval, Audio Retrieval, Video Retrieval, the Database community and, notably, some subfields of Artificial Intelligence like Knowledge Representation and Reasoning and Machine Learning. This variety reveals that many different aspects are involved in MIR, each requiring a specific background and methodology, and that there may be different approaches not only within the same discipline, but also across different ones.

A MIR system is composed of several subsystems, but the heart of the system is always the retrieval engine it is based on. A principled approach to the design of a retrieval engine should always start from the identification of a suitable *retrieval model*, *i.e.* of a formal specification of the three basic entities of retrieval:

- the representation d of multimedia documents D;
- the representation q (called query) of users' information needs Q; and
- the retrieval function $\mathcal{R}$, assigning a set of documents d to each information need q.

More often than not, to each retrieved document d w.r.t. a query q a degree of (system) relevance is given, called *retrieval status value* (denoted by $RSV(d, q)$), indicating the confidence the system has in being a document d relevant to the query q.

Therefore, we can characterise a retrieval model formally as follows:

- let L_{Doc} be the language for representing multimedia documents D as $d \in L_{Doc}$;
- let L_{Query} be the language for representing users' information needs Q as $q \in L_{Query}$;
- let $dc \subseteq L_{Doc}$ be a collection of documents $d \in dc$,

then a *retrieval function* $\mathcal{R}$ may be seen as a function

$$\mathcal{R}: 2^{L_{Doc}} \times L_{Query} \rightarrow 2^{(L_{Doc} \times [0,1])}, \tag{1.1}$$

i.e. given a document collection dc and a query q, $\mathcal{R}(dc, q)$ returns a set of pairs (d, n), where to each document d the confidence, n, the system has in being the document d relevant to the query q is associated, *i.e.* $n = RSV(d, q)$.

Given a retrieval model, the interaction with a simple MIR system may be described as follows (see Figure 1.1).

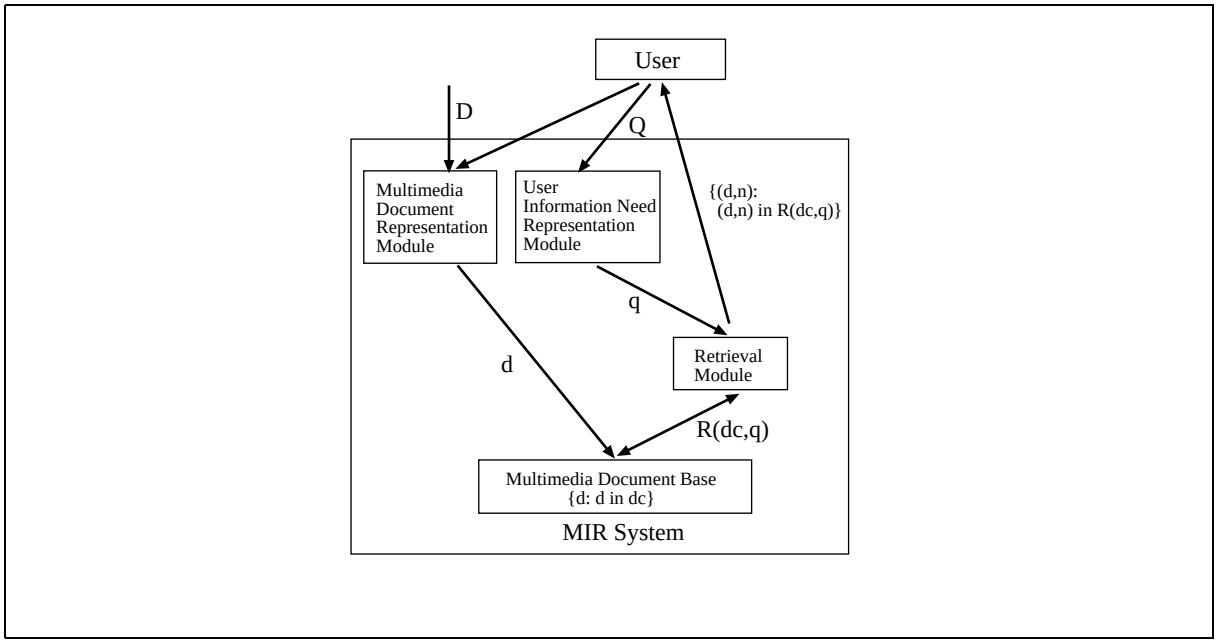


Figure 1.1: A simple MIR system.

- A document D is submitted to the MIR system in order to be stored into the multimedia document base. Its representation d is manually, semi-automatically or automatically computed and stored into the document collection dc. A user may help during this phase.
- A query Q is submitted to the MIR system by a user. Its representation q is determined. The result list $\mathcal{R}(dc, q)$ is shown to the user.

Being more concrete,

- in *Information Retrieval* (IR) systems, a text document D and an information need Q are represented as a vector of weighted terms, *i.e.* a vector in $[0,1]^r$, *i.e.* $L_{Doc} = L_{Query} = [0,1]^r$. The retrieval function $\mathcal{R}$ is defined in terms of a similarity function $f(q,d)$ between a document representation and a query representation. The similarity function $f(q,d)$ is then determined according to some relatedness measure between vectors $q, d \in [0,1]^r$, *e.g.* euclidian distance, cosine, etc. (see *e.g.* [242] for a text book on IR and SMART [244, Chapter 4] an example for an IR system);

- in image retrieval systems (see *e.g.* Virage [44]) an image is represented through automatically extracted low-level features like color histogram, texture and shape. Queries are images too, *i.e.* a query could be of the form "find those images from an image collection similar to a given one" and, thus, are represented through the same set of features ($L_{Doc} = L_{Query}$). The retrieval function $\mathcal{R}$ is defined in terms of a similarity function f between image representations. The similarity function f is then determined according to some relatedness measure between feature attributes of the images.

Of course, the adequacy of all retrieval models for MIR purposes strongly depends on the adequacy of their components, L_{Doc}, L_{Query} and $\mathcal{R}$, where, generally, each of them plays an equally important role.

We claim that designing retrieval models for MIR is an inherently multidimensional problem which should involve an appropriate combination of notions and techniques from a number of different disciplines. For instance, a text is a syntactically correct arrangement of symbols carrying information with a meaning, called the *meaning* of the text. In the case of text the symbols are words and the correctness of their arrangement is fixed by the *syntax* of natural language. Similarly, w.r.t. an image, the symbols are color regions, whose disposition in space may or may not depict a scene; in the latter case the "image" is indeed meaningless, in the same way a text can be. We thus can characterise documents as having (at least) two orthogonal dimensions, that of *form* (or *syntax*) and that of *semantics* (or *meaning*). The *form* of a document is a collective name for all those features of the document that pertain to the medium that carries the document, and thus, are *media dependent*, whereas the *semantics* of a document is a collective name for those features that pertain to the slice of the real world being represented, which exists independently of the existence of a document referring to it. Unlike form, the semantics of a document is thus *media independent*.

Corresponding to the two dimensions of a document just introduced, there are three categories of retrieval: one for each dimension (*form-based retrieval* and *semantics-based retrieval*) and one concerning the combination of both of them. The retrieval of information based on form addresses, of course, the syntactical properties of documents. For instance, form-based retrieval methods automatically create the document representations to be used in retrieval by extracting low-level features from documents, such as the number of occurrences of words in text, or color distributions in images. User queries are themselves documents from which a representation is constructed which is analogous to those of documents, *i.e.* $L_{Doc} = L_{Query}$. The document and query representation are then compared with the aim of assessing their relatedness.

To the contrary, semantics-based retrieval methods rely on a symbolic representation of the meaning of documents, that is descriptions formulated in some suitable formal language. Typically, meaning representations are constructed manually, perhaps with the assistance of some automatic tool.

The main thrust of this thesis is that a data model for MIR not only needs both dimensions to be taken into account, but also requires that each of them be tackled by means of the tools most appropriate to it, and that these two sets of tools be integrated in a principled way. As a consequence, all the components, L_{Doc}, L_{Query} and $\mathcal{R}$, of a MIR model have to address this distinction. In particular, we will view L_{Doc} as a triple

$$L_{Doc} = (L_{Form}, L_{Semantics}, Interpretation), \tag{1.2}$$

where

- L_{Form} is a language for representing each "object" (or "part") of interest in a document at the form dimension;
- $L_{Semantics}$ is a language for describing the semantics of these objects; and
- $Interpretation$: $L_{Form} \rightarrow L_{Semantics}$ is a mapping which associates a meaning description to each "object" of interest at the form level.

Note that the *interpretation function* is a bridge between the form dimension and the semantics dimension determining the semantical meaning of the relevant objects identified at the form level. It is well known that interpretation functions are, in general, a *subjective* and *imprecise* matter: *e.g.* an object in an image can be interpreted in several different ways and each of these interpretation has its own degree of confidence. The definition of an automatic interpretation step is rather a hard job. In most cases this step is simply impossible. Nevertheless it is an exciting research topic and some success has been obtained in very specialised application domains (see *e.g.* [83, 214]). Therefore, is it not surprising that in most cases this step is simply done *manually*.

The main goal of this thesis is to present a MIR model which shows

1. how documents can be represented at the form level, as sets of physical features of the objects *representing* a slice of the world;
2. how documents can be represented at the semantics level, as sets of properties of the real-world objects *being represented*;
3. how these two representations can be integrated in a principled way; and
4. how the above mentioned three categories of retrieval can be addressed in a simply way.

Our data model is based on *mathematical logic* in the sense that $L_{Semantics}$ is a logic, L_{Query} is a logic too[1], the integration between L_{Form} and $L_{Semantics}$ is defined in logical terms and, consequently, $\mathcal{R}$ can be defined in terms of logical entailment. Features of documents pertaining to form are not represented explicitly in the logic, as they are best dealt with outside it. However, they impact on logical reasoning through a mechanism of "procedural attachments" [205], which implements the connection between (logical) reasoning about semantics and (non-logical) reasoning about form, thus allowing a unified logic capable of addressing both dimensions. In practice, the formalism L_{Form} we have chosen for representing multimedia documents at the form level is *object oriented* [1, 162], as the principles of object-oriented

[1] More precisely, $L_{Semantics} = L_{Query}$.

design, namely *aggregation*, *classification* and *generalisation*, has widely been used in the context of multimedia document representation [6, 138, 158, 215, 235, 288, 296], revealing its appropriateness.

The rational of the above choices relies primarily on the fact that mathematical logic, among other things, is universally acknowledged as a formal counterpart of natural language, thus the most apt tool to make concepts precise. In addition, logic has proven most successful in capturing the essential nature of information systems, and a MIR system, as expected, is no exception. More precisely, the particular mathematical logic we adopt is based on a horn extension of *Description Logics* (DLs – see *e.g.* [55, 186, 207]). DLs, which can vary from very small-scale languages to very expressive languages, are contractions of *First-Order Logic* (FOL) and have an object-oriented character (facilitating the integration between form and semantics). In fact, they provide a logical reconstruction of the so-called frame-based knowledge representation languages, with a simple well-established denotational semantics in order to capture the meaning of the most popular features of structured representation of knowledge [15, 59, 61, 58, 64, 101, 207, 286], and, as such, allow the same principles of object-oriented design (aggregation, classification and generalisation). As a consequence, it is not surprising that they have been shown to be adequate for modelling a very large class of facts relevant to information systems [67, 75], and a number of them have also proven to have decidedly better computational properties than FOL (decidability and, in some cases, also polynomiality; see e.g. [95]). In sum, DLs represent a thoroughly investigated field, and a wide range of results and automated reasoning techniques are now known which make them a viable tool for practical use.

Our MIR model subsumes all approaches developed to date, in the sense that they can be formalised in our framework. This means that our work is not only a proposal for logic-based MIR, but can also be viewed as a step forward in developing a unified framework for the various existing approaches, capturing all kinds of retrieval on multimedia documents that have been deemed as useful, and therefore investigated in the various areas of computer science mentioned above. The specific choice we have made in selecting the formalism directly relates to MIR, and is dictated by the requirement, typical of information modelling, of using a formalism which is a good compromise w.r.t. the tradeoff between expressive power and computational complexity (see *e.g.* [181, 183, 184]).

Our overall endeavour is to be understood as a contribution to the research trail initiated by van Rijsbergen's proposal that document retrieval may be identified as logical inference, *i.e.* query answering is defined in terms of logical entailment. In his seminal 1986 papers [273, 274], van Rijsbergen argued that an approach combining conditional reasoning and reasoning about uncertainty should be used, leading to the determination through logical inference of $P(d \rightarrow q)$ ("the probability that document d implies query q") as an estimation of the probability of relevance of document d to query q.

Moreover, besides the above mentioned facts, the present work brings concrete benefits to the various actors of the MIR stage:

- to the designer of a MIR system, the model provides the guidelines for the conception of systems that are able to provide a generalised retrieval service, where the existing forms of retrieval not only coexist in harmony, but can be combined in any desired manner. Following these guidelines, and using the appropriate tool that implements the model, a designer can quickly build a prototype of the system to be developed, and use such a prototype to test its adequacy to the user's functional requirements;

- to researchers working on different aspects of MIR, the model will provide the possibility of recognising that what they are doing is part of a larger endeavour. Hopefully, this will increase awareness of the limitations of the current approaches and will stimulate improvement by integration with complementary approaches. As a further benefit to researchers, the formal specification of the model, by viewing MIR as a special form of uncertain implication, may be used as a basis for formal investigations on specific aspects of MIR, not last the extension to the present model.

1.2 An introductory example

In the following section we characterise the above mentioned concepts, form and semantics dimensions of multimedia data, form-based retrieval and semantics-based retrieval, through a concrete example.

We will concentrate our attention on the case of image data only, as it best highlights the above points. Consider the image in Figure 1.2.

Figure 1.2: Snoopy and Woodstock, sweetly embracing.

It shows Snoopy and Woodstock, sweetly embracing. In the image Snoopy is white, whereas Woodstock is yellow[2]. It is well known that Snoopy is a dog, that Woodstock is a bird and that both dogs as well as birds are animals.

According to our view, it is quite clear that features like color, shape and texture of the two objects of interest (Snoopy and Woodstock) pertain to the form dimension and are media dependent. On the other hand, features like "Woodstock is a bird.", "Snoopy is dog.", "Dogs are animals." and "Birds are animals." pertain to the semantics dimension, as they describe a slice of the real world and, thus, are "true" independently of the existence of a document referring to them. Of course, these features are media independent.

In Figure 1.3, the distinction between form features and semantics features is presented in more detail. The form dimension is on the left hand side and the semantics dimension is on the right hand side. Note that, in general, multimedia data contains a large amount of significant information that cannot easily be managed unless its parts and the logical

[2]Well, due to the grey scale printing, Woodstock is the darker one. The background is red in the original picture and plays no role in our example.

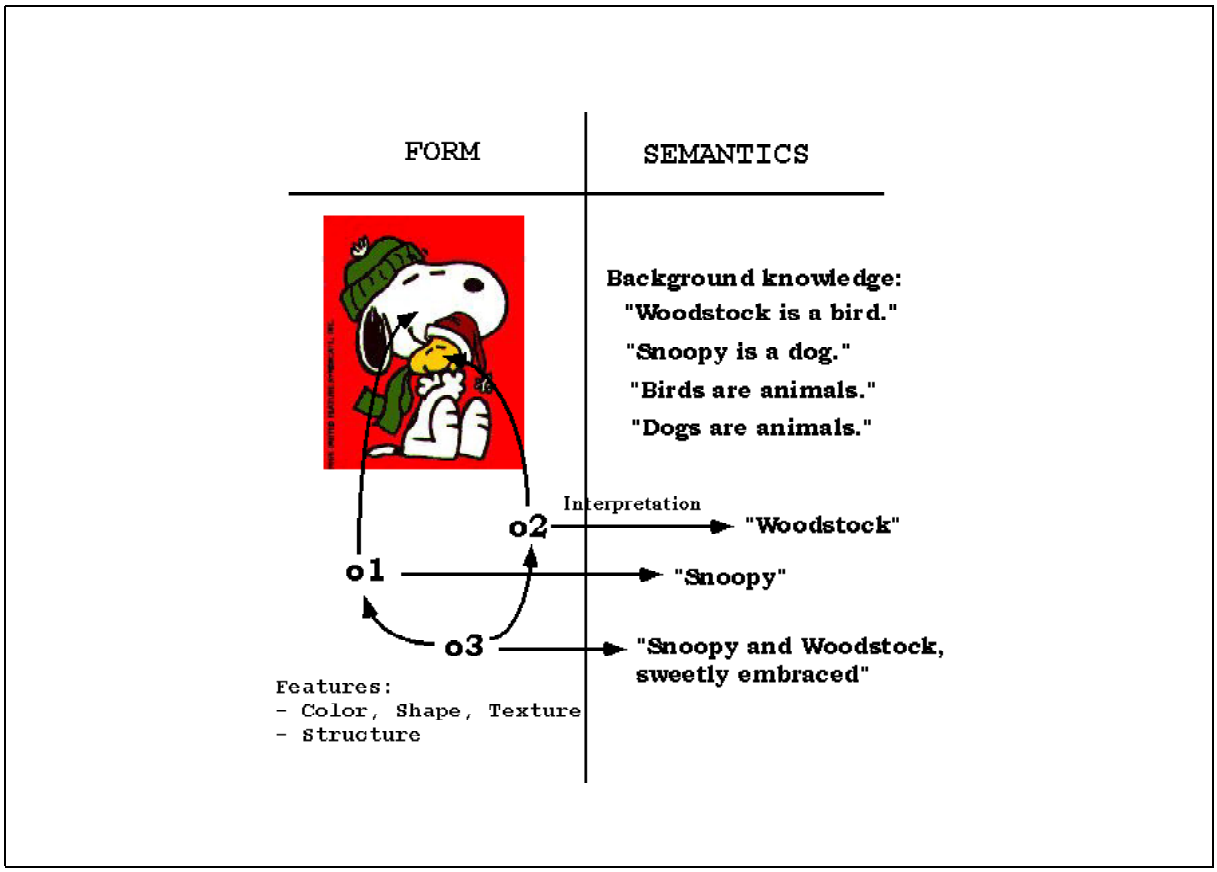


Figure 1.3: Form and semantics dimension in images.

structure between the parts are explicitly represented. This is quite clear for the text case: *e.g.* a LaTeX typed book [177] has a (automatically computable) logical structure determined by *e.g.* chapters, sections, subsections, paragraphs, etc. Generally, the logical structures are not automatically computable, but mechanisms to explicit them are provided or are desirable. The logical structures will pertain to the form dimension too. In our example we explicate a logical structure among some parts of interest: they are identified by the *objects* o1, o2 and o3. This means that at the form level we will consider the three objects o1, o2 and o3 and each of them has an automatically computed set of features pertaining to the form dimension. From a structural point of view, we specify that o3 is "composed of", *i.e.* is an aggregation of, o1 and o2.

Concerning the semantics dimension, it contains the expressions "Woodstock is a bird.", "Snoopy is a dog.", "Dogs are animals." and "Birds are animals.", which constitutes our background knowledge, and describe some properties about the real world entities Woodstock and Snoopy, respectively. The bridge between the form dimension and the semantics dimension is determined by an interpretation function, which determines the meaning of the relevant objects identified at the form level. In our example we may interpret o1 and o2 as Snoopy and Woodstock (represented through the expressions "Snoopy" and "Woodstock"), respectively, whereas o3 may be interpreted as the event represented through the expression "Snoopy and Woodstock, sweetly embracing.". In particular, we may represent these expres-

sions in logical terms[3] as follows: we will use two *individual constants*, snoopy and woodstock for representing the two real world entities Snoopy and Woodstock, respectively. We represent with the three unary predicates $\mathsf{Dog}(\cdot)$, $\mathsf{Bird}(\cdot)$ and $\mathsf{Animals}(\cdot)$, the class of dogs, birds and animals, respectively. As a consequence, the expressions "Woodstock is a bird." and "Snoopy is a dog.", can be represented through the formulae $\mathsf{Dog}(\mathsf{snoopy})$ and $\mathsf{Bird}(\mathsf{woodstock})$, respectively. The expressions "Dogs are animals." and "Birds are animals." can be represented through the formulae $\forall X.\mathsf{Dog}(X) \rightarrow \mathsf{Animal}(X)$ and $\forall X.\mathsf{Bird}(X) \rightarrow \mathsf{Animal}(X)$, respectively. Finally, the expression "Snoopy and Woodstock, sweetly embracing." is represented through, by relying on a binary predicate $\mathsf{SweetlyEmbracing}(\cdot,\cdot)$, the formula $\mathsf{SweetlyEmbracing}(\mathsf{snoopy}, \mathsf{woodstock})$. All the above formulae are allowed in our logic[4].

We will see that queries expressed in our logic can be arbitrarily complex and, in particular, can address both the form and the semantics dimension of multimedia data. For instance, concerning the form dimension we may ask

> **Find those images in which there is a white identified object.**

Such a query involves non-logical reasoning about features at the form level. In particular, a similarity measure f between colors should be given. In our logic-based framework this query is formalised through a procedural attachment [205], *i.e.* a procedure call to the function f which, for instance, is implemented in a underlying image retrieval system like Virage [44].

Concerning the semantics dimension we may ask

> **Find those images in which there is an identified object interpreted as animal.**

Here, in order to answer to the above request, it is quite obvious that logical reasoning about semantics must be performed, *i.e.* from "Snoopy is a dog" and from "Dogs are animals." we infer that "Snoopy is an animal.". These steps are performed through a theorem prover for the chosen logic.

Finally, the combination of both may be

> **Find those images in which there is a white identified object interpreted as animal, *i.e.* find those images in which there is a white animal.**

which involves both non-logical reasoning about form and logical reasoning about semantics.

1.3 Relation to previous works

In terms of the information used to represent multimedia documents, we can roughly classify the main approaches into three categories (see *e.g.* [7]).

Keyword based: In case of media like images, audio and video, the semantics of multimedia data is described through annotations provided by users, like free text or keywords.

[3] For ease, we will use FOL.

[4] Except syntactical differences.

Feature based: In this case a set of features is directly extracted, *i.e.* computed, from the machine readable representation of multimedia data. Typical features are values that represent either general information, such as, weighted term vector (case text data), color, texture, shape (case image data), position, motion (case video data), etc. or are specific for a particular application, such as face recognition, trademarks [288], and medical images [229]. Feature extraction is either performed through the supervision and support of a user, or automatically.

Concept based: In this case application domain knowledge is used to interpret a multimedia object's semantics. This interpretation leads to the recognition of concepts which are used to retrieve the object itself. Usually this process is application domain specific and may require user intervention.

Of course, retrieval systems of the first category are mainly based on manual description of multimedia objects. In this category fall models like those described in [3, 137, 193, 255, 257].

In the second category, apart from all the various text IR systems, concerning images there are commercial systems like QBIC [122], Virage [44], and experimental systems such as Photobook [228] and VisualSeek [258]. For instance, in QBIC [122], global features are color histogram, global texture and average values of color distribution. Object features include color, texture and shape. The features associated with shapes are their area, circularity, eccentricity and axis-orientation. Moreover, queries can be formulated through full image queries, which are based on global features, or by image prototypes in order to describe restrictions on objects (*e.g.* retrieve images with an object "like this"). The retrieval is performed by measuring the similarity between the query and the image database. Concerning video data, they are separated into shots, which consist of a set of contiguous frames. Each shot is represented by a single frame, r-frame, which is treated as a still image: features are extracted and stored in the database.

Finally, systems and approaches that belong to the third category are *e.g.* OVID [215], CORE [288], Infoscopes [158], and [7, 66, 198, 236, 296]. For instance, [7] extends the approaches taken in CORE and VIMSYS [138, 158]: a multimedia object is represented through a set of features and concepts, identified through an interpretation process. The same object may have multiple features and multiple interpretations. The model is object-oriented and takes the structure of an object into account too, *i.e.* the composition of a multimedia object in terms of other objects can explicitly be represented. More formally, both L_{Form} and $L_{Semantics}$ are the same object-oriented formalism L_O. L_{Query} is a SQL-like query language and *Interpretation* are fuzzy membership functions, assigning a membership degree to an object w.r.t. a concept. Unfortunately, in [7], as in almost all MIR models, the distinction between form dimension and semantics dimension has been not clearly defined, *i.e.* they are missing, or both dimensions are undistinguishable[5].

Logic, and in particular DLs, has already been used in the context of document semantics representation and information source description [134, 89, 140, 165, 197, 235, 237, 285] but they do not address all the document dimensions. As already argued, the retrieval functionality should be able to address all the document dimensions and, most importantly, be able to address each dimension *in its own modality*. In order to fulfil this goal, integration

[5]As a consequence, *e.g.* in [7] it is quite simple to get into trouble in submitting queries like "Find objects which are chapters". In this case it is not clear whether the retrieved objects are chapters, *i.e.* structured objects at the form level, or whether they are objects which are *about* chapters.

is a key concept and, indeed, the basis of our approach. In this sense, our framework, which in a large sense can be classified as belonging to the third category, can be seen as a framework in which all the current MIR methods can not only be formalised, but coexist in harmony. In particular, our framework provides the basis for integrating retrieval methods pertaining to different media and document dimensions.

1.4 Outline

The overall organisation of the work is as follows. The main parts of this work are Part I, Part II and Part III.

Part I concerns the form dimension of multimedia documents. In particular, we will formally specify an object-oriented data model for multimedia objects which is general enough in order to capture all their relevant aspects like, multimedia document structure, features, etc.

Part II, the heart of this work, presents a logic for representing the semantics of multimedia objects. This logic will also be the tool for the integration issue. The logic, called fuzzy HORN-$\mathcal{ALC}$, is characterised by

- a description logic component which allows the representation of the structured objects (of interest) in the real world;
- a horn rule component which allows us to reason about structured objects;
- a non-classical, four-valued, semantics [48, 182, 217, 222] which allows us to deal both with (*i*) possible inconsistencies arising from the representation of document semantics; and (*ii*) the caption of a simple form of relevance entailment in query answering [8];
- a fuzzy component which allows for the treatment of the inherent imprecision in multimedia document representation and retrieval.

In Part II, we will proceed step by step until our final logic is defined in terms of its syntax and its semantics.

In Chapter 6 an overview on DLs will be given. In Chapter 7 we try to go one step further in addressing the "conditional reasoning" issue, and propose a modification to the semantics for DLs in order to try and better mirror in the logic the classic document retrieval notion of "relevance". The issue we tackle in particular is that of accepting as indicative of relevance only those implications $d \rightarrow q$ in which the premise d contains information relevant to the conclusion q. This condition is identified as the requirement that evidence supporting the conclusion be explicitly present in the premise. This switch of focus is accomplished by abandoning classical logic in favour of relevance logic, which in turn implies abandoning classical two-valued semantics in favour of four-valued semantics. For readability purposes, we will first present the semantics within propositional logic, discussing several basic properties, and then extend it to the DL (first-order) case. We will conclude Chapter 7 by formally specifying the horn extension of DLs.

The logic defined in Chapter 7 does not allow for the treatment of the imprecision inherent in multimedia document representation and retrieval. This will be the topic of Chapter 8. In particular, a fuzzy extension of the logic developed in Chapter 7 will be specified. Chapter 8 follows the organisation of Chapter 7: we will first present our fuzzy extension within

propositional logic and then we will extend it to the DL case. We will conclude Chapter 8 by formally specifying our final logic: the logic fuzzy Horn-$\mathcal{ALC}$.

Finally, Part III addresses the integration issue. We will show how reasoning and retrieval about form and semantics of multimedia objects may be formalised through the logic fuzzy Horn-$\mathcal{ALC}$ defined in Part II.

For all logics presented here, syntax, semantics, and sound and complete reasoning algorithms will be provided. In most cases computational complexity results will be given.

The thesis concludes with Part IV of a chapter containing the contributions of the thesis and pointers to areas for further work.

One of the first papers that was published on a DL based approach to multimedia document retrieval was [201]. In this paper, a description logic, called Mirtl, has been defined in order to represent structured documents. Document retrieval has been defined in terms of logical entailment. The connection of this work to our thesis is limited to the idea of using DLs for multimedia document representation and retrieval. In [203, 266] a four-valued DL has been presented. [203] shows its usefulness to multimedia document retrieval, whereas in [266] a Gentzen-style sequent calculus for reasoning in the logic described in [203] is presented and related computational complexity results have been given. Chapter 7 extends the works [203, 266] by providing a more powerful DL: from an expressive power point of view horn rules will be added and more efficient reasoning algorithms will be provided. In [265], a four-valued fuzzy propositional logic is presented. Chapter 8 inherits some aspects of the propositional logic defined in [265], but a slightly different fuzzy semantics will be defined: [265] is based on Gougen's fuzzy implication, whereas we will adopt the well known Zadeh semantics [298]. Further, some ideas presented in [267], where a classical fuzzy semantics for DLs is presented, are inherited in Chapter 8 and are substantially modified and extended.

1.5 Conventions

Throughout this thesis we assume that every introduced metavariable has an optional subscript or superscript.

The knowledge bases and formulae are always assumed to be *finite*.

*Sections and *Subsections marked with a single asterisk contain in depth studies on particular aspects which can be skipped by those who are not interested in such details.

**Sections and **Subsections marked with a double asterisk contain related aspects which can be skipped, as not necessarily of general interest.

Part I

Representing the form of multimedia objects

Chapter 2

Preview

In this part our focus will be given to the definition of a *multimedia data model* which allows the representation of multimedia documents along the form dimension.

The model we will present here inherits most of the ideas presented in the literature (especially [7]) and is based on the representation of multimedia objects through features and other properties. The aim of the model is to be quite general such that most approaches can be fit into our model. The easiest way in doing this is to rely on an object-oriented model.

The advantages of the model presented here can be summarised as follows:

- the representation of the *structure* of multimedia data is allowed. This means that the composition of multimedia objects in terms of other multimedia objects can be explicitly be represented, and that restrictions on the structure can be expressed in queries;
- features and their characteristics are not predefined. New features can be created according to the application needs. New features can be customised by defining specific extraction functions and functions for measuring the similarity of the values of the features.

In order to give a formal foundation of our work, we will formally define the object-oriented database model we will rely on. This will be the topic of Chapter 3. Chapter 4, the core of this part, formally specifies the representation of multimedia documents through the object-oriented model presented in Chapter 3.

Chapter 3

Preliminaries: an object-oriented data model

Let us present here some basic definitions about the *object-oriented data model* we rely on. These are not meant to be very detailed at all, but are sufficient to let the presentation to be self contained. For a more detailed and formal presentation see *e.g.* [1, 162].

Consider an alphabet of *atomic types* $\textsc{Type}_A$ (denoted by T), *e.g.* Integer, String, bool, float. Given an atomic type $T \in \textsc{Type}_A$, the corresponding set of *values* is indicated with $\textsc{Value}(T)$. For instance, $\textsc{Value}(\mathsf{Integer}) = \{-3, -2, -1, 0, 1, 2, 3\}$. The set $\textsc{Value}_A$ of *atomic values* is

$$\textsc{Value}_A = \bigcup_{T \in \textsc{Type}_A} \textsc{Value}(T). \tag{3.1}$$

The elements of $\textsc{Value}_A$ are also called *constants*.

We also assume an alphabet ObjectId of *Object Identifiers* (OIDs) (denoted by o), an alphabet Class of *Class Names* (CNs) (denoted by C), and an alphabet AttrName of *Attribute Names* (ANs) (denoted by A). With OID, CN and AN we indicate a set of object identifiers, class names and attribute names, respectively.

Given ObjectId, the alphabet Value of *values* (denoted by v) over ObjectId is defined so that

1. *nil*, each element of $\textsc{Value}_A$, and each element of ObjectId are values over ObjectId; and
2. if $v_1, \ldots, v_n$ are values over ObjectId, and $A_1, \ldots, A_n$ are distinct attribute names, then the tuple $[A_1{:}v_1, \ldots, A_n{:}v_n]$ and the set $\{v_1, \ldots, v_n\}$ are values over ObjectId.

An *object* is a pair (o, v), where o is an OID and v is a value. For instance,

$$(\mathsf{o12}, [\mathsf{Name}{:}\text{“Umberto Straccia”}, \mathsf{Hobby}{:}\{\text{“Football”}, \text{“Music”}\}])$$

is an object.

Objects are grouped into classes. All objects in a class have complex values of the same type. The type corresponding to each class is specified by the Object-Oriented Data Base (OODB) schema.

Types are defined with respect to a given set of class names CN. The set TY of *types* (denoted by T) over CN is defined so that

1. each atomic type is a type, *i.e.* Integer, String, bool, float, ... are types;
2. the class names in CN are types;
3. if T is a type, then $\{T\}$ is a (set) type;
4. if $T_1, \ldots, T_n$ are types and $A_1, \ldots, A_n$ are distinct attribute names, then the tuple $[A_1{:}T_1, \ldots, A_n{:}T_n]$ is a (tuple) type.

With Type we will indicate the alphabet of types over Class. Just notice the close resemblance with types used in the complex value model. For example, the type of object o12 above is the following:

$$[\mathsf{Name{:}String}, \mathsf{Hobby{:}\{String\}}].$$

Generally, one may want to give a name to types *e.g.* Person. We will write in these cases $C = T$, where C is a CN and T is a type. For instance,

$$\mathsf{Person} = [\mathsf{Name{:}String}, \mathsf{Hobby{:}\{String\}}].$$

In an OODB schema we associate with each concept name C a type T, $C = T$, which dictates the type of the objects in this class. In particular, for each object (o, v) in class C, v must have the exact structure described by T. Moreover, any such schema includes an ISA hierarchy among the classes of the schema. The class hierarchy has three components: (i) a set of classes, (ii) the types associated with these classes, and (iii) a specification of the ISA relationship between the classes. Formally, a *class hierarchy* (denoted by CH) is a triple $\mathsf{CH} = (\mathsf{CN}, =, \prec)$, where CN is the set of class names, $=$ is a type assignment[1]

$$=: \mathsf{CN} \to \mathsf{TY},$$

where TY is a set of types over CN, and $\prec$ is a partial order on CN. Of course, in a class hierarchy the type associated with a subclass should be a refinement of the type associated with its superclass. For instance, a class Student is expected to refine the information on its superclass Person by providing additional attributes.

Let $\mathsf{CH} = (\mathsf{CN}, =, \prec)$ be a class hierarchy. An OID *assignment* over CH is a function π mapping each class name in CN to a disjoint finite set of OIDs, *i.e.*

$$\pi: \mathsf{CN} \to 2^{\textsc{ObjectId}}$$

such that $\pi(C_1) \cap \pi(C_2) = \emptyset$, if $C_1 \neq C_2$.

The *disjoint extension* over CH of $C_1 \in \mathsf{CN}$ is $\pi(C_1)$, whereas the *extension* of C_1, denoted $\pi^*(C_1)$, is

$$\pi^*(C_1) = \bigcup\{\pi(C_2) : C_2 \in \mathsf{CN}, C_2 \prec C_1\} \tag{3.2}$$

[1] We use infix notation. Hence, we will write $C = T$ in place of $=(C, T)$.

Hence, if π is an OID assignment over CH, then $\pi^*(C_2) \subseteq \pi^*(C_1)$ whenever $C_2 \prec C_1$. This should be understood as a formalization of the fact that an object of a subclass C_2 may be viewed also as an object of a superclass C_1. Just notice that, unlike many semantic models, the definition of OID assignment for OODB schemas implies that extensions of classes of an ISA hierarchy without common subclass are *necessary disjoint*.

The final ingredient of the generic OODB model are *methods*. A method has three components:

1. a *name*;
2. a *signature*;
3. an *implementation* (or *body*).

There is no problem in specifying the names and signature of methods in an OODB schema. To specify the implementation of methods, a language for methods is needed. We do not consider specific languages in the generic OODB model. Therefore only names and signature of methods are specified at the schema level.

Let METHOD be the alphabet of *Method Names* (MN) (denoted by m). With MET we indicate a set of method names. Let $\mathsf{CH} = (\mathsf{CN}, =, \prec)$ be a class hierarchy. For a method m, a *signature* is an expression of the form

$$m\colon C \times T_1 \times \ldots \times T_{n-1} \to T_n, \tag{3.3}$$

where $C \in \mathsf{CN}$ is a class name and T_i are types over CN. This signature is associated with class C; we say that method m *applies* to objects of class C and to objects of classes that inherit m from C.

Examples of methods of the class Person may be

$$\begin{aligned}&\mathsf{GetName}\colon \mathsf{Person} \to \mathsf{String}\\ &\mathsf{GetNameLength}\colon \mathsf{Person} \to \mathsf{Integer}\\ &\mathsf{GetHobby}\colon \mathsf{Person} \to \{\mathsf{String}\}\\ &\mathsf{GetNumberofHobbies}\colon \mathsf{Person} \to \mathsf{Integer},\end{aligned}$$

and for the object $\mathsf{o12}$ described above may have

$$\begin{aligned}&\mathsf{GetName}(\mathsf{o12}) = \text{“}\mathsf{Umberto\ Straccia}\text{”}\\ &\mathsf{GetNameLength}(\mathsf{o12}) = 16\\ &\mathsf{GetHobby}(\mathsf{o12}) = \{\mathsf{Football}, \mathsf{Music}\}\\ &\mathsf{GetNumberofHobbies}(\mathsf{o12}) = 2.\end{aligned}$$

As usual, we assume that for each class C of type $[A_1{:}T_1, \ldots, A_n{:}T_n]$, there are *implicit methods* $A_1, \ldots, A_n$ each of these are of signature $A_i{:}C \to T_i$. For instance, for the class Person there are the two implicit methods

$$\begin{aligned}&\mathsf{Name}\colon \mathsf{Person} \to \mathsf{String}\\ &\mathsf{Hobby}\colon \mathsf{Person} \to \{\mathsf{String}\}.\end{aligned}$$

such that

$$\mathsf{Name(o12)} = \text{“Umberto Straccia”}$$
$$\mathsf{Hobby(o12)} = \{\mathsf{Football}, \mathsf{Music}\}.$$

Let (o, v) be an object of a class C. The application of a method m to (o, v) is also indicated with $o.m$. For instance,

$$\mathsf{o12.Name} = \text{“Umberto Straccia”}$$
$$\mathsf{o12.Hobby} = \{\mathsf{Football}, \mathsf{Music}\}.$$

Now, we are ready to define what an object-oriented database is. An *Object-Oriented DataBase* (OODB) (denoted by OODB) is a tuple

$$\mathsf{OODB} = (\mathsf{CH}, \mathsf{TY}, \mathsf{MET}, \pi, \mathsf{OID}) \tag{3.4}$$

where $\mathsf{CH} = (\mathsf{CN}, =, \prec)$ is a class hierarchy, TY is the set of types over CN, MET is a set of method names m with signature $m{:}\, C \times T_1 \times \ldots \times T_{n-1} \to T_n$ such that $C, T_1, \ldots, T_n \in \mathsf{TY}$, π is an OID assignment over CN and OID is a set of objects (o, v) such that

$$\begin{aligned} \mathsf{OID} = \{&(o, v)\text{: there is unique class } C \in \mathsf{CN} \text{ such that} \\ &C = T, T \in \mathsf{TY}, o \in \pi(C) \text{ and } v \in \textsc{Value}(T)\}. \end{aligned} \tag{3.5}$$

Essentially, in a object-oriented database the components CN specifies the classes considered, $=$ specifies the type of the classes, $\prec$ specifies the hierarchy between the classes, MET specifies the set of methods of the classes, π specifies the set of instances of each class and OID is the set of all objects considered.

A particular form of OODB we will consider later on in Part III concerns OODB in normal form. We will say that an OODB is in *normal form* iff TY, MET, and OID are in normal form. A set of types TY over sf CN is in *normal form* iff it is a set of normal types. A *normal type* over sf CN is defined inductively as follows

1. an atomic type $T \in \textsc{Type}_A$ is a basic type;
2. a class name $C \in \mathsf{CN}$ is a basic type;
3. if T is an atomic type or a class name, then $\{T\}$ is a basic type;
4. a basic type is a normal type;
5. if $T_1, \ldots T_n$ are basic types and $A_1, \ldots A_n$ are attribute names, then $[A_1{:}T_1, \ldots, A_n{:}T_n]$ is a normal type.

For instance,

$$[\mathsf{Code{:}Integer}, \mathsf{Info{:}Person}],$$

where

$$\mathsf{Person} = [\mathsf{Name{:}String}, \mathsf{Age{:}Integer}]$$

are both normal types. On the other hand side,

$$[\mathsf{Code{:}Integer}, \mathsf{Info{:}}[\mathsf{Name{:}String}, \mathsf{Age{:}Integer}]]$$

is not a normal type.

The set OID is in *normal form* iff it is a set of normal objects. An object (o, v) is in *normal form* iff v is a normal value. A *normal value* is defined inductively as follows

1. *nil* or an atomic value $v \in \textsc{Value}_A$ is a basic value;
2. an OID o is a basic value;
3. if $v_1, \ldots, v_n$ are atomic values or OIDs, then $\{v_1, \ldots, v_n\}$ is a basic value;
4. a basic value is a normal value;
5. if $v_1, \ldots v_n$ are basic values and $A_1, \ldots A_n$ are attribute names, then $[A_1{:}v_1, \ldots, A_n{:}v_n]$ is a normal value.

It is quite easy to see that there is a strict correlation between normal types and normal values. Finally, MET is in *normal form* iff all $m \in \mathsf{MET}$ are of the form

$$m{:}\, C \times T_1 \times \ldots \times T_{n-1} \to T_n,$$

where C, $T_1 \times \ldots \times T_{n-1}$ are basic types not being set types, *i.e.* atomic types or classes.

Chapter 4

Representing the form

4.1 Model Overview

The model we will adopt inherits the ideas from [2, 7, 81, 140, 297]. Mainly, the model allows the representation of three types of objects, namely, *media data objects*, *complex single media objects* and *complex multimedia objects*. Roughly, a media data object represents the raw multimedia data, like a text, an image, a video and an audio. These are the real pieces of multimedia data. None of these data contain any specification regarding *e.g.* their structure and features. The unstructured media data object can be conveniently structured by representing portions of it as basic objects and then assembling such basic objects into a complex objects. For instance, in a book, the sections may be the basic objects which can be assembled yielding the chapters and the chapters assembled together form the book. These complex objects are called complex *single* media objects, as each of their components refers to the same media data object, *e.g.* in the book case, each component of a complex object refers to a piece of text within the *same* book. The natural generalisation of the concept of complex object over the same media data object is the extension of the aggregation principle to the case in which an object could be an aggregation of objects referring to pieces in different media data objects. For instance, a book made by text parts and images can be represented through such *complex multimedia objects*. Similarly, complex multimedia objects allow to represent HTML documents (see *e.g.* [279]), in which several media types could be involved.

The overall organisation of this chapter is as follows. In the following section, media data objects are defined. Section 4.3 addresses the definition of complex single media objects, whereas Section 4.4 concerns complex multimedia objects. Section 4.5 concludes in defining a multimedia databases about form as an OODB collecting all types of multimedia objects defined so far.

4.2 Media data objects

At the lowest level we have the representation of *raw multimedia data*, like a text, an image, a video and an audio, which are acquired in some way (scanner, video camera, ...). We simply view raw data as a sequence of bytes B. Typically, a portion of such data contains information about their physical encoding and the remaining data consists of an unstructured linear stream of bytes. It is not our interest to investigate how this data is stored and accessed. The interested reader can consult *e.g.* [130].

At first, we assume that there is an atomic type ByteSeq $\in$ TYPE$_A$ whose set of values, VALUE(ByteSeq), is the set of *sequences of bytes* (denoted by B), also viewed as *ordered set of bytes*. Moreover, let us assume that there are class names for each media type we are interested in, *i.e.* let Media, Text, Image, Video and Audio be class names such that Text $\prec$ Media, Image $\prec$ Media, Video $\prec$ Media, Audio $\prec$ Media, ... Finally, let Location be a class name.

A multimedia database is populated by *Media Data Objects* (MDOs) (denoted by *mdo*) which represents multimedia raw data. The definition of media data object is given by the specification of the class of media data objects MDO:

$$\mathsf{MDO} = [A_1{:}T_1, \ldots, A_n{:}T_n]. \tag{4.1}$$

Let (o, v) be a media data object (*i.e.* $o \in \pi^*(\mathsf{MDO})$). The tuple $[A_1{:}T_1, \ldots, A_n{:}T_n]$ typically contains the following attributes:

Data:Location. o.Data $\in$ VALUE(Location) specifies the *location* where the real data is stored. It could be a pointer, file name, URL, etc. We do not specify the type definition of Location, which is application dependent. We assume is that there is a method GetData with signature

$$\mathsf{GetData}\colon \mathsf{Location} \to \mathsf{ByteSeq} \tag{4.2}$$

which given an object $(\mathsf{o}', \mathsf{v}')$, instance of class Location, returns the raw object: GetData(o′) $\in$ VALUE(ByteSeq)), *i.e.* GetData(o′.Data) $\in$ VALUE(ByteSeq)) is the real multimedia raw data addressed by (o, v). This attribute is mandatory.

Size:Integer. o.Size indicates the size of the byte sequence o.Data.

CreationDate:Date. o.Data is the creation date of the MDO (o, v). For instance, o.CreationDate may be "Thu Dec 4 10:00:19 MET 1997".

MediaType:Media. o.MediaType $\in$ VALUE(Media) is the type of the MDO (o, v). For instance, o.MediaType = o13, where o13 $\in \pi^*$(Image) indicates that (o, v) represents an image given by o.Data.

Format:String. o.Format identifies the format of the encoding o.Data. For instance, o.Format = "gif" indicates that (o, v) is a `gif` encoded image. Of course for each media type T there are several possible encodings. For instance, o.Format may assume one of the following values ascii, rtf, tex, latex, doc, html, sgml, ... (case Text), gif, tiff, eps, jpg, pict, ... (case Image), mpg, mov, ... (case Video) and au, mid, aif, wav, ... (case Video).

Author:String. o.Author identifies the author of the MDO (o, v). For instance, o.Author = "Umberto Straccia" and o.MediaType = o13 $\in \pi^*$(Image) indicate that the author of the image identified by o.Data is "Umberto Straccia".

Of course, no restrictions are made in principle on the set of attributes of MDO.

Let us conclude this section with a simple example about MDOs and a possible class hierarchy.

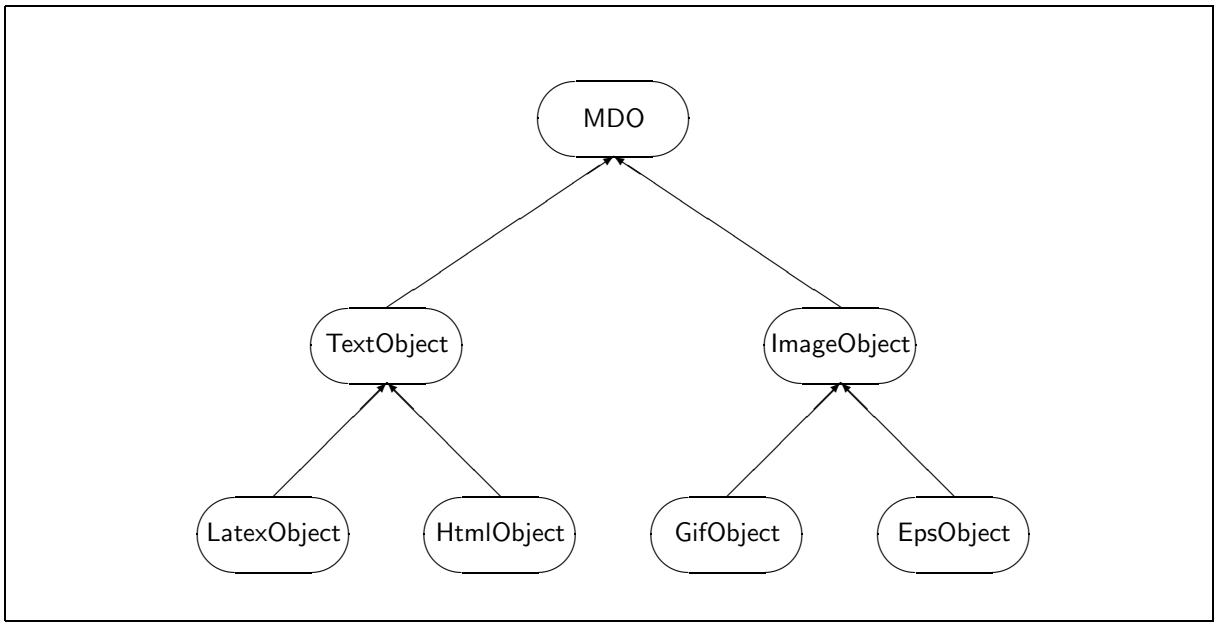


Figure 4.1: Media data object type taxonomy.

Example 1 Suppose that the set of class names we are considering contains the classes MDO, TextObject, ImageObject, LatexObject, HtmlObject, GifObject and EpsObject hierarchically organised as in Figure 4.1, according to their media type and format.
More formally, $\prec$ is defined as

$$\mathsf{TextObject} \prec \mathsf{MDO}, \quad \mathsf{ImageObject} \prec \mathsf{MDO}, \quad \mathsf{LatexObject} \prec \mathsf{TextObject},$$

$$\mathsf{HtmlObject} \prec \mathsf{TextObject}, \quad \mathsf{GifObject} \prec \mathsf{ImageObject}, \quad \mathsf{EpsObject} \prec \mathsf{ImageObject}.$$

The types of the classes are:

MDO	=	[Data	:	Location,	TextObject	=	[Data	:	Location,
		Size	:	Integer,			Size	:	Integer,
		CreationDate	:	Date,			CreationDate	:	Date,
		MediaType	:	Media,			MediaType	:	Text,
		Format	:	String,			Format	:	String,
		Author	:	String]			Author	:	String]
ImageObject	=	[Data	:	Location,	LatexObject	=	[Data	:	Location,
		Size	:	Integer,			Size	:	Integer,
		CreationDate	:	Date,			CreationDate	:	Date,
		MediaType	:	Image,			MediaType	:	Text,
		Format	:	String,			Format	:	Latex,
		Author	:	String]			Author	:	String]

HtmlObject	=	[Data	:	Location,	HtmlObject	=	[Data	:	Location,
		Size	:	Integer,			Size	:	Integer,
		CreationDate	:	Date,			CreationDate	:	Date,
		MediaType	:	Text,			MediaType	:	Image,
		Format	:	Html,			Format	:	Gif,
		Author	:	String]			Author	:	String]
EpsObject	=	[Data	:	Location,					
		Size	:	Integer,					
		CreationDate	:	Date,					
		MediaType	:	Image,					
		Format	:	Eps,					
		Author	:	String]					

where Latex = String, Html = String, Gif = String and Eps = String.

Our database contains two MDOs $(o1, v1)$ and $(o2, v2)$ which are objects of the classes LatexObject and EpsObject, respectively. They are defined as

(o1,	[Data	:	o_{L1},
	Size	:	414,
	CreationDate	:	"Thu Dec 4 10:17:31 MET 1997",
	MediaType	:	text,
	Format	:	"latex",
	Author	:	"Straccia"])

where GetData(o_{L1}) is the text[1]

```
\documentstyle{article}
\begin{document}
    Don't worry, be happy.
\end{document}
```

and

(o2,	[Data	:	o_{L2},
	Size	:	61510,
	CreationDate	:	"Fri Dec 5 11:36:48 MET 1997",
	MediaType	:	image,
	Format	:	"eps",
	Author	:	"Straccia"])

where GetData(o_{L2}) is the image

[1]For simplicity, in the following we will put the byte sequence into a readable form.

Of course, other kinds of MDOs, like video and audio, may be defined similarly.

■

4.3 Complex single media objects

So far, we have described the lowest level of representation, that is the representation of raw multimedia data. But, a multimedia data contains a large amount of significant information that cannot easily be managed unless its parts and the logical structure between these parts are explicitly represented. For instance, a LaTeX typed book [177] has a (automatically computable) logical structure determined by its parts, *i.e.* chapters, sections, subsections, paragraphs, etc. More often, the parts and logical structures among themselves are not automatically computable, but mechanisms to explicit them are provided or are desirable. For instance, it is desirable to allow to identify a house in a picture by starting to define it as composed of some basic objects, *e.g.* the roof, the front, the window and its door.

Object oriented data models are well known to satisfy this kind of desiderata as they allow the principles of *aggregation*, *classification* and *generalisation* [14]. As it happens in almost all multimedia models (see *e.g.* [6, 66, 138, 158, 214, 215, 236, 288, 296]), we allow the description of a *Complex Single Media Object* (CSMO) (denoted by *csmo*). A CSMO is meant to refer to a relevant portion of data of a given MDO and allows its aggregation with other objects. For instance, in the book example, the parts of the book (chapters, sections, etc.) may be modelled as CSMOs, which refer to the corresponding sequence of bytes, and these objects are related among themselves according the logical structure of the book. Whereas, in the image example, we have five CSMOs (which refers to five regions/parts of the image): one refers to the region representing the house, one to the roof, one to the front, one to the window and one to the door, respectively. Similarly, in case of MDOs of type video and audio, parts are simply sequences of video frames and timed intervals of audio streams, respectively.

4.3.1 Regions

A CSMO represents a piece of data of a MDO. We will call this piece of data *region*. The topic of this section is, thus, to formally specify what we mean with "part" of a text, an image, a video, etc., which a CSMO refers to.

Let us assume that there is a CN Region, the class of regions, *i.e.* each object (o, v) of class Region allows us to identify uniquely a region in a MDO. In practice, each (o, v) encodes a membership function. The class Region has type

$$\mathsf{Region} = [A_1{:}T_1, \ldots, A_n{:}T_n], \tag{4.3}$$

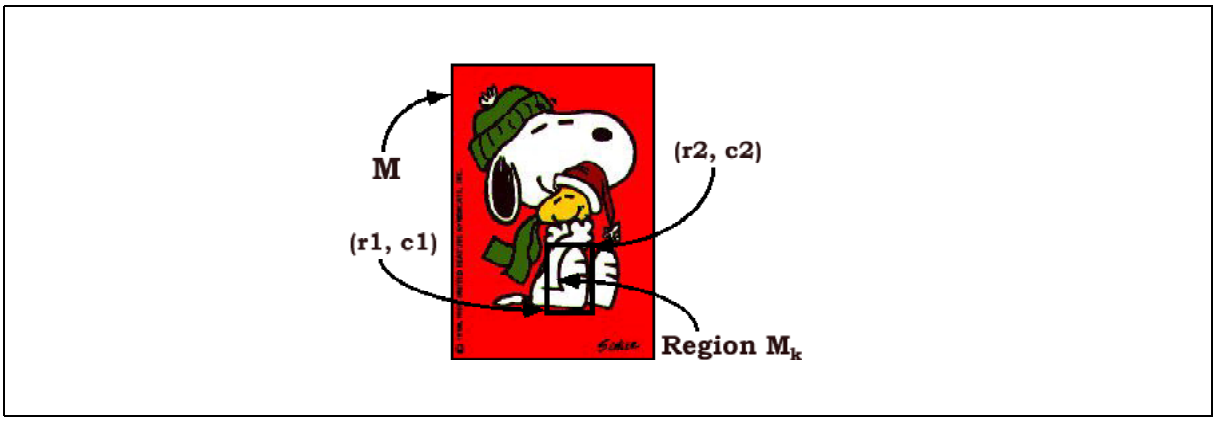


Figure 4.2: Regions in an image.

where $A_1, \ldots, A_n$ are attributes useful for determining a region. We also assume that there is a membership method m_μ of the class Region with signature

$$m_\mu\colon \mathsf{Region} \times \mathsf{MDO} \times \mathsf{Byte} \to \{0,1\}, \tag{4.4}$$

where $\mathsf{Byte} \in \textsc{Type}_A$ is an atomic type such that $\textsc{Value}(\mathsf{Byte})$ is the set of bytes. The purpose of method m_μ is to implement the membership function of a region in a MDO, *i.e.* m_μ determines whether a give byte belongs to a specified region of a MDO. This can best be explained by means of the following example.

Example 2 Let us consider the MDO $(\mathsf{o2}, \mathsf{v})$ of the class EpsObject specified in Example 1. Suppose the image (sequence of bytes) GetData(o2.Data) is viewed abstractly as a matrix M of bytes. A region of it is identified as a submatrix M_k of M, *i.e.* a region is identified as a "rectangle" (see Figure 4.2).
It is not important here to know how M is encoded in terms of byte sequence $B = \mathsf{o2.Data}$. Just let us assume that the class EpsObject has two methods m_{row} and m_{col} of signature

$$m_{row}\colon \mathsf{EpsObject} \times \mathsf{Byte} \to \mathsf{Integer}$$
$$m_{col}\colon \mathsf{EpsObject} \times \mathsf{Byte} \to \mathsf{Integer}$$

which given an object (o, v) of class EpsObject and a byte $b \in o.\mathsf{Data}$ return b's row and column position in the matrix of bytes M, respectively. Of course, in order to identify a region M_k of M it is sufficient to consider the "coordinates" of the left bottom edge (r_1, c_1) and the top right edge (r_2, c_2) of the matrix M_k. As a consequence, we could define the class Region having four attributes like:

$$\mathsf{Region} = [\mathsf{R_1}\text{:Integer}, \mathsf{C_1}\text{:Integer}, \mathsf{R_2}\text{:Integer}, \mathsf{C_2}\text{:Integer}]$$

and a method $m_\mu\colon \mathsf{Region} \times \mathsf{EpsObject} \times \mathsf{Byte} \to \{0,1\}$. The method m_μ is defined as follows: let (o, v) be an object of class Region, let (o, v) be an object of the class EpsObject and let $b \in o.\mathsf{Data}$ a byte. With $Row(b) = m_{row}(o', b)$ we indicate b's row position and with $Col(b) = m_{col}(o', b)$ we indicate b's column position. Then

$$m_\mu(o, o', b) = \begin{cases} 1 & \text{if both } o.\mathsf{R}_2 \leq Row(b) \leq o.\mathsf{R}_1, \text{ and} \\ & \quad o.\mathsf{C}_1 \leq Col(b) \leq o.\mathsf{C}_2 \\ 0 & otherwise \end{cases}$$

It is easily verified that region R (set of bytes) identified by o w.r.t. o' is

$$R = \{b \in o'.\mathsf{Data}\colon m_\mu(o, o', b) = 1\}.$$

Of course, rather than rectangular regions, other geometrical forms of regions can be defined as well, like polygons etc. Our model does not impose any particular restriction. Moreover a class hierarchy of regions can be defined too, like GeometricalRegion $\prec$ Region, PolygonRegion $\prec$ GeometricalRegion, CircleRegion $\prec$ GeometricalRegion, RectangleRegion $\prec$ PolygonRegion, SquareRegion $\prec$ PolygonRegion, and so on. For instance, we may have

$$\mathsf{RectangleRegion} = [\mathsf{R_1}\text{:Integer}, \mathsf{C_1}\text{:Integer}, \mathsf{R_2}\text{:Integer}, \mathsf{C_2}\text{:Integer}],$$

and

$$\mathsf{CircleRegion} = [\mathsf{Radius}\text{:Float}, \mathsf{CenterX}\text{:Integer}, \mathsf{CenterY}\text{:Integer}],$$

By relying on the introduced objects, as example, in case of object $(\mathsf{o2}, \mathsf{v})$, the region M_k in Figure 4.2 is identified by the object $(\mathsf{o}_{\mathsf{M_k}}, \mathsf{v}_{\mathsf{M_k}})$ of class RectangleRegion, and is defined as:

$$(\mathsf{o}_{\mathsf{M_k}}, [\mathsf{R_1}\colon \mathsf{r1}, \mathsf{C_1}\colon \mathsf{c1}, \mathsf{R_2}\colon \mathsf{r2}, \mathsf{C_2}\colon \mathsf{c2}])$$

■

4.3.1.1 **Topological space of regions

We have seen that a region is viewed as a set (of bytes). Therefore, it is quite useful to assume that there are methods of the class Region which are typical operations on sets, like union, intersection, complementation, subregion, superregion, ... with signature

$$\begin{array}{l} \mathsf{union}\colon \mathsf{Region} \times \mathsf{Region} \times \mathsf{MDO} \rightarrow \mathsf{Region} \\ \mathsf{intersection}\colon \mathsf{Region} \times \mathsf{Region} \times \mathsf{MDO} \rightarrow \mathsf{Region} \\ \mathsf{complement}\colon \mathsf{Region} \times \mathsf{MDO} \rightarrow \mathsf{Region} \\ \mathsf{subregion}\colon \mathsf{Region} \times \mathsf{MDO} \rightarrow \{\mathsf{Region}\} \\ \mathsf{superregion}\colon \mathsf{Region} \times \mathsf{MDO} \rightarrow \{\mathsf{Region}\} \\ \ldots \end{array} \tag{4.5}$$

Of course, these are not meant to be the only existing methods of class Region. Typically, for each media type there are methods of class Region which automatically, semi-automatically or manually determine the regions of a MDO. For instance, if a MDO is of class LatexObject then certainly a large amount of regions (document's structural parts) can be computed automatically. In case of images, for instance a method determining the boundaries of objects in it may help an user to select meaningful regions, and so on.

The above methods are called *topological operators*. The reason relies on the fact that the regions referred by CSMOs w.r.t. a MDO determine a topological space.

Formally, let $B \in$ VALUE(ByteSeq) be a sequence of bytes. A *region* (denoted by R) w.r.t. B is any subset $R \subseteq B$. In order to identify each region R, we assume that each region R w.r.t. B has a membership function $\mu_R\colon B \to \{0,1\}$ such that $\forall x \in B.\mu(x) = 1$ iff $x \in R$. Of course, given a membership function μ w.r.t. B, the region R identified by μ is $R = \mu(1)^{-1}$.

A *topology* τ on B (see [263, Chapter 5]) is a set of regions $\tau \subseteq 2^B$ such that

1. $B, \emptyset \in \tau$,
2. if $R_1, R_2 \in \tau$ then $R_1 \cap R_2 \in \tau$, and
3. if $R_i \in \tau$ for $i \in I$ then $\bigcup_{i \in I} R_i \in \tau$ (I is an arbitrary index set).

A *topological space* is a pair (B, τ), where B is a byte sequence and τ is a topology on B. Essentially, with a topology τ on B we will identify the set of relevant regions (parts) of a byte sequence B, as we will see later on.

Example 3 Examples of topologies are the following:

1. Consider the byte sequence $B \in$ VALUE(ByteSeq). Then $\tau = \{\emptyset, B\}$ is a topology on B, called the *trivial* topology on B.
2. Consider the byte sequence $B \in$ VALUE(ByteSeq). Then $\tau = 2^B$ is also a topology on B, called the *discrete* topology on B.

■

It is an established mathematical practice to present mathematical objects as simple as possible. Hence, we will represent topologies in terms of topological bases, a subset of the topology generating it, which may be of practical convenience. Let $B \in$ VALUE(ByteSeq) be a byte sequence. A set of regions $\mathcal{R} \subseteq 2^B$ is a *topological base* on B if

1. $\bigcup_{R \in \mathcal{R}} R = B$, and
2. if $R_1, R_2 \in \mathcal{R}$ and $x \in R_1 \cap R_2$ then there is $R_3 \in \mathcal{R}$ such that $x \in R_3 \subseteq R_1 \cap R_2$.

It is worth noting that a topological base $\mathcal{R}$ on B generates a topology $\tau_{\mathcal{R}}$ on B by letting

$$\tau_{\mathcal{R}} = \{R : R = R_1 \cup R_2, R_1, R_2 \in \mathcal{R}\}. \tag{4.6}$$

Example 4 Consider the three regions identified by three circles in Figure 4.3 (we may imaging that they are regions of an image). Let B be the union of these three circles. These three regions determine 7 pairwise disjoint subregions R1, R2, ..., R7. The set

$$\mathcal{R} = \{\mathsf{R1}, \mathsf{R2}, \mathsf{R3}, \mathsf{R4}, \mathsf{R5}, \mathsf{R6}, \mathsf{R7}\}$$

is a topological base on B.

⊣

Of course, the class definition for topological bases is quite simple. Consider the CN TopoBase, the class of topological bases, with type

$$\mathsf{TopoBase} = \{\mathsf{Region}\}, \tag{4.7}$$

i.e. each object (o, v) of class TopoBase is such that the value v is a set of OIDs of Region objects, that is $v \subseteq \pi^*(\mathsf{Region})$.

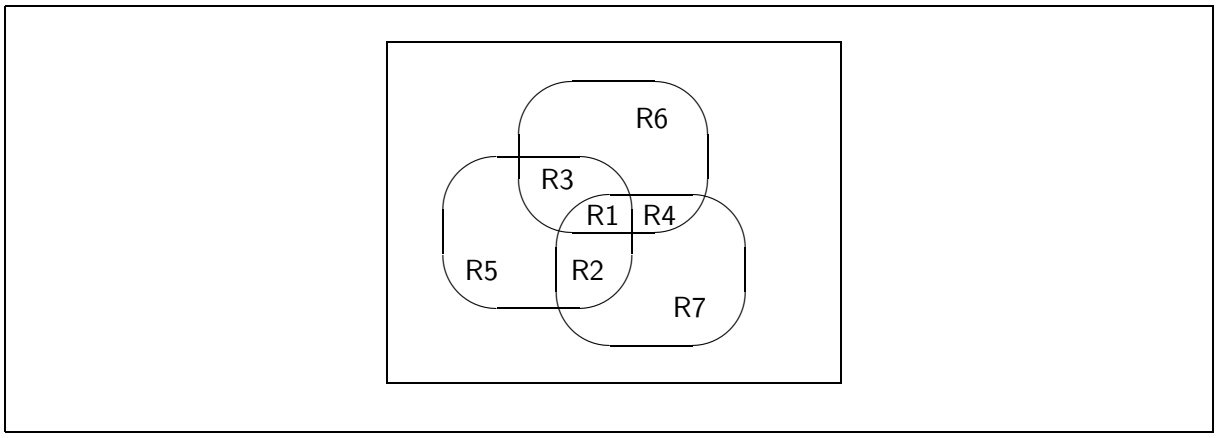


Figure 4.3: Topology of regions.

4.3.2 Complex objects

In this section we will define *Complex Single Media Objects* (CSMO) (denoted by *csmo*). Roughly, CSMOs are structured objects (aggregates of objects) with the characteristics that they refer to the *same* MDO. For instance, given a MDO of media type image, like the one identified by $(\mathsf{o2}, \mathsf{v})$ in Figure 4.2, we are interested in building a CSMO (o, v) concerning a face as composed of three parts: one referring to its eyes, one to its nose and another one to its mouth. A common characteristics of (o, v) is that (o, v) and its parts refer to regions of the *same* MDO $(\mathsf{o2}, \mathsf{v})$. Essentially, our model supports the aggregation of objects, referring to regions of the same MDO.

From a formal point of view, we will consider the CN CSMO. The type definition of class CSMO is quite general and is of the form

$$\mathsf{CSMO} = [A_1{:}T_1, \ldots, A_n{:}T_n]. \tag{4.8}$$

As for MDO, there are no special restrictions on the number of attributes and the methods defined for the class CSMO. Essentially, these are application dependent. As we did for MDO, we describe rather which attributes and methods have to be there and which may be of interest.

Let (o, v) be a CSMO (*i.e.* $\mathsf{o} \in \pi^*(\mathsf{CSMO})$). The tuple $[A_1{:}T_1, \ldots, A_n{:}T_n]$ has to contain the following attributes:

MediaDataObj:MDO. $\mathsf{o.MediaDataObj} \in \pi^*(\mathsf{MDO})$ is the MDO identifier pointing to the whole real data o refers to.

ComposedOf:{CSMO}. $\mathsf{o.ComposedOf} = \{\mathsf{o}_1, \ldots, \mathsf{o}_n\} \subseteq \pi^*(\mathsf{CSMO})$ is a set of OIDs of CSMOs which are the "structural components" of o. A restriction on this attribute is that each component o_i and o have to refer to the same MDO, *i.e.* $\forall 1 \leq i \leq n$,

$$\mathsf{o.MediaDataObj} = \mathsf{o}_i\mathsf{.MediaDataObj}. \tag{4.9}$$

Region:Region. $\mathsf{o.Region} = \mathsf{o}_\mathsf{r} \in \pi^*(\mathsf{Region})$ is the OID of the object of class Region which identifies the region (portion of bytes) o refers to. That is, the object (o, v) of class

CSMO is related to the MDO identified by $\mathsf{o_m} = \mathsf{o.MediaDataObj}$. Its raw data is $\mathsf{B} = \mathsf{GetData}(\mathsf{o_m.Data})$ and the addressed region $\mathsf{R_o} \subseteq \mathsf{B}$ is the set of bytes

$$\mathsf{R_o} = \{b \in \mathsf{B}\colon m_\mu(\mathsf{o_r}, \mathsf{o_m}, b) = 1\}. \tag{4.10}$$

It is worth noting that not necessarily $\mathsf{R_o}$ has to be the union of the regions referred by o's components. For instance, the CSMO referring to a face is composed of objects referring to its eyes, its nose and its mouth, but the "union" of these regions does not necessarily constitutes the region of face. Nevertheless, we impose that $\mathsf{R_o}$ has to contain at least the union of the regions referred by o's components, *i.e.*

$$\{b \in \mathsf{B}\colon \exists \mathsf{o}_i \in \mathsf{o.ComposedOf} \text{ such that } m_\mu(\mathsf{o}_{\mathsf{r}_i}, \mathsf{o_m}, b) = 1\} \subseteq \mathsf{R_o}, \tag{4.11}$$

where $\mathsf{o}_{\mathsf{r}_i}$ is $\mathsf{o}_i\mathsf{.Region}$. If not specified otherwise, we assume that the *default value* of o.Region is the union of these regions, *i.e.*

$$\mathsf{R_o} = \{b \in \mathsf{B}\colon \exists \mathsf{o}_i \in \mathsf{o.ComposedOf} \text{ such that } m_\mu(\mathsf{o}_{\mathsf{r}_i}, \mathsf{o_{mdo}}, b) = 1\}, \tag{4.12}$$

A final remark is that the region addressed by $\mathsf{R_o}$ may be the whole data $\mathsf{B} = \mathsf{GetData}(\mathsf{o_m.Data})$, according to the fact that by definition B is a region too.

Let us now present some examples explicating the notions introduced right now.

Example 5 Let us consider the already defined object $(\mathsf{o2}, \mathsf{v})$ of class EpsObject. We rely our example on Figure 4.4.

Consider the CN CIO, the class of complex image objects. We impose that $\mathsf{CIO} \prec \mathsf{CSMO}$, *i.e.* a complex image object is a CSMO. CIO is defined as:

$$\mathsf{CIO} \quad = \quad [\mathsf{MediaDataObj}\colon \mathsf{ImageObject}, \mathsf{ComposedOf}\colon \{\mathsf{CIO}\}, \mathsf{Region}\colon \mathsf{Region}].$$

We have five CSMOs whose OIDs are o3, o4, o5, o6 and o7, respectively, all of which are of class CIO, *i.e.* $\mathsf{o3}, \mathsf{o4}, \mathsf{o5}, \mathsf{o6}, \mathsf{o7} \in \pi^*(\mathsf{CIO})$. The first four are "atomic" in the sense that they have no aggregates, *i.e.* $\mathsf{o}i\mathsf{.ComposedOf} = nil$, for $3 \leq i \leq 6$. Each object $(\mathsf{o}i, \mathsf{v}i)$ is of the form

$$(\mathsf{o}i, [\mathsf{MediaDataObj}\text{:}\mathsf{o2}, \mathsf{ComposedOf}\text{:}nil, \mathsf{Region}\text{:}\mathsf{o}_{\mathsf{r}i}]),$$

where $\mathsf{o}_{\mathsf{r}i}$ is the OID of the correspondent region depicted in $\mathsf{GetData(o2.Data)}$ in Figure 4.4. They correspond to the left eye, the mouth, the right eye and the nose of the face, respectively.

The only object which is an aggregation of other CSMOs is the one identified by o7. In fact, for it $\mathsf{o7.ComposedOf} = \{\mathsf{o3}, \mathsf{o4}, \mathsf{o5}, \mathsf{o6}\}$ holds, and thus, $(\mathsf{o7}, \mathsf{v7})$ is of the form

$$(\mathsf{o7}, [\mathsf{MediaDataObj}\text{:}\mathsf{o2}, \mathsf{ComposedOf}\text{:}\{\mathsf{o3}, \mathsf{o4}, \mathsf{o5}, \mathsf{o6}\}, \mathsf{Region}\text{:}\mathsf{o}_{\mathsf{r7}}]),$$

where o_{r7} is the OID of the correspondent white face region. It is worth noting to observe that in fact, relying on Equation (4.10),

$$\bigcup_{3 \leq i \leq 6} \mathsf{R}_{\mathsf{o}i} \subset \mathsf{R}_{\mathsf{o7}},$$

according to Equation (4.11).

■

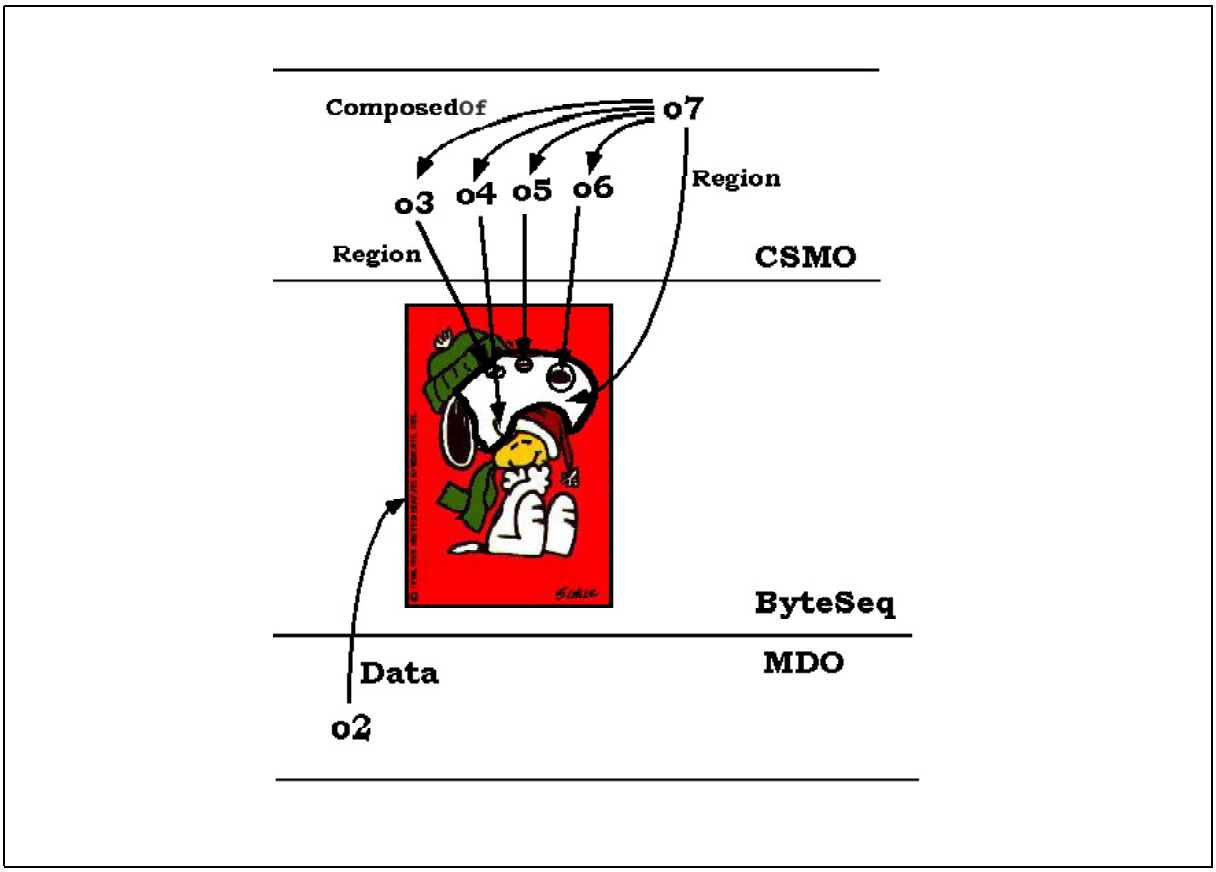


Figure 4.4: Complex single media objects involving image data.

Example 6 The following example concerns the case where the media type of the common MDO is text (and the format is latex).

Consider the CN CTO, the class of complex text objects. We impose that $\mathsf{CTO} \prec \mathsf{CSMO}$, *i.e.* a complex text object is a CSMO. CTO is defined as:

$$\mathsf{CTO} \quad = \quad [\mathsf{MediaDataObj}\colon \mathsf{TextObject}, \mathsf{ComposedOf}\colon \{\mathsf{CTO}\}, \mathsf{Region}\colon \mathsf{TextRegion}].$$

Suppose $(\mathsf{o1}_{\mathsf{txt}}, \mathsf{v1}_{\mathsf{txt}})$ is a MDO of class TextObject such that its raw data $\mathsf{B} = \mathsf{GetData}(\mathsf{o1}_{\mathsf{txt}}.\mathsf{Data})$ is

```
\documentstyle{book}
\title{Complex objects}
   \author{Umberto Straccia}
\begin{document}
\maketitle
\chapter{Media data objects}
    In this chapter we will discuss ...
\section{Background}
    In this section we will address ...
\chapter{Complex single media objects}
```

```
    We have seen that ...
\section{Data model specification}
    We are now redy to formally ...
\end{document}
```

We rely our example on Figure 4.5.

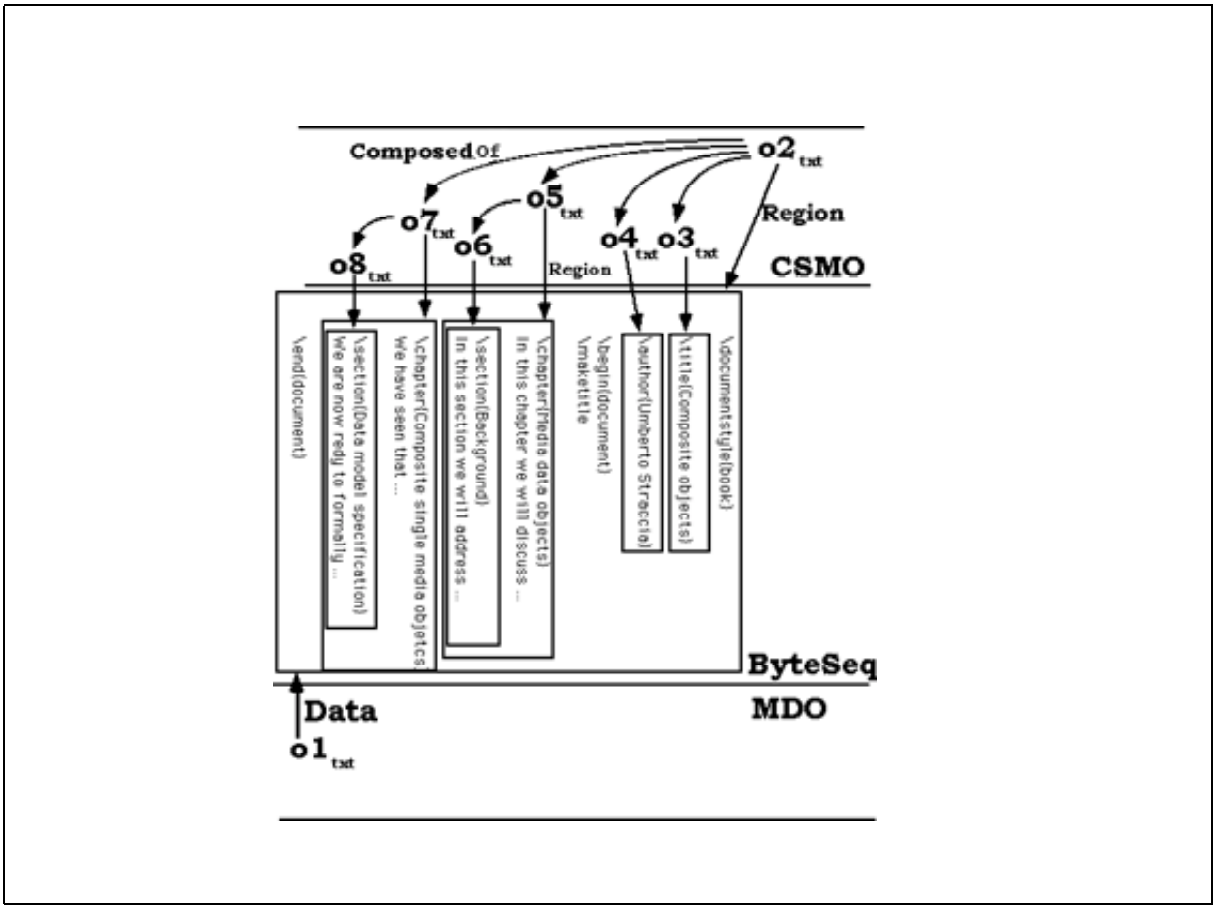


Figure 4.5: Complex single media objects involving text data.

Consider CNs Book, Title, Author, Chapter, and Section, all members of the set class names CN, and the class hierarchy $\mathsf{CH}(\mathsf{CN}, \prec, =)$ such that

$$\begin{array}{lll} \mathsf{Book} \prec \mathsf{CTO}, & \mathsf{BookTitle} \prec \mathsf{CTO}, & \mathsf{BookAuthor} \prec \mathsf{CTO}, \\ \mathsf{Chapter} \prec \mathsf{CTO}, & \mathsf{Section} \prec \mathsf{CTO}. & \end{array}$$

Then, the definition of the objects in Figure 4.5 is immediate. The OID $\mathsf{02}_{\mathsf{txt}}$ is a Book, *i.e.* $\mathsf{02}_{\mathsf{txt}} \in \pi^*(\mathsf{Book})$, composed of four parts, *i.e.* $\mathsf{02}_{\mathsf{txt}}.\mathsf{ComposedOf} = \{\mathsf{03}_{\mathsf{txt}}, \mathsf{04}_{\mathsf{txt}}, \mathsf{05}_{\mathsf{txt}}, \mathsf{07}_{\mathsf{txt}}\}$: a title, $\mathsf{03}_{\mathsf{txt}} \in \pi^*(\mathsf{BookTitle})$, an author, $\mathsf{04}_{\mathsf{txt}} \in \pi^*(\mathsf{BookAuthor})$, and two chapters $\mathsf{05}_{\mathsf{txt}}, \mathsf{07}_{\mathsf{txt}} \in \pi^*(\mathsf{Chapter})$. Each chapter $\mathsf{05}_{\mathsf{txt}}, \mathsf{07}_{\mathsf{txt}}$ is composed of a section $\mathsf{06}_{\mathsf{txt}} \in \pi^*(\mathsf{Section})$ and $\mathsf{08}_{\mathsf{txt}} \in \pi^*(\mathsf{Section})$, respectively. ■

The examples show that it is rather easy to describe arbitrarily complex text and image objects and their structural aggregation.

Finally, the following example shows how video can be represented in our framework. Rather inventing some new model we will show how an existing model fits into our framework. The video model of concern is OVID [215].

Example 7 In OVID [215], a data model for video is presented. Essentially, a video is viewed a sequence of frames (and the contents of the video-frame sequence is described by a tuple of attribute/value pairs). Complex video objects corresponds to a video-frame sequence. An interval is represented by a pair (i_1, i_2), where i_1 is the starting frame number and i_2 is the ending frame number and denotes a continuous sequence of video-frames. In OVID, since a complex video object corresponds to more than one video-frame sequence, a set of intervals is associated with the corresponding complex video objects. This is because a meaningful scene does not always correspond to a single continuous sequence of video frames.

We now show how the video model of OVID can be mapped into our framework. At first, we consider the CN VideoDataObject. VideoDataObject is the class of *video data objects*. Hence, we impose that VideoDataObject $\prec$ MDO and define VideoDataObject as

$$\begin{array}{lcll} \mathsf{VideoDataObject} & = & [\mathsf{Data} & : \mathsf{Location}, \\ & & \mathsf{Size} & : \mathsf{Integer}, \\ & & \mathsf{CreationDate} & : \mathsf{Date}, \\ & & \mathsf{MediaType} & : \mathsf{Video}, \\ & & \mathsf{Format} & : \mathsf{String}, \\ & & \mathsf{Author} & : \mathsf{String}]. \end{array}$$

Regarding intervals, we consider a CN VideoRegion and impose that VideoRegion $\prec$ Region. An interval is a video-frame sequence. Therefore, we define

$$\mathsf{VideoRegion} \quad = \quad [\mathsf{Startframe}\colon \mathsf{Integer}, \mathsf{Endframe}\colon \mathsf{Integer}].$$

Moreover, the membership function of video regions has signature

$$m_\mu\colon \mathsf{VideoRegion} \times \mathsf{VideoObject} \times \mathsf{Byte} \rightarrow \{0, 1\},$$

which given a video region OID $\mathsf{o_{vr}}$ and a video object OID $\mathsf{o_{vo}}$, returns the sequence of bytes corresponding to the video-frame sequence determined by the interval $\mathsf{o_{vr}}$ in the video $\mathsf{o_{vo}}$.

Finally, complex video objects are defined in terms of the class CVO. We impose that CVO $\prec$ CSMO, *i.e.* a complex video object is a CSMO. Finally, CVO is defined as:

$$\mathsf{CVO} \quad = \quad [\mathsf{MediaDataObj}\colon \mathsf{VideoDataObject}, \mathsf{ComposedOf}\colon \{\mathsf{CVO}\}, \mathsf{Region}\colon \{\mathsf{VideoRegion}\}].$$

■

Similarly an audio stream, and of course, a relational and an object-oriented database can be represented.

4.3.3 Media dependent topological operators

In Section 4.3.1 and Section 4.3.1.1 concerned topology on regions, the definition of the class Region and topological operators (like union, intersection). A characteristics of these methods is that they are quite general and are media type independent. Besides these topological operators, additionally any particular media brings with it a set of (application dependent) specific operators. For instance,

- if we assume that there is a CN ImageRegion such that ImageRegion $\prec$ Region and membership function

$$m_\mu\colon \mathsf{ImageRegion} \times \mathsf{ImageObject} \times \mathsf{Byte} \to \{0,1\},$$

then w.r.t. image regions there could be topological operators (methods) like isLeftof, isRightof, isAboveof, isBelowof, ..., with signature

$$\begin{aligned}
&\mathsf{isLeftof}\colon \mathsf{ImageRegion} \times \mathsf{ImageRegion} \times \mathsf{ImageObject} \to [0,1]\\
&\mathsf{isRightof}\colon \mathsf{ImageRegion} \times \mathsf{ImageRegion} \times \mathsf{ImageObject} \to [0,1]\\
&\mathsf{isAboveof}\colon \mathsf{ImageRegion} \times \mathsf{ImageRegion} \times \mathsf{ImageObject} \to [0,1]\\
&\mathsf{isBelowof}\colon \mathsf{ImageRegion} \times \mathsf{ImageRegion} \times \mathsf{ImageObject} \to [0,1]\\
&\ldots
\end{aligned}$$

For instance, $\mathsf{isLeftof}(o_{r_1}, o_{r_2}, o_i) = 1$ means that the region identified by o_{r_1} is *spatially to the left* of the region identified by o_{r_2} in the image identified by o_i, whereas $\mathsf{isLeftof}(o_{r_1}, o_{r_2}, o_i) = .8$ means that the region identified by o_{r_1} is likely to the left of the region identified by o_{r_2}.

- w.r.t. video regions, *i.e.* video-frames sequences, there could be topological operators (methods) like before, after, next, previous, ..., with signature

$$\begin{aligned}
&\mathsf{before}\colon \mathsf{VideoRegion} \times \mathsf{VideoRegion} \times \mathsf{VideoObject} \to [0,1]\\
&\mathsf{after}\colon \mathsf{VideoRegion} \times \mathsf{VideoRegion} \times \mathsf{VideoObject} \to [0,1]\\
&\mathsf{next}\colon \mathsf{VideoRegion} \times \mathsf{VideoObject} \to \mathsf{VideoRegion}\\
&\mathsf{previous}\colon \mathsf{VideoRegion} \times \mathsf{VideoObject} \to \mathsf{VideoRegion}\\
&\ldots
\end{aligned}$$

For instance, $\mathsf{before}(o_{r_1}, o_{r_2}, o_v) = 1$ iff the video-frames sequence identified by o_{r_1} *starts before* of the video-frames sequence identified by o_{r_2} in the video identified by o_i.

The set of topological operators defined for the class Region and its subclasses is essentially application dependent. For some mathematical background about spatial operators and calculi on regions see *e.g.* [82, 230].

Finally, we can assume that all these methods m are defined for the class CSMO and its subclasses too. Most of these can be defined by relying on the Region attribute of class CSMO and its related method. For instance, for class of complex image objects CIO, isLeftof is a method of this class with signature

$$\mathsf{isLeftof}\colon \mathsf{CIO} \times \mathsf{CIO} \to [0,1],$$

such that for complex image objects (o1, v1) and (o2, v2) of class CIO,

$$\mathsf{isLeftof}(\mathsf{o1}, \mathsf{o2}) = \mathsf{isLeftof}(\mathsf{o1.Region}, \mathsf{o2.Region}, \mathsf{o1.MediaDataObject})$$

whenever the condition o1.MediaDataObject = o2.MediaDataObject holds. Just notice that the above condition is necessary, as topological spatial relations between regions of different MDOs are rarely defined. The other topological operators can be defined similarly.

The above example shows us that, in general, methods defined at the MDO level can be inherited at the CSMO level too.

4.3.3.1 **Induced topological space of regions

We briefly show that the set of regions of a MDO (o, v), referred by several CSMOs, induces a topological base on $\mathsf{GetData}(o.\mathsf{Data})$. Consider, $\pi^*(\mathsf{CSMO})$, *i.e.* the set of all OIDs of complex single media objects of class CSMO, and $\pi^*(\mathsf{MDO})$, *i.e.* the set of all OIDs of media data objects of class MDO. For each media data object OID $\mathsf{o_m} \in \pi^*(\mathsf{MDO})$, let $\mathsf{CSMO}(\mathsf{o_m})$ be the subset of $\pi^*(\mathsf{CSMO})$ such that each CSMO in it refers to the MDO $\mathsf{o_m}$, *i.e.*

$$\mathsf{CSMO}(\mathsf{o_m}) = \{\mathsf{o_c} \in \pi^*(\mathsf{CSMO}) : \mathsf{o_c}.\mathsf{MediaDataObj} = \mathsf{o_m}\}. \tag{4.13}$$

Let $\mathsf{Region}(\mathsf{o_m}) \subseteq 2^{\mathsf{GetData}(\mathsf{o_m}.\mathsf{Data})}$ be the set of all regions of $\mathsf{GetData}(\mathsf{o_m}.\mathsf{Data})$ referred by OIDs in $\mathsf{CSMO}(\mathsf{o_m})$, *i.e.* according to Equation (4.10)

$$\mathsf{Region}(\mathsf{o_m}) = \{\mathsf{R_{o_c}} \subseteq \mathsf{GetData}(\mathsf{o_m}.\mathsf{Data}) : \mathsf{o_c} \in \mathsf{CSMO}(\mathsf{o_m})\}. \tag{4.14}$$

Example 8 Consider Example 5 and Example 6. Suppose that our data base contains the two MDOs

$$\pi^*(\mathsf{MDO}) = \{\mathsf{o3}, \mathsf{02_{txt}}\}$$

and that

$$\pi^*(\mathsf{CSMO}) = \{\mathsf{o3}, \mathsf{o4}, \mathsf{o5}, \mathsf{o6}, \mathsf{o7}, \mathsf{02_{txt}}, \mathsf{03_{txt}}, \mathsf{04_{txt}}, \mathsf{05_{txt}}, \mathsf{06_{txt}}, \mathsf{07_{txt}}\}.$$

By definition of Equation (4.13), $\mathsf{CSMO}(\mathsf{o2})$ is

$$\mathsf{CSMO}(\mathsf{o2}) = \{\mathsf{o3}, \mathsf{o4}, \mathsf{o5}, \mathsf{o6}, \mathsf{o7}\},$$

and thus, from Equation (4.14), $\mathsf{Region}(\mathsf{o2})$ is the set of regions $\mathsf{R_{o}}i$, depicted in $\mathsf{GetData}(\mathsf{o2}.\mathsf{Data})$ in Figure 4.4, and identified by OIDs o_{ri}, where $3 \leq i \leq 7$:

$$\mathsf{Region}(\mathsf{o2}) = \{\mathsf{R_{o3}}, \mathsf{R_{o4}}, \mathsf{R_{o5}}, \mathsf{R_{o6}}, \mathsf{R_{o7}}\}.$$

■

Let $B_{\mathsf{o_m}} = \mathsf{GetData}(\mathsf{o_m}.\mathsf{Data})$. Let $\mathcal{R}_{\mathsf{o_m}} \subseteq 2^{B_{\mathsf{o_m}}}$ be the set of regions w.r.t. $B_{\mathsf{o_m}}$ defined as

$$\mathcal{R}_{\mathsf{o_m}} = \{B_{\mathsf{o_m}}, \emptyset\} \cup \wp(\mathsf{Region}(\mathsf{o_m})),$$

where $\wp(\mathsf{Region}(\mathsf{o_m}))$ is the partition of

$$\bigcup_{R_i \in \mathsf{Region}(\mathsf{o_m})} R_i$$

into atomic subsets (see Figure 4.3). It can be verified that $\mathcal{R}_{\mathsf{o_m}}$ is a topological base on $B_{\mathsf{o_m}}$, and the generated topology $\tau_{\mathcal{R}_{\mathsf{o_m}}}$ on $B_{\mathsf{o_m}}$ is obtained from Equation (4.6):

$$\tau_{\mathcal{R}_{\mathsf{o_m}}} = \{R : R = R_1 \cup R_2, R_1, R_2 \in \tau_{\mathcal{R}_{\mathsf{o_m}}}\}. \tag{4.15}$$

4.3.4 Feature attributes

In this section we will address the description of content at the form level, *i.e.* physical features of CSMOs. In particular, we will show how content of the form level is represented. The description of content at the semantics level will be the topic of Chapter 10.

Each CSMO may contain several feature attributes: each of them may be measured starting from the region the CSMO refers to. Of course, features are media dependent and, thus, for each media type Text, Image, Video and Audio (see Section 4.2) there is a specific set of feature types. Therefore, among the attributes $A_1, \ldots, A_n$ of the class of CSMOs,

$$\mathsf{CSMO} = [A_1{:}T_1, \ldots, A_n{:}T_n],$$

there may be some feature attributes A_j of type T_j, where T_j is a feature type of some specific media.

A feature is mainly characterised by a *feature extraction function* and by a *feature similarity function*. The feature extraction function extracts and materialises useful intrinsic properties of a CSMO. Color distributions, shapes, textures (media type: image), mosaiced video-frame sequences (media type:video) and text index term weights (media type:text) are examples of features that are extracted from regions. Each extracted *feature value* has its own type. The feature similarity function measures the similarity between two feature values of the same type.

Formally, we will consider CN Feature, the class of all features. Moreover, we will assume that there is a feature CN for each media type: TextFeature, ImageFeature, VideoFeature and AudioFeature. These are hierarchically ordered according to:

$$\begin{array}{ll} \mathsf{TextFeature} \prec \mathsf{Feature}, & \mathsf{ImageFeature} \prec \mathsf{Feature}, \\ \mathsf{VideoFeature} \prec \mathsf{Feature}, & \mathsf{AudioFeature} \prec \mathsf{Feature}. \end{array}$$

Finally, for each of the above media dependent feature classes, we assume that there are feature CNs for each specific feature between the same media type. For instance, w.r.t. images, ColorDistr, Shape, Texture, ..., are CNs which are subclasses of the class ImageFeature.

Each feature class is of type

$$F = [A_1{:}T_1, \ldots, A_n{:}T_n], \tag{4.16}$$

such that

1. there is a method m_F^{ext}, the *feature extraction function*, with signature

$$m_F^{ext}{:}\,\mathsf{MDO} \times \mathsf{Region} \rightarrow F; \tag{4.17}$$

2. for any object (o, v) of class CSMO, v is the feature value extracted from region o.Region of the MDO o.MediaDataObj, *i.e.*

$$m_F^{ext}(\mathsf{o.MediaDataObj}, \mathsf{o.Region}) = \mathsf{v};\ \text{and}$$

3. there is a method m_F^{sim}, the *feature similarity function*, with signature

$$m_F^{sim}{:}\,F \times F \rightarrow [0,1]: \tag{4.18}$$

given two objects (o_1, v_1) and (o_2, v_2) of class F, then $m_F^{sim}(o_1, o_2)$ measures the *similarity* (according to some similarity measure) between feature values v_1 and v_2.

Example 9 Following the schema above, a definition of the class of complex image objects CIO may be:

$$\begin{array}{lcll} \mathsf{CIO} & = & [\ \mathsf{MediaDataObj} & : \ \mathsf{ImageObject}, \\ & & \mathsf{ComposedOf} & : \ \{\mathsf{CIO}\}, \\ & & \mathsf{Region} & : \ \mathsf{RectangleRegion}, \\ & & \mathsf{HasColor} & : \ \mathsf{ColorDistr}, \\ & & \mathsf{HasShape} & : \ \mathsf{Shape}, \\ & & \mathsf{HasTexture} & : \ \mathsf{Texture}\ ], \end{array}$$

whereas a definition of the class of complex text objects CTO may be:

$$\begin{array}{lcll} \mathsf{CTO} & = & [\ \mathsf{MediaDataObj} & : \ \mathsf{TextObject}, \\ & & \mathsf{ComposedOf} & : \ \{\mathsf{CTO}\}, \\ & & \mathsf{Region} & : \ \mathsf{TextRegion}, \\ & & \mathsf{TermFrequency} & : \ \mathsf{TermVector}\ ], \end{array}$$

where TermVector is a binary vector of weighted terms[2], *i.e.*

$$\mathsf{TermVector} \ = \ [\mathsf{TermWeight}\colon \mathsf{Array}[1, \mathsf{n}] \ \mathsf{of} \ [0, 1]].$$

■

Notice that we assume that there is a similarity function (method) between two features of the same type. Of course, it is highly desirable that similarity function (or several similarity functions) comparing CSMOs for similarity are given. Multimedia systems provide such functions which are used in order to answer the user's queries. For instance, in an image retrieval system, the user may provide an image and asks for a ranked list of all images known to the system to be similar to the given one; in traditional text information retrieval systems, the user's query is a text, and the system looks for documents relevant to the user's query (here "relevant" is used in place of "similar"). Similarly, in case of video retrieval systems and speech retrieval systems.

Given two CSMOs (o_1, v_1) and (o_2, v_2), instances of the same class C, where C is a tuple type containing feature attributes/types $A_1{:}F_1, \ldots, A_n{:}F_n$, then a similarity function is usually a method of signature

$$m^{sim}\colon C \times C \rightarrow [0, 1], \tag{4.19}$$

where $m^{sim}(o_1, o_2)$ strongly depends on the values $o_1.A_1, \ldots, o_1.A_n, o_2.A_1, \ldots, o_2.A_n$: more precisely, on the values $m_{F_1}^{sim}(o_1.A_1, o_2.A_1), \ldots, m_{F_n}^{sim}(o_1.A_n, o_2.A_n)$. We call the similarity between CSMOs *form similarity* as it refers to the form dimension of multimedia data and the *form similarity functions*. As we will see in Chapter 10, there may be similarity functions pertaining to the semantics dimension, which will be called *semantics similarity functions*. Essentially, they specify the similarity between concepts, called *content similarity*.

[2] VALUE([0, 1]) is the set of reals in range $[0, 1]$.

4.4 Complex multimedia objects

So far, we allowed the specification of aggregation of objects which refers to the *same* MDO. A natural extension of the aggregation principle is to extend it to the case in which an object can be the aggregation of objects which refer to regions of different MDOs. Objects of this type are called *Complex Multimedia Objects* (CMOs) (denoted by *cmo*). From a formal point of view, we consider the CN CMO and that CSMO $\prec$ CMO, *i.e.* each complex single media object is a complex multimedia object. The type definition of class CMO is quite general and is of the form

$$\mathsf{CMO} = [A_1{:}T_1, \ldots, A_n{:}T_n]. \tag{4.20}$$

As for CSMO, there are no special restrictions on the number of attributes and the methods defined for the class CMO. Essentially, these are application dependent. But, one attribute is mandatory. Let (o, v) be a CMO (*i.e.* $\mathsf{o} \in \pi^*(\mathsf{CMO})$). The tuple $[A_1{:}T_1, \ldots, A_n{:}T_n]$ has to contain the following attribute for aggregation:

ComposedOf:{CMO}. o.ComposedOf $= \{\mathsf{o}_1, \ldots, \mathsf{o}_n\} \subseteq \pi^*(\mathsf{CMO})$ is a set of OIDs of CMOs which are the "structural components" of o.

This property may be useful in the case of HTML documents [279]. HTML documents are structured text documents in which text can be combined with references to images, video, audio and links to other HTML documents. It might be useful to create an unique object collecting these three pieces of data together.

Example 10 The following text is a simple HTML document.

```
<HTML><HEAD>
<TITLE>A Simple HTML Demo Page </TITLE>
</HEAD>
<BODY BGCOLOR="#FFFFFF" TEXT="#000000">
   Hi, that's about Snoopy.
<P>
<IMG SRC="snoopy.jpg" WIDTH="100" HEIGHT="150" ALIGN="BOTTOM" BORDER="0">
<P>
    In the above Image, Snoopy and Woodstock are sweetly embraced.
    Snoopy likes all sort of activities like Golf, Baseball, daydreaming....
    If you are interested in then do not hesitate to go
    to the official Snoopy home page.
<A HREF="http://www.snoopy.com"> Go there. </A>
</BODY></HTML>
```

whose aim is to mix a text with a reference to an image by means of the HTML command `<A HREF="http://www.snoopy.com"> Go there. </A>`.

The output of the above document through a HTML browser can be seen in Figure 4.6. Let us assume that the text region

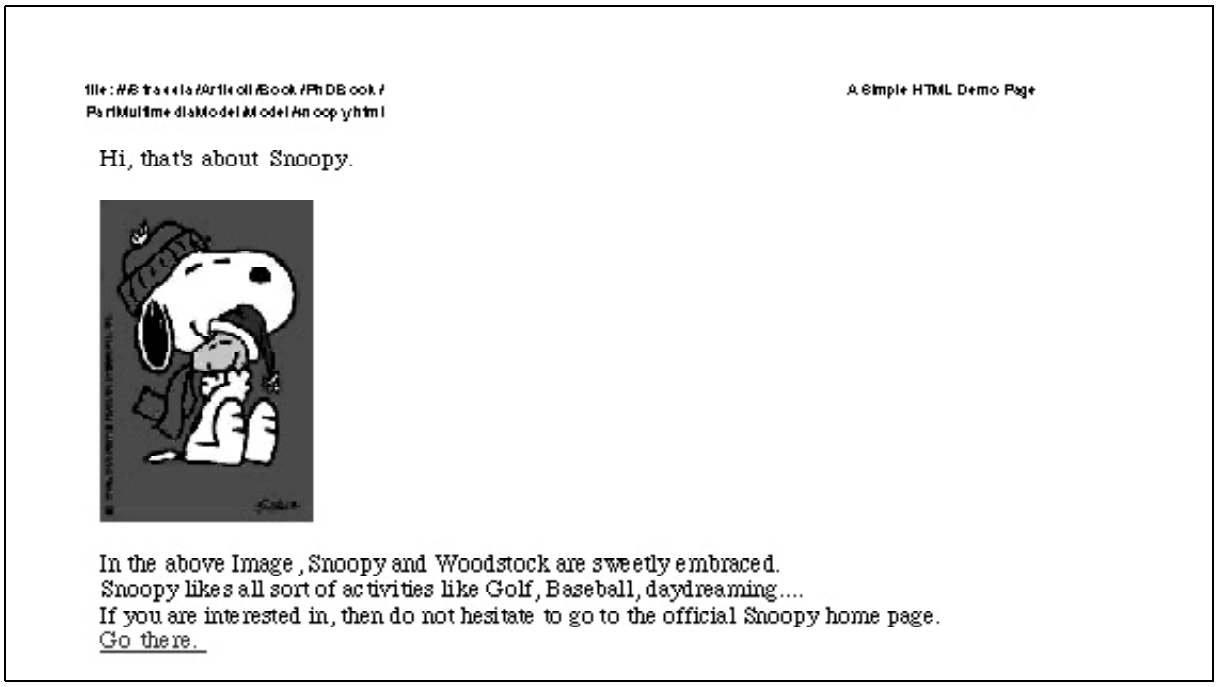


Figure 4.6: HTML file viewed from a browser.

```
In the above Image, Snoopy and Woodstock are sweetly embraced.
Snoopy likes all sort of activities like Golf, Baseball, daydreaming....
If you are interested in then do not hesitate to go
to the official Snoopy home page.
```

is referred by the CSMO $(\mathsf{o1}, \mathsf{v1})$, whereas the CSMO $(\mathsf{o2}, \mathsf{v2})$ refers to the whole image `snoopy.jpg`[3]. Notice that

$$\begin{aligned} \mathsf{o1.MediaDataObj.MediaType} &= \mathsf{Text}, \text{ and} \\ \mathsf{o2.MediaDataObj.MediaType} &= \mathsf{Image}. \end{aligned}$$

Then

$$(\mathsf{o}, [\mathsf{Composedby}.\{\mathsf{o1}, \mathsf{o2}\}])$$

is a CMO.

■

Finally, w.r.t. features, essentially features attributes of an CMO with OID o, composed of objects with OID $o_1, \ldots, o_n$, are given by the feature attributes of $o_1, \ldots, o_n$. As for CSMO, there may be a similarity function testing two CMO for similarity: *i.e.* for each subclass C of CMO there may be methods of signature

$$m^{sim}\colon C \times C \to [0, 1]. \tag{4.21}$$

[3] o2.Region refers to the whole image.

It is worth noticing that the concept of CMOs may profitably be used in indexing (at the form level) CSMOs. In fact, suppose we have an image i about a person. The features usually extracted from i concern color distribution, texture and shape detection. As it is quite difficult to retrieve that picture by relying only on image features, often image retrieval systems provide annotation methods, *i.e.* an image is annotated with a text describing the content of the picture, and thus, image retrieval is performed via text retrieval (see *e.g.* [3, 137, 193, 255, 257]).

It is quite easy to see that the concept of annotated images may be implemented in our framework in a straightforward way. In fact, we accomplish this through the aggregation property of CMO. Given the image i, we (*i*) create an image object (a CIO) (o1, v1) which refers to the region (or whole image) of the person in i, (*ii*) create a text object (a CTO) (o2, v2) which refers to the text describing the image's content. Thereafter, we create a complex multimedia object with OID o with aggregates o1 and o2, *i.e.* we create the CMO

$$(\mathsf{o}, [\mathsf{ComposedOf}{:}\{\mathsf{o1}, \mathsf{o2}\}]).$$

Now, retrieval may be performed through the similarity functions for images, text or the combination of both. Of course, the above method may be generalised to any kind of CMO, *i.e.* it may be used in order to index a CMO through annotations of any media type.

4.5 Multimedia databases about form

Let us recall the main constituents of our simple object-oriented data model for multimedia data. The main classes (concepts) we introduced are the following.

Media Data Object: An object of class MDO represents the *raw multimedia data*, like a text, an image, a video and an audio, which are simply viewed as a sequence of bytes. As shown in Figure 4.1, there may be a media data object type taxonomy.

Region: An object of class Region represents uniquely a region in a piece of data, *i.e.* a portion of data in a MDO. There may be a type taxonomy on regions, representing different kinds of regions, and topological operators, *i.e.* methods like union, intersection, ...

Complex Single Media Object: An object of class CSMO is a structured object, *i.e.* an aggregate of CSMOs, with the characteristics that they rely on the *same* MDO (see *e.g.* Figure 4.4). A CSMO (o, v) represents a region in a MDO which includes the union of the regions referred by the components of (o, v). The aggregation determines the structure of the object. There may be a type taxonomy on CSMOs, like Chapter $\prec$ CTO and CTO $\prec$ CSMO, and topological operators, *i.e.* methods like isLeftof, isRightof, ..., (case image objects). Usually, methods for determining similarity between CSMOs are provided.

Feature attributes: The form level addresses the description of content in terms of the usual *physical features*, like color distribution, shape and texture (case image objects), which are *media dependent*. The class of features, Feature, is characterised by a feature extraction function and by a feature similarity function.

Complex Multimedia Object: An object of class CMO is a structured object, *i.e.* an aggregation of objects which may refer to regions of *different* MDOs (see *e.g.* Figure 4.6). The concept of CMOs is nothing else than a generalisation of the concept of CSMOs to the case of multiple media. There may be a type taxonomy on CMOs and methods for determining similarity between CMOs.

Given the classes above, a *multimedia database about form* (denoted by DBF) is an object-oriented database (see Equation (3.4)

$$\mathsf{DBF} \quad = \quad (\mathsf{CH}, \mathsf{TY}, \mathsf{MET}, \pi, \mathsf{OID}) \tag{4.22}$$

where $\mathsf{CH} = (\mathsf{CN}, =, \prec)$ is a class hierarchy, CN is a set of class names which includes those specified above, TY is the set of types over CN, MET is a set of methods m of classes in CN, π is an OID assignment over CN and OID is a set of objects (o, v) instances of classes in CN.

4.6 Summary

We have defined the object-oriented multimedia data model we will rely on. Essentially, we identified three types of data objects through which we will be able to represent all the relevant information of multimedia data pertaining to the form dimension. Media data objects represents the raw multimedia data, like a text, an image, a video or an audio. Complex single media objects allow us to aggregate together parts of a media data object, forming more complex objects. Finally, complex multimedia objects allow us to aggregate parts of different media data objects. A collection of media data objects, complex single media objects and complex multimedia objects forms a multimedia database about form. To complex single media objects and to complex multimedia objects a semantics may be associated through a logic which will be defined in the next part of this thesis.

Part II

A logic for representing the semantics of multimedia objects

Chapter 5

Preview

In this part, the heart of our work, we present a logic for representing the semantics of complex multimedia objects defined in the previous part. The logic will be used as query language in Part III, too. The presentation of the logic proceeds step by step until our final logic is defined. The logic, called fuzzy HORN-$\mathcal{ALC}$, is characterised through:

- a Description Logic (DL) component which allows the representation of structured objects of the real world. In particular, through it we will model the real domain of interest, *i.e.* it will define the ontology;
- a horn rule component which allows us to reason about these structured objects;
- a non-classical, four-valued, semantics [48, 182, 217, 222] which allows us to deal both with (*i*) possible inconsistencies arising from the representation of document semantics; and (*ii*) the caption of a simple form of relevance entailment in query answering [8];
- a fuzzy component which allows the treatment of the inherent uncertainty in multimedia document representation and retrieval.

In the next chapter, a brief overview on DLs will be given. In Chapter 7 a four-valued semantics will be defined for $\mathcal{ALC}$, a significant representative for DLs. For readability purposes, the semantics will be given at propositional level first and thereafter extended to the first-order case. In Chapter 8, a fuzzy extension of the logic developed in Chapter 7 will be specified, yielding our final logic fuzzy HORN-$\mathcal{ALC}$.

In Appendix C and Appendix D all decision algorithms of the various logics are presented.

Chapter 6

An overview on description logics

6.1 Introduction

In the last decade a substantial amount of work has been carried out in the context of *Description Logics* (DLs, for short)[1]. DLs have their origin in semantic networks (*e.g.* [261]) and frame-based-systems (*e.g.* [234]). In particular, they are a logical reconstruction of the so-called frame-based knowledge representation languages, with the aim of providing a simple well-established Tarski-style declarative semantics to capture the meaning of the most popular features of structured representation of knowledge [15, 59, 61, 58, 64, 101, 207, 286]. *Concepts, roles* and *individuals* are the basic building blocks of these logics.

- *Concepts* are expressions which collects the properties, described by means of roles, of a set of individuals. From a logical point of view, concepts can be seen as unary predicates, interpreted as sets of objects over a domain, whereas roles are interpreted as binary predicates. Examples of concepts are Team and Person;
- *Roles* are considered as binary predicates which are interpreted as binary relations between the objects over a domain. An example of role is Member which may represent a relation between a team and the persons belonging to the team.
- *Individuals* are interpreted as objects in the domain. For instance, a particular member of a team would be represented by an individual.

In order to build a knowledge base one starts with the definition and construction of the taxonomy of concepts, by means of *specialisations* (denoted by $C \Rightarrow D$, where C, D are concepts) using the language connectives (such as intersection, union, role quantification). An example of specialisation is

$$\mathsf{DreamTeam} \Rightarrow \mathsf{Team} \sqcap \forall \mathsf{Member}.\mathsf{SuperStar}$$

which specifies that a DreamTeam is a Team such that each Member is a SuperStar.

Also information about individuals can be told through *assertions*. An assertion states either that an individual a is an instance of a concept C (denoted by $C(a)$) or that two individuals a and b are related by means of a role R (denoted by $R(a, b)$). Examples of

[1]Description Logics have also been referred to as Terminological Logics, Concept Logics, KL-ONE-like languages. The web page of the description logic community is found at address `http://dl.kr.org/dl`.

assertions are DreamTeam(chicago) and Member(chicago,jordan). A DL *knowledge base* is a set of specialisations and assertions.

Some of the reasoning tasks that a description logic systems supports, are

subsumption checking: typically concepts defined in a knowledge base are structured into a hierarchy, induced by the subsumption relation and interpreted as set containment: a concept C subsumes a concept D, iff the set of individuals denoted by the concept D is a subset of the set of individuals denoted by C, *i.e.* in symbols $D \preceq C$. It is a common opinion that subsumption checking is an important reasoning task in DLs. This has motivated a large body of research on the problem of subsumption checking in different DLs [54, 62, 70, 95, 150, 152, 249, 253].

classification : classification is the process to build the is-a relationship, determined by the subsumption relation, between the concepts defined in the knowledge base;

instance checking: a basic inference task with knowledge bases. It amounts to verify whether the individual a is an instance of the concept C with respect to the knowledge base Σ, *i.e.* in symbols $\Sigma \models C(a)$.

The first DL system was KL-ONE [64]. Nowadays, a whole family of knowledge representation systems has been build using DLs and for the most of them complexity results for the subsumption and instance checking algorithm are known. The systems differ with respect to expressiveness of the DL and the complexity and completeness of the algorithms. Some of the most used systems nowadays are BACK [227], CLASSIC [59, 58], KRIS [20], LOOM [196], FACT [151] and CRACK [123].

DL systems has been used for building a variety of applications including (see [91]) systems supporting software management [90], browsing and querying of networked information sources [112], knowledge mining [9], data archeology [65], planning [284], learning [175], natural language understanding [51], clinical information system [134], digital libraries [285], software configuration management system [300], web source integration [165] and information retrieval [197].

Experience in using DLs in applications has also shown that in many cases we would like to extend the representational and reasoning capabilities of the DL with other types of reasoning. Therefore, work has begun on extending DLs (see *e.g.* [11, 31, 96, 149, 157, 176, 185, 202, 250, 251, 264] and Appendix A). This lead to considering DLs as to be attractive logics in knowledge based applications as they are a good compromise between expressive power [23, 56, 184] and computational complexity [60, 62, 68, 95, 253, 277].

6.2 A quick look to $\mathcal{ALC}$

The specific DL we employ in our model is the logic $\mathcal{ALC}$, a significant representative of the best-known and most important family of DLs, the $\mathcal{AL}$ family. The reason why we opt for $\mathcal{ALC}$ is that $\mathcal{ALC}$ is universally considered the "standard" DL (as much as **K** is considered the "standard" modal logic) and is therefore regarded as the most convenient testbed for carrying out logical extensions and, in general, logical work of an experimental nature. Reverting to one's DL of choice may be taken as the very last (and usually straightforward) step in the development of a logical DL-based model.

We assume two alphabets of symbols, called *primitive concepts* and *primitive roles*. The letter A will denote a primitive concept and the letter R will denote a primitive role. The *concepts* (denoted by C and D) of the language $\mathcal{ALC}$ are formed out of primitive concepts according to the following syntax rules:

$$\begin{array}{rcrl} C, D & \longrightarrow & A \mid & \text{(primitive concept)} \\ & & C \sqcap D \mid & \text{(concept conjunction)} \\ & & C \sqcup D \mid & \text{(concept disjunction)} \\ & & \neg C \mid & \text{(concept negation)} \\ & & \forall R.C \mid & \text{(universal quantification)} \\ & & \exists R.C & \text{(existential quantification)} \end{array} \tag{6.1}$$

We will use $\top$ (top concept) as a macro for $A \sqcup \neg A$ and $\bot$ (bottom concept) as a macro for $A \sqcap \neg A$.

Description logics have a clean, model-theoretic semantics, based on the notion of *interpretation*. An interpretation $\mathcal{I}$ consists of a non empty set $\Delta^{\mathcal{I}}$ (called the *domain*) of *domain objects* (denoted by the letter d) and of an *interpretation function* $\cdot^{\mathcal{I}}$ mapping different individuals into different elements of $\Delta^{\mathcal{I}}$, primitive concepts into functions from $\Delta^{\mathcal{I}}$ to the set of truth values $\{t, f\}$ and primitive roles into functions from $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ to $\{t, f\}$. The assumption that different individuals denote different objects of the domain is called the *unique name assumption*. In compliance with the style of model-theoretic semantics, the interpretation of complex concepts and roles is obtained by appropriately combining the interpretations of their components. The semantics of $\mathcal{ALC}$ concepts is the following:

$$\begin{array}{rcl} (C \sqcap D)^{\mathcal{I}}(d) = t & \text{iff} & C^{\mathcal{I}}(d) = t \text{ and } D^{\mathcal{I}}(d) = t \\ (C \sqcup D)^{\mathcal{I}}(d) = t & \text{iff} & C^{\mathcal{I}}(d) = t \text{ or } D^{\mathcal{I}}(d) = t \\ (\neg C)^{\mathcal{I}}(d) = t & \text{iff} & C^{\mathcal{I}}(d) = f \\ (\forall R.C)^{\mathcal{I}}(d) = t & \text{iff} & \text{for all } d' \in \Delta^{\mathcal{I}}, \text{ if } R^{\mathcal{I}}(d, d') = t \text{ then } C^{\mathcal{I}}(d') = t \\ (\exists R.C)^{\mathcal{I}}(d) = t & \text{iff} & \text{for some } d' \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(d, d') = t \text{ and } C^{\mathcal{I}}(d') = t \end{array} \tag{6.2}$$

Notice that $\top^{\mathcal{I}}(d) = t$ for all d and $\bot^{\mathcal{I}}(d) = t$ for no d.

It can thus be verified that the interpretation of the concept $\mathsf{Thesis} \sqcap \exists \mathsf{Author.Italian}$ is such that: $(\mathsf{Thesis} \sqcap \exists \mathsf{Author.Italian})^{\mathcal{I}}(d) = t$ iff $\mathsf{Thesis}^{\mathcal{I}}(d) = t$ and for some $d' \in \Delta^{\mathcal{I}}$, $\mathsf{Author}^{\mathcal{I}}(d, d') = t$ and $\mathsf{Italian}^{\mathcal{I}}(d') = t$, describing the set of Thesises for which there is an Italian Author.

Note that each concept C and role R can be mapped into an equivalent open first-order formula $F_C(x)$ and $F_R(x, y)$, respectively:

$$F_A(x) = A(x) \tag{6.3}$$
$$F_R(x, y) = R(x, y) \tag{6.4}$$
$$F_{C \sqcap D}(x) = F_C(x) \wedge F_D(x) \tag{6.5}$$
$$F_{C \sqcup D}(x) = F_C(x) \vee F_D(x) \tag{6.6}$$
$$F_{\neg C}(x) = \neg F_C(x) \tag{6.7}$$
$$F_{\forall R.C}(x) = \forall y.F_R(x, y) \rightarrow F_C(y) \tag{6.8}$$
$$F_{\exists R.C}(x) = \exists y.F_R(x, y) \wedge F_C(y) \tag{6.9}$$

Two concepts C and D are said to be *equivalent* (denoted by $C \equiv_2 D$) when $t = C^{\mathcal{I}}(d)$ iff $t = D^{\mathcal{I}}(d)$ for all $d \in \Delta^{\mathcal{I}}$ and for all interpretations $\mathcal{I}$.

This definition allows us to point to some *duality* in our set of connectives. We may notice, for instance, that $\top$ and $\bot$ are dual, *i.e.* $\top \equiv_2 \neg\, \bot$; similarly, $\sqcap$ is the dual of $\sqcup$, as $(C \sqcap D) \equiv_2 \neg(\neg C \sqcup \neg D)$, and $\forall$ is the dual of $\exists$, as $(\forall R.C) \equiv_2 (\neg \exists R.\neg C)$ (use Equations (6.3)–(6.9) above).

Sometimes we will consider complex roles according the following
(i) syntax

$$\begin{array}{rcrl} R, Q & \longrightarrow & P \mid & \text{(primitive role)} \\ & & R \mid C \mid & \text{(role restriction)} \\ & & R \circ Q & \text{(role composition)} \end{array} \tag{6.10}$$

(ii) semantics

$$\begin{array}{lll} (R|C)^{\mathcal{I}}(d, d') = t & \text{iff} & R^{\mathcal{I}}(d, d') = t \text{ and } C^{\mathcal{I}}(d') = t \\ (R \circ Q)^{\mathcal{I}}(d, d') = t & \text{iff} & \text{for some } d'' \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(d, d'') = t \text{ and } Q^{\mathcal{I}}(d'', d') = t. \end{array} \tag{6.11}$$

and (iii) first-order transformation

$$R|C \;\mapsto\; R(x, y) \wedge C(y) \tag{6.12}$$

$$R \circ Q \;\mapsto\; \exists z.R(x, z) \wedge Q(z, y). \tag{6.13}$$

Role composition and role restrictions are reduced to basic $\mathcal{ALC}$ using the following equivalences:

$$\exists(R \circ Q).C \;\equiv_2\; \exists R.\exists Q.C \tag{6.14}$$

$$\forall(R \circ Q).C \;\equiv_2\; \forall R.\forall Q.C \tag{6.15}$$

$$\exists(R|C).D \;\equiv_2\; \exists R.(C \sqcap D) \tag{6.16}$$

$$\forall(R|C).D \;\equiv_2\; \forall R.(\neg C \sqcup D) \tag{6.17}$$

Notice that more expressive DLs may introduce other concept-forming connectives as well as role-forming connectives. See Appendix B.

A concept C is in *Negation Normal Form* (NNF) iff every negated concept occurring in C has a primitive concept as its argument. It is easily verified that each concept C can be transformed in linear space into an equivalent concept C'. Hence, without loss of generality we will restrict our attention to those concepts already in NNF. Just note that roles all already in NNF.

Let $\mathcal{O}$ be a new alphabet of symbols called *individuals*, denoted by a and b. An *assertion* is an expression of type $C(a)$ (meaning that a is an instance of C), or an expression of type $R(a, b)$ (meaning that a is related to b by means of R). For instance, the assertion Thesis(d) states that d is a Thesis, while Author(d,umberto) states that umberto is an author of d. An assertion made out by a primitive symbol is called a *primitive assertion*. An assertion made out by a negated primitive symbol is called a *negated primitive assertion*. Note that $R(a, b)$ is always a primitive assertion. Primitive assertions or negated primitive assertions are called *atomic assertions*.

The semantics of assertions is specified by saying that the assertion $C(a)$ (resp. $R(a,b)$) is *satisfied* by $\mathcal{I}$ iff $C^{\mathcal{I}}(a^{\mathcal{I}}) = t$ (resp. $R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) = t$).

Specialisations allow instead to state the existence of a specialisation ("more specific than") relation between concepts or between roles. For instance, the specialisation

$$\mathsf{Father} \Rightarrow \mathsf{Man} \sqcap \exists \mathsf{HasChild.Child} \tag{6.18}$$

states that a father is a man having at least a child. The semantics of specialisations is specified by saying that the specialisation $C \Rightarrow D$ (resp. $R_1 \Rightarrow R_2$) is *satisfied* by $\mathcal{I}$ iff $C^{\mathcal{I}}(d) = t$ implies that $D^{\mathcal{I}}(d) = t$ for all $d \in \Delta^{\mathcal{I}}$ (resp. ${R_1}^{\mathcal{I}}(d, d') = t$ implies that ${R_2}^{\mathcal{I}}(d, d') = t$ for all $d, d' \in \Delta^{\mathcal{I}}$). We will use metavariables A for assertions and specialisations.

A set Σ of assertions and specialisations will be called a *Knowledge Base* (KB). With Σ_F we will denote the set of assertions in Σ, whereas with Σ_T we will denote the set of specialisations in Σ.

Often, DLs based systems support only a special form of specialisations. The form of specialisations they allow is defined as follows. A *concept definition* is an expression of the form

$$A := C, \tag{6.19}$$

where A is a primitive concept, called *concept name*, and C is an $\mathcal{ALC}$ concept. The goal of a concept definition $A := C$ is to define concept A as being equivalent to concept C, *i.e.* A is the *name* of concept C. Essentially, a concept definition $A := C$ is a macro which can be expressed in terms of generic concept specialisations: each occurrence of $A := C$ will be replaced by considering both specialisations $A \Rightarrow C$ and $C \Rightarrow A$. An example of concept definition is the following

$$\begin{aligned} \mathsf{PirateMovie} := \ & \mathsf{Movie} \sqcap \\ & (\forall \mathsf{HasMainCharacter.}(\mathsf{Pirate} \sqcap \mathsf{Captain})) \sqcap \\ & (\forall \mathsf{HasMainLocation.SailingShip}). \end{aligned} \tag{6.20}$$

Consider a KB Σ such that $\Sigma_T \neq \emptyset$. Suppose Σ_T contains only concept definitions $A := C$ and specialisations of the form $A \Rightarrow C$, where A is a primitive concept. We will say that A *directly uses* primitive concept B, if either (i) there is a concept definition $A := C \in \Sigma_T$ such that B occurs in C; or (ii) there is a specialisation $A \Rightarrow C \in \Sigma_T$ such that B occurs in C. Let *uses* be the transitive closure of the relation directly uses in Σ. Σ_T is *cyclic* iff there is A such that A uses A through Σ_T.

Finally, we will say that an $\mathcal{ALC}$ KB Σ is *well formed* iff if $\Sigma_T \neq \emptyset$ then

1. Σ_T contains only concept definitions $A := C$ and specialisations of the form $A \Rightarrow C$, where A is a primitive concept;

2. no A appears more than once on the left hand side of concept definitions or specialisations in Σ_T; and

3. Σ_T is not cyclic.

An interpretation $\mathcal{I}$ *satisfies* (*is a model of*) a KB Σ iff $\mathcal{I}$ satisfies each element in Σ. A KB Σ *entails* an assertion $C(a)$ (denoted by $\Sigma \models_2 C(a)$) iff every model of Σ also satisfies $C(a)$. In

this case, we will also say that $C(a)$ is a *logical consequence* of Σ. It is worth noticing that an assertion $R(a, b)$ is satisfied by all models of Σ iff $R(a, b) \in \Sigma$. A KB Σ *entails* a specialisation $C \Rightarrow D$ (denoted by $\Sigma \models_2 C \Rightarrow D$) iff every model of Σ also satisfies $C \Rightarrow D$. In this case, we say that C *is subsumed by* D in Σ, and we will write $C \preceq_2^{\Sigma} D$. For example, the concept Father is subsumed by the concept Man in any KB containing specialisation (6.18). If C *is subsumed by* D in an empty KB, we simply say that C *is subsumed by* D, and we will write $C \preceq_2 D$. For example, the concept Man $\sqcap \exists$HasChild.Child is subsumed by the concept Man.

The problem of determining whether $C \not\equiv_2 \bot$ is called the *concept coherence* problem; the problem of determining whether C is subsumed by D is called *subsumption problem*; the problem of determining whether C is subsumed by D in Σ is called *hybrid subsumption problem*; the problem of determining whether Σ is satisfiable is called *knowledge base satisfiability problem*, and the problem of determining whether $\Sigma \models_2 C(a)$ is called *instance checking problem*. It can easily be verified that the following relations hold:

$$C \text{ is not satisfiable} \quad \text{iff} \quad C \preceq_2 \bot \tag{6.21}$$
$$C \text{ satisfiable} \quad \text{iff} \quad \{C(a)\} \text{ is satisfiable} \tag{6.22}$$
$$C \preceq_2 D \quad \text{iff} \quad C \sqcap \neg D \text{ is not satisfiable} \tag{6.23}$$
$$C \preceq_2 D \quad \text{iff} \quad \emptyset \models_2 C \Rightarrow D \tag{6.24}$$
$$C \preceq_2 D \quad \text{iff} \quad \{C(a)\} \models_2 D(a) \tag{6.25}$$
$$C \preceq_2^{\Sigma} D \quad \text{iff} \quad \Sigma \models_2 C \Rightarrow D \tag{6.26}$$
$$\Sigma \models_2 C \Rightarrow D \quad \text{iff} \quad \Sigma \cup \{(C \sqcap \neg D)(a)\} \text{ is not satisfiable} \tag{6.27}$$
$$\Sigma \models_2 C(a) \quad \text{iff} \quad \Sigma \cup \{\neg C(a)\} \text{ is not satisfiable} \tag{6.28}$$
$$\Sigma \text{ is not satisfiable} \quad \text{iff} \quad \Sigma \not\models_2 \top \Rightarrow \bot \tag{6.29}$$
$$\Sigma \text{ is not satisfiable} \quad \text{iff} \quad \Sigma \not\models_2 \bot\ (a) \tag{6.30}$$

As a consequence, all the above problems can be reduced to the knowledge base satisfiability problem. There exists a well known technique based on constraint propagation, which solves this problem. The interested reader can consult *e.g.* [32, 69, 93, 95, 99, 100].

As specified early, given the object-oriented character of DLs and $\mathcal{ALC}$ in particular, they are mainly used in order to model the domain of interest in terms of a taxonomy of concepts (classes) and their properties.

Example 11 Consider the following taxonomy.

$$\begin{array}{lcl}
\mathsf{SportsKind} & \Rightarrow & \top \\
\mathsf{IndividualSports} & \Rightarrow & \mathsf{SportsKind} \\
\mathsf{TeamSports} & \Rightarrow & \mathsf{SportsKind} \\
\mathsf{SportsTool} & \Rightarrow & \top \\
\mathsf{Football} & \Rightarrow & \mathsf{SportsTool} \\
\mathsf{Basketball} & \Rightarrow & \mathsf{SportsTool} \\
\mathsf{TennisRacket} & \Rightarrow & \mathsf{SportsTool} \\
\mathsf{SportsMovie} & \Rightarrow & \top \\
\mathsf{TeamSportsMovie} & := & \mathsf{SportsMovie} \sqcap \\
 & & (\exists \mathsf{KindOfSports}.\top) \sqcap \\
 & & (\forall \mathsf{KindOfSports}.\mathsf{TeamSports})
\end{array}$$

$$
\begin{array}{lcl}
\mathsf{IndividualSportsMovie} & := & \mathsf{SportsMovie} \sqcap \\
 & & (\exists \mathsf{KindOfSports}.\top) \sqcap \\
 & & (\forall \mathsf{KindOfSports}.\mathsf{IndividualSports}) \\
\mathsf{FootballMovie} & := & \mathsf{TeamSportsMovie} \sqcap \\
 & & (\exists \mathsf{HasSportsTool}.\top) \sqcap \\
 & & (\forall \mathsf{HasSportsTool}.\mathsf{Football}) \\
\mathsf{BasketMovie} & := & \mathsf{TeamSportsMovie} \sqcap \\
 & & (\exists \mathsf{HasSportsTool}.\top) \sqcap \\
 & & (\forall \mathsf{HasSportsTool}.\mathsf{Basket}) \\
\mathsf{TennisMovie} & := & \mathsf{IndividualSportsMovie} \sqcap \\
 & & (\exists \mathsf{HasSportsTool}.\top) \sqcap \\
 & & (\forall \mathsf{HasSportsTool}.\mathsf{TennisRacket})
\end{array}
$$

Now suppose that there are two video frame sequences identified by (o1,v1) and (o2,v2), instances of the CVO class specified in Section 4.3, Example 7. Suppose that they are about basket and tennis, respectively. We may represent the semantics of them through the assertions

$$(\exists \mathsf{Int\_As}.\mathsf{BaskedMovie})(\mathsf{o1}), (\exists \mathsf{Int\_As}.\mathsf{TennisMovie})(\mathsf{o2})$$

stating the o1 is interpreted as a basket movie, whereas o2 is interpreted as a tennis movie. Let Σ be the resulting KB. It is quite easy to verify that

$$\Sigma \models_2 \mathsf{BaskedMovie} \Rightarrow \mathsf{SportsMovie}$$

stating that a BaskedMovie is a SportsMovie, *i.e.* SportsMovie subsumes BaskedMovie w.r.t. Σ. Similarly,

$$\Sigma \models_2 \mathsf{TennisMovie} \Rightarrow \mathsf{SportsMovie}$$

holds. Therefore, if an user is interested in retrieving movies about sport, it may query Σ through the query concept

$$\mathsf{Q} = \exists \mathsf{Int\_As}.\mathsf{SportsMovie}.$$

The answer will be the list containing both (o1,v1) and (o2,v2), as

$$\begin{array}{l} \Sigma \models_2 \mathsf{Q}(\mathsf{o1}), \text{ and} \\ \Sigma \models_2 \mathsf{Q}(\mathsf{o2}) \end{array}$$

hold. Finally, given the query concept

$$\mathsf{Q}' = \exists \mathsf{Int\_As}.\mathsf{SportsMovie} \sqcap \exists \mathsf{KindOfSports}.\mathsf{IndividualSports},$$

i.e. "retrieve movies about individual sports", it follows that

$$\begin{array}{l} \Sigma \not\models_2 \mathsf{Q}'(\mathsf{o1}), \text{ whereas} \\ \Sigma \models_2 \mathsf{Q}'(\mathsf{o2}), \end{array}$$

that is, only $(\mathsf{o2}, \mathsf{v2})$ will be retrieved.

■

Chapter 7

A four-valued horn description logic

7.1 About relevance logic

To characterise information retrieval, one often relies on the notion of *relevance*: given a set of documents and a query, the task of document retrieval is to retrieve those documents, and only those, whose information content is relevant to the information content of the query (aka user information need). The centrality of relevance is the main reason why the logical formalisation of document retrieval is a non trivial problem: what is relevant, that is, is decided by the user from session to session and from time to time, and is then heavily dependent on judgments where highly subjective and scarcely reproducible factors are brought to bear [45, 246]. The possibility of a logical theory of document retrieval is then dependent on the possibility of giving a *formal* definition of relevance capable of approximating the *operational* definition of relevance given above. In order to do so, it is of fundamental importance to at least identify part or all of those subjective and contingent factors that contribute to relevance, and wire them into one's adopted logic.

In his seminal 1986 papers [273, 274], van Rijsbergen argued that an approach combining conditional reasoning and reasoning about imprecision should be used, leading to the determination through logical inference of $P(d \rightarrow q)$ ("the probability that d implies q") as an estimation of the probability of relevance of document d to query q. The rationale of using conditional reasoning is clear: if we were able to give to a document and a query *perfect* representations d and q of the information content that the user attributes to them, document retrieval could be seen just as the task of establishing the validity of the formula $d \rightarrow q$ in a logic in which "$\rightarrow$" mirrors "information containment". The rationale of using reasoning about imprecision is also clear: such perfect representations cannot be obtained because of the above-mentioned elusive character of relevance, and the system can then only make a subjective estimation of how likely it is that the user will deem the document relevant to her information need.

Concerning this point, the addition of imprecision on top of a calculus for conditional reasoning can indeed work as a "correction factor" for bridging the gap between the rigidity of logical calculi and the flexible, human-centered notion of relevance, as in principle it allows to fine-tune the system estimation of relevance as a function of contextual factors, user preferences and so on. Moreover, however, that in order to arrive at a successful logical model of document retrieval *some effort should be made in order to wire as much relevance as possible into the implication connective*, *i.e.* to design a calculus for (without imprecision) conditional

reasoning where the factors that influence relevance, as perceived by the user, are taken into account. This will be the topic of this Chapter.

The history of logic has seen a flurry of logics motivated by the need to give a natural account of the implication connective. Quite interestingly, the accounts proposed by classical, modal and other logics, have been criticised on the account that they license, as theorems of the pure calculus, sentences that suffer from *fallacies of relevance*. In other words, some conditional sentences are theorems of the given logic *even if their premise is not relevant to their conclusion*. For instance, the sentence $(\alpha \rightarrow (\beta \rightarrow \alpha))$ (asserting that a true proposition is implied by any proposition) is a theorem of classical logic. And this should strike one as peculiar, in that the fact that β holds does not have any "relevance" to the fact that α holds.

Among the first to put forth such a criticism, Nelson [210] argued that, in order for any conditional notion "$\rightarrow$" to be adequate, a sentence such as $\alpha \rightarrow \beta$ should be valid only if there be "some connection of meaning between α and β" (and this consideration should strike the document retrieval theorist as familiar!). To the surprise of many orthodox logicians who considered these issues to more properly belong to rhetoric rather than logic, the idea of a "connection of meaning between α and β" (or, more generally, the idea of α being *relevant* to β) has been shown to be amenable to formal treatment by a circle of logicians who defined a class of logical calculi called *relevance* (or *relevant*) *logics* [8, 111]. Relevance logics attempt to formalise a conditional notion in which relevance is a primary concern. By doing this, they challenge classical logic and its extensions in a number of ways, *i.e.* by introducing a new, non truth-functional connective (denoted by "$\rightarrow$") into the syntactic apparatus of classical logic, by rejecting some classical rules of inference for classical connectives, and by changing the notion of validity itself by "wiring" into it considerations of relevance.

Although relevance logics might not be a panacea for all the problems concerning the logical formalisation of document retrieval, the insights provided by relevance logics are valuable to information retrieval. In fact, even a brief analysis of the motivations put forth by relevance logicians and by document retrieval theorists, respectively, indicates a surprising coincidence of underlying tenets and purposes (see *e.g.* [139, Chapter 10]), much beyond the case of omonimy. Therefore, it seems just natural to think that, if we view retrieval as essentially consisting of a disguised form of logical inference [275], relevance logic and document retrieval might constitute the theoretical side and the applied side of the same coin. This eventually calls for the *adoption of a relevance logic as the kernel of a full-blown logic for document retrieval*. Given that the description logic we have presented in the previous chapter is essentially based on classical logic, we intend to propose the switch to a *relevance description logic*.

As with modal logics, there are many relevance logics, each formalising a different notion of relevance. The relevance logic that seems to best comply with the requirements of the document retrieval world is the logic $\mathbf{E_{fde}}$, also called the *logic of first degree (tautological) entailments* [110]. This consists of the fragment of the famous relevance logics $\mathbf{E}$ and $\mathbf{R}$ that deals with *first degree entailments* only, *i.e.* pairs of propositional (classical) formulae separated by one "$\rightarrow$" symbol. This logic seems well suited to formalise a state of affairs in which both document and query have a boolean representation, and in which the relevance of one to the other is the parameter of interest. In addition, $\mathbf{E_{fde}}$ has a *four-valued* denotational semantics (independently developed by Belnap [47] and Dunn [110]). Compliance with the denotational approach makes it amenable to the various extensions (*e.g.* to reasoning about imprecision) needed for modelling document retrieval. The logic $\mathbf{E_{fde}}$ has also been investigated from the standpoint of its computational properties: while decision in the general case

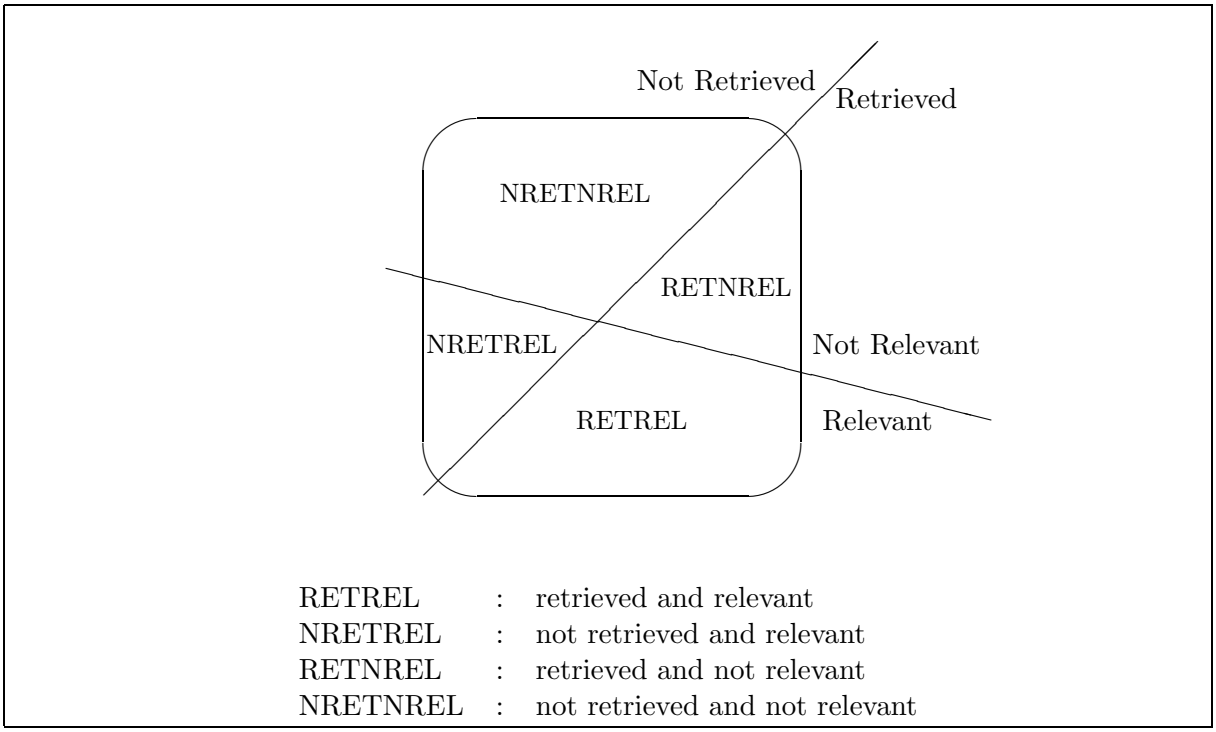


Figure 7.1: Partition of document collection.

is co-NP-complete [219], whenever α and β are formulae in Conjunctive Normal Form there exists an $O(|\alpha| \cdot |\beta|)$ algorithm that tests the validity of $\alpha \rightarrow \beta$ [182]. Relevance description logics based on a four-valued semantics have already proposed by Patel-Schneider for use in knowledge representation, and have been proven to possess a generally better computational behaviour than their two-valued analogues [216, 217, 218, 220, 221, 222, 223].

We conclude by noticing that the goal of relevance logic can be reformulated in terms of the well-known notions of *precision* and *recall* [244], too. Precision is the percentage of relevant documents among the retrieved ones, while recall is the percentage of the retrieved documents among the relevant ones. If a cut is made through the document collection to distinguish retrieved documents from non retrieved ones on the one hand as shown in Figure 7.1, the standard recall R and standard precision P may be defined as

$$\mathrm{P} = \frac{|\mathrm{RETREL}|}{|RET|} \tag{7.1}$$

$$\mathrm{R} = \frac{|\mathrm{RETREL}|}{|REL|}, \tag{7.2}$$

where $\mathrm{RET} = \mathrm{RETREL} \cup \mathrm{RETNREL}$ and $\mathrm{REL} = \mathrm{RETREL} \cup \mathrm{NRETREL}$ are the retrieved documents and the relevant documents in the collection, respectively.

In other words[1],

[1] $|S|$ is the cardinality of set S.

$$\mathrm{P} = P(\mathrm{REL}|\mathrm{RET}) = \frac{|\mathrm{REL} \cap \mathrm{RET}|}{|\mathrm{RET}|} \tag{7.3}$$

and

$$\mathrm{R} = P(\mathrm{RET}|\mathrm{REL}) = \frac{|\mathrm{REL} \cap \mathrm{RET}|}{|\mathrm{REL}|}, \tag{7.4}$$

respectively.

Let RET_{Cl} be the set of retrieved documents through classical logic, and RET_R be the set of retrieved documents with respect to a relevance logic. It can be shown that according to four-valued semantics, $\mathrm{RET}_R \subset \mathrm{RET}_{Cl}$ holds.

The empirical assumption of relevance logics is that a document $d \in \mathrm{RET}_R$ is more likely to be relevant to the user's information need, than a document $d \in \mathrm{RET}_{Cl} \setminus \mathrm{RET}_R$, *i.e.* in probabilistic terms

$$P(\mathrm{REL}|\mathrm{RET}_R) \geq P(\mathrm{REL}|(\mathrm{RET}_{Cl} \setminus \mathrm{RET}_R)). \tag{7.5}$$

Using Bayes' Theorem, this may be rewritten as

$$P(\mathrm{REL}|\mathrm{RET}_R) \geq P(\mathrm{REL}|\mathrm{RET}_{Cl}), \tag{7.6}$$

which simply says that the probability of a document to be relevant is higher for documents retrieved by means of a relevance logic than for the ones retrieved according to classical logic, *i.e.* we have higher precision. On the other hand, since $\mathrm{RET}_R \subset \mathrm{RET}_{Cl}$, it follows easily that $P(\mathrm{RET}_R|\mathrm{REL}) < P(\mathrm{RET}_{Cl}|\mathrm{REL})$, *i.e.* we have less recall. But, the aim of relevance logics is to define tautological entailment in such a way that only few of the documents of $\mathrm{RET}_{Cl} \setminus \mathrm{RET}_R$ are really considered relevant. This is expressed by

$$P(\mathrm{REL}|(\mathrm{RET}_{Cl} \setminus \mathrm{RET}_R)) \approx 0. \tag{7.7}$$

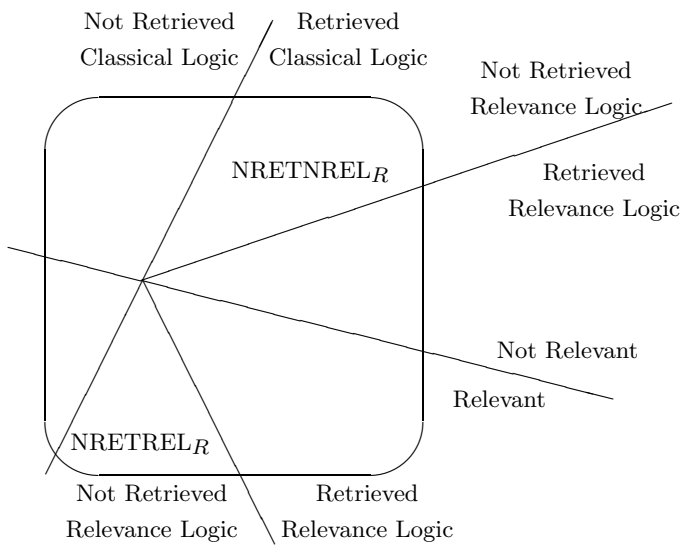
Consequently,

$$P(\mathrm{RET}_R|\mathrm{REL}) \approx P(\mathrm{RET}_{Cl}|\mathrm{REL}) \tag{7.8}$$

follows, *i.e.* we have similar recall. The two relations in Equation (7.5) and (7.7) can be explained through Figure 7.2 below. Equation (7.7) says us that the set $\mathrm{NRETREL}_R$ is quite small w.r.t. $\mathrm{RET}_{Cl} \setminus \mathrm{RET}_R$, *i.e.* $\mathrm{NRETREL}_R \cup \mathrm{NRETNREL}_R$. On the other hand, Equation (7.5) says us that the set $\mathrm{NRETNREL}_R$ is huge w.r.t. $\mathrm{RET}_{Cl} \setminus \mathrm{RET}_R$.
Certainly, the relations would probably be subscribed by a relevance logician, but no "statistical" results, to the best of our knowledge, have been given. It is worth noticing that these relations can be hopefully verified experimentally on statistically significant large collections of documents.

7.2 About inconsistencies

It has long been recognised that inconsistencies may easily arise in knowledge based information processing [48, 110]. This may be due to some rules or data being recorded in the knowledge base system. Removing inconsistencies from a knowledge base is difficult and expensive since, as we know, inconsistencies may not lie on the surface and in most cases



NRETNREL$_R$: not retrieved according relevance logic and not relevant, retrieved according classical logic and not relevant

NRETREL$_R$: not retrieved according relevance logic and relevant, retrieved according classical logic and relevant

Figure 7.2: Partition of document collection: classical vs. relevance logic.

there is no single solution to eliminate them. Furthermore, before a knowledge base is even discovered to be inconsistent, the user might have been using them for quite some time.

Example 12 To be more concrete, in the context of the representation of the semantics of multimedia data, we may have a fiction video v showing the earth with the sun and moon moving around it. Now, if there are rules in our knowledge base saying that *e.g.* "earth is a planet having only one satellite", an inconsistency in classical terms arises. Formally, one could have the following: suppose we use the set Σ_1 of first-order formulae

$$\begin{aligned}\Sigma_1 \;=\; & \{\mathsf{Satellite}(\mathsf{sun}), \mathsf{Satellite}(\mathsf{moon}), \mathsf{Star}(\mathsf{sun}),\\ & \mathsf{MovingAround}(\mathsf{earth}, \mathsf{sun}), \mathsf{MovingAround}(\mathsf{earth}, \mathsf{moon})\}\end{aligned} \tag{7.9}$$

representing the semantics of video v, and background knowledge Σ_2

$$\begin{aligned}\Sigma_2 \;=\; & \{\forall x \forall y.\mathsf{MovingAround}(x, y) \wedge \mathsf{Satellite}(y) \rightarrow \mathsf{Planet}(x),\\ & \forall x.\mathsf{Planet}(x) \rightarrow \neg\mathsf{Star}(x),\\ & \forall x.\mathsf{Satellite}(x) \rightarrow \neg\mathsf{Star}(x),\\ & \forall x.\mathsf{Star}(x) \rightarrow \mathsf{VeryBigObj}(x)\}\end{aligned} \tag{7.10}$$

stating that a satellite is moving around a planet, a planet is not a star, a satellite is not a star and a star is a very big object, respectively. It is quite straightforward to see that $\Sigma_1 \cup \Sigma_2$ is inconsistent in classical terms, as both $\mathsf{Star}(\mathsf{sun})$ and $\neg\mathsf{Star}(\mathsf{sun})$ follows. A consequence of this inconsistency is that if we are looking for very big objects and formulate the query through the determination of the set

$$\{a : a \text{ occurs in } \Sigma_1 \cup \Sigma_2, \text{ and } \Sigma_1 \cup \Sigma_2 \models \mathsf{VeryBigObj}(a)\} \tag{7.11}$$

then the answer set is

$$AS_1 = \{\mathsf{earth}, \mathsf{sun}, \mathsf{moon}\} \tag{7.12}$$

which is certainly not what we want. The right answer should be

$$AS_2 = \{\mathsf{sun}\}. \tag{7.13}$$

It is not immediate how to deal with such kind of situations. ■

A great deal of research has been devoted to constructing systems that allow reasoning in the presence of inconsistency. Most of them are based on some kind of multivalued logic (see [8, 48, 77, 160, 190, 191, 235, 280], to name just a view). These systems avoid to falling into triviality, *i.e.* concluding every sentence in the underlying language, in the presence of inconsistency. They vary in their semantics and, in particular, in their behaviour. So, the question is: which one is the most appropriate in the case of multimedia document retrieval?

As we will see below, the four-valued semantics developed in [48, 110, 182] not only specifies a notion of relevance, but has the property to deal with inconsistencies in the right way, simplifying our formal treatment in the following sections. In fact, consider the inconsistent set of formulae

$$\Sigma \quad = \quad \{p, \neg p, q, q \rightarrow r, p \rightarrow s\} \tag{7.14}$$

where $p, \neg p, q$ express the semantics of multimedia objects (see (7.9) above), and $q \rightarrow r, p \rightarrow s$ is some background knowledge (see (7.10) above). According to *e.g.* [48], from Σ we infer the facts, q, and both p and $\neg p$. This last point seems to be correct in the context of multimedia document retrieval as both p and $\neg p$ represents object's semantics, and thus, there is no reason to disregard them. An important point concerns the ability to reason about facts, *i.e.* (i) whether from $q, q \rightarrow r$ we should infer r and (ii) whether from $p, p \rightarrow s$ we should infer s. We want to be able to to conclude r from the "safe" fact q, since normal reasoning about q and r should not be disrupted by the inconsistency in p. According to [182], Σ entails r. For instance, by considering Example 12 above, we infer *e.g.* $\mathsf{Planet}(\mathsf{earth})$ through the first rule. Point (ii) is more controversial. Should we infer s from $p, p \rightarrow s$ notwithstanding there is an inconsistency about p? In our context the answer is yes. Again, this is motivated by the fact that the fact formulae represent multimedia object's semantics, and thus, if an object has been indexed to be about p and $p \rightarrow s$, then the object is about s too, no matter whether the could be an inconsistency about p. For instance, by considering Example 12 above, from $\mathsf{Star}(\mathsf{sun})$ and $\forall x.\mathsf{Star}(x) \rightarrow \mathsf{VeryBigObj}(x)$ we infer, according to [48], $\mathsf{VeryBigObj}(\mathsf{sun})$. Just note that this last inference is not allowed in some systems, like *e.g.* [191]. In conclusion, according to [48], we infer correctly the facts $\{p, \neg p, q, r, s\}$, *i.e.* in terms of Example 12, the answer set is AS_2.

7.3 Preliminaries: four-valued propositional logic

7.3.1 The logic $\mathcal{L}$ and its properties

At first, we start with propositional tautological entailment [8, 47, 48, 182].

Let $\mathcal{L}$ be the language of propositional logic, with connectives $\wedge, \vee$ and $\neg$. We will use metavariables $A, B, C, \ldots$ and $p, q, r, \ldots$ for propositions and propositional letters, respectively. Propositions in *negation normal form* (NNF), *Conjunctive Normal Form* (CNF) and *Disjunctive Normal Form* (DNF) are defined as usual.

The key difference between classical propositional logic and propositional tautological entailment is that, while the semantics of the former relies on the classical set of truth values $\{t, f\}$, the semantics of the latter relies on its *powerset* $2^{\{t,f\}}$, which contains the four values $\{t\}$, $\{f\}$, $\{t, f\}$ and $\emptyset$. These values may be understood as representing the status of a proposition in the epistemic state of a reasoning agent. Under this view, if the value of a proposition contains t, then the agent has evidence to the effect – or beliefs – that the proposition is true. Similarly, if it contains f, then the agent has evidence to the effect that the proposition is false. The value $\emptyset$ corresponds to a lack of evidence, while the truth value $\{t, f\}$ corresponds to the possession of contradictory evidence.

A *four-valued interpretation* $\mathcal{I}$ maps a proposition into an element of $2^{\{t,f\}}$ and has to satisfy the following equations:

$$\begin{array}{lll} t \in (A \wedge B)^{\mathcal{I}} & \text{iff} & t \in A^{\mathcal{I}} \text{ and } t \in B^{\mathcal{I}} \\ f \in (A \wedge B)^{\mathcal{I}} & \text{iff} & f \in A^{\mathcal{I}} \text{ or } f \in B^{\mathcal{I}} \\ t \in (A \vee B)^{\mathcal{I}} & \text{iff} & t \in A^{\mathcal{I}} \text{ or } t \in B^{\mathcal{I}} \\ f \in (A \vee B)^{\mathcal{I}} & \text{iff} & f \in A^{\mathcal{I}} \text{ and } f \in B^{\mathcal{I}} \\ t \in (\neg A)^{\mathcal{I}} & \text{iff} & f \in A^{\mathcal{I}} \text{ and} \\ f \in (\neg A)^{\mathcal{I}} & \text{iff} & t \in A^{\mathcal{I}}. \end{array} \tag{7.15}$$

It is worth noting that a two-valued interpretation is just a four-valued interpretation $\mathcal{I}$ such that $A^{\mathcal{I}}$ is either $\{t\}$ or $\{f\}$, for each A.

Example 13 Consider $C = (A \vee B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B)$. The following table enumerates the possible models of C.

Models of C	A	B
$\mathcal{I}_1$	$\{t, f\}$	$\emptyset$
$\mathcal{I}_2$	$\{t\}$	$\{t, f\}$
$\mathcal{I}_3$	$\{f\}$	$\{t\}$
$\mathcal{I}_4$	$\emptyset$	$\{t, f\}$
$\mathcal{I}_5$	$\{t, f\}$	$\{t\}$
$\mathcal{I}_6$	$\{t, f\}$	$\{f\}$
$\mathcal{I}_7$	$\{t, f\}$	$\{t, f\}$
$\mathcal{I}_8$	$\{f\}$	$\{t, f\}$

As we can notice, in a four-valued setting there are more models than two-valued models w.r.t. proposition C. In fact, only $\mathcal{I}_3$ is a two-valued model of C. ■

We might characterize the distinction between two-valued and four-valued semantics as the distinction between *implicit* and *explicit* falsehood: in a two-valued logic a formula is (implicitly) false in an interpretation iff it is not true, while in a four-valued logic this need not be

the case. Note that our truth conditions are always given in terms of belongings $\in$ (and never in terms of non belongings $\notin$) of truth values to interpretations.

Let $\mathcal{I}$ be an interpretation, let A, B be two propositions and let Σ be a set of propositions (*i.e.* a knowledge base). $\mathcal{I}$ *satisfies* (*is a model of*) A iff $t \in A^{\mathcal{I}}$; A and B are *equivalent* (denoted by $A \equiv_4 B$) iff they have the same models; $\mathcal{I}$ *satisfies* (*is a model of*) Σ iff $\mathcal{I}$ is a model of A, for all $A \in \Sigma$; Σ *entails* A (denoted by $\Sigma \models_4 A$) iff all models of Σ are models of A.

For ease of notation, we will often omit braces, thus writing *e.g.* $A, B \models_4 C$ in place of $\{A, B\} \models_4 C$ and $\models_4 A$ in place of $\emptyset \models_4 A$.

Without loss of generality, we can restrict our attention to propositions in NNF only. In fact, by means of the following equivalences, every proposition A can be transformed in linear time and space into an equivalent proposition in NNF, denoted by A_{NNF}:

$$\neg\neg A \equiv_4 A \tag{7.16}$$
$$\neg(A \wedge B) \equiv_4 \neg A \vee \neg B \tag{7.17}$$
$$\neg(A \vee B) \equiv_4 \neg A \wedge \neg B \tag{7.18}$$

It is easy to see that every proposition A can be put into an equivalent CNF, denoted by A_{CNF} (similarly for propositions in DNF). In fact, just transform A into an A_{NNF} and thereafter by applying to A_{NNF} the equivalences (7.19) and (7.20) below we obtain an A_{CNF}.

$$A \wedge (B \vee C) \equiv_4 (A \wedge B) \vee (A \wedge C) \tag{7.19}$$
$$A \vee (B \wedge C) \equiv_4 (A \vee B) \wedge (A \vee C) \tag{7.20}$$

The following relations can easily be verified.

$$A \models_4 A_{CNF} \text{ and } A_{CNF} \models_4 A \tag{7.21}$$
$$A \models_4 B \text{ and } B \models_4 C \text{ implies } A \models_4 C \tag{7.22}$$
$$A \wedge B \models_4 A \text{ and } A \models_4 A \vee B \tag{7.23}$$

It is well known that the above Equations (7.21), (7.22) and (7.23) constitutes a simple axiomatization of $\mathcal{L}$ (see, *e.g.* [48]).

In the following we list some properties of $\mathcal{L}$. For each of such cases below, a interpretation $\mathcal{I}$ confirming the property is given.

$$A \wedge (\neg A \vee B) \not\models_4 B \text{ (no modus ponens)} \tag{7.24}$$
$$A^{\mathcal{I}} = \{t, f\}, B^{\mathcal{I}} = \emptyset$$
$$A \not\models_4 B \vee \neg B \text{ (\textit{i.e.} no tautologies)} \tag{7.25}$$
$$A^{\mathcal{I}} = \{t\}, B^{\mathcal{I}} = \emptyset$$
$$A \wedge \neg A \not\models_4 B \text{ (\textit{i.e.} every KB is satisfiable)} \tag{7.26}$$
$$A^{\mathcal{I}} = \{t, f\}, B^{\mathcal{I}} = \emptyset$$
$$\Sigma \models_4 A \text{ implies } \Sigma \models_2 A \tag{7.27}$$
every two-valued model of Σ is a four-valued model of Σ

(7.24) states that modus ponens is not a valid rule of inference (which is certainly an undesired property), whereas (7.25) and (7.26) states that the so-called paradoxes of logical implication do not hold, *e.g.* $A \wedge \neg A \not\models_4 B$ states that from a classical inconsistent KB not every thing can be inferred. It is worth noticing that in $\mathcal{L}$ *every* KB is satisfiable. Moreover, a consequence of the fact that every KB is four-valued satisfiable, the classical relation "$A \models_2 B$ iff $A \wedge \neg B$ not two-valued satisfiable" can not be applied in a four-valued setting, *i.e.* "$A \models_4 B$ iff $A \wedge \neg B$ not four-valued satisfiable" does not hold, as $A \wedge \neg B$ is four-valued satisfiable. Finally, (7.27) states that tautological entailment is sound with respect to classical logic, *i.e.* every inference that can be drawn within four-valued semantics can also be drawn within two-valued semantics. In other words, those inferences that conform to four-valued intuitions also conform to two-valued ones, which means that a two-value-reasoning user does not run the risk of being offered a conclusion she does not subscribe to. This is a consequence of the fact that the set of two-valued interpretations is a (proper) subset of the set of four-valued interpretations. Hence, from (7.25) and (7.27) it follows that $\models_4 \subset \models_2$.

We conclude this section by pointing out that in *e.g.* [48] entailment has been defined as follows: $A \models_4 B$ iff for all $\mathcal{I}$,

1. if $t \in A^{\mathcal{I}}$ then $t \in B^{\mathcal{I}}$, and
2. if $f \in B^{\mathcal{I}}$ then $f \in A^{\mathcal{I}}$.

Just note that with respect to our definition of entailment, condition 2 is missing. Notwithstanding, it is well known that the two definitions are equivalent. This is a direct consequence of the following proposition.

Proposition 1 *In $\mathcal{L}$, $A \models_4 B$ iff $\neg B \models_4 \neg A$.* ⊣

Proof: We show only $A \models_4 B$ implies $\neg B \models_4 \neg A$. The other direction can simply be obtained by replacing in $A \models_4 B$, A with $\neg B$ and B with $\neg A$ and applying relation (7.16).

Suppose $A \models_4 B$. Assume to the contrary that there is model $\mathcal{I}$ of $\neg B$ not being a model of $\neg A$. Hence, $f \in B^{\mathcal{I}}$ and $f \notin A^{\mathcal{I}}$. Let $\tilde{\mathcal{I}}$ be the following interpretation: for all propositional letters p, let

$$\begin{array}{lll} t \in p^{\tilde{\mathcal{I}}} & \text{iff} & f \notin p^{\mathcal{I}} \\ f \in p^{\tilde{\mathcal{I}}} & \text{iff} & t \notin p^{\mathcal{I}} \end{array}$$

We show on induction on the numbers of connectives in A that $t \in A^{\tilde{\mathcal{I}}}$.

A **is a literal:** Suppose A is a letter. Hence, $f \notin A^{\mathcal{I}}$ implies $t \in A^{\tilde{\mathcal{I}}}$, by definition. If A is a literal $\neg p$, then $f \notin A^{\mathcal{I}}$ implies $t \notin p^{\mathcal{I}}$. By definition, $f \in p^{\tilde{\mathcal{I}}}$ and, thus, $t \in A^{\tilde{\mathcal{I}}}$ follows.

induction step: Suppose A is $C \wedge D$. Hence, $f \notin A^{\mathcal{I}}$ implies $f \notin C^{\mathcal{I}}$ and $f \notin D^{\mathcal{I}}$. By induction on C and D, $t \in C^{\tilde{\mathcal{I}}}$ and $t \in D^{\tilde{\mathcal{I}}}$ follow. Therefore, $t \in A^{\tilde{\mathcal{I}}}$. The case A is $C \vee D$ is similar.

In a similar way, it can be shown (by induction on the number of connectives in B) that $t \notin B^{\tilde{\mathcal{I}}}$, *e.g.* if B is $\neg C$, then from $f \in B^{\mathcal{I}}$, $t \in p^{\mathcal{I}}$ follows, and thus, $f \notin p^{\tilde{\mathcal{I}}}$. Therefore, $t \notin B^{\tilde{\mathcal{I}}}$. As a consequence, $\tilde{\mathcal{I}}$ is a model of A but not a model of B, contrary to our assumption. Q.E.D.

Hence, if $A \models_4 B$ then condition 1 is satisfied and by Proposition 1, condition 2 is satisfied too (and vice-versa). A simple consequence of the above proposition is the following.

Proposition 2 *In $\mathcal{L}$, $A \equiv_4 B$ iff $\neg B \equiv_4 \neg A$.* ⊣

Proof: $A \equiv_4 B$ iff ($A \models_4 B$ and $B \models_4 A$) iff ($\neg B \models_4 \neg A$ and $\neg A \models_4 \neg B$) iff $\neg B \equiv_4 \neg A$. `Q.E.D.`

7.3.2 The logic $\mathcal{L}_+$ and its properties

In Section 7.3.1 we have seen that modus pones is not a valid inference rule in $\mathcal{L}$ (see relation (7.24). This property is certainly a limitation of $\mathcal{L}$ from an inference power point of view. In this section we will overcome to this limitation by introducing a new connective $\rightarrow$. Let $\mathcal{L}_+$ be $\mathcal{L}$ extended to the set of propositions of type $A \rightarrow B$, where $\rightarrow$ can be nested. For instance, $(A \vee (B \wedge C)) \rightarrow (B \vee E)$ is a proposition in $\mathcal{L}_+$, so as $A \rightarrow (B \rightarrow (C \vee E))$

From a semantics point of view, an interpretation $\mathcal{I}$ has also to satisfy the following conditions:

$$\begin{array}{lll} t \in (A \rightarrow B)^{\mathcal{I}} & \text{iff} & t \in A^{\mathcal{I}} \text{ implies } t \in B^{\mathcal{I}} \\ f \in (A \rightarrow B)^{\mathcal{I}} & \text{iff} & t \in A^{\mathcal{I}} \text{ and } f \in B^{\mathcal{I}} \end{array} \tag{7.28}$$

Notice, that now

$$A, A \rightarrow B \models_4 B \tag{7.29}$$

holds. Most of the properties of $\mathcal{L}$, *i.e.* (7.16)–(7.27), hold for $\mathcal{L}_+$ too. In particular, all $\mathcal{L}_+$ KBs are satisfiable. Moreover, the following relations can easily be verified.

$$\neg(A \rightarrow B) \equiv_4 A \wedge \neg B \tag{7.30}$$
$$(A \rightarrow B) \wedge (B \rightarrow C) \models_4 A \rightarrow C \tag{7.31}$$
$$(A \rightarrow B) \wedge (A \rightarrow C) \equiv_4 A \rightarrow (B \wedge C) \tag{7.32}$$
$$(A \rightarrow C) \wedge (B \rightarrow C) \equiv_4 (A \vee B) \rightarrow C \tag{7.33}$$

As for $\mathcal{L}$, some properties, which hold in a two-valued framework, do not hold in a $\mathcal{L}_+$. For instance, in the following we list some of these and give also a counterexample interpretation.

$$A \rightarrow B \not\models_4 \neg B \rightarrow \neg A \tag{7.34}$$
$$t \in A^{\mathcal{I}} \text{ and } B^{\mathcal{I}} = \{t, f\}$$
$$A \rightarrow B \not\models_4 \neg A \vee B \tag{7.35}$$
$$A^{\mathcal{I}} = B^{\mathcal{I}} = \emptyset$$
$$\neg A \vee B \not\models_4 A \rightarrow B \tag{7.36}$$
$$A^{\mathcal{I}} = \{t, f\} \text{ and } B^{\mathcal{I}} = \emptyset$$
$$A \rightarrow B \not\equiv_4 \neg A \vee B \tag{7.37}$$
$$A^{\mathcal{I}} = B^{\mathcal{I}} = \emptyset$$

Here, relation (7.34) states that *contraposition* is not valid in $\mathcal{L}_+$.

Further, note that Proposition 1 and Proposition 2 do not hold in $\mathcal{L}_+$, *e.g.* $\neg(A \rightarrow B) \models_4 A \wedge \neg B$ does not imply $\neg A \vee B \models_4 A \rightarrow B$.

An axiomatization can simply be obtained by extending the axioms of $\mathcal{L}$ with the modus ponens schema rule (7.29), *i.e.*

$$A \models_4 A_{CNF} \text{ and } A_{CNF} \models_4 A \tag{7.38}$$
$$A \models_4 B \text{ and } B \models_4 C \text{ implies } A \models_4 C \tag{7.39}$$
$$A \wedge B \models_4 A \text{ and } A \models_4 A \vee B \tag{7.40}$$
$$A, A \rightarrow B \models_4 B \tag{7.41}$$

is a sound and complete axiomatization of $\mathcal{L}_+$.

Of course, for all two-valued interpretations $\mathcal{I}$ $(A \rightarrow B)^{\mathcal{I}} = (\neg A \vee B)^{\mathcal{I}}$ holds. Hence, in a two-valued setting $A \rightarrow B$ is noting else as a macro in place of $\neg A \vee B$.

Finally, as for the two-valued case, the deduction theorem holds in $\mathcal{L}^+$, too.

Proposition 3 *In* $\mathcal{L}_+$, $A, B \models_4 C$ *iff* $A \models_4 B \rightarrow C$. ⊣

Proof: Straightforward. Q.E.D.

For example, it is easily verified that

$$A, A \rightarrow B, C \models_4 B \wedge C \text{ iff } A, A \rightarrow B \models_4 C \rightarrow (B \wedge C) \tag{7.42}$$

holds. A direct consequence of this fact is that, to the contrary to what happens in $\mathcal{L}$, in $\mathcal{L}_+$ there are tautologies. For instance, since $A \models_4 B \rightarrow A$, from the deduction theorem it follows that

$$\models_4 A \rightarrow (B \rightarrow A). \tag{7.43}$$

Example 14 Let Σ be the KB

$$\Sigma = \{ \begin{array}{l} \mathsf{Gil} \rightarrow \mathsf{Adult}, \\ \mathsf{Adult} \rightarrow \mathsf{Tall}, \\ \mathsf{Karl} \rightarrow \mathsf{Child}, \\ \mathsf{Child} \rightarrow \mathsf{Tall} \\ \mathsf{Gil} \vee \mathsf{Karl} \ \}. \end{array}$$

It can easily be verified that,

$$\Sigma \models_4 \mathsf{Tall}.$$

■

7.3.3 The logic Horn-$\mathcal{L}$

We conclude the part about four-valued propositional logic by considering a special case of the logic $\mathcal{L}_+$ on which our final logic is founded: Horn-$\mathcal{L}$.

Horn-$\mathcal{L}$ is a restriction of $\mathcal{L}_+$ to the case where the implications of the form $A \rightarrow B$ are horn rules. Formally, let Horn-$\mathcal{L}$ be $\mathcal{L}$ extended with *horn rules* (denoted by R) of the form

$$A \leftarrow A_1, \ldots, A_n, \tag{7.44}$$

with $n \geq 1$ and $A, A_1, \ldots, A_n$ propositional letters. A is called *head* and $A_1, \ldots, A_n$ is called *body*.

A *fact* is any proposition $A \in \mathcal{L}$. It is worth noting that *e.g.* $A \vee (B \wedge \neg C)$ is a fact: we do not limit ourself to facts which are horn propositions.

A *goal* (denoted by G) is an expression of the form

$$\leftarrow A_1, \ldots, A_n, \tag{7.45}$$

with $n \geq 0$ and $A_1, \ldots, A_n$ propositional letters. The case $n = 0$ is called *empty goal* and is indicated with $\leftarrow \blacksquare$.

A *query* (denoted by Q) is an expression of the form

$$A_1 \wedge \ldots \wedge A_n, \tag{7.46}$$

with $n \geq 1$ and $A_1, \ldots, A_n$ propositional letters.

A HORN-$\mathcal{L}$ KB Σ is a finite set of horn rules and facts. With $\Sigma_R \subseteq \Sigma$ we indicate the set of horn rules in Σ and with $\Sigma_F \subseteq \Sigma$ we indicate the set of facts in Σ. A *horn* KB is a finite set of horn rules and horn facts.

We will say that a HORN-$\mathcal{L}$ KB Σ is *safe* iff no head A of a rule $R \in \Sigma_R$ appears in Σ_F. For instance, the KB in Example 14 is a safe HORN-$\mathcal{L}$ KB, whereas $\{A \vee B, A \leftarrow B\}$ is not a safe HORN-$\mathcal{L}$ KB ($\Sigma_F = \{A \vee B\}$, $\Sigma_R = \{A \leftarrow B\}$ and head A of $A \leftarrow B$ appears in Σ_F).

Finally, the definition of *recursive* HORN-$\mathcal{L}$ KBs Σ is as usual: for all horn rules $A \leftarrow A_1, \ldots, A_n \in \Sigma$ we will say that A *directly uses* A_i for all $1 \leq i \leq n$. Let *uses* be the transitive closure of the relation directly uses in Σ. We will say that a HORN-$\mathcal{L}$ KB Σ is *recursive* iff there is A such that A uses A trough the rules in Σ. The KBs we will interested in are *safe and non recursive* HORN-$\mathcal{L}$ KBs.

From a semantics point of view, we rely on the semantics of expressions $A \rightarrow B \in \mathcal{L}_+$. Remember that $\mathcal{I}$ satisfies $A \rightarrow B$ iff if $t \in A^{\mathcal{I}}$ then $t \in B^{\mathcal{I}}$. Therefore, an interpretation $\mathcal{I}$

1. *satisfies* a horn rule $A \leftarrow A_1, \ldots, A_n$ iff if $t \in (A_1 \wedge \ldots \wedge A_n)^{\mathcal{I}}$ then $t \in A^{\mathcal{I}}$;
2. *satisfies* a goal $\leftarrow A_1, \ldots, A_n$ iff $t \notin (A_1 \wedge \ldots \wedge A_n)^{\mathcal{I}}$ and $n \geq 1$.

Notice that the empty goal is never satisfied.

An interpretation $\mathcal{I}$ *satisfies* (*is a model of*) a HORN-$\mathcal{L}$ KB Σ iff $\mathcal{I}$ satisfies each element of it. Satisfiability is extended to an arbitrary set S of horn rules, facts and goals, as usual. Finally, a HORN-$\mathcal{L}$ KB Σ *entails* a query Q (denoted by $\Sigma \models_4 Q$) iff every model of Σ satisfies Q.

Let Q be a query of the form $A_1 \wedge \ldots \wedge A_n$. The *associated goal* of Q (denoted by G_Q) is $\leftarrow A_1, \ldots, A_n$. It is easily verified that

$$\Sigma \models_4 Q \quad \text{iff} \quad \Sigma \cup \{G_Q\} \text{ not satisfiable.} \tag{7.47}$$

Just let us mention that in case of horn KB, equivalence between four-valued entailment and two-valued entailment holds.

Proposition 4 *In* HORN-$\mathcal{L}$*, let* Σ *be a horn KB and let* Q *be a query. Then* $\Sigma \models_4 Q$ *iff* $\Sigma \models_2 Q$. ⊣

In the general case where Σ is a HORN-$\mathcal{L}$ KB, then $\Sigma \models_4 A_1 \wedge \ldots \wedge A_n$ implies $\Sigma \models_2 A_1 \wedge \ldots \wedge A_n$, but not vice-versa. In fact, consider $\Sigma = \{\neg A \vee B, A\}$. Then $\Sigma \models_2 B$, but $\Sigma \not\models_4 B$.

Example 15 Let Σ be the KB

$$\Sigma = \{ \begin{array}{l} \mathsf{Adult} \leftarrow \mathsf{Gil}, \\ \mathsf{Tall} \leftarrow \mathsf{Adult}, \\ \mathsf{Gil} \ \}. \end{array}$$

It can easily be verified that, Σ is safe, non-recursive, horn and that

$$\Sigma \models_4 \mathsf{Tall} \text{ iff } \Sigma \models_2 \mathsf{Tall}$$

holds, confirming Proposition 4. ■

7.4 Four-valued horn $\mathcal{ALC}$

7.4.1 Four-valued $\mathcal{ALC}$

The four-valued semantics for $\mathcal{ALC}$ is similar to the four-valued semantics described in [222].

7.4.1.1 Syntax and semantics

Syntax: From a syntax point of view, we refer to the definitions of Section 6.1 about $\mathcal{ALC}$.

Semantics: From a semantics point of view, as usual, the four truth values are the elements of $2^{\{t,f\}}$. An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non empty set $\Delta^{\mathcal{I}}$ (the *domain* of $\mathcal{I}$) and a function $\cdot^{\mathcal{I}}$ (the *interpretation function* of $\mathcal{I}$) such that

1. $\cdot^{\mathcal{I}}$ maps every concept into a function from $\Delta^{\mathcal{I}}$ to $2^{\{t,f\}}$;
2. $\cdot^{\mathcal{I}}$ maps every role into a function from $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ to $2^{\{t,f\}}$;
3. $\cdot^{\mathcal{I}}$ maps every individual into $\Delta^{\mathcal{I}}$;
4. $a^{\mathcal{I}} \neq b^{\mathcal{I}}$, if $a \neq b$.

The interpretation function can best be understood as an extension function of two separate two-valued extensions – the positive extension and the negative extension – defined next.

Given an interpretation $\mathcal{I}$, the *positive extension* of a concept C (denoted by $[\![C^{\mathcal{I}}]\!]^+$) is defined as the set $\{d \in \Delta^{\mathcal{I}} : t \in C^{\mathcal{I}}(d)\}$, whereas the *negative extension* of a concept C (denoted by $[\![C^{\mathcal{I}}]\!]^-$) is defined as the set $\{d \in \Delta^{\mathcal{I}} : f \in C^{\mathcal{I}}(d)\}$. The positive and negative extension of a role is defined similarly. It is easily verified that a two-valued interpretation is an interpretation $\mathcal{I}$ such that for every concept C, $[\![C^{\mathcal{I}}]\!]^- = \Delta^{\mathcal{I}} \setminus [\![C^{\mathcal{I}}]\!]^+$ and for all roles R, $[\![R^{\mathcal{I}}]\!]^- = \Delta^{\mathcal{I}} \setminus [\![R^{\mathcal{I}}]\!]^+$. Unlike two-valued semantics, the positive extension and the negative extension need not to be complement of each other. Domain elements that are members of neither set are not known to belong to the concept and are not known not to belong to the concept. This is a perfectly reasonable state for a system that is not a perfect reasoner or does not have complete information. Domain elements that are members of both sets can be thought of as inconsistent with respect to that concept in that there is evidence to indicate

that they are in the extension of the concept and, at the same time, not in the extension of the concept. This is a slightly harder state to rationalize but can be considered a possibility in the light of inconsistent information.

The extensions of concepts have to meet certain restrictions, designed so that the formal semantics respects the informal meaning of concepts and roles.

For example, the positive extension of the concept $\mathsf{A} \sqcap \mathsf{B}$ must be the intersection of the positive extension of A and B and its negative extension must be the union of their negative extensions, thus formalizing the intuitive notion of conjunction in the context of the four-valued semantics.

Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation. The interpretation function $\cdot^{\mathcal{I}}$ has to meet the following equations:

$$
\begin{aligned}
[\![(C \sqcap D)^{\mathcal{I}}]\!]^{+} &= [\![C^{\mathcal{I}}]\!]^{+} \cap [\![D^{\mathcal{I}}]\!]^{+} && (7.48)\\
[\![(C \sqcap D)^{\mathcal{I}}]\!]^{-} &= [\![C^{\mathcal{I}}]\!]^{-} \cup [\![D^{\mathcal{I}}]\!]^{-} && (7.49)\\
[\![(C \sqcup D)^{\mathcal{I}}]\!]^{+} &= [\![C^{\mathcal{I}}]\!]^{+} \cup [\![D^{\mathcal{I}}]\!]^{+} && (7.50)\\
[\![(C \sqcup D)^{\mathcal{I}}]\!]^{-} &= [\![C^{\mathcal{I}}]\!]^{-} \cap [\![D^{\mathcal{I}}]\!]^{-} && (7.51)\\
[\![(\neg C)^{\mathcal{I}}]\!]^{+} &= [\![C^{\mathcal{I}}]\!]^{-} && (7.52)\\
[\![(\neg C)^{\mathcal{I}}]\!]^{-} &= [\![C^{\mathcal{I}}]\!]^{+} && (7.53)\\
[\![(\exists R.C)^{\mathcal{I}}]\!]^{+} &= [\![(\neg \forall R.\neg C)^{\mathcal{I}}]\!]^{+} && (7.54)\\
[\![(\exists R.C)^{\mathcal{I}}]\!]^{-} &= [\![(\neg \forall R.\neg C)^{\mathcal{I}}]\!]^{-} && (7.55)
\end{aligned}
$$

Just notice that the above equations are equivalent to the following reformulation:

$$
\begin{aligned}
t \in (C \sqcap D)^{\mathcal{I}}(d) &\text{ iff } t \in C^{\mathcal{I}}(d) \text{ and } t \in D^{\mathcal{I}}(d) && (7.56)\\
f \in (C \sqcap D)^{\mathcal{I}}(d) &\text{ iff } f \in C^{\mathcal{I}}(d) \text{ or } f \in D^{\mathcal{I}}(d) && (7.57)\\
t \in (C \sqcup D)^{\mathcal{I}}(d) &\text{ iff } t \in C^{\mathcal{I}}(d) \text{ or } t \in D^{\mathcal{I}}(d) && (7.58)\\
f \in (C \sqcup D)^{\mathcal{I}}(d) &\text{ iff } f \in C^{\mathcal{I}}(d) \text{ and } f \in D^{\mathcal{I}}(d) && (7.59)\\
t \in (\neg C)^{\mathcal{I}}(d) &\text{ iff } f \in C^{\mathcal{I}}(d) && (7.60)\\
f \in (\neg C)^{\mathcal{I}}(d) &\text{ iff } t \in C^{\mathcal{I}}(d) && (7.61)\\
t \in (\exists R.C)^{\mathcal{I}}(d) &\text{ iff } t \in (\neg \forall R.\neg C)^{\mathcal{I}}(d) && (7.62)\\
f \in (\exists R.C)^{\mathcal{I}}(d) &\text{ iff } f \in (\neg \forall R.\neg C)^{\mathcal{I}}(d) && (7.63)
\end{aligned}
$$

It is worth noting that the semantics for the $(\exists)$ connective is given in terms of the $(\forall)$ connective (see Equations (7.54) and (7.55)).

For it, we present two different semantics:

Type A: for each $d, d' \in \Delta^{\mathcal{I}}$

$$
\begin{aligned}
t \in (\forall R.C)^{\mathcal{I}}(d) &\text{ iff } \forall\, e \in \Delta^{\mathcal{I}}, t \in R^{\mathcal{I}}(d,e) \text{ implies } t \in C^{\mathcal{I}}(e)\\
f \in (\forall R.C)^{\mathcal{I}}(d) &\text{ iff } \exists\, e \in \Delta^{\mathcal{I}}, t \in R^{\mathcal{I}}(d,e) \text{ and } f \in C^{\mathcal{I}}(e)
\end{aligned}
\qquad (7.64)
$$

Type B: for each $d, d' \in \Delta^{\mathcal{I}}$

$$\begin{array}{lll} t \in (\forall R.C)^{\mathcal{I}}(d) & \text{iff} & \forall\ e \in \Delta^{\mathcal{I}}, f \in R^{\mathcal{I}}(d,e) \text{ or } t \in C^{\mathcal{I}}(e) \\ f \in (\forall R.C)^{\mathcal{I}}(d) & \text{iff} & \exists\ e \in \Delta^{\mathcal{I}}, t \in R^{\mathcal{I}}(d,e) \text{ and } f \in C^{\mathcal{I}}(e) \end{array} \tag{7.65}$$

Notice that type B semantics is used in [222], whereas type A semantics has been proposed in [203, 266]. It is worth noting that the type A semantics for the $\forall$ connective follows the schema of the semantics for the $\rightarrow$ connective in $\mathcal{L}^+$. In particular, with respect to type A semantics, the concept $\forall R.C$ is viewed as the formula

$$\forall y.R(x,y) \rightarrow C(y),$$

while with respect to the type B semantics $\forall R.C$ is viewed as the formula

$$\forall y.\neg R(x,y) \vee C(y).$$

In the following, the default will be the semantics of type A. A concept C is *equivalent* to a concept D (denoted by $C \equiv_4^A D$) iff $[\![C^{\mathcal{I}}]\!]^+ = [\![D^{\mathcal{I}}]\!]^+$, for every interpretation $\mathcal{I}$. A concept C *subsumes* a concept D (denoted by $D \preceq_4^A C$) iff $[\![D^{\mathcal{I}}]\!]^+ \subseteq [\![C^{\mathcal{I}}]\!]^+$, for every interpretation $\mathcal{I}$. A concept C is *coherent* iff there is an interpretation $\mathcal{I}$ such that $[\![C^{\mathcal{I}}]\!]^+ \neq \emptyset$.

With respect to assertions, we have the following definitions. An interpretation $\mathcal{I}$ *satisfies an assertion* $C(a)$, iff $t \in C^{\mathcal{I}}(a^{\mathcal{I}})$, whereas $\mathcal{I}$ *satisfies an assertion* $R(a,b)$ iff $t \in R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}})$. Given two $\mathcal{ALC}$ assertions A and B, A is *equivalent* to B (denoted by $A \equiv_4^A B$) iff for every interpretation $\mathcal{I}$, $\mathcal{I}$ satisfies A iff $\mathcal{I}$ satisfies B.

The semantics of specializations is specified by saying that the specialization $C \Rightarrow D$ (resp. $R_1 \Rightarrow R_2$) is *satisfied* by $\mathcal{I}$ iff $[\![C^{\mathcal{I}}]\!]^+ \subseteq [\![D^{\mathcal{I}}]\!]^+$ (resp. $[\![{R_1}^{\mathcal{I}}]\!]^+ \subseteq [\![{R_2}^{\mathcal{I}}]\!]^+$).

An interpretation $\mathcal{I}$ *satisfies* (*is a model of*) a knowledge base Σ (a set of assertions and specializations) iff $\mathcal{I}$ satisfies all elements in Σ. A knowledge base Σ *entails* an assertion $C(a)$ (denoted by $\Sigma \models_4^A C(a)$) iff all models of Σ satisfy $C(a)$. Similarly, a knowledge base Σ *entails* a specialization $C \Rightarrow D$ (denoted by $\Sigma \models_4^A C \Rightarrow D$) iff all models of Σ satisfy $C \Rightarrow D$.

Finally, all the above definitions are given for the case of type B semantics too. In this case we will use $\cdot_4^B$, in place of $\cdot_4^A$. For instance, we will write $\Sigma \models_4^B C(a)$, if Σ entails $C(a)$ with respect to type B semantics.

7.4.1.2 Discussion of the semantics

At first, let us resume which of the properties (6.21) – (6.30) hold for four-valued $\mathcal{ALC}$ too. It can easily be verified that the following properties hold[2]:

$$C \preceq_4^A D \quad \text{iff} \quad \emptyset \models_4^A C \Rightarrow D \tag{7.66}$$

$$C \preceq_4^A D \quad \text{iff} \quad \{C_1(a)\} \models_4^A D(a) \tag{7.67}$$

$$C \preceq_4^{A,\Sigma} D \quad \text{iff} \quad \Sigma \models_4^A C \Rightarrow D \tag{7.68}$$

[2]They hold for type B semantics too.

Since every KB is (four-valued) satisfiable, *i.e.* all concepts are coherent, properties (6.21), (6.22), (6.23), (6.27), (6.28), (6.29) and (6.30) do not hold in a four-valued setting. As the above table shows, unlike classical DLs, the only interesting problems in a four-valued setting is the subsumption problem and the instance checking problem. Moreover, the subsumption problem can easily be reduced to the instance checking problem. This property holds for all four-valued DLs we will analyse in this thesis.

It easy to see that all the properties with respect to four-valued propositional logic, as (7.16)-(7.27) and (7.29)-(7.33), can be reformulated in a four-valued DL setting. These properties can be used in order to formulate subsumption relations and entailment relations both holding and not holding in our four-valued DL. Rather to address all these points, we show only some of these.

Soundness of the semantics: Reasoning in our logic is *sound* with respect to two-valued semantics. In fact, it is easily verified that for all knowledge bases Σ, for all assertions A and for all concepts C and D, $D \preceq_4^A C$ implies $D \preceq_2 C$ and $\Sigma \models_4^A A$ implies $\Sigma \models_2 A$. From $A \sqcap \neg A \preceq_2 B$ and $A \sqcap \neg A \not\preceq_4^A B$, it follows that $\preceq_4^A \subset \preceq_2$ and $\models_4^A \subset \models_2$ hold. Similarly for type B semantics.

A subsumption relation: In [203, 222, 266] it has already shown that $\preceq_4^B$ and $\models_4^A$ capture an *interesting* subset of $\preceq_2$ and $\models_2$, respectively. For instance,

$$(\mathsf{A} \sqcup \mathsf{B}) \sqcap \forall \mathsf{R}.(\mathsf{C} \sqcap \mathsf{D}) \preceq_4^A (\mathsf{A} \sqcap \forall \mathsf{R}.\mathsf{C}) \sqcup (\mathsf{B} \sqcap \forall \mathsf{R}.\mathsf{D}) \tag{7.69}$$

The above relation is obtained by applying the DL variants of (7.19) and (7.32), and by viewing $\forall R.C$ as the formula $\forall y.R(x,y) \rightarrow C(y)$.

Modus ponens on specialisations: Our semantics allows bottom up propagation through a taxonomy. In fact, the following relations hold.

$$\{C(a), C \Rightarrow D\} \models_4^B D(a) \tag{7.70}$$
$$\{A(a), A := C\} \models_4^B C(a) \tag{7.71}$$
$$\{C(a), A := C\} \models_4^B A(a)\ . \tag{7.72}$$

Please, note that contraposition does not hold, *i.e.*

$$\{\neg D(a), C \Rightarrow D\} \not\models_4^A C(a)\ . \tag{7.73}$$

Not licensed relations: In what follows we will show which inferences, licensed in two-valued semantics, are left out by our semantics. It is quite obvious that the so-called paradoxes of logical implication, *i.e.* $\{C(a), \neg C(a)\} \models_2 D(b)$ and $\models_2 (C \sqcup \neg C)(a)$ do not hold in our type A and type B semantics (see (7.25), (7.26)).

Generally, modus ponens is not a valid inference rule in our logic, *i.e.* $\{(C \sqcap (\neg C \sqcup D))(a)\} \not\models_4^A D(a)$ (see (7.24).

But, the semantics of type A allows a restricted form of modus ponens, called *modus ponens on roles*: for all concepts C, D, for any role R, and for all individuals a, b,

$$\{(\forall R.C)(a), R(a,b)\} \models_4^A C(b) \tag{7.74}$$

and

$$(\exists R.C) \sqcap (\forall R.D) \preceq_4^A \exists R.(C \sqcap D) \tag{7.75}$$

For instance,

$$(\forall \mathsf{Author.Italian})(\mathsf{d}), \mathsf{Author}(\mathsf{d}, \mathsf{umberto}) \models_4^A \mathsf{Italian}(\mathsf{umberto}),$$

asserts that "if d is a document whose Authors are Italian and umberto is an Author of d, then umberto is Italian". On the other hand,

$$(\exists \mathsf{Author.Researcher}) \wedge (\forall \mathsf{Author.Italian}) \preceq_4^A \exists \mathsf{Author.}(\mathsf{Researcher} \wedge \mathsf{Italian})$$

asserts that "each document having Italian Authors such that one of it is Researcher, is also a document of an Italian Researcher".

This kind of inference is a direct consequence of viewing viewing $\forall R.C$ as $\forall y.R(x,y) \rightarrow C(y)$ and, thus, $R(a,b) \wedge (\forall y.R(a,y) \rightarrow C(y)) \models_4 C(b)$. This kind of inference is not allowed in type B semantics, since[3] $R(a,b) \wedge (\forall y.\neg R(a,y) \vee C(y)) \not\models_4 C(b)$. Hence, $\models_4^B \neq \models_4^A$ and $\preceq_4^B \neq \preceq_4^A$ hold. Finally, in Section C.4.4 we will show that type B semantics is weaker than type A semantics, *i.e.* $\preceq_4^B \subset \preceq_4^A$ and $\models_4^B \subset \models_4^A$ hold.

Unfortunately, the above additional inference has a cost in terms of computational complexity. In fact, as it has been shown in [266], checking whether $\Sigma \models_4^A C(a)$ is harder than checking whether $\Sigma \models_4^B C(a)$ if Σ is a set of assertions.

Reasoning by cases (puzzle mode reasoning): *Reasoning by cases* does not work within our type A and type B semantics. Consider the following situation (see Figure 7.3). There is

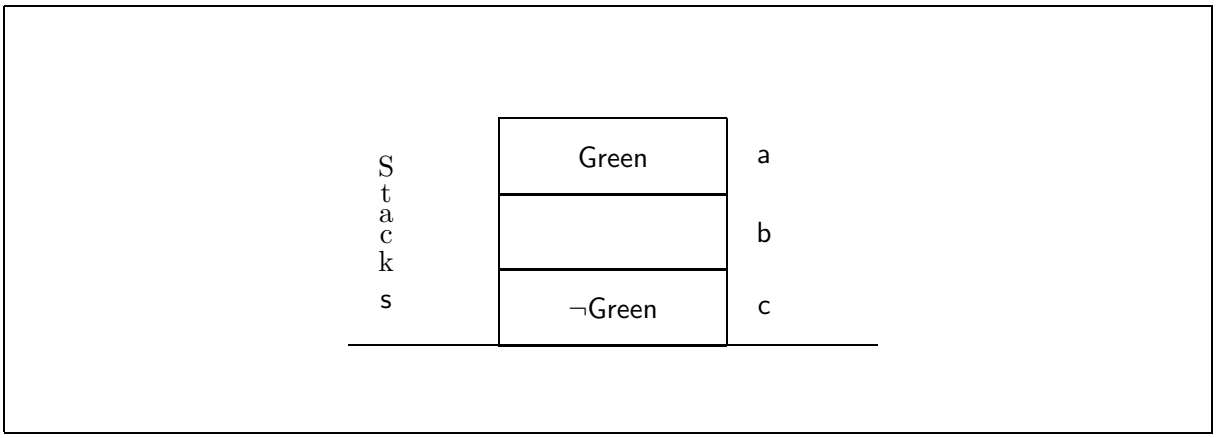


Figure 7.3: The block example.

a stack s with three blocks a, b and c. Block a is green, block c is not green and the colour of block b is unknown. Suppose we represent this situation by means of the following KB:

[3]Just consider an interpretation which maps $R(a,b)$ into $\{t,f\}$ and $C(b)$ into $\emptyset$.

$$\begin{aligned}\Sigma_B \;=\; & \{\mathsf{Onstack(s,a)}, \mathsf{Onstack(s,b)}, \mathsf{Onstack(s,c)},\\ & \mathsf{Block(a)}, \mathsf{Block(b)}, \mathsf{Block(c)},\\ & \mathsf{On(a,b)}, \mathsf{On(b,c)},\\ & \mathsf{Green(a)}, \neg\mathsf{Green(c)}\}.\end{aligned} \tag{7.76}$$

The question is: is there a green block on a non-green block in the stack s? We may represent the query A_B by means of the assertion

$$(\exists \mathsf{Onstack.Block} \sqcap \mathsf{Green} \sqcap (\exists \mathsf{On.}(\mathsf{Block} \sqcap \neg\mathsf{Green})))(\mathsf{s}). \tag{7.77}$$

By reasoning by cases (puzzle mode reasoning), it can be verified that $\Sigma_B \models_2 A_B$ holds. In fact, $\Sigma_B \models_2 A_B$ since in each two-valued interpretation $\mathcal{I}$ satisfying Σ_B, either $\mathsf{Green(b)}$ is true or $\mathsf{Green(b)}$ is false. In the former case, there is a green block (b) on a non-green block (c). In the latter case, there is green block (a) on a non-green block (b). Therefore, in both cases the query is satisfied.

On the other hand, $\Sigma_B \not\models_4^A A_B$ holds. In fact, $\Sigma_B \not\models_4^A A_B$ holds since it could be the case $\mathsf{Green}^{\mathcal{I}}(\mathsf{b}^{\mathcal{I}}) = \emptyset$. That is, we are uncertain about b's color, *i.e.* $\mathsf{Green(b)}$ is neither true nor false in $\mathcal{I}$, which blocks the preceding piece of reasoning. In other words, the absence of evidence supporting the truth of $\mathsf{Green(b)}$ and the simultaneous absence of evidence supporting the falsity of $\mathsf{Green(b)}$ prevents the inference from being drawn.

It is worth noting that there is no "top" concept within type A and type B semantics, *i.e.* there is no concept C such that $\emptyset \models_4^A C(a)$, for all individuals a. This plays an important role in the example above. In fact, suppose that we would admit a "top" concept $\top$ with semantics $[\![\top^{\mathcal{I}}]\!]^+ = \Delta^{\mathcal{I}}$ and $[\![\top^{\mathcal{I}}]\!]^- = \emptyset$. If we replace in Σ_B and A_B above, Green and $\neg\mathsf{Green}$ with $\exists\mathsf{R}.\top$ and $\forall\mathsf{R.A}$, respectively, then it can be verified that $\Sigma'_B \models_4^A A'_B$ and $\Sigma'_B \not\models_4^B A'_B$ hold, where Σ'_B and A'_B are the result of the substitutions. This is due to the fact that $\Sigma'_B \models_4^A A'_B$ relies on the relations $\mathsf{Green} \sqcup \neg\mathsf{Green} \equiv_4^A \top$ and $\exists\mathsf{R}.\top \sqcup \forall\mathsf{R.A} \equiv_4^A \top$, whereas $\exists\mathsf{R}.\top \sqcup \forall\mathsf{R.A} \not\equiv_4^B \top$. As it has been shown in [266], admitting a top concept changes the computational complexity of instance checking.

Example 16 Consider Example 11, *i.e.*

$$\begin{array}{lcl}
\mathsf{SportsKind} & \Rightarrow & \top \\
\mathsf{IndividualSports} & \Rightarrow & \mathsf{SportsKind} \\
\mathsf{TeamSports} & \Rightarrow & \mathsf{SportsKind} \\
\mathsf{SportsTool} & \Rightarrow & \top \\
\mathsf{Football} & \Rightarrow & \mathsf{SportsTool} \\
\mathsf{Basketball} & \Rightarrow & \mathsf{SportsTool} \\
\mathsf{TennisRacket} & \Rightarrow & \mathsf{SportsTool} \\
\mathsf{SportsMovie} & \Rightarrow & \top \\
\mathsf{TeamSportsMovie} & := & \mathsf{SportsMovie} \sqcap \\
 & & (\exists \mathsf{KindOfSports}.\top) \sqcap \\
 & & (\forall \mathsf{KindOfSports.TeamSports})
\end{array}$$

$$
\begin{array}{lcl}
\mathsf{IndividualSportsMovie} & := & \mathsf{SportsMovie} \sqcap \\
 & & (\exists \mathsf{KindOfSports}.\top) \sqcap \\
 & & (\forall \mathsf{KindOfSports}.\mathsf{IndividualSports}) \\
\mathsf{FootballMovie} & := & \mathsf{TeamSportsMovie} \sqcap \\
 & & (\exists \mathsf{HasSportsTool}.\top) \sqcap \\
 & & (\forall \mathsf{HasSportsTool}.\mathsf{Football}) \\
\mathsf{BasketMovie} & := & \mathsf{TeamSportsMovie} \sqcap \\
 & & (\exists \mathsf{HasSportsTool}.\top) \sqcap \\
 & & (\forall \mathsf{HasSportsTool}.\mathsf{Basket}) \\
\mathsf{TennisMovie} & := & \mathsf{IndividualSportsMovie} \sqcap \\
 & & (\exists \mathsf{HasSportsTool}.\top) \sqcap \\
 & & (\forall \mathsf{HasSportsTool}.\mathsf{TennisRacket})
\end{array}
$$

$(\exists \mathsf{Int\_As}.\mathsf{BaskedMovie})(\mathsf{o1})$,
$(\exists \mathsf{Int\_As}.\mathsf{TennisMovie})(\mathsf{o2})$.

It is quite easy to verify that

$$\Sigma \models_4^B \mathsf{BaskedMovie} \Rightarrow \mathsf{SportsMovie}$$

stating that a BaskedMovie is a SportsMovie. Similarly,

$$\Sigma \models_4^B \mathsf{TennisMovie} \Rightarrow \mathsf{SportsMovie}$$

holds. Therefore, given the query concept

$$\mathsf{Q} = \exists \mathsf{Int\_As}.\mathsf{SportsMovie},$$

$$\begin{array}{l}
\Sigma \models_4^B \mathsf{Q}(\mathsf{o1}), \text{ and} \\
\Sigma \models_4^B \mathsf{Q}(\mathsf{o2})
\end{array}$$

hold. It is worth noting that, given the query concept

$$\mathsf{Q}' = \exists \mathsf{Int\_As}.\mathsf{SportsMovie} \sqcap \exists \mathsf{KindOfSports}.\mathsf{IndividualSports}$$

it follows that

$$\begin{array}{l}
\Sigma \not\models_4^A \mathsf{Q}'(\mathsf{o1}), \\
\Sigma \not\models_4^B \mathsf{Q}'(\mathsf{o2}), \text{ whereas} \\
\Sigma \models_4^A \mathsf{Q}'(\mathsf{o2})
\end{array}$$

confirming the adequacy of $\models_4^A$.

■

To sum up, what kind of relevance relation is captured by $\models_4$? A first answer is that, roughly speaking, a KB Σ entails everything that is in the transitive closure of Σ by means of modus ponens on roles (in case of type A semantics) and the connectives $\sqcap, \sqcup, \neg, \exists$. All

other inferences are left out. More precisely, in order for $\Sigma \models_4 A$ to hold, the structural components of A must have an analogue in Σ, modulo modus ponens on roles (in case of type A semantics). A second answer is that a knowledge base Σ entails everything for which there is *explicit support* (or, we might say, everything for which there are *relevant premises*), where explicit support means the presence of supporting evidence rather than the absence of evidence supporting the contrary. A third, equally true but even more intuitive answer may be given by recalling that, in our last example, *at a first, superficial look* it would have seemed that there was no entailment between Σ and A. We may say that four-valued semantics models *superficial, shallow reasoning*, *i.e.* a mode of reasoning in which the agent only draws quick inferences that do not overtax her brain cells and cognitive resources. Those inferences that two-valued semantics licenses and four-valued semantics does not are those for drawing which the agent must reason, as Levesque says [183], in *puzzle mode*, *i.e.* in the style that we adopt once we try to prove a challenging mathematical problem or a logic puzzle. This interpretation clearly shows that the four-valued setting of $\mathcal{ALC}$, rather than two-valued $\mathcal{ALC}$, is the interesting tool for document retrieval, unless we are prepared to defend the thesis that the user of an IR system reasons in "puzzle mode" when she must decide whether a retrieved document is relevant or not to her query.

7.4.2 The logic HORN-$\mathcal{ALC}$

Although $\mathcal{ALC}$ is an expressive DL, we will extend it from an expressive power point of view, as it will be of use in our multimedia context. In particular, we extend $\mathcal{ALC}$ with horn rules following the line of CARIN [186, 187, 188] (see [29, 97] for similar extensions). Essentially, this section is an adaption to the $\mathcal{ALC}$ case of Section 7.3.3.

Expressions of the logic HORN-$\mathcal{ALC}$ are defined as follows. Consider a new alphabet of n-ary predicates, called *ordinary predicates*. A HORN-$\mathcal{ALC}$ *predicate* (denoted by P) is either a primitive concept, a role or an ordinary predicate.

Consider a new alphabet of *horn variables* (denoted by X, Y, Z).

A *horn rule* (denoted by R) is an expression of the form

$$P(\vec{X}) \leftarrow P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n), \tag{7.78}$$

where $n \geq 1$ and $\vec{X}_1, \ldots, \vec{X}_n, \vec{X}$ are tuples of horn variables or individuals. Of course, horn rules are universally quantified. We require that a horn variable which appears in $\vec{X}$ also appears in $\vec{X}_1, \ldots, \vec{X}_n$. The predicates $P, P_1, \ldots P_n$ are HORN-$\mathcal{ALC}$ predicates. $P(\vec{X})$ is called *head* and $P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$ is called *body*.

For instance, suppose that we have the following specialisation

$$\mathsf{SuperVisor} := \mathsf{Professor} \sqcap \exists \mathsf{Teaching.AdvancedCourse}$$

then the horn rule

$$\mathsf{MayDoThesis}(X, Y, Z) \leftarrow \mathsf{Student}(X), \mathsf{SuperVisor}(Y), \mathsf{Expert}(Y, Z)$$

establishes that a student X may do thesis about Z with a professor Y, teaching an advanced course, which is Expert on topic Z. Here, MayDoThesis and Expert are ordinary predicates.

A *fact* (denoted by $P(\vec{a})$) is either an $\mathcal{ALC}$ assertion (where an $\mathcal{ALC}$ variable can occur) or a ground instance of some ordinary predicate. It is worth noting that *e.g.* (Professor $\sqcap$

$\exists$Teaching.Course)(paul) and Expert(paul, kr) are facts: we do not limit ourself to facts which are horn. It is worth anticipating that the semantics of a complex multimedia object will be described through a set of facts.

A *goal* is an expression of the form

$$\leftarrow P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n) \tag{7.79}$$

with $n \geq 0$ and $P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$ are HORN-$\mathcal{ALC}$ predicates. The case $n = 0$ is called *empty goal* and is indicated with $\leftarrow \blacksquare$.

A *query* is an expression of the form

$$\exists \vec{X}.P_1(\vec{X}_1) \wedge \ldots \wedge P_n(\vec{X}_n), \tag{7.80}$$

with $n \geq 1$, $\vec{X}$ is the tuple of variables appearing in $\vec{X}_1, \ldots, \vec{X}_n$ and $P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$ HORN-$\mathcal{ALC}$ predicates.

A HORN-$\mathcal{ALC}$ KB Σ is a finite set of $\mathcal{ALC}$ specialisations and definitions, horn rules and facts. With $\Sigma_T \subseteq \Sigma$ we indicate the set of specialisations and definitions in Σ, with $\Sigma_R \subseteq \Sigma$ we indicate the set of horn rules in Σ and with $\Sigma_F \subseteq \Sigma$ we indicate the set of facts in Σ. A *horn* KB is a finite set of horn rules and horn facts. Note that in a horn KB Σ, the terminology Σ_T is empty.

We will say that a HORN-$\mathcal{ALC}$ KB Σ is *safe* iff for all rules $P(\vec{X}) \leftarrow P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$ in Σ_R, P is an ordinary predicate. It should be noted that a safe HORN-$\mathcal{ALC}$ KB Σ does not allow primitive concepts and roles to appear in the head of any rule $R \in \Sigma_R$ because of the underlying assumption that the terminological component *completely* describes the hierarchical structure of classes in the domain, and thus, the horn rules should not allow to make new inferences about that structure.

Finally, the definition of *recursive* HORN-$\mathcal{ALC}$ KBs Σ is as usual: for all horn rules $P(\vec{X}) \leftarrow P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n) \in \Sigma$ we will say that P *directly uses* P_i for all $1 \leq i \leq n$. Let *uses* be the transitive closure of the relation directly uses in Σ. We will say that a HORN-$\mathcal{ALC}$ KB Σ is *recursive* iff there is P such that P uses P trough rules in Σ. The KBs which will be of our interest are the *safe and non recursive* HORN-$\mathcal{ALC}$ KBs.

From a semantics point of view, we extend the semantics of horn rules defined in Section 7.3.3. Let $\mathcal{I}$ be an interpretation. $\mathcal{I}$ maps horn variables into elements of the domain $\Delta^{\mathcal{I}}$ and maps a n-ary ordinary predicate P into a function from $(\Delta^{\mathcal{I}})^n$ into $2^{\{t,f\}}$. Let $\vec{X} = (X_1, \ldots, X_n)$ be a tuple of distinct horn variables and let $\vec{d} = (d_1, \ldots, d_n)$ be a tuple of elements of the domain $\Delta^{\mathcal{I}}$. The interpretation $\mathcal{I}^{\vec{d}}_{\vec{X}}$ is as $\mathcal{I}$, except that each variable X_i in $\vec{X}$ is mapped into d_i, *i.e.*

$$Y^{\mathcal{I}^{\vec{d}}_{\vec{X}}} = \begin{cases} Y^{\mathcal{I}} & \text{if for all } i,\ Y \neq X_i; \\ d_i & \text{if for some } i,\ Y = X_i. \end{cases} \tag{7.81}$$

We will say that an interpretation $\mathcal{I}$

1. *satisfies* a horn rule $P(\vec{X}) \leftarrow P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$ iff for all $\vec{d} \in (\Delta^{\mathcal{I}})^k$, if $\mathcal{I}^{\vec{d}}_{\vec{Y}}$ does satisfy all $P_1(\vec{X}_1)$, ..., $P_n(\vec{X}_n)$, then $\mathcal{I}^{\vec{d}}_{\vec{Y}}$ does satisfy $P(\vec{X})$, where $\vec{Y}$ is the tuple of all the k variables occurring in the rule;

2. *satisfies* a goal $\leftarrow P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$ iff $n \geq 1$ and for all $\vec{d} \in (\Delta^{\mathcal{I}})^k$, $\mathcal{I}^{\vec{d}}_{\vec{Y}}$ does not satisfy some $P_i(\vec{X}_i)$, $1 \leq i \leq n$, where $\vec{Y}$ is the tuple of all the k variables occurring in the goal;

3. *satisfies* a query $\exists \vec{X}.P_1(\vec{X}_1) \wedge \ldots \wedge P_n(\vec{X}_n)$ iff for some $\vec{d} \in (\Delta^{\mathcal{I}})^k$, $\mathcal{I}^{\vec{d}}_{\vec{X}}$ does satisfy all $P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n))$, where $\vec{X}$ is the tuple of all the k variables occurring in the query.

Note that the empty goal is never satisfied.

An interpretation $\mathcal{I}$ *satisfies* (*is a model of*) a HORN-$\mathcal{ALC}$ KB Σ iff $\mathcal{I}$ satisfies each element of it. Satisfiability is extended to an arbitrary set S of horn rules, facts and goals, as usual. Finally, a HORN-$\mathcal{ALC}$ KB Σ *entails* a query $Q = \exists \vec{X}.P_1(\vec{X}_1) \wedge \ldots \wedge P_n(\vec{X}_n)$ (denoted by $\Sigma \models_4 Q$) iff every model of Σ satisfies Q.

Let $Q = \exists \vec{X}.P_1(\vec{X}_1) \wedge \ldots \wedge P_n(\vec{X}_n)$ be a query. The *associated goal* of Q (denoted by G_Q) is $\leftarrow P_1(\vec{x}_1), \ldots, P_n(\vec{x}_n)$. It is easily verified that

$$\Sigma \models_4 Q \quad \text{iff} \quad \Sigma \cup \{G_Q\} \text{ not satisfiable.} \tag{7.82}$$

Finally, an *answer* to the query Q is a substitution θ of all variables in Q. An answer is *correct* w.r.t. a HORN-$\mathcal{ALC}$ KB Σ iff $\Sigma \models_4 Q\theta$, where $Q\theta$ is $(P_1(\vec{X}_1) \wedge \ldots \wedge P_n(\vec{X}_n))\theta$[4]. As usual, the *answer set* of a query Q w.r.t. a KB Σ (denoted by $AnswerSet(\Sigma, Q)$) is the set of correct answers, *i.e.*

$$AnswerSet(\Sigma, Q) \quad = \quad \{\theta : \Sigma \mid\!\approx_4 Q\theta\}. \tag{7.83}$$

As for Proposition 4, in case of horn KBs, equivalence between four-valued entailment and two-valued entailment can be established.

Proposition 5 *In* HORN-$\mathcal{ALC}$*, let* Σ *be a horn KB. Let* Q *be a query. Then* $\Sigma \models_4 Q$ *iff* $\Sigma \models_2 Q$. ⊣

As pointed out in Section 7.3.3, the proposition is not true in case of general HORN-$\mathcal{ALC}$ KBs with empty terminology. Moreover, the proposition is not true in case of not empty terminology. For instance, consider $\Sigma = \{A \Rightarrow B, \neg A \Rightarrow B\}$. It is easily verified that $\Sigma \models_2 B(a)$, whereas $\Sigma \not\models_4 B(a)$ (note that even $\Sigma_R = \emptyset$).

Example 17 Consider the following simple HORN-$\mathcal{ALC}$KB.

$$\begin{aligned}\Sigma \quad = \quad & \{\mathsf{SuperVisor} := \mathsf{Professor} \sqcap \exists \mathsf{Teaching}.\mathsf{AdvancedCourse},\\ & \\ & \mathsf{MayDoThesis}(X, Y, Z) \leftarrow \mathsf{Student}(X), \mathsf{SuperVisor}(Y), \mathsf{Expert}(Y, Z),\\ & \\ & \mathsf{Professor}(\mathsf{paul}), \mathsf{Teaching}(\mathsf{paul}, \mathsf{ai}), \mathsf{AdvancedCourse}(\mathsf{ai}),\\ & \mathsf{Student}(\mathsf{tom}), \mathsf{Expert}(\mathsf{paul}, \mathsf{kr})\}\end{aligned}$$

Consider the query

$$Q \quad = \quad \exists Y, Z.\mathsf{MayDoThesis}(\mathsf{tom}, Y, Z)$$

[4]The definition of satisfiability for expressions of the form $Q\theta$ is obvious.

i.e. we are asking whether tom is doing a thesis. It can be verified that

$$\Sigma \models_4 \exists Y, Z.\mathsf{MayDoThesis}(\mathsf{tom}, Y, Z)$$

holds and a correct answer is $\theta = \{Y/\mathsf{paul}, Z/\mathsf{kr}\}$.

■

Chapter 8

A four-valued fuzzy horn description logic

8.1 Introduction

The logic we have described so far is still insufficient for describing *real* retrieval situations, as retrieval is usually not only a yes-no question:

- the representations of documents and queries which the system (and the logic) have access to are inherently imperfect;
- and the relevance of a document to a query can thus be established only up to a limited degree of imprecision.

Because of this, we need a framework in which, rather than deciding *tout court* whether a document satisfies a query or not, we are able to *rank* documents according to how strongly the system believes in their relevance to a query. To this end, we will extend our DL with *fuzzy assertions*.

Fuzzy assertions take inspiration from Zadeh's work on fuzzy sets [113, 239, 290, 291, 292, 298]. A *fuzzy set* A with respect to a set X is characterized by a *membership function* $\mu_A : X \to [0,1]$, assigning an A-membership degree, $\mu_A(x)$, to each element x in X. This membership degree gives us an estimation of the belonging of x to A. Typically, if $\mu_A(x) = 1$ then x definitely belongs to A, while $\mu_A(x) = 0.8$ means that x is "likely" to be an element of A. Moreover, according to Zadeh, the membership function has to satisfy three well-known restrictions, for all $x \in X$ and for all fuzzy sets A, B with respect to X:

$$\begin{aligned} \mu_{A\cap B}(x) &= \min\{\mu_A(x), \mu_B(x)\}, \\ \mu_{A\cup B}(x) &= \max\{\mu_A(x), \mu_B(x)\}, \text{ and} \\ \mu_{\overline{A}}(x) &= 1 - \mu_A(x), \end{aligned}$$

where $\overline{A}$ is the complement of A in X. Other membership functions have been proposed, but it is not our aim to investigate them here (the interested reader may consult *e.g.* [105, 170, 292]).

When we switch to logic, and to DLs in particular, we have concepts which are fuzzy sets and, thus, speak about of membership degrees.

For instance, the assertion that individual a is an instance of concept C, formally denoted by $\mathsf{C}(\mathsf{a})$, may have as a degree of membership any real number in between 0 and 1: if the degree of membership of $\mathsf{C}(\mathsf{a})$ is 1, then a is definitely an instance of C, while if the degree of membership of $\mathsf{C}(\mathsf{a})$ is 0.8 then a is likely to be an instance of C. Similarly for role assertions. Hence, in a fuzzy DL, concepts become *imprecise* (or *vague*). As a consequence, given *e.g.* a query concept Q, the retrieval process produces a ranking of individuals: the rank of a, for each individual a, is the degree of membership of $\mathsf{Q}(\mathsf{a})$, and will be interpreted as the degree of relevance of the document identified by a to the query Q.

The choice of fuzzy set theory as a way of endowing a DL with the capability to deal with imprecision is not uncommon [84, 295, 267] and can be motivated both from the syntactical and the semantical point of view.

- From a semantical point of view, fuzzy logics capture the notion of vague concept, that is a concept that is intrinsically *imprecise* and for which a clear and precise definition is not possible. For instance, "hot" and "tall" are vague concepts. The key fact about vague concepts is that while they are not well defined, assertions involving them may be quite well defined. For instance, the boundaries of the Mount Everest are ill-defined, whereas the assertion stating that the Mount Everest is the highest mountain of the world is clearly definite, and its definiteness is not compromised by the ill-definiteness of the exact boundaries of the mountain. It is easy to see that fuzzy assertions play a key role in meaning descriptions of documents (most of human's concepts are vague). For example, in the context of images, the semantics of an image region r may be described by means of a fuzzy assertion like "r represents the Mount Everest with degree 0.8".

- From a proof theoretical point of view, there exist well-known techniques for reasoning in fuzzy logics (see *e.g.* [78, 156, 179, 195, 290]). This is not the case for alternative logics, such as, for instance, probabilistic DLs [146, 157, 252]. In particular, [142, 143] shows that probabilistic reasoning is computationally more difficult than non-probabilistic reasoning, and in most cases a complete axiomatization is missing.

Fuzzy logic is not appropriate to deal with *uncertain assertions*, that is assertions which are only true or false, but, due to the lack of precision of the available information, one can only estimate to what extent it is possible or necessary that they are true. For instance, "line", and "polygon" are precise concepts, but due to the lack of precision of the available information we may only be able to estimate to what degree a certain object in an image is *e.g.* a polygon. The logics dealing with this kind of uncertainty have been called *Possibilistic Logics* [103, 106, 108, 109, 153, 178]. Possibilistic DLs are discussed in [149].

The combination of possibilistic and fuzzy logic would lead to the treatment of *uncertain fuzzy assertions*, *i.e.* fuzzy assertions for which the available reference information is not precise. While this combination is possible, and may even be desirable for retrieval purposes, our model only provides fuzzy assertions. A DL allowing uncertain fuzzy assertions can be obtained by combining the approach in [149] with our logic.

The Chapter is organised as follows. At first, in the following sections, we present the four-valued fuzzy propositional logic $\mathcal{L}^f$ which extends $\mathcal{L}$ to the fuzzy case: syntax, semantics and examples will be given. In Section 8.3 we extend the propositional case to the first-order case. In particular, we will specify the extension of $\mathcal{ALC}$ to the fuzzy case. Finally, in Section 7.4.2 our final logic will be presented, *i.e.* fuzzy Horn-$\mathcal{ALC}$. Essentially, fuzzy Horn-$\mathcal{ALC}$ extends Horn-$\mathcal{ALC}$ by dealing with imprecision.

8.2 Preliminaries: four-valued fuzzy propositional logic

Following Chapter 7, we start first with propositional logic. We first recall briefly Zadeh's standard two-valued fuzzy propositional logic [78, 179] and, thereafter, we present the four-valued counterpart.

8.2.1 The logic $\mathcal{L}^f$ and its properties

In a two-valued semantics framework (see [78, 179]), a *fuzzy valuation* is a function $|\cdot|$ mapping propositions of $\mathcal{L}$ into $[0,1]$. $|A|$ will naturally be interpreted as the *degree of truth* of A. Following Zadeh, $|\cdot|$ has also to satisfy the well known equations:

$$\begin{array}{lcl} |A \wedge B| & = & \min\{|A|, |B|\}, \\ |A \vee B| & = & \max\{|A|, |B|\}, \text{ and} \\ |\neg A| & = & 1 - |A|. \end{array} \tag{8.1}$$

Switching to the four-valued case, consistently with our approach of distinguishing explicit from implicit falsehood (*i.e.* distinguishing $f \in A^{\mathcal{I}}$ from $t \notin A^{\mathcal{I}}$), we will use rather two fuzzy valuations, $|\cdot|^t$ and $|\cdot|^f$: $|A|^t$ will naturally be interpreted as the *degree of truth* of A, whereas $|A|^f$ will analogously be interpreted as the *degree of falsity* of A. Whereas in the classical "two-valued" fuzzy case, $|\cdot|^t$ and $|\cdot|^f$ are such that $|A|^f = 1 - |A|^t$, for each A, we might well have $|A|^t = .6$ and $|A|^f = .8$. This is a natural consequence of our four-valued approach.

Now, a *fuzzy proposition* is an expression of type $(A \geq n)$, where A is a proposition in $\mathcal{L}$ and $n \in [0,1]$; $\mathcal{L}^f$ is just the set of fuzzy propositions. We will use γ as metavariable for fuzzy propositions. For instance, $(\mathsf{ItsCold} \geq .7)$ is a fuzzy proposition and its intended meaning is that the degree of truth of $\mathsf{ItsCold}$ is at least .7, while $(\mathsf{ItsCold} \geq 1)$ means that it is definitely cold. On the other hand $(\neg\mathsf{ItsCold} \geq .7)$ means that it is likely to be not cold, while $(\neg\mathsf{ItsCold} \geq 1)$ may be interpreted as saying that it is definitely not cold.

A *fuzzy interpretation*[1] $\mathcal{I}$ is a triple $\mathcal{I} = ((\cdot)^{\mathcal{I}}, |\cdot|^t, |\cdot|^f)$, where $|\cdot|^t$ and $|\cdot|^f$ are fuzzy valuations and $(\cdot)^{\mathcal{I}}$ maps each fuzzy proposition into an element of $2^{\{t,f\}}$. Additionally, $(\cdot)^{\mathcal{I}}, |\cdot|^t$ and $|\cdot|^f$ have to satisfy the following equations

$$\begin{array}{lcl} |A \wedge B|^t & = & \min\{|A|^t, |B|^t\} \\ |A \wedge B|^f & = & \max\{|A|^f, |B|^f\} \\ \\ |A \vee B|^t & = & \max\{|A|^t, |B|^t\} \\ |A \vee B|^f & = & \min\{|A|^f, |B|^f\} \\ \\ |\neg A|^t & = & |A|^f \\ |\neg A|^f & = & |A|^t \end{array} \tag{8.2}$$

which are the four-valued counterpart of Equations (8.1), and

$$\begin{array}{lll} t \in (A \geq n)^{\mathcal{I}} & \text{iff} & |A|^t \geq n \\ f \in (A \geq n)^{\mathcal{I}} & \text{iff} & |A|^f \geq n. \end{array} \tag{8.3}$$

It is easy to see that, *e.g.* $|A \wedge B|^t = |\neg A \vee \neg B|^f$. Similarly for $|A \vee B|^t$.

[1] In the following called interpretation.

It is worth noting that there is a simple connection between the four-valued semantics given in Section 7.3 and the fuzzy counterpart. In fact, the above conditions can be reformulated as *e.g.*

$$\begin{array}{ll} t \in (A \wedge B \geq n)^{\mathcal{I}} & \text{iff} \quad t \in (A \geq n)^{\mathcal{I}} \text{ and } t \in (B \geq n)^{\mathcal{I}}; \\ f \in (A \wedge B \geq n)^{\mathcal{I}} & \text{iff} \quad f \in (A \geq n)^{\mathcal{I}} \text{ or } f \in (B \geq n)^{\mathcal{I}} \end{array} \tag{8.4}$$

(the other cases, $\vee$ and $\neg$ are quite similar). Just note that if both $|A|^f = 1 - |A|^t$ and $(A \geq n)^{\mathcal{I}} \in \{\{t\}, \{f\}\}$, classical "two-valued" fuzzy logic is obtained.

Fuzzy satisfiability, fuzzy equivalence and fuzzy entailment are defined as the natural extensions of the non fuzzy case. Let $\mathcal{I}$ be an interpretation, let $(A \geq n), (B \geq m)$ be two fuzzy propositions and let Σ be a set of fuzzy propositions: $\mathcal{I}$ *satisfies* (*is a model of*) $(A \geq n)$ iff $t \in (A \geq n)^{\mathcal{I}}$; $(A \geq n)$ and $(B \geq m)$ are *equivalent* (denoted by $(A \geq n) \approx_4 (B \geq m)$) iff they have the same models; $\mathcal{I}$ *satisfies* (*is a model of*) Σ iff $\mathcal{I}$ is a model of $(A \geq n)$, for all $(A \geq n) \in \Sigma$; Σ *entails* $(A \geq n)$ (denoted by $\Sigma \mid\!\!\approx_4 (A \geq n)$) iff all models of Σ are models of $(A \geq n)$. Since $\mid\!\!\approx_4 (A \geq 0)$, we will not consider those $(A \geq n)$ for $n = 0$.

Given a KB Σ and a proposition A, we define the *maximal degree of truth* of A with respect to Σ (denoted by $Maxdeg(\Sigma, A)$) to be $\max\{n > 0 : \Sigma \mid\!\!\approx_4 (A \geq n)\}$ ($\max \emptyset = 0$). Notice that $\Sigma \mid\!\!\approx_4 (A \geq n)$ iff $Maxdeg(\Sigma, A) \geq n$.

A fuzzy proposition $(A \geq n)$ is in *Negation Normal Form* – respectively, in *Conjunctive Normal Form* (CNF) – iff A is in NNF – respectively in CNF. As for $\mathcal{L}$, without loss of generality, we can restrict our attention to fuzzy propositions in NNF only. In fact, it is easy to verify that the corresponding fuzzy version of the relations (7.16) – (7.18) hold too, *e.g.*

$$(\neg(A \vee B) \geq n) \quad \approx_4 \quad (\neg A \wedge \neg B \geq n) \tag{8.5}$$

Thus,

$$(A \geq n) \quad \approx_4 \quad (A_{NNF} \geq n) \tag{8.6}$$

Similarly for the CNF case: it can easily be verified that

$$(A \geq n) \quad \approx_4 \quad (A_{CNF} \geq n) \tag{8.7}$$

There is a strict relation between fuzzy propositions and propositions. Given a KB Σ, let $\overline{\Sigma}$ be the (crisp) KB

$$\overline{\Sigma} = \{A : (A \geq n) \in \Sigma\}. \tag{8.8}$$

Proposition 6 *Let $\Sigma \subseteq \mathcal{L}^f$ and $A \in \mathcal{L}$. For all $n > 0$, if $\Sigma \mid\!\!\approx_4 (A \geq n)$ then $\overline{\Sigma} \models_4 A$.* ⊣

Proof: Assume $\Sigma \mid\!\!\approx_4 (A \geq n)$. Now, suppose to the contrary that $\overline{\Sigma} \not\models_4 A$. Therefore there is a (four-valued) model $\mathcal{I}$ of $\overline{\Sigma}$ not being a model of A. Let $\tilde{\mathcal{I}} = ((\cdot)^{\mathcal{I}}, |\cdot|^t, |\cdot|^f)$ be the following fuzzy interpretation. For all propositional letters B,

$$\begin{array}{ll} |B|^t = 1 & \text{if } t \in B^{\mathcal{I}} \\ |B|^t = 0 & \text{if } t \notin B^{\mathcal{I}} \\ |B|^f = 1 & \text{if } f \in B^{\mathcal{I}} \\ |B|^f = 0 & \text{if } f \notin B^{\mathcal{I}} \end{array}$$

It is easy to see that $\tilde{\mathcal{I}}$ is in effect a fuzzy interpretation. Now, we will show that $\tilde{\mathcal{I}}$ is a fuzzy interpretation satisfying Σ, but not satisfying $(A \geq n)$ which is contrary to our assumption.

First of all, we show on induction on the number of connectives of B that if $(B \geq m) \in \Sigma$ then $t \in (B \geq m)^{\tilde{\mathcal{I}}}$ and, thus, $\tilde{\mathcal{I}}$ satisfies Σ.

B is a literal: If $(B \geq m) \in \Sigma$ and B is a letter, then $B \in \overline{\Sigma}$ follows and, thus, $t \in B^{\mathcal{I}}$. By definition of $\tilde{\mathcal{I}}$, $|B|^t = 1 \geq m$. Hence, $t \in (B \geq m)^{\tilde{\mathcal{I}}}$. If B is $(\neg p \geq m) \in \Sigma$, then $\neg p \in \overline{\Sigma}$ follows and, thus, $f \in p^{\mathcal{I}}$. By definition of $\tilde{\mathcal{I}}$, $|p|^f = 1 \geq m$. Hence, $t \in (\neg p \geq m)^{\tilde{\mathcal{I}}}$.

Induction step: Suppose B is $(C \wedge D \geq m) \in \Sigma$. Therefore, $C \wedge D \in \overline{\Sigma}$ and $t \in C \wedge D^{\mathcal{I}}$. Therefore, $t \in C^{\mathcal{I}}$ and $t \in D^{\mathcal{I}}$. By induction on C and D, $t \in (C \geq m)^{\tilde{\mathcal{I}}}$ and $t \in (D \geq m)^{\tilde{\mathcal{I}}}$. Hence, $t \in (C \wedge D \geq m)^{\tilde{\mathcal{I}}}$. The case B is $(C \vee D \geq m) \in \Sigma$ is similar.

Therefore, $\tilde{\mathcal{I}}$ satisfies Σ.

By induction on the number of connectives of A we show that if $t \notin A^{\mathcal{I}}$ then $|A|^t = 0$ and, thus, $\tilde{\mathcal{I}}$ does not satisfy $(A \geq n)$.

A is a literal: If A is a letter, then $t \notin A^{\mathcal{I}}$ implies $|A|^t = 0$, by definition. Otherwise, A is $\neg p$. $t \notin \neg p^{\mathcal{I}}$ implies $f \notin p^{\mathcal{I}}$. Therefore, we have $|p|^f = 0$, by definition. Hence, $|A|^t = |\neg p|^t = 0$.

Induction step: If A is $B \wedge C$ then $t \notin B \wedge C^{\mathcal{I}}$ implies $t \notin B^{\mathcal{I}}$ or $t \notin C^{\mathcal{I}}$. By induction, $|B|^t = 0$ or $|C|^t = 0$. Therefore, $|A|^t = \min\{|B|^t, |C|^t\} = 0$. The case A is $B \vee C$ is similar.

Q.E.D.

Proposition 6 states that *there cannot be fuzzy entailment without entailment.* For instance,

$$\{(p \geq .7), (\neg p \geq .5)\} \not\approx_4 (q \geq n), \text{ for all } n > 0.$$

In fact, consider an interpretation $\mathcal{I}$ such that $|p|^t = .7$, $|p|^f = .5$, $|q|^f = 0$ and $|q|^t = \frac{n}{2}$.

Example 18 Let Σ be the set

$$\Sigma = \{(p \geq .1), (p \wedge q \geq .5), (q \vee r \geq .6)\}$$

Let A be $p \vee r$. One may check that $\Sigma \not\approx_4 (A \geq .5)$ and $Maxdeg(\Sigma, A) = .5$. $\overline{\Sigma}$ is

$$\overline{\Sigma} = \{p, p \wedge q, q \vee r\}$$

and $\overline{\Sigma} \models_4 A$ is easily verified, thereby confirming Proposition 6. ■

A simple version of the converse of Proposition 6 states the following:

Proposition 7 *Let $A, B \in \mathcal{L}$. If $A \models_4 B$ then for all $n > 0$ $(A \geq n) \not\approx_4 (B \geq n)$.* ⊣

Proof: Assume $A \models_4 B$. Suppose to the contrary that for some $n > 0$, $(A \geq n) \not\approx_4 (B \geq n)$. Therefore, there is a (fuzzy) interpretation $\mathcal{I} = ((\cdot)^{\mathcal{I}}, |\cdot|^t, |\cdot|^f)$ such that $|A|^t \geq n$ and $|B|^t < n$. Let $\tilde{\mathcal{I}}$ be the following four-valued interpretation: for all letters p

$$t \in p^{\tilde{\mathcal{I}}} \quad \text{iff } |p|^t \geq n$$
$$f \in p^{\tilde{\mathcal{I}}} \quad \text{iff } |p|^f \geq n.$$

Since $|A|^t \geq n$, it can easily be verified, by induction on the number of connectives in A, that $t \in A^{\mathcal{I}}$. In a similar way, it can be verified that from $|B|^t < n$, $t \notin B^{\mathcal{I}}$ follows. Therefore, $A \not\models\!\!\!\approx_4 B$, contrary to our assumption. Q.E.D.

The following proposition can easily be shown.

Proposition 8 *Let $A, B \in \mathcal{L}$. For all $n > 0$, Maxdeg($\{(A \geq n)\}, B$) $= n$ iff $A \models_4 B$.* ⊣

Proof:

⇒ .) If Maxdeg($\{(A \geq n)\}, B$) $= n$, then $(A \geq n) \not\approx_4 (B \geq n)$, by definition. Therefore, from Proposition 6, $A \models_4 B$.

⇐ .) If $A \models_4 B$ then by Proposition 7, $(A \geq n) \not\approx_4 (B \geq n)$ follows. Therefore, Maxdeg($\{$ $(A \geq n)$ $\}$, B) $\geq n$. Let $\tilde{\mathcal{I}}$ be the following fuzzy interpretation: for all letters p

$$|p|^t = |p|^f = n$$

$\tilde{\mathcal{I}}$ satisfies both $(A \geq n)$ and $(B \geq n)$, but $\tilde{\mathcal{I}}$ does not satisfy any $(B \geq m)$ with $m > n$. Therefore it cannot be the case $(A \geq n) \not\approx_4 (B \geq m)$, for some $m > n$. Hence, Maxdeg($\{(A \geq n)\}, B$) $= n$. Q.E.D.

Proposition 9 *Let $A, B \in \mathcal{L}$. The following hold: for all $n > 0$*

1. *$(A \geq n) \not\approx_4 (B \geq n)$ iff $(\neg B \geq n) \not\approx_4 (\neg A \geq n)$;*
2. *$(A \geq n) \approx_4 (B \geq n)$ iff $A \equiv_4 B$.* ⊣

Proof: The proof is simple. By using Propositions 1, 6 and 7 we have

1. $(A \geq n) \not\approx_4 (B \geq n)$ iff $A \models_4 B$ iff $\neg B \models_4 \neg A$ iff $(\neg B \geq n) \not\approx_4 (\neg A \geq n)$;
2. $(A \geq n) \approx_4 (B \geq n)$ iff $((A \geq n) \not\approx_4 (B \geq n)$ and $(B \geq n) \not\approx_4 (A \geq n))$ iff $(A \models_4 B$ and $B \models_4 A)$ iff $A \equiv_4 B$

Q.E.D.

Given these strict analogies between the fuzzy case and the not fuzzy case, most properties are the fuzzy analogous of those of $\mathcal{L}$ ((7.19) − (7.27)) and are resumed here for readability:

$$(A \geq n) \not\approx_4 (A \geq m), \quad \forall n \geq m \tag{8.9}$$
$$(A \geq n) \not\approx_4 (A_{CNF} \geq n) \text{ and } (A_{CNF} \geq n) \not\approx_4 (A \geq n) \tag{8.10}$$
$$(A \geq n) \not\approx_4 (B \geq l) \text{ and } (B \geq l) \not\approx_4 (C \geq m) \text{ implies } (A \geq n) \not\approx_4 (C \geq m) \tag{8.11}$$
$$(A \wedge B \geq n) \not\approx_4 (A \geq n) \text{ and } (A \geq n) \not\approx_4 (A \vee B \geq n) \tag{8.12}$$

Similarly as for $\mathcal{L}$, the above Equations (8.9), (8.10), (8.11) and (8.12) constitutes a simple axiomatization of $\mathcal{L}^f$. Additionally,

$$(A \geq n), (\neg A \vee B \geq m) \not\approx_4 (B \geq k),\ \forall k > 0 \text{ (no modus ponens)} \quad (8.13)$$
$$|A|^t = n, |A|^f = m, |B|^t = |B|^f = 0$$
$$(A \geq n) \not\approx_4 (B \vee \neg B \geq m),\ \forall m > 0 \ (i.e. \text{ no tautologies}) \quad (8.14)$$
$$|A|^t = |A|^f = n, |B|^t = |B|^f = 0$$
$$(A \wedge \neg A \geq n) \not\approx_4 (B \geq m),\ \forall m > 0 \ (i.e. \text{ every KB is satisfiable}) \quad (8.15)$$
$$|A|^t = |A|^f = n, |B|^t = |B|^f = 0$$
$$\Sigma \not\approx_4 (A \geq n) \text{ implies } \Sigma \not\approx_2 (A \geq n) \quad (8.16)$$
every two-valued fuzzy model is a four-valued fuzzy model

hold.

8.2.2 The logic $\mathcal{L}^f_+$ and its properties

In this section we extend $\mathcal{L}^f$ to the case where a form of modus ponens is allowed.

From a syntax point of view, we extend $\mathcal{L}^f$ to $\mathcal{L}^f_+$, where $\mathcal{L}^f_+$ is $\mathcal{L}^f$ union the set of fuzzy proposition of type $(A \rightarrow B \geq n)$, where $A, B \in \mathcal{L}^f$ and $n \in [0, 1]$. It is worth noting that no nesting of $\rightarrow$ is allowed.

From a semantics point of view, the restriction to the fuzzy valuation functions is given as a natural extension to the four-valued case of Zadeh's view of implication: in a two-valued setting, we have

$$|A \rightarrow B| = \max\{1 - |A|, |B|\} \quad (8.17)$$

which is motivated by viewing $A \rightarrow B$ as $\neg A \vee B$. Hence, in case of a four-valued setting, the restrictions which the fuzzy valuation functions, $|\cdot|^t$ and $|\cdot|^f$, have to satisfy are:

$$|A \rightarrow B|^t = \max\{1 - |A|^t, |B|^t\} \quad (8.18)$$
$$|A \rightarrow B|^f = \min\{|A|^t, |\neg B|^t\} \quad (8.19)$$

Equation (8.18) enables us a simple form of modus ponens:

$$(A \geq m), (A \rightarrow B \geq n) \not\approx_4 (B \geq n) \text{ if } m > 1 - n \quad (8.20)$$

whereas Equation (8.19) establishes the obvious equivalence

$$(\neg(A \rightarrow B) \geq n) \approx_4 (A \wedge \neg B \geq n) \quad (8.21)$$

Equation (8.20) is certainly not the unique definition which can be given for the $\rightarrow$ connective. In the literature, a long list of alternative definitions can be found (see *e.g.* [170]) and a snapshot is given in Table 8.1 ($|\neg A|$ is always $1 - |A|$).
The discussion about which of them is the most appropriate one in our context is beyond our purposes.

Type	$\|A \to B\|$	Motivation
Zadeh	$\max\{1 - \|A\|, \|B\|\}$	$A \to B = \neg A \vee B$
Gougen	$\min\{1, \frac{\|B\|^t}{\|A\|^t}\}$	$A \to B = \frac{\|B \cap A\|}{\|A\|}$
Lukasiewicz	$\min\{1, 1 - \|A\| + \|B\|\}$	$A \to B = \neg A \vee B$ $\|A \vee B\| = \min\{1, \|A\| + \|B\|\}$
Mamdani	$\min\{\|A\|, \|B\|\}$	
Yager	$\|B\|^{\|A\|}$	
Max-Min	$\max\{1 - \|A\|, \min\{\|A\|, \|B\|\}\}$	$A \to B = \neg A \vee (A \wedge B)$

Table 8.1: Some alternative definitions for the conditional implication connective.

As for $\mathcal{L}^f$, a simple axiomatization of $\mathcal{L}^f_+$ is obtained by adding axiom 8.20 to the axioms of $\mathcal{L}^f$, *i.e.* axiom (8.9), (8.10), (8.11) and (8.12).

As for $\mathcal{L}_+$, most of the properties are inherited by $\mathcal{L}^f_+$:

$$\not\approx_4(A \to A \geq .5), \text{ (}i.e.\text{ there are tautologies)} \tag{8.22}$$

$$\text{every KB is satisfiable} \tag{8.23}$$

$$\Sigma \not\approx_4(A \geq n) \text{ implies } \Sigma \not\approx_2(A \geq n) \tag{8.24}$$

Example 19 Let Σ be the fuzzy KB

$$\begin{array}{rll} \Sigma = & \{ & (\mathsf{Jon} \to \mathsf{Student} \wedge \mathsf{Adult} \geq .6), \\ & & (\mathsf{Gil} \to \mathsf{Adult} \geq .7), \\ & & (\mathsf{Adult} \to \mathsf{Tall} \geq .8), \\ & & (\mathsf{Karl} \to \mathsf{Child} \geq .9), \\ & & (\mathsf{Child} \to \mathsf{Tall} \geq .2), \\ & & (\mathsf{Student} \to \mathsf{Tall} \geq .5)\} \end{array}$$

By applying the simple modus ponens inference schema (8.20), it can easily be verified that

$$\Sigma \cup \{(\mathsf{Gil} \geq .8)\} \not\approx_4 (\mathsf{Tall} \geq .8),$$

$$\Sigma \cup \{(\mathsf{Jon} \geq .3)\} \not\approx_4 (\mathsf{Tall} \geq n), \text{ for all } n > 0, \textit{and}$$

$$\Sigma \cup \{(\mathsf{Gil} \vee \mathsf{Karl} \geq .4)\} \not\approx_4 (\mathsf{Tall} \geq .2).$$

It is worth noting that the values are maximal. ■

As for $\mathcal{L}^f$, there are some relations between fuzzy entailment in $\mathcal{L}^f_+$ and entailment in $\mathcal{L}_+$. By a straightforward extension of the proof, it can be verified that Proposition 6 holds in $\mathcal{L}^f_+$ too.

Proposition 10 *Consider* $\Sigma \subseteq \mathcal{L}^f_+$, $A \in \mathcal{L}_+$ *and* $n > 0$. *If* $\Sigma \not\approx_4(A \geq n)$ *and* $n > 0$ *then* $\overline{\Sigma} \models_4 A$. ⊣

By relying on Example 19, it can be easily verified that

$$\overline{\Sigma \cup \{(\mathsf{Gil} \geq .8)\}} \models_4 \mathsf{Tall}, \text{ and}$$

$$\overline{\Sigma \cup \{(\mathsf{Gil} \vee \mathsf{Karl} \geq .4)\}} \models_4 \mathsf{Tall},$$

confirming Proposition 10. Moreover, the KB in Example 19 shows us immediately that the converse of Proposition 6 does not hold. In fact,

$$\overline{\Sigma \cup \{(\mathsf{Jon} \geq .3)\}} \models_4 \mathsf{Tall}, \text{ but}$$

$$\Sigma \cup \{(\mathsf{Jon} \geq .3)\} \not\models_4 (\mathsf{Tall} \geq n), \text{ for all } n > .$$

We conclude this section by showing that a version of the deduction theorem similar to Proposition 3 holds. This is stated by means of the following two theorems.

Proposition 11 *Let $A, B, C \in \mathcal{L}$. For all $0 < n, m, k \leq 1$, if $(A \geq n), (B \geq m) \models_4 (C \geq k)$ then $(A \geq n) \models_4 (B \rightarrow C \geq l)$, where $l = \min\{k, 1 - m\}$.* ⊣

Proof: Let $\mathcal{I}$ be an interpretation such that $|A|^t \geq n$. By definition, $|B \rightarrow C|^t = \max\{1 - |B|^t, |C|^t\} = h$. There are two cases to be analyzed: (i) if $|B|^t \geq m$ then, from hypothesis $|C|^t \geq k \geq l$. Hence, $h \geq l$; (i) if $|B|^t < m$ then $1 - |B|^t \geq 1 - m \geq l$. Hence, $h \geq l$. Therefore, $|B \rightarrow C|^t = h \geq l$. Q.E.D.

For example, (see Equation 7.42 for the crisp case)

$$\begin{array}{c} (A \geq .4), (A \rightarrow B \geq .7), (C \geq .8) \models_4 (B \wedge C \geq .7) \\ \text{iff} \\ (A \geq .4), (A \rightarrow B \geq .7) \models_4 (C \rightarrow (B \wedge C) \geq .2) \end{array} \tag{8.25}$$

where $.2 = \min\{.7, 1 - .8\}$.

Proposition 12 *Let $A, B, C \in \mathcal{L}$ and $n > 0$. For all $0 < n, k \leq 1$ and $0 < \epsilon \leq k$, if $(A \geq n) \models_4 (B \rightarrow C \geq k)$ then $(A \geq n), (B \geq m) \models_4 (C \geq k)$, where $m = 1 - k + \epsilon$.* ⊣

Proof: Let $\mathcal{I}$ be an interpretation such that $|A|^t \geq n$ and $|B|^t \geq m$. By hypothesis, $|B \rightarrow C|^t = \max\{1 - |B|^t, |C|^t\} \geq k$. Therefore, from $|B|^t \geq m = 1 - k + \epsilon$, $k > k - \epsilon \geq 1 - |B|^t$ follows. Hence, $k \leq \max\{1 - |B|^t, |C|^t\} = |C|^t$. Q.E.D.

For instance,

$$\begin{array}{c} (A \geq .4), (A \rightarrow B \geq .7) \models_4 (C \rightarrow (B \wedge C) \geq .2) \\ \text{iff for all } 0 < \epsilon \leq .2 \\ (A \geq .4), (A \rightarrow B \geq .7), (C \geq .8 + \epsilon) \models_4 (B \wedge C \geq .2) \end{array} \tag{8.26}$$

where $.8 + \epsilon = 1 - .2 + \epsilon$.

8.2.3 The logic HORN-$\mathcal{L}^f$

Consider $(A \to B \geq n)$. A drawback of the semantics we have given (according to Zadeh) to the connective $\to$ is that, for al interpretations $\mathcal{I}$ satisfying $(A \to B \geq n)$, $|B|^t$ has constant value n, if $m = |A|^t > 1 - n$. This means that $|B|^t$ *does not depend on the value of* $|A|^t$. This is a characteristic of Zadeh's semantics for the implication connective. As we have seen, there are several different semantics for the $\to$ (see *e.g.* Table 8.1 and [290]). Some of these are such that $|B|^t$ has to depend on $|A|^t$. For instance, the Gougen semantics is such that

$$(A \geq m), (A \to B \geq n) \models_4 (B \geq nm)$$

which models a sort of conditional probability: if $P(A) \geq m$ and $P(B|A) \geq n$ then $P(A \wedge B) \geq nm$, which is a consequence of the hypothesis: "the probability of the conditional $\to$ is the same as the conditional probability", *i.e.*

$$P(A \to B) = P(B|A), \text{ whenever } P(A) > 0.$$

This hypothesis has come to be known as Stalnaker's Hypothesis. But, notwithstanding it seems to be a quite reasonable hypothesis, there are several arguments against it. See, for instance, [141] in which it is shown that it cannot be right in general, and that it in fact fails in all situations in which it would have the most obvious applicability.

In logic, the different semantics for $\to$ aim to model some sort of modus pones: *i.e.* for some predefined function $f\colon [0,1] \times [0,1] \to [0,1]$, for all interpretations $\mathcal{I}$ satisfying $(A \to B \geq n)$, if $|A|^t \geq m$ then $|B|^t \geq f(n,m)$. The drawback of those approaches is that, ones the semantics for $\to$ has been chosen, then the function f is fixed.

The aim of this section is to define the semantics in such a way that any arbitrary function f can be chosen, as it is in works like [114, 161, 163, 171, 172, 254, 272].

Consider a new alphabet of variables, called *fuzzy variables* (denoted by V, W). A *fuzzy value* (denoted by T, U) is either a fuzzy variable or a real in $[0,1]$.

Let $\mathcal{F}$ be a set of *degree functions* such that

$$\begin{array}{rcll} \mathcal{F} & = & \{f\colon [0,1]^n \to [0,1]: & n = 1, 2, \ldots, \\ & & & f \text{ defined on } [0,1]^n, \\ & & & f \text{ nondecreasing on all arguments}\}. \end{array}$$

An n-ary function $f \in \mathcal{F}$ determines how to combine the degrees of its arguments. For instance,

$$\begin{array}{l} f^1(m_1, \ldots, m_n) = \min\{m_1, \ldots, m_n\}, \\ f^2(m_1, \ldots, m_n) = \max\{m_1, \ldots, m_n\}, \\ f^3(m_1, \ldots, m_n) = m_1 \cdot \ldots \cdot m_n, \\ f^4(m_1, \ldots, m_n) = \frac{m_1 + \ldots + m_n}{n}, \end{array}$$

and so on, are all degree functions. The condition of being nondecreasing is quite natural (see *e.g.* [46]). Moreover, we assume that for all $n \in [0,1]$, $n \in \mathcal{F}$, *i.e.* the constant function returning n is a degree function.

A *horn rule* in HORN-$\mathcal{L}^f$ is an expression of the form

$$(A \geq V) \leftarrow (A_1 \geq V_1), \ldots, (A_n \geq V_n), \langle V, f(V_1, \ldots, V_n)\rangle, \tag{8.27}$$

with $n \geq 1$, $A, A_1, \ldots, A_n$ propositional letters, $V, V_1, \ldots, V_n$ distinct fuzzy variables and f a degree function. $(A \geq V)$ is called *head* and $(A_1 \geq V_1)$, ..., $(A_n \geq V_n)$, $\langle V, f(V_1, \ldots, V_n)\rangle$ is called *body*. Of course, horn rules are universally quantified on the fuzzy variables $V, V_1, \ldots, V_n$. For instance, suppose f is the degree function

$$f(n, m) = n \cdot m.$$

Then the horn rule

$$(\mathsf{Tall} \geq V) \leftarrow (\mathsf{Adult} \geq V_1), (\mathsf{Male} \geq V_2), \langle V, f(V_1, V_2)\rangle$$

establishes that male adults are Tall to a degree depending on f, *i.e.* on the degree of being adult and male.

A *fact* is any fuzzy proposition $(A \geq n) \in \mathcal{L}^f$. It is worth noting that *e.g.* $(A \vee (B \wedge \neg C) \geq .4)$ is a fact: we do not limit ourself to facts which are *horn*, *i.e.* expressions of type $(p \geq n) \in \mathcal{L}^f$ (p is a propositional letter).

A *set of fuzzy value restrictions* VR is a set $\{U \leq T : U, T \text{ fuzzy values}\}$. We will write $U = T$ in place of $U \leq T$ and $T \leq U$.

A *goal* is an expression of the form

$$\begin{array}{rl} \leftarrow & (A_1 \geq V_1), \ldots, (A_n \geq V_n), \\ & \langle V_{f^1}, f^1(V_{f^1_1}, \ldots, V_{f^1_{n_1}})\rangle, \ldots, \\ & \langle V_{f^k}, f^k(V_{f^k_1}, \ldots, V_{f^k_{n_k}})\rangle : VR \end{array} \tag{8.28}$$

with $n \geq 0$, $A_1, \ldots, A_n$ propositional letters, f^i fuzzy degree functions and VR a set of fuzzy value restrictions. The case $n = 0, k = 0$ is called *empty goal* and is indicated with $\leftarrow \blacksquare : VR$. Moreover, the following conditions have to be satisfied. Let $\vec{V}_{f^i}$ be the set $\{V_{f^i_1}, \ldots, V_{f^i_{n_i}}\}$. Then

1. the fuzzy variables $V_1, \ldots, V_n, V_{f^1}, \ldots, V_{f^k}$ are all distinct;
2. for all $1 \leq i < j \leq k$, $\vec{V}_{f^i} \cap \vec{V}_{f^j} = \emptyset$;
3. there is $1 \leq l \leq k$ such that V_{f^l} occurs in no $\vec{V}_{f^i}$, $1 \leq i \leq k$;
4. for each $1 \leq i \leq k$, $i \neq l$, there is exactly one l_i such that $V_{f^i} \in \vec{V}_{f^{l_i}}$.
5. $\{V_1, \ldots, V_n\} = (\bigcup_{1 \leq i \leq k} \vec{V}_{f^i}) \setminus \{V_{f^1}, \ldots, V_{f^k}\}$.

An example of goal is

$$\leftarrow (\mathsf{Adult} \geq V_1), (\mathsf{Student} \geq V_3), \langle V, f^1(V_1, V_2)\rangle, \langle V_2, f^2(V_3)\rangle.$$

Notice that, the above points are satisfied. Let $\vec{V}_{f^1} = \{V_1, V_2\}$ and let $\vec{V}_{f^2} = \{V_3\}$.

1. the fuzzy variables V_1, V_3, V, V_2 are all distinct;
2. $\vec{V}_{f^1} \cap \vec{V}_{f^2} = \emptyset$;
3. for $l = 1$, V_{f^l} occurs in no $\vec{V}_{f^i}$, $1 \leq i \leq 2$;

4. for $i = 2$, there is exactly one $l_i = 1$ such that $V_{f^i} \in \vec{V}_{f^{l_i}}$.

5. $\{V_1, V_3\} = (\bigcup_{1 \leq i \leq 2} \vec{V}_{f^i}) \setminus \{V_{f^1}, V_{f^2}\}$.

A *query* is an expression of the form

$$\exists V_1, \ldots, V_n.(A_1 \geq V_1) \wedge \ldots \wedge (A_n \geq V_n), \tag{8.29}$$

with $n \geq 1$, $A_1, \ldots, A_n$ propositional letters and $V_1, \ldots, V_n$ distinct fuzzy variables.

The *associated* goal G_Q of a query $Q = \exists V_1, \ldots, V_n.(A_1 \geq V_1) \wedge \ldots \wedge (A_n \geq V_n)$ is the goal

$$\leftarrow (A_1 \geq V_1), \ldots, (A_n \geq V_n), \langle V, 1\rangle : \emptyset,$$

where V new fuzzy variable.

A HORN-$\mathcal{L}^f$ KB Σ is a finite set of horn rules and facts. With $\Sigma_R \subseteq \Sigma$ we indicate the set of horn rules in Σ and with $\Sigma_F \subseteq \Sigma$ we indicate the set of facts in Σ. A *horn* KB is a finite set of horn rules and horn facts.

We will say that HORN-$\mathcal{L}^f$ KB Σ is *safe* iff no A of head $(A \geq V)$ of a rule $R \in \Sigma_R$ appears in Σ_F. Finally, the definition of *recursive* HORN-$\mathcal{L}^f$ KBs Σ is as usual: for all horn rules $(A \geq V) \leftarrow (A_1 \geq V_1), \ldots, (A_n \geq V_n), \langle V, f(V_1, \ldots, V_n)\rangle \in \Sigma$ we will say that A *directly uses* A_i for all $1 \leq i \leq n$. Let *uses* be the transitive closure of the relation directly uses in Σ. We will say that a HORN-$\mathcal{L}^f$ KB Σ is *recursive* iff there is A such that A uses A trough rules in Σ.

From a semantics point of view, let $\mathcal{I}$ be an interpretation. $\mathcal{I}$ maps fuzzy variables into $[0, 1]$ and for $n \in [0, 1]$, $n^{\mathcal{I}} = n$. Moreover, if VR is a set of fuzzy value restrictions then $\mathcal{I}$ *satisfies* VR iff for all $U \leq T \in VR$, $U^{\mathcal{I}} \leq T^{\mathcal{I}}$ holds.

Let f be a degree function. Then $\mathcal{I}$ has to satisfy

$$t \in \langle V, f(V_1, \ldots, V_n)\rangle^{\mathcal{I}} \quad \text{iff} \quad V^{\mathcal{I}} \leq f({V_1}^{\mathcal{I}}, \ldots, {V_n}^{\mathcal{I}}). \tag{8.30}$$

$\mathcal{I}$ *satisfies* $\langle V, f(V_1, \ldots, V_n)\rangle$ iff $t \in \langle V, f(V_1, \ldots, V_n)\rangle^{\mathcal{I}}$.

Let $\vec{V} = (V_1, \ldots, V_n)$ be a tuple of distinct fuzzy variables and let $\vec{m} = (m_1, \ldots, m_n)$ be a tuple of elements in $[0, n]^n$. The interpretation $\mathcal{I}^{\vec{m}}_{\vec{V}}$ is as $\mathcal{I}$, except that each fuzzy variable V_i in $\vec{V}$ is mapped into m_i, *i.e.*

$$W^{\mathcal{I}^{\vec{m}}_{\vec{V}}} = \begin{cases} W^{\mathcal{I}} & \text{if for all } i,\ W \neq V_i; \\ m_i & \text{if for some } i,\ W = V_i. \end{cases} \tag{8.31}$$

With respect to horn rules, we extend the semantics of $A \leftarrow A_1, \ldots, A_n \in$ HORN-$\mathcal{L}$. Just remember that $\mathcal{I}$ satisfies satisfies $A \leftarrow A_1, \ldots, A_n$ iff if $\mathcal{I}$ satisfies each $A_1, \ldots, A_n$ then $\mathcal{I}$ satisfies A. Equivalently, $\mathcal{I}$ satisfies $A \leftarrow A_1, \ldots, A_n$ iff $\mathcal{I}$ satisfies A or $\mathcal{I}$ does not satisfy some A_i. An interpretation $\mathcal{I}$

1. *satisfies* a horn rule

$$(A \geq V) \leftarrow (A_1 \geq V_1), \ldots, (A_n \geq V_n), \langle V, f(V_1, \ldots, V_n)\rangle$$

iff for all $\vec{m} \in [0, 1]^{n+1}$ such that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies $\langle V, f(V_1, \ldots, V_n)\rangle$, if $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies all $(A_1 \geq V_1)$, ..., $(A_n \geq V_n)$, then $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies $(A \geq V)$, where $\vec{V}$ is the tuple of all the $n + 1$ fuzzy variables appearing in the rule;

2. *satisfies* a goal

$$\begin{array}{rl} \leftarrow & (A_1 \geq V_1), \ldots, (A_n \geq V_n), \\ & \langle V_{f^1}, f^1(V_{f^1_1}, \ldots, V_{f^1_{n_1}})\rangle, \ldots, \\ & \langle V_{f^k}, f^k(V_{f^k_1}, \ldots, V_{f^k_{n_k}})\rangle : VR \end{array}$$

iff $n + k \geq 1$ and for all $\vec{m} \in [0,1]^l$ such that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies VR and $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies all $\langle V_{f^j}, f^j(V_{f^j_1}, \ldots, V_{f^j_{n_j}})\rangle$, $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy some $(A_i \geq V_i)$, $1 \leq i \leq n$, where $\vec{V}$ is the tuple of all the l fuzzy variables appearing in the goal;

3. *satisfies* a query

$$\exists V_1, \ldots, V_n.(A_1 \geq V_1) \wedge \ldots \wedge (A_n \geq V_n),$$

iff for some $\vec{m} \in [0,1]^n$, $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies all $(A_1 \geq V_1)$, $\ldots$, $(A_n \geq V_n)$, where $\vec{V}$ is the tuple of all the n fuzzy variables appearing in the query.

Note that the empty goal is never satisfied.

An interpretation $\mathcal{I}$ *satisfies* (*is a model of*) a HORN-$\mathcal{L}^f$ KB Σ iff $\mathcal{I}$ satisfies each element of it. Satisfiability is extended to an arbitrary set S of horn rules, facts and goals, as usual. Finally, a HORN-$\mathcal{L}^f$ KB Σ *entails* a query Q (denoted by $\Sigma \models_4 Q$) iff every model of Σ satisfies Q.

Let Q be a query $\exists V_1, \ldots, V_n.(A_1 \geq V_1) \wedge \ldots \wedge (A_n \geq V_n)$. An *answer* to the query Q is a substitution θ of all variables in Q, *i.e.* $\theta = \{V_1/m_1, \ldots, V_n/m_n\}$. If θ is an answer, with $\vec{\theta}$ we will denote the tuple $(m_1, \ldots, m_n)$.

An answer is *correct* w.r.t. a HORN-$\mathcal{L}^f$ KB Σ iff $\Sigma \models_4 Q\theta$, where $Q\theta$ is $(A_1 \geq m_1) \wedge \ldots \wedge (A_n \geq m_n)$. As usual, the *answer set* of a query Q w.r.t. a KB Σ (denoted by $AnswerSet(\Sigma, Q)$) is the set of correct answers, *i.e.*

$$AnswerSet(\Sigma, Q) \quad = \quad \{\theta : \Sigma \models_4 Q\theta\}. \tag{8.32}$$

Now, let $l, h \in [0,1]$ be two reals, let $\vec{n} = (n_1, \ldots, n_k)$ and $\vec{m} = (m_1, \ldots, m_k)$ be two tuples of reals in $[0,1]$, let $N = \{\vec{n}_1, \ldots, \vec{n}_r, \ldots\}$ be a set of reals and tuple of reals in $[0,1]$, let $\theta_1 = \{V_1/n_1, \ldots, V_k/n_k\}$ and $\theta_2 = \{V_1/m_1, \ldots, V_k/m_k\}$ be two answers of query Q w.r.t. KB Σ and let $\Theta = \{\theta_1, \ldots, \theta_q, \ldots\}$ be a set of answers of query Q w.r.t. KB Σ. We define

$$\begin{array}{lcl} l \uparrow h & = & \sup\{l, h\} \\ \vec{n} \uparrow \vec{m} & = & (n_1 \uparrow m_1, \ldots, n_k \uparrow m_k) \\ \uparrow N & = & \vec{n}_1 \uparrow \ldots \uparrow \vec{n}_r \uparrow \ldots \\ & & \\ l \downarrow h & = & \inf\{l, h\} \\ \vec{n} \downarrow \vec{m} & = & (n_1 \downarrow m_1, \ldots, n_k \downarrow m_k) \\ \downarrow N & = & \vec{n}_1 \downarrow \ldots \downarrow \vec{n}_r \downarrow \ldots \\ & & \\ \vec{n} \geq \vec{m} & \text{iff} & \vec{n} = \vec{m} \uparrow \vec{n} \\ \vec{n} \leq \vec{m} & \text{iff} & \vec{m} \geq \vec{n} \\ \vec{n} > \vec{m} & \text{iff} & \vec{n} \geq \vec{m}, \vec{n} \neq \vec{m} \\ \vec{n} < \vec{m} & \text{iff} & \vec{m} > \vec{n} \end{array} \tag{8.33}$$

$$\begin{array}{lcl}
\theta_1 \uparrow \theta_2 & = & \{V_1/n_1 \uparrow m_1, \ldots, V_k/n_k \uparrow m_k\} \\
\uparrow \Theta & = & \theta_1 \uparrow \ldots \uparrow \theta_q \uparrow \ldots \\
\theta_1 \downarrow \theta_2 & = & \{V_1/n_1 \downarrow m_1, \ldots, V_k/n_k \downarrow m_k\} \\
\downarrow \Theta & = & \theta_1 \downarrow \ldots \downarrow \theta_q \downarrow \ldots \\
 & & \\
\theta_1 \geq \theta_2 & \text{iff} & \theta_1 = \theta_1 \uparrow \theta_2 \\
\theta_1 \leq \theta_2 & \text{iff} & \theta_2 \geq \theta_1 \\
\theta_1 > \theta_2 & \text{iff} & \theta_1 \geq \theta_2, \theta_1 \neq \theta_2 \\
\theta_1 < \theta_2 & \text{iff} & \theta_2 > \theta_1
\end{array} \tag{8.34}$$

Let Σ be a HORN-$\mathcal{L}^f$ KB, let Q be a query and let $\theta_1, \theta_2 \in AnswerSet(\Sigma, Q)$. It is easily verified that $\theta_1 \uparrow \theta_2 \in AnswerSet(\Sigma, Q)$ too. Therefore, $AnswerSet(\Sigma, Q)$ has an unique maximal correct answer according to $\leq$ which is $\uparrow AnswerSet(\Sigma, Q)$. We will define

$$Maxdeg(\Sigma, Q) \quad = \quad \uparrow AnswerSet(\Sigma, Q). \tag{8.35}$$

Notice that if $Maxdeg(\Sigma, Q) = \theta$ and $\vec{\theta} = \vec{m}$, then it is possible to define the maximal degree of the query Q to be *e.g.* $\min\{m_1, \ldots, m_n\}$ (of course, any other degree function can be used). In a more principled way, if we were interested in to determine a degree value $m \in [0, 1]$ in terms of $f(m_1, \ldots, m_n)$, then it is sufficient to add the rule

$$(A \geq V) \leftarrow (A_1 \geq V_1), \ldots, (A_n \geq V_n), \langle V, f(V_1, \ldots, V_n \rangle$$

to Σ, where A is a new propositional letter. Thereafter we have to consider the new query

$$\exists V.(A \geq V).$$

Let $Q = \exists V_1, \ldots, V_n.(A_1 \geq V_1) \wedge \ldots \wedge (A_n \geq V_n)$. Let G_Q be the associated goal. It is easily verified that

$$\Sigma \mid\!\approx_4 Q \quad \text{iff} \quad \Sigma \cup \{G_Q\} \text{ not satisfiable.} \tag{8.36}$$

Just let us mention that in case of horn KB, equivalence between four-valued fuzzy entailment and two-valued fuzzy entailment holds.

Proposition 13 *In* HORN-$\mathcal{L}^f$*, let* Σ *be a horn KB and let* Q *be a query. Then* $\Sigma \mid\!\approx_4 Q$ *iff* $\Sigma \mid\!\approx_2 Q$. ⊣

In the general case where Σ is a HORN-$\mathcal{L}^f$ KB, then $\Sigma \mid\!\approx_4 Q$ implies $\Sigma \mid\!\approx_2 Q$, but not vice-versa. In fact, consider $\Sigma = \{(\neg A \vee B \geq 1), (A \geq 1)\}$. Then $\Sigma \mid\!\approx_2 (B \geq 1)$, but $\Sigma \not\mid\!\approx_4 (B \geq 1)$.

Example 20 Consider the following HORN-$\mathcal{L}^f$ KB

$$\begin{array}{lcl}
\Sigma & = & \{(\mathsf{Hungry} \geq V) \leftarrow (\mathsf{Adult} \geq V_1), (\mathsf{Male} \geq V_2), \langle V, \max\{0, V_1 + V_2 - 1\}\rangle, \\
 & & (\mathsf{Adult} \geq V) \leftarrow (\mathsf{Murray} \geq V_1), \langle V, V_1\rangle, \\
 & & (\mathsf{Male} \geq V) \leftarrow (\mathsf{Murray} \geq V_1), \langle V, .9\rangle, \\
 & & (\mathsf{Adult} \geq V) \leftarrow (\mathsf{Murphy} \geq V_1), \langle V, V_1\rangle, \\
 & & (\mathsf{Male} \geq V) \leftarrow (\mathsf{Murphy} \geq V_1), \langle V, .7\rangle, \\
 & & (\mathsf{Murphy} \vee \mathsf{Murray} \geq .6)\}.
\end{array}$$

Just notice here that $\max\{0, V_1 + V_2 - 1\}$ is the degree of truth of a conjunction according to Lukasiewicz (see Table 8.1). In principle, there is no restriction on degree functions. Any arbitrary degree function can be used and mixed together. It is easily verified that $\Sigma \mid\!\!\approx_4 (\mathsf{Hungry} \geq .3)$. Moreover, $Maxdeg(\Sigma, \exists V.(\mathsf{Hungry} \geq V)) = \{V/.3\}$. ■

8.3 Four-valued fuzzy horn $\mathcal{ALC}$

The four-valued fuzzy semantics for $\mathcal{ALC}$ is a simple first-order extension of the semantics developed in Section 8.2. Most considerations and propositions are a straightforward extension of those seen in Section 8.2. Hence, we will not be to verbose in this section.

8.3.1 Four-valued fuzzy $\mathcal{ALC}$

8.3.1.1 Syntax and semantics of fuzzy assertions

Syntax: A *fuzzy assertion* is an expression of type $(A \geq n)$, where A is an assertion in $\mathcal{ALC}$ and $n \in [0, 1]$. We will use γ as metavariable for fuzzy assertions. An *atomic fuzzy assertion* is of the form $(A \geq n)$, where A is an atomic assertion.

In fuzzy $\mathcal{ALC}$, a concept is interpreted as a fuzzy set. Therefore, concepts and roles become *imprecise* (or *vague*). According to this view, the intended meaning of *e.g.* $(C(a) \geq n)$ we will adopt is: "the membership degree of individual a being an instance of concept C is at least n". Similarly for roles.

Similarly for roles: the intended meaning of *e.g.* $(R(a, b) \geq n)$ we will adopt is: "the membership degree of individual a being related to b by means of R is at least n" For instance, (i) $(\mathsf{About}(\mathsf{i1}, \mathsf{basket}) \geq .7)$ means that the degree of being "image" i1 About basket is at least .7, *i.e.* i1 is likely About basket; $(\mathsf{About}(\mathsf{i1}, \mathsf{basket}) \geq .7)$ means that i1 is definitely about basket; and (ii) $(\mathsf{Tall}(\mathsf{umberto}) \geq .7)$ means that the degree of tom being Tall is at least .7, *i.e.* umberto is likely tall; $(\mathsf{Tall}(\mathsf{umberto}) \geq 1)$ means that umberto is tall, whereas $(\neg\mathsf{Tall}(\mathsf{umberto}) \geq 1)$ means that umberto is not tall. Hence, by switching to DLs, the degree of truth of $C(a)$ will be interpreted as degree of membership of the individual a to the *fuzzy concept* (set) C.

Semantics: With respect to $\mathcal{ALC}$, a *fuzzy valuation* is now is a function $|\cdot|$ mapping (i) $\mathcal{ALC}$ concepts into a membership degree function $\Delta \to [0, 1]$ (Δ is the domain); and (ii) $\mathcal{ALC}$ roles into a membership degree function $\Delta \times \Delta \to [0, 1]$. If C is a concept then $|C|$ will naturally be interpreted as the *membership degree function* of the fuzzy concept (set) C, *i.e.* if $d \in \Delta$ is an object of the domain Δ then $|C|(d)$ gives us the degree of being the object d an element of the fuzzy concept C. Similar arguments holds for roles.

As for four-valued fuzzy propositional case, we will have two valuation functions: $|\cdot|^t$ and $|\cdot|^f$: $|C|^t$ will naturally be interpreted as the *membership degree function* of C, whereas $|C|^f$ will analogously be interpreted as the *non-membership degree function* of C. For instance, $|\mathsf{Tall}|^t(\mathsf{umberto})$ gives us the degree of umberto being Tall, whereas $|\mathsf{Tall}|^f(\mathsf{umberto})$ gives us the degree of umberto being not Tall.

The classical "two-valued" fuzzy case is obtained, as usual, whenever $|C|^f = 1 - |C|^t$, for each concept C. Similarly for roles.

A *fuzzy interpretation*[2] in the context of $\mathcal{ALC}$, is a tuple $\mathcal{I} = ((\cdot)^{\mathcal{I}}, |\cdot|^t, |\cdot|^f, \Delta^{\mathcal{I}})$, where

[2] In the following called interpretation.

1. $|\cdot|^t$ and $|\cdot|^f$ are fuzzy valuations, *i.e.* $|\cdot|^t$ and $|\cdot|^f$ map concepts into a function from $\Delta^{\mathcal{I}}$ into $[0,1]$ and map roles into a function from $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ into $[0,1]$;
2. $\cdot^{\mathcal{I}}$ maps every fuzzy assertion into an element of $2^{\{t,f\}}$;
3. $\cdot^{\mathcal{I}}$ maps every individual into $\Delta^{\mathcal{I}}$;
4. $a^{\mathcal{I}} \neq b^{\mathcal{I}}$, if $a \neq b$.

Additionally, $|\cdot|^t$ and $|\cdot|^f$ have to satisfy the following equations: for all $d \in \Delta^{\mathcal{I}}$

$$\begin{array}{lcl}
|C \sqcap D|^t(d) & = & \min\{|C|^t(d), |D|^t(d)\} \\
|C \sqcap D|^f(d) & = & \max\{|C|^f(d), |D|^f(d)\} \\
 & & \\
|C \sqcup D|^t(d) & = & \max\{|C|^t(d), |D|^t(d)\} \\
|C \sqcup D|^f(d) & = & \min\{|C|^f(d), |D|^f(d)\} \\
 & & \\
|\neg C|^t(d) & = & |C|^f(d) \\
|\neg C|^f(d) & = & |C|^t(d) \\
 & & \\
|\exists R.C|^t(d) & = & |(\neg\forall R.\neg C)|^t(d) \\
|\exists R.C|^f(d) & = & |(\neg\forall R.\neg C)|^f(d) \\
 & & \\
|\forall R.C|^t(d) & = & \inf_{d' \in \Delta^{\mathcal{I}}}\{\max\{1 - |R|^t(d,d'), |C|^t(d')\}\} \\
|\forall R.C|^f(d) & = & \sup_{d' \in \Delta^{\mathcal{I}}}\{\min\{|R|^t(d,d'), |C|^f(d')\}\}.
\end{array} \tag{8.37}$$

These equations are the standard interpretation of conjunction, disjunction, and negation. Just note that that the semantics of the $\exists$ connective has been given in terms of the $\forall$ connective. For the $\forall$ connective,

$$|\forall R.C|^t(d) = \inf_{d' \in \Delta^{\mathcal{I}}}\{\max\{1 - |R|^t(d,d'), |C|^t(d')\}\} \tag{8.38}$$

is the result of viewing $\forall R.C$ as the open first order formula $\forall y.R(x,y) \rightarrow C(y)$. Now, the universal quantifier $\forall$ is viewed as a conjunction over the elements of the domain and, thus, $|\forall x.P(x)|^t = \inf_{d' \in \Delta^{\mathcal{I}}}\{|P|^t(d')\}$, where P is an unary predicate, whereas an implication $F \rightarrow G$ follows the semantics of $\mathcal{L}^f_+$ and, thus, $|F \rightarrow G|^t = \max\{1 - |F|^t, |G|^t\}$ (see Equation (8.17)). The combination of these two points of views yields

$$|\forall R.C|^t(d) = \inf_{d' \in \Delta^{\mathcal{I}}}\{\max\{1 - |R|^t(d,d'), |C|^t(d')\}\}.$$

Observe that the semantics of the $\forall$ connective gives us a sort of modus ponens over roles as for any $n \in [0,1]$, $|\forall R.C|^t(d) \geq n$ iff for all $d' \in \Delta^{\mathcal{I}}$ if $|R|^t(d,d') > 1 - n$ then $|C|^t(d') \geq n$. Thus, $|\forall R.C|^t(d)$ reflects the type A semantics for the $\forall$ connective too (see Equation (7.64)). By the way, in order to reflect the type B semantics for the $\forall$ connective (see Equation(7.65)), just define

$$|\forall R.C|^t(d) = \inf_{d' \in \Delta^{\mathcal{I}}}\{\max\{|R|^f(d,d'), |C|^t(d')\}\}. \tag{8.39}$$

In the following we will not consider this case.

By definition, $|\exists R.C|^t(d)$ is $|(\neg\forall R.\neg C)|^t(d)$ which is $|\forall R.\neg C|^f(d)$ and which leads to

$$|\exists R.C|^t(d) = \sup_{d' \in \Delta^{\mathcal{I}}} \{\min\{|R|^t(d,d'), |C|^t(d')\}\}. \tag{8.40}$$

The above equation is the result of viewing $\exists R.C$ as the open first order formula $\exists y.R(x,y) \wedge C(y)$. Now, the existential quantifier $\exists$ is viewed as a disjunction over the elements of the domain and, thus, $|\exists x.P(x)|^t = \sup_{d' \in \Delta^{\mathcal{I}}}\{|P|^t(d')\}$, where P is an unary predicate. Hence, this view yields to (8.40).

The definitions for $|\cdot|^t$ and $|\cdot|^f$ in (8.37) are, thus, complementary with respect to the $\neg$ connective.

Moreover, $(\cdot)^{\mathcal{I}}$ has to satisfy (see Equation (8.3))

$$\begin{array}{lll} t \in (C(a) \geq n)^{\mathcal{I}} & \text{iff} & |C|^t(a^{\mathcal{I}}) \geq n \\ f \in (C(a) \geq n)^{\mathcal{I}} & \text{iff} & |C|^f(a^{\mathcal{I}}) \geq n \\ t \in (R(a,b) \geq n)^{\mathcal{I}} & \text{iff} & |R|^t(a^{\mathcal{I}}, b^{\mathcal{I}}) \geq n \\ f \in (R(a,b) \geq n)^{\mathcal{I}} & \text{iff} & |R|^f(a^{\mathcal{I}}, b^{\mathcal{I}}) \geq n. \end{array} \tag{8.41}$$

As for the propositional case $\mathcal{L}^f_+$, there is a simple connection between the four-valued semantics given in Section 7.4.1 (Equations (7.56)–(7.63)) and the fuzzy counterpart. In fact, the above conditions can be reformulated as *e.g.*

$$\begin{array}{lll} t \in ((C \sqcap D)(a) \geq n)^{\mathcal{I}} & \text{iff} & t \in (C(a) \geq n)^{\mathcal{I}} \text{ and } t \in (D(a) \geq n)^{\mathcal{I}}; \\ f \in ((C \sqcap D)(a) \geq n)^{\mathcal{I}} & \text{iff} & f \in (C(a) \geq n)^{\mathcal{I}} \text{ or } f \in (D(a) \geq n)^{\mathcal{I}} \end{array} \tag{8.42}$$

(the other cases, $\sqcup, \neg, \forall$ and $\exists$ are quite similar).

A concept C is *equivalent* to a concept D (denoted by $C \approx_4 D$) iff $|C|^t = |D|^t$, for every interpretation $\mathcal{I}$. A concept C is *coherent* iff there is an interpretation $\mathcal{I}$ and an object $d \in \Delta^{\mathcal{I}}$ such that $|C|^t(d) > 0$.

An interpretation $\mathcal{I}$ *satisfies a fuzzy assertion* γ, iff $t \in \gamma^{\mathcal{I}}$. Given two $\mathcal{ALC}$ fuzzy assertions γ_1 and γ_2, γ_1 is *equivalent* to γ_2 (denoted by $\gamma_1 \approx_4 \gamma_2$) iff for every interpretation $\mathcal{I}$, $\mathcal{I}$ satisfies γ_1 iff $\mathcal{I}$ satisfies γ_2.

An interpretation $\mathcal{I}$ *satisfies* (*is a model of*) a knowledge base Σ (a set of fuzzy assertions) iff $\mathcal{I}$ satisfies all elements in Σ. A knowledge base Σ *entails* a fuzzy assertion γ (denoted by $\Sigma \mid\!\approx_4 \gamma$) iff all models of Σ satisfy γ.

Given a KB Σ and an assertion A, we define the *maximal degree of truth* of A with respect to Σ (denoted by $Maxdeg(\Sigma, A)$) to be $\max\{n > 0 : \Sigma \mid\!\approx_4 (A \geq n)\}$ ($\max \emptyset = 0$). Notice that $\Sigma \mid\!\approx_4 (A \geq n)$ iff $Maxdeg(\Sigma, A) \geq n$.

8.3.1.2 Syntax and semantics of fuzzy specialisations

We have seen that in $\mathcal{ALC}$ a generic specialisation is of the form $C \Rightarrow D$ whose first-order view is of the form $\forall x.C(x) \rightarrow D(x)$. A natural extension to the fuzzy case is the following. From a syntax point of view, a fuzzy specialisation is an expression of type $(C \Rightarrow D \geq n)$, where C, D are $\mathcal{ALC}$ concepts and $n \in [0,1]$. From a semantics point of view, we have that

$$\begin{array}{lll} |C \Rightarrow D|^t & = & \inf_{d \in \Delta^{\mathcal{I}}}\{\max\{1 - |C|^t(d), |D|^t(d)\}\} \\ |C \Rightarrow D|^f & = & \sup_{d \in \Delta^{\mathcal{I}}}\{\min\{|C|^t(d), |D|^f(d)\}\}. \end{array} \tag{8.43}$$

A consequence of this is that if $n \in [0,1]$ then $|C \Rightarrow D|^t \geq n$ iff for all $d \in \Delta^{\mathcal{I}}$ if $|C|^t(d) > 1-n$ then $|D|^t(d) \geq n$. In other words, by abuse of notation, $(C \Rightarrow D \geq n)$ is viewed as the FOL formula

$$\forall x.((\neg C(x) \vee (D(x)) \geq n),$$

i.e. $(C \Rightarrow D \geq n)$ is

$$\forall x.(C(x) \leq 1-n) \vee (D(x) \geq n).$$

Therefore,

$$\{(C(a) \geq m), (C \Rightarrow D \geq n)\} \mid\!\approx_4 (D(a) \geq n) \text{ if } m > 1-n. \tag{8.44}$$

A drawback of the above property is that whatever the degree m is (as long as $m > 1-n$), from $(C(a) \geq m)$ and $(C \Rightarrow D \geq n)$ we infer $(D(a) \geq n)$, where n is a priori fixed value. In particular, for $m_1 > m_2$ and $m_1, m_2 > 1-n$ it follows that

$$\begin{array}{rcll} \text{if } \Sigma & = & \{(C(a) \geq m_1), (C(b) \geq m_2), (C \Rightarrow D \geq n)\} & \\ \text{then} & & & \\ & & Maxdeg(\Sigma, D(a)) = n \text{ and} & \\ & & Maxdeg(\Sigma, D(b)) = n, & \end{array} \tag{8.45}$$

contrary to the intuition that $Maxdeg(\Sigma, D(a))$ should be greater than $Maxdeg(\Sigma, D(b))$. As it is easily verified, the problem relies on the semantics of fuzzy specialisations.

Example 21 Suppose we have two images $\mathsf{i1}$ and $\mathsf{i2}$. In image $\mathsf{i1}$ we recognise with degree .6 a Ferrari, whereas in image $\mathsf{i2}$ we recognise with degree .8 a Porsche. Of course, both are cars. Let us consider

$$\begin{array}{rcl} \Sigma & = & \{(\mathsf{About}(\mathsf{i1}, \mathsf{c1}) \geq .6), (\mathsf{About}(\mathsf{i2}, \mathsf{c2}) \geq .8), \\ & & (\mathsf{Ferrari}(\mathsf{c1}) \geq 1), (\mathsf{Porsche}(\mathsf{c2}) \geq 1), \\ & & (\mathsf{Ferrari} \Rightarrow \mathsf{Car} \geq 1), \\ & & (\mathsf{Porsche} \Rightarrow \mathsf{Car} \geq 1)\}. \end{array}$$

If we are looking for images in which there is a car, then from Σ we may infer that $\Sigma \mid\!\approx_4 ((\exists \mathsf{About.Car})(\mathsf{i1}) \geq 1)$ and $\Sigma \mid\!\approx_4 ((\exists \mathsf{About.Car})(\mathsf{i2}) \geq 1)$, contrary to our intuition on considering image $\mathsf{i2}$ about a car to a stronger extent than image $\mathsf{i1}$. ■

In the following we will present an alternative semantics for specialisations which overcomes the above problem.

Syntax: A *fuzzy specialisation* is an expression of type $C \rightarrow D$, where C and D are $\mathcal{ALC}$ concepts. We will use γ as metavariable for fuzzy specialisation. The intended meaning of a fuzzy specialisation $C \rightarrow D$ is: for all instances a of concept C, if $C(a)$ has membership degree n then $D(a)$ has membership degree n too.

Semantics: Given a fuzzy interpretation $\mathcal{I}$, additionally $(\cdot)^{\mathcal{I}}$ has to satisfy

$$\begin{array}{lll} t \in (C \to D)^{\mathcal{I}} & \text{iff} & \forall d \in \Delta^{\mathcal{I}}\ \forall n \in (0,1].\ |C|^t(d) \geq n \text{ implies } |D|^t(d) \geq n \\ f \in (C \to D)^{\mathcal{I}} & \text{iff} & \exists d \in \Delta^{\mathcal{I}}\ \exists n \in [0,1].\ |C|^t(d) \geq n \text{ and } |D|^t(d) < n. \end{array} \tag{8.46}$$

It is easily verified that the above semantics is an adaption of the semantics of HORN-$\mathcal{L}^f$ rules to the DL case. In fact, a first-order point of view, $C \to D$ is viewed as the formula

$$\forall x \forall y \in [0,1].(C(x) \geq y) \to (D(x) \geq y),$$

where $F \to G$ is interpreted in terms of the horn connective $\leftarrow$, *i.e.* as $G \leftarrow F$ (rather than as $\neg F \vee G$). Therefore, $C \to D$ is

$$\forall x \forall y \in [0,1].(C(x) < y) \vee (D(x) \geq y). \tag{8.47}$$

The definitions of *satisfiability* and entailment are extended to specialisations in the usual way.

It is straightforward to verify that for $m_1 > m_2$

$$\begin{array}{lll} \Sigma & = & \{(C(a) \geq m_1), (C(b) \geq m_2), C \to D\} \\ & \text{implies} & \\ & & Maxdeg(\Sigma, D(a)) = m_1 \text{ and} \\ & & Maxdeg(\Sigma, D(b)) = m_2, \end{array} \tag{8.48}$$

according to our intuition about specialisations.

Example 22 Consider Example 21, *i.e.* in image $\mathsf{i1}$ we recognise with degree .6 a Ferrari, whereas in image $\mathsf{i2}$ we recognise with degree .8 a Porsche. We assume both that a Ferrari and a Porsche are a car. Let us consider

$$\begin{array}{lll} \Sigma & = & \{(\mathsf{About}(\mathsf{i1}, \mathsf{c1}) \geq .6), (\mathsf{About}(\mathsf{i2}, \mathsf{c2}) \geq .8), \\ & & (\mathsf{Ferrari}(\mathsf{c1}) \geq 1), (\mathsf{Porsche}(\mathsf{c2}) \geq 1), \\ & & \mathsf{Ferrari} \to \mathsf{Car}, \\ & & \mathsf{Porsche} \to \mathsf{Car}\}. \end{array}$$

If we are looking for images in which there is a car, then from Σ we may infer that $\Sigma \models_4 ((\exists \mathsf{About}.\mathsf{Car})(\mathsf{i1}) \geq .6)$ and $\Sigma \models_4 ((\exists \mathsf{About}.\mathsf{Car})(\mathsf{i2}) \geq .8)$, and thus, reasonably image $\mathsf{i2}$ is considered about a car to a stronger extent than image $\mathsf{i1}$. ■

8.3.1.3 Properties of four-valued fuzzy $\mathcal{ALC}$

A fuzzy assertion $(A \geq n)$ is in *Negation Normal Form* – respectively, in *Conjunctive Normal Form* (CNF) – iff A is in NNF – respectively in CNF. Similarly, a fuzzy specialisation $C \to D$ is in *Negation Normal Form* – respectively, in *Conjunctive Normal Form* (CNF) – iff C and D are in NNF – respectively in CNF. Since every concept can be transformed into an equivalent concept in NNF (respectively in CNF), as for the non fuzzy case, without loss of generality, we can restrict our attention to fuzzy assertions and fuzzy specialisations in NNF only.

Let Σ be a fuzzy $\mathcal{ALC}$ KB. With $\overline{\Sigma}$ we indicate the (crisp) KB (see also (8.8))

$$\overline{\Sigma} = \{A : (A \geq n) \in \Sigma\} \cup \{C \Rightarrow D : C \to D \in \Sigma\}. \tag{8.49}$$

Concerning the part about fuzzy assertions, it is straight obvious that fuzzy $\mathcal{ALC}$ is nothing else than an extension of $\mathcal{L}_+^f$ to the DLs case. Therefore, as $\mathcal{L}_+^f$ inherits mostly all the properties of its non fuzzy counter-part $\mathcal{L}_+$, we will see that fuzzy $\mathcal{ALC}$ inherits mostly all the properties of its non fuzzy counter-part $\mathcal{ALC}$.

At first, a similar proposition as Proposition 10 holds in fuzzy $\mathcal{ALC}$ too.

Proposition 14 *Let Σ be an $\mathcal{ALC}$ fuzzy KB and A an $\mathcal{ALC}$ assertion. For all $n > 0$, if $\Sigma \mid\!\!\approx_4 (A \geq n)$ then $\overline{\Sigma} \models_4^A A$.* ⊣

Proof: Straightforward extension of proof of Proposition 6. Q.E.D.

As for the propositional case, Proposition 14 states that *there cannot be fuzzy entailment without entailment in $\mathcal{ALC}$*.

Example 23 Consider, the reasoning by cases example at Page 89. Consider the following fuzzy KB

$$\begin{array}{rcl} \Sigma_B & = & \{(\mathsf{Onstack}(\mathsf{s},\mathsf{a}) \geq .6), (\mathsf{Onstack}(\mathsf{s},\mathsf{b}) \geq .7), (\mathsf{Onstack}(\mathsf{s},\mathsf{c}) \geq .8), \\ & & (\mathsf{Block}(\mathsf{a}) \geq .9), (\mathsf{Block}(\mathsf{b}) \geq .8), (\mathsf{Block}(\mathsf{c}) \geq .7), \\ & & (\mathsf{On}(\mathsf{a},\mathsf{b}) \geq .6), (\mathsf{On}(\mathsf{b},\mathsf{c}) \geq .7), \\ & & (\mathsf{Green}(\mathsf{a}) \geq .8), (\neg\mathsf{Green}(\mathsf{c}) \geq .9)\} \end{array} \tag{8.50}$$

consider the query

$$\mathsf{C}(\mathsf{s}) = (\exists \mathsf{Onstack}.\mathsf{Block} \sqcap \mathsf{Green})(\mathsf{s}) \tag{8.51}$$

asking whether there is a green block on the stack s, and the query

$$\mathsf{C}_B(\mathsf{s}) = (\exists \mathsf{Onstack}.\mathsf{Block} \sqcap \mathsf{Green} \sqcap (\exists \mathsf{On}.(\mathsf{Block} \sqcap \neg\mathsf{Green})))(\mathsf{s}) \tag{8.52}$$

asking whether there is a green block on a non-green block in the stack s.

Consider $\mathsf{C}(\mathsf{s})$. Since

$$\{(\mathsf{Onstack}(\mathsf{s},\mathsf{a}) \geq .6), (\mathsf{Block}(\mathsf{a}) \geq .9), (\mathsf{Green}(\mathsf{a}) \geq .8)\} \subset \Sigma_B,$$

from $\min\{.6, .9, .8\} = .6$ it follows that $\Sigma_B \mid\!\!\approx_4 (\mathsf{C}(\mathsf{s}) \geq .6)$. Moreover, it is easy to see that $Maxdeg(\Sigma_B, \mathsf{C}(\mathsf{s})) = .6$.

With respect to $\mathsf{C}_B(\mathsf{s})$ we have, $\Sigma_B \not\mid\!\!\approx_4 (\mathsf{C}_B(\mathsf{s}) \geq n)$, for all $n > 0$. This is a direct consequence of Proposition 14 and of the fact that reasoning by cases does not hold in crisp four-valued $\mathcal{ALC}$.

It worth noticing that the two-valued case is more sophisticated. In fact, let $n > 0$. Consider a two-valued fuzzy interpretation $\mathcal{I}$ satisfying Σ_B. In $\mathcal{I}$ either $|\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}}) \geq n$ or $|\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}}) < n$, *i.e.* $|\neg\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}}) > 1 - n$ (note that $|\cdot|^f = 1 - |\cdot|^t$).

If $|\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}}) \geq n$, then

$$|\mathsf{C}_B|^t(\mathsf{s}^{\mathcal{I}}) \geq \min\{.7, |\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}})\}$$

whereas, if $|\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}}) < n$ then

$$|\mathsf{C}_B|^t(\mathsf{s}^{\mathcal{I}}) \geq \min\{.6, |\neg\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}})\}.$$

Therefore,

$$\Sigma_B \not\approx_2 (\mathsf{C}_B(\mathsf{s}) \geq \min\{.6, .7, |\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}}), |\neg\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}})\}).$$

Since,

$$\min\{.6, .7, |\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}}), |\neg\mathsf{Green}|^t(\mathsf{b}^{\mathcal{I}})\} \geq .5$$

it follows that with respect to two-valued semantics,

$$Maxdeg(\Sigma_B, \mathsf{C}_B(\mathsf{s})) = .5.$$

■

As already specified, our four-valued fuzzy semantics for the $\forall$ connective follows the type A semantics. In particular, there is a restricted form of modus ponens, called *modus ponens on roles*: for all concepts C, for any role R, and for all individuals a, b,

$$\{((\forall R.D)(a) \geq n), (R(a,b) \geq m)\} \not\approx_4 (D(b) \geq n) \text{ if } m > 1 - n \tag{8.53}$$

$$\{((\forall R.D)(a) \geq n), ((\exists R.C)(a) \geq m)\} \not\approx_4 ((\exists R.C \sqcap D)(a) \geq \min\{n, m\}) \text{ if } m > 1 - n \tag{8.54}$$

which are quite similar to Equation (7.74) and Equation (7.75) of the crisp $\mathcal{ALC}$ case, and is strictly related to Equation (8.20). As a consequence, the inverse of Proposition 14 does not hold. In fact just consider that

$$\{((\forall R.D)(a) \geq .6), (R(a,b) \geq .2)\} \not\approx_4 (D(b) \geq n), \tag{8.55}$$

for all $n > 0$, whereas

$$\{(\forall R.C)(a), R(a,b)\} \models_4 C(b).$$

We can give a weaker form of the inverse of Proposition 14, by observing that, if $\overline{\Sigma} \models_4^B A$, *i.e.* no modus ponens over roles has been applied, then $\Sigma \not\approx_4 (A \geq n)$, for some $n > 0$.

Proposition 15 *Let Σ be an $\mathcal{ALC}$ fuzzy KB and A an $\mathcal{ALC}$ assertion. If $\overline{\Sigma} \models_4^B A$ then $\Sigma \not\approx_4 (A \geq n)$ for some $n > 0$.* ⊣

Finally, Proposition 14 and Proposition 15 are applicable to fuzzy specialisations, too. In particular, it follows that

Proposition 16 *Let Σ be an $\mathcal{ALC}$ fuzzy KB and $C \rightarrow D$ a fuzzy specialisation. Then $\Sigma \not\approx_4 C \rightarrow D$ iff $\overline{\Sigma} \models_4^B C \Rightarrow D$.* ⊣

which allows us to reduce fuzzy subsumption to subsumption w.r.t. type B semantics.

8.3.2 The logic fuzzy HORN-$\mathcal{ALC}$

We conclude this chapter by presenting the logic fuzzy HORN-$\mathcal{ALC}$. Essentially, this logic is an extension of HORN-$\mathcal{ALC}$ to the fuzzy case, *i.e.* with features we have seen for HORN-$\mathcal{L}^f$. In particular, almost all properties are the result of the combination HORN-$\mathcal{L}^f$ and HORN-$\mathcal{ALC}$ (see Section 8.2.3 and Section 7.4.2). The logic fuzzy HORN-$\mathcal{ALC}$ will be our final logic.
At first, we extend the definition of fuzzy degree function.

Let $\mathcal{W}$ be a set of $\mathcal{ALC}$ objects ($\mathcal{ALC}$ individuals or variables) and let $\mathcal{F}$ be a set of *degree functions* such that

$$\begin{array}{rcll} \mathcal{F} & = & \{f\colon [0,1]^k \times \mathcal{W}^m \to [0,1]: & k = 1,2,\ldots, \\ & & & m = 1,2,\ldots, \\ & & & f \text{ defined on } [0,1]^k \times \mathcal{W}^m, \\ & & & f \text{ nondecreasing on the first } k \text{ arguments}\}. \end{array}$$

A $k+m$-ary degree function $f \in \mathcal{F}$ takes as input k degrees $n_1, \ldots, n_k$ in $[0,1]$ and m objects $w_1, \ldots, w_m$. $f(n_1, \ldots, n_k, w_1, \ldots, w_m)$ gives us a degree in $[0,1]$ depending on the values of $n_1, \ldots, n_k, w_1, \ldots, w_m$. Moreover, we assume that for all $n \in [0,1]$, $n \in \mathcal{F}$, *i.e.* the constant function returning n is a degree function.

For instance, suppose we would like to specify a degree function $f(h)$ which defines the degree $n = f(h)$ of being tall in terms of the height $h = \mathsf{height}(x)$ of a person x. A simple function may be (the height is expressed in cm)

$$f_{\mathsf{height}}(h) = \min\{1, (h/200)^2\}. \tag{8.56}$$

The above function specifies that a person whose height is greater or equal than 200 is definitely a tall person: *e.g.* if $180 = \mathsf{height}(\mathsf{tom})$ then $.81 = f_{\mathsf{height}}(180)$, *i.e.* tom is likely tall. On the other hand, if $225 = \mathsf{height}(\mathsf{tom})$ then $1 = f_{\mathsf{height}}(225)$, *i.e.* tom is definitely tall. Just notice that $f_{\mathsf{height}}(h)$ depends only on the value (object) h. Now suppose we are unable to determine precisely the height of tom. For instance, we guess its height from a picture, *e.g.* we determine that $180 = \mathsf{height}(\mathsf{tom})$ only to a degree n, say .7. It is certainly desirable to define f_{height} not only parametric w.r.t. height h but also parametric w.r.t. the degree of confidence we have on the value h. A function which does the job is

$$f_{\mathsf{height}}(n, h) = \min\{1, n \cdot (h/200)^2\}. \tag{8.57}$$

Just notice that $f_{\mathsf{height}}(n, h)$ is nondecreasing on n, which is compatible with our intuition that if the degree of confidence n w.r.t. the data h grows, then the degree of being tall $f_{\mathsf{height}}(n, h)$ does not decrease.

A *horn rule* in fuzzy HORN-$\mathcal{ALC}$ is an expression of the form

$$(P(\vec{X}) \geq V) \leftarrow (P_1(\vec{X}_1) \geq V_1), \ldots, (P_n(\vec{X}_n) \geq V_n), \langle V, f(\vec{V}, \vec{Y})\rangle, \tag{8.58}$$

with $n \geq 1$, and

1. $\vec{X}_1, \ldots, \vec{X}_n, \vec{X}$ are tuples of horn variables or individuals;
2. a horn variable which appears in $\vec{X}$ also appears in $\vec{X}_1, \ldots, \vec{X}_n$;
3. $\vec{Y}$ is the tuple of objects appearing in $\vec{X}_1, \ldots, \vec{X}_n$;

4. $V, V_1, \ldots, V_n$ are distinct fuzzy variables and f is a degree function;
5. $\vec{V}$ is the tuple of variables $V_1, \ldots, V_n$;
6. the predicates $P, P_1, \ldots P_n$ are HORN-$\mathcal{ALC}$ predicates, *i.e.* they are either primitive concepts, roles or ordinary predicates.
7. $(P(\vec{X}) \geq V)$ is called *head* and $(P_1(\vec{X}_1) \geq V_1)$, ..., $(P_n(\vec{X}_n) \geq V_n)$, $\langle V, f(V_1, \ldots, V_n)\rangle$ is called *body*.

Of course, horn rules are universally quantified on the fuzzy variables $V, V_1, \ldots, V_n$ and on the horn variables occurring in $\vec{X}_1, \ldots, \vec{X}_n, \vec{X}$.

For instance, the horn rule

$$(\mathsf{Tall}(X) \geq V) \leftarrow (\mathsf{Person}(X) \geq V_1), (\mathsf{Height}(X, Y) \geq V_2), \langle V, f_{\mathsf{height}}(V_2, Y)\rangle \tag{8.59}$$

establishes that a person X is tall to a degree $V = f_{\mathsf{height}}(V_2, Y)$ if its height is Y with precision V_2 and f_{height} is defined as in Equation 8.57.

A *fact* (denoted by $(P(\vec{w}) \geq n)$) is an expression of the form $(A \geq n)$, such that A is either an $\mathcal{ALC}$ assertion (where an $\mathcal{ALC}$ variable can occur) or a A is ground instance $P(w_1, \ldots, w_n)$ of some ordinary predicate and $w_1, \ldots, w_n$ are objects. For instance, $(\mathsf{Person} \sqcap \exists \mathsf{Friend}.\mathsf{President})(\mathsf{tom}) \geq .7$ and $(\mathsf{Height}(\mathsf{tom}, 180) \geq .6)$ are facts. It is worth anticipating that the semantics of multimedia objects will be described through fuzzy HORN-$\mathcal{ALC}$ facts.

A *goal* is an expression of the form

$$\begin{array}{rl} \leftarrow & (P_1(\vec{X}_1) \geq V_1), \ldots, (P_n(\vec{X}_n) \geq V_n), \\ & \langle V_{f^1}, f^1(\vec{V}_{f^1}, \vec{Y}_{f^1})\rangle, \ldots, \\ & \langle V_{f^k}, f^k(\vec{V}_{f^k}, \vec{Y}_{f^k})\rangle : VR \end{array} \tag{8.60}$$

with $n \geq 0$, $P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$ HORN-$\mathcal{ALC}$ predicates, f^i fuzzy degree functions and VR a set of fuzzy value restrictions. The case $n = 0, k = 0$ is called *empty goal* and is indicated with $\leftarrow \blacksquare : VR$. Moreover, the following conditions are satisfied:

1. for all $1 \leq i < j \leq k$, $\vec{Y}_{f^i} \cap \vec{Y}_{f^j} = \emptyset$;
2. the fuzzy variables $V_1, \ldots, V_n, V_{f^1}, \ldots, V_{f^k}$ are all distinct;
3. for all $1 \leq i < j \leq k$, $\vec{V}_{f^i} \cap \vec{V}_{f^j} = \emptyset$;
4. there is $1 \leq l \leq k$ such that V_{f^l} occurs in no $\vec{V}_{f^i}$, $1 \leq i \leq k$;
5. for each $1 \leq i \leq k$, $i \neq l$, there is exactly one l_i such that V_{f^i} occurs in $vec V_{f^{l_i}}$.
6. $\{V_1, \ldots, V_n\} = (\bigcup_{1 \leq i \leq k} \vec{V}_{f^i}) \setminus \{V_{f^1}, \ldots, V_{f^k}\}$.

An example of goal is

$$\leftarrow (\mathsf{Person}(X) \geq V_1), (\mathsf{Height}((X, Y) \geq V_3), \langle V, f^1(V_1, V_2, X)\rangle, \langle V_2, f^2(V_3, Y)\rangle.$$

Notice that, the above points are satisfied.

A *query* is an expression of the form

$$\exists \vec{X} \exists \vec{V}.(P_1(\vec{X}_1) \geq V_1) \wedge \ldots \wedge (P_n(\vec{X}_n) \geq V_n), \tag{8.61}$$

with $n \geq 1$, $\vec{X}$ is the tuple of horn variables appearing in $\vec{X}_1, \ldots, \vec{X}_n$, $\vec{V}$ is the tuple of distinct fuzzy variables $V_1, \ldots, V_n$ and $P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$ are HORN-$\mathcal{ALC}$ predicates.

The *associated* goal G_Q of a query $Q = \exists \vec{X} \exists \vec{V}.(P_1(\vec{X}_1) \geq V_1) \wedge \ldots \wedge (P_n(\vec{X}_n) \geq V_n)$ is the goal

$$\leftarrow (P_1(\vec{X}_1) \geq V_1), \ldots, (P_n(\vec{X}_n) \geq V_n), \langle V, 1 \rangle : \emptyset,$$

where V is a new fuzzy variable.

As usual, a fuzzy HORN-$\mathcal{ALC}$ KB Σ is a finite well formed set of $\mathcal{ALC}$ specialisations and definitions, horn rules and facts. With $\Sigma_T \subseteq \Sigma$ we indicate the set of specialisations and definitions in Σ, with $\Sigma_R \subseteq \Sigma$ we indicate the set of horn rules in Σ and with $\Sigma_F \subseteq \Sigma$ we indicate the set of facts in Σ. A *horn* KB is a finite set of horn rules and horn facts. Note that in a horn KB Σ, the terminology Σ_T is empty. We will say that a HORN-$\mathcal{ALC}$ KB Σ is *safe* iff for all rules R, the head of R involves ordinary predicates only. Finally, the definition of *recursive* fuzzy HORN-$\mathcal{ALC}$ KBs Σ is as usual.

From a semantics point of view, we combine the definitions of Section 8.2.3 with those of Section 7.4.2. Let $\mathcal{I}$ be an interpretation and let f be a degree function. Then $\mathcal{I}$ has to satisfy the condition

$$t \in \langle V, f(\vec{V}, \vec{Y}) \rangle^{\mathcal{I}} \quad \text{iff} \quad V^{\mathcal{I}} \leq f(\vec{V}^{\mathcal{I}}, \vec{Y}^{\mathcal{I}}). \tag{8.62}$$

$\mathcal{I}$ *satisfies* $\langle V, f(\vec{V}, \vec{Y}) \rangle$ iff $t \in \langle V, f(\vec{V}, \vec{Y}) \rangle^{\mathcal{I}}$. Moreover, we extend $|\cdot|^t$ and $|\cdot|^f$ to n-ary ordinary predicate P, as usual:

$$|P|^t \colon \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}} \to [0,1]$$
$$|P|^f \colon \Delta^{\mathcal{I}} \times \ldots \times \Delta^{\mathcal{I}} \to [0,1]$$

and

$$t \in (P(\vec{w}) \geq n)^{\mathcal{I}} \text{ iff } |P|^t(\vec{w}^{\mathcal{I}}) \geq n$$
$$f \in (P(\vec{w}) \geq n)^{\mathcal{I}} \text{ iff } |P|^f(\vec{w}^{\mathcal{I}}) \geq n.$$

$\mathcal{I}$ *satisfies* a horn fact $(P(\vec{w}) \geq n)$ iff $t \in (P(\vec{w}) \geq n)^{\mathcal{I}}$.

With respect to horn rules, goals and queries we have:

1. let R be the horn rule

$$(P(\vec{X}) \geq V) \leftarrow (P_1(\vec{X}_1) \geq V_1), \ldots, (P_n(\vec{X}_n) \geq V_n), \langle V, f(\vec{V}, \vec{Y}) \rangle.$$

where $\vec{Y}$ is the tuple of all the k horn variables occurring in R and $\vec{W}$ is the tuple of all the $n+1$ fuzzy variables appearing in R. An interpretation $\mathcal{I}$ *satisfies* horn rule R iff for all $\vec{m} \in [0,1]^{n+1}$, for all $\vec{d} \in (\Delta^{\mathcal{I}})^k$ such that $\mathcal{I}^{\vec{m},\vec{d}}_{\vec{W},\vec{Y}}$ satisfies $\langle V, f(\vec{V}, \vec{Y}) \rangle$, if $\mathcal{I}^{\vec{m},\vec{d}}_{\vec{W},\vec{Y}}$ satisfies all $(P_1(\vec{X}_1) \geq V_1)$, …, $(P_n(\vec{X}_n) \geq V_n)$, then $\mathcal{I}^{\vec{m},\vec{d}}_{\vec{W},\vec{Y}}$ satisfies $(P(\vec{X}) \geq V)$;

2. let G be the goal

$$\begin{array}{rl} \leftarrow & (P_1(\vec{X}_1) \geq V_1), \ldots, (P_n(\vec{X}_n) \geq V_n), \\ & \langle V_{f^1}, f^1(\vec{V}_{f^1}, \vec{Y}_{f^1})\rangle, \ldots, \\ & \langle V_{f^k}, f^k(\vec{V}_{f^k}, \vec{Y}_{f^k})\rangle : VR. \end{array}$$

Let $\vec{Y}$ be the tuple of all the k horn variables occurring in R and let $\vec{V}$ be the tuple of all the l fuzzy variables appearing in R. An interpretation $\mathcal{I}$ *satisfies* goal G iff $n + k \geq 1$ and for all $\vec{m} \in [0,1]^l$ such that $\mathcal{I}_{\vec{V}}^{\vec{m}}$ satisfies VR, for all $\vec{d} \in (\Delta^{\mathcal{I}})^k$ such that $\mathcal{I}_{\vec{V},\vec{Y}}^{\vec{m},\vec{d}}$ satisfies all $\langle V_{f^j}, f^j(\vec{V}_{f^j}, \vec{Y}_{f^j})\rangle$, $\mathcal{I}_{\vec{V},\vec{Y}}^{\vec{m},\vec{d}}$ does not satisfy some $(P_i(\vec{X}_i) \geq V_i)$, $1 \leq i \leq n$;

3. let Q be the query

$$\exists \vec{X} \exists \vec{V}.(P_1(\vec{X}_1) \geq V_1) \wedge \ldots \wedge (P_n(\vec{X}_n) \geq V_n),$$

where $\vec{V}$ is an l-tuple and $\vec{X}$ is an k-tuple. An interpretation $\mathcal{I}$ *satisfies* a query Q iff for some $\vec{m} \in [0,1]^l$ and for some $\vec{d} \in (\Delta^{\mathcal{I}})^k$, $\mathcal{I}_{\vec{V},\vec{Y}}^{\vec{m},\vec{d}}$ satisfies all $(P_1(\vec{X}_1) \geq V_1)$, $\ldots$, $(P_n(\vec{X}_n) \geq V_n)$.

Note that the empty goal is never satisfied.

An interpretation $\mathcal{I}$ *satisfies* (*is a model of*) a fuzzy HORN-$\mathcal{ALC}$ KB Σ iff $\mathcal{I}$ satisfies each element of it. Satisfiability is extended to an arbitrary set S of horn rules, facts and goals, as usual. Finally, a fuzzy HORN-$\mathcal{ALC}$ KB Σ *entails* a query Q (denoted by $\Sigma \mid\!\approx_4 Q$) iff every model of Σ satisfies Q. We extend the definition entailment to ground facts as follows: Σ *entails* a ground fact $(P(\vec{w}) \geq n)$ (denoted by $\Sigma \mid\!\approx_4 (P(\vec{w}) \geq n)$) iff every model of Σ satisfies $(P(\vec{w}) \geq n)$.

Let Q be a query $\exists \vec{X} \exists \vec{V}.\ (P_1(\vec{X}_1) \geq V_1) \wedge \ldots \wedge (P_n(\vec{X}_k) \geq V_k)$. An *answer* to the query Q is a substitution θ of all horn variables and fuzzy variables in Q, *i.e.* $\theta = \{X_1/w_1, \ldots, X_q/w_q, V_1/n_1, \ldots, V_k/n_k\}$. We define $\theta^{\vec{X}} = \{X_1/w_1, \ldots, X_q/w_q\}$ and $\theta^{\vec{V}} = \{V_1/n_1, \ldots, V_k/n_k\}$. Of course, $\theta^{\vec{X}} \cap \theta^{\vec{V}} = \emptyset$ and $\theta = \theta^{\vec{X}} \cup \theta^{\vec{V}}$. If θ is an answer, with $\vec{\theta}$ we will denote the tuple $(n_1, \ldots, n_k)$.

An answer is *correct* w.r.t. a fuzzy HORN-$\mathcal{ALC}$ KB Σ iff $\Sigma \mid\!\approx_4 Q\theta$. As usual, the *answer set* of a query Q w.r.t. a fuzzy KB Σ (denoted by $AnswerSet(\Sigma, Q)$) is the set of correct answers, *i.e.*

$$AnswerSet(\Sigma, Q) \quad = \quad \{\theta : \Sigma \mid\!\approx_4 Q\theta\}. \tag{8.63}$$

Now, let $l, h \in [0,1]$ be two reals, let $\vec{n} = (n_1, \ldots, n_k)$ and $\vec{m} = (m_1, \ldots, m_k)$ be two tuples of reals in $[0,1]$, let $N = \{\vec{n}_1, \ldots, \vec{n}_r, \ldots\}$ be a set of reals and tuples of reals in $[0,1]$. We assume, $l \uparrow h$, $\vec{n} \uparrow \vec{m}$, $\uparrow N$, $l \downarrow h$, $\vec{n} \downarrow \vec{m}$, $\downarrow N$, $\vec{n} \geq \vec{m}$, $\vec{n} \leq \vec{m}$, $\vec{n} > \vec{m}$ and $\vec{n} < \vec{m}$ defined as in Equation (8.33). The analogue definitions concerning answers θ is more complex as objects are involved in θ. So, let $\theta_1 = \{X_1/w_1, \ldots, X_q/w_q, V_1/n_1, \ldots, V_k/n_k\}$ and $\theta_2 = \{X_1/v_1, \ldots, X_q/v_q, V_1/m_1, \ldots, V_k/m_k\}$ be two *non empty* answers of query Q w.r.t. KB Σ. We define[3]

[3]See Equation (8.34) for the definitions of $\theta_1^{\vec{V}} \uparrow \theta_2^{\vec{V}}$ and $\theta_1^{\vec{V}} \downarrow \theta_2^{\vec{V}}$.

$$\begin{array}{lcl}
\theta_1 \uparrow \theta_2 & = & \left\{ \begin{array}{ll} \emptyset & \text{if } \theta_1^{\vec{X}} \neq \theta_2^{\vec{X}} \\ \theta_1^{\vec{X}} \cup (\theta_1^{\vec{V}} \uparrow \theta_2^{\vec{V}}) & \text{otherwise} \end{array} \right. \\
\theta_1 \downarrow \theta_2 & = & \left\{ \begin{array}{ll} \emptyset & \text{if } \theta_1^{\vec{X}} \neq \theta_2^{\vec{X}} \\ \theta_1^{\vec{X}} \cup (\theta_1^{\vec{V}} \downarrow \theta_2^{\vec{V}}) & \text{otherwise} \end{array} \right.
\end{array} \tag{8.64}$$

$$\begin{array}{lll}
\theta_1 \geq \theta_2 & \text{iff} & \theta_1 = \theta_1 \uparrow \theta_2 \\
\theta_1 \leq \theta_2 & \text{iff} & \theta_2 \geq \theta_1 \\
\theta_1 > \theta_2 & \text{iff} & \theta_1 \geq \theta_2, \theta_1 \neq \theta_2 \\
\theta_1 < \theta_2 & \text{iff} & \theta_2 > \theta_1.
\end{array}$$

Unlike the case of HORN-$\mathcal{L}^f$, if Σ is a HORN-$\mathcal{L}^f$ KB, Q is a query and $\theta_1, \theta_2 \in AnswerSet(\Sigma, Q)$ then it could be that neither $\theta_1 \geq \theta_2$ nor $\theta_2 \geq \theta_1$. So, let $\Theta = \{\theta_1, \ldots, \theta_s, \ldots\}$ be a set of answers of query Q w.r.t. KB Σ. We define,

$$\begin{array}{lcl}
\uparrow \Theta & = & \{\theta \in \Theta : \theta \text{ maximal according to } \leq\} \\
\downarrow \Theta & = & \{\theta \in \Theta : \theta \text{ minimal according to } \leq\}
\end{array} \tag{8.65}$$

Example 24 Let Σ be the following fuzzy HORN-$\mathcal{ALC}$ KB.

$$\begin{array}{lcl}
\Sigma & = & \{(A(a) \geq .2), (B(a) \geq .7), \\
 & & (A(b) \geq .4), (B(b) \geq .6)\}
\end{array}$$

and consider the query Q

$$Q = \exists X \exists V_1, V_2 . (A(X) \geq V_1) \wedge (B(X) \geq V_2).$$

Consider the two correct answers

$$\begin{array}{lcl}
\theta_1 & = & \{X/a, V_1/.2, V_2/.7\} \\
\theta_2 & = & \{X/b, V_1/.4, V_2/.6\}.
\end{array}$$

It is easily verified that each answer θ such that $\theta \leq \theta_1$ or $\theta \leq \theta_2$ is a correct answer, *i.e.* $\theta = \{X/a, V_1/m, V_2/n\}$ with $m \leq .2, n \leq .7$, or $\theta = \{X/b, V_1/m, V_2/n\}$ with $m \leq 4, n \leq .6$. As a consequence, the answer set is the set

$$AnswerSet(\Sigma, Q) \quad = \quad \{\theta : \theta \leq \theta_1\} \cup \{\theta : \theta \leq \theta_2\}.$$

Notice that neither $\theta_1 \leq \theta_2$ nor $\theta_2 \leq \theta_1$. In fact, $\theta_1^X = \{X/a\} \neq \{X/b\} = \theta_2^X$. Essentially, θ_1 and θ_2 are incomparable. As a consequence, the maximal elements of $AnswerSet(\Sigma, Q)$ w.r.t. $\leq$ is not unique. In fact,

$$\uparrow AnswerSet(\Sigma, Q) = \{\theta_1, \theta_2\}.$$

■

Finally, let us define

$$Maxdeg(\Sigma, Q) \quad = \quad \uparrow AnswerSet(\Sigma, Q), \tag{8.66}$$

which defines $Maxdeg(\Sigma, Q)$ as the *set of correct and maximal answers.*

Notice that if $\theta \in Maxdeg(\Sigma, Q)$ and $\vec{\theta} = \vec{m}$, then it is possible to define a maximal degree of the query Q to be *e.g.* $\min\{m_1, \ldots, m_n\}$ (of course, any other degree function can be used). In a more principled way, if we were interested in to determine a degree value $m \in [0, 1]$ in terms of $f(m_1, \ldots, m_n)$, then it is sufficient to add rule

$$(P(\vec{X}) \geq V) \leftarrow (P_1(\vec{X}_1) \geq V_1), \ldots, (P_n(\vec{X}_n) \geq V_n), \langle V, f(V_1, \ldots, V_n\rangle$$

to Σ, where P is a new ordinary predicate and $\vec{X}$ is the tuple of all horn variables appearing in the body. Thereafter we have to consider the new query

$$\exists \vec{X} \exists V.(P(\vec{X}) \geq V).$$

Given a query Q and its associated goal G_Q then it is easily verified that

$$\Sigma \models_4 Q \quad \text{iff} \quad \Sigma \cup \{G_Q\} \text{ not satisfiable.} \tag{8.67}$$

As in all horn extensions, we have that in case of horn KBs, equivalence between four-valued fuzzy entailment and two-valued fuzzy entailment holds.

Proposition 17 *Let Σ be a horn fuzzy* HORN-$\mathcal{ALC}$ *KB and let Q be a query. Then $\Sigma \mid\!\approx_4 Q$ iff $\Sigma \mid\!\approx_2 Q$.* ⊣

In the general case where Σ is a fuzzy HORN-$\mathcal{ALC}$ KB, then $\Sigma \mid\!\approx_4 Q$ implies $\Sigma \mid\!\approx_2 Q$, but not vice-versa. In fact, consider $\Sigma = \{((\neg A \vee B)(a) \geq 1), (A(a) \geq 1)\}$. Then $\Sigma \mid\!\approx_2 (B(a) \geq 1)$, but $\Sigma \not\mid\!\approx_4 (B(a) \geq 1)$.

We conclude this section with an example.

Example 25 Consider the following fuzzy HORN-$\mathcal{ALC}$ KB

$$\begin{array}{rcl}
\Sigma & = & \{(\mathsf{Tall}(X) \geq V) \leftarrow (\mathsf{Person}(X) \geq V_1), (\mathsf{Height}(X, Y) \geq V_2), \langle V, f_{\mathsf{height}}(V_2, Y)\rangle, \\
& & \mathsf{GoodFriend} := \mathsf{Person} \sqcap \exists \mathsf{Friend}.\mathsf{President}, \\
& & (\mathsf{Person}(\mathsf{tom}) \geq 1), \\
& & (\mathsf{Friend}(\mathsf{tom}, \mathsf{clinton}) \geq .7), \\
& & (\mathsf{Height}(\mathsf{tom}, 180) \geq .6), \\
& & (\mathsf{President}(\mathsf{clinton}) \geq 1), \\
& & (\mathsf{Person}(\mathsf{tim}) \geq 1), \\
& & (\mathsf{Friend}(\mathsf{tim}, \mathsf{nixon}) \geq .3), \\
& & (\mathsf{Height}(\mathsf{tim}, 225) \geq .6), \\
& & (\mathsf{President}(\mathsf{nixon}) \geq 1)\}.
\end{array}$$

Notice that in the above KB $(\mathsf{Friend}(\mathsf{tom}, \mathsf{clinton}) \geq .7)$ expresses a kind of imprecision whether clinton is tom's friend. Similarly, $(\mathsf{Height}(\mathsf{tom}, 180) \geq .6)$ says us that we are not completely sure that tom's high is 180. The case for tim is similar.

Let Q be the query

$$\exists X \exists V_1, V_2.(\mathsf{Tall}(X) \geq V_1) \wedge (\mathsf{GoodFriend}(X) \geq V_2).$$

i.e. we are asking whether there is tall good friend.

It is easily verified that

$$\Sigma \mid\!\approx_4 (\mathsf{Tall}(\mathsf{tom}) \geq .486)$$
$$\Sigma \mid\!\approx_4 (\mathsf{GoodFriend}(\mathsf{tom}) \geq .7)$$
$$\Sigma \mid\!\approx_4 (\mathsf{Tall}(\mathsf{tim}) \geq .66)$$
$$\Sigma \mid\!\approx_4 (\mathsf{GoodFriend}(\mathsf{tom}) \geq .3)$$

i.e. $\theta_1 = \{X/\mathsf{tom}, V_1/.486, V_2/.7\}$ and $\theta_2 = \{X/\mathsf{tim}, V_1/.66, V_2/.3\}$ are correct answers. From the fact that neither $\theta_1 \leq \theta_2$ nor $\theta_2 \leq \theta_1$ it is easily verified that

$$Maxdeg(\Sigma, Q) =\uparrow AnswerSet(\Sigma, Q) = \{\theta_1, \theta_2\}.$$

■

8.4 Summary

In this part we have described the logic fuzzy HORN-$\mathcal{ALC}$ which will be used in the following Part III in order to describe the semantic meaning of complex multimedia objects defined in the previous Part I. The logic fuzzy HORN-$\mathcal{ALC}$ has several components addressing different topics of modelling:

- a description logic component which allows the representation of structured objects of the real world;
- a horn rule component which allows us to reason about these structured objects;
- a four-valued semantics which allows us to deal both with inconsistency and relevance entailment;
- a fuzzy component which allows the treatment of the inherent imprecision in complex multimedia objects meaning representation and their retrieval.

Part III

Reasoning about form and semantics of multimedia objects

Chapter 9

Preview

In Chapter 4 we formally described a simple object-oriented model for representing multimedia data along the *form* dimension. In particular, we have defined complex single media objects (CSMOs). A CSMO represents nothing else than a portion of data of interest in a single media document. Moreover, CSMOs can be combined together in order to form more complex objects, called complex multimedia objects (CMOs), which allow the aggregation of portions of data of different documents. As a consequence, a CMO allows the representation a portion of data of a multimedia document. A collection of such CMOs constitutes a multimedia database about form (see Section 4.5). Now, given a multimedia database about form DBF (see Equation (4.22)), at least three questions arise.

1. How can we query the database DBF?
2. How can we represent the semantics of the complex multimedia objects in DBF?
3. How can we query a database containing both information about form and information about semantics of complex multimedia objects?

These three questions will be the topic of the following chapters. In particular, *concerning Point 1.*, in order to query DBF we have to formally specify a query language and successively specify what the results of a query w.r.t. DBF are. This will be the topic of the next chapter. We will see that querying the form will be done through fuzzy Horn-$\mathcal{ALC}$, *i.e.* a query w.r.t. DBF will be a fuzzy Horn-$\mathcal{ALC}$ query. In particular, querying DBF through a fuzzy Horn-$\mathcal{ALC}$ query Q will formally be specified by viewing DBF as a concrete domain [31].

On the other hand, *Point 2. and Point 3.* will be the topic of Chapter 11. We will see that the semantics of a CMO can simply be specified through a set of fuzzy Horn-$\mathcal{ALC}$ facts. As a consequence, fuzzy Horn-$\mathcal{ALC}$ queries allow us to query a database along both the form and the semantics dimension in an uniform and principled way. The expressive power of fuzzy Horn-$\mathcal{ALC}$ will allow us to perform a huge variety of queries which go behind the capabilities of the to date existing multimedia systems.

Chapter 10

Reasoning about form

10.1 Formalisation

Given a multimedia database about form DBF, essentially it contains the objects which can be retrieved through user's queries. The natural question is: how can we query DBF and what kind of queries can we perform? We answer to these questions by providing a query language and specifying the semantics of it. There are at least two ways in doing this.

1. The first solution is to define a mapping Ω such that $\Omega(\mathsf{DBF})$ is viewed as deductive database (see *e.g.* [1]). In particular, $\Omega(\mathsf{DBF})$ may be a horn fuzzy HORN-$\mathcal{ALC}$ KB. A query Q is a query in fuzzy HORN-$\mathcal{ALC}$ and, thus, querying DBF by means of query Q is defined through $\Omega(\mathsf{DBF})\mid\!\approx_4 Q$. Hence, the main effort is to formally specify the function $\Omega(\cdot)$ which maps multimedia databases about form into horn fuzzy HORN-$\mathcal{ALC}$ KBs.

2. A second solution is to rely on concrete domains and predicates on them [31]. Examples of concrete domains are integers, string, reals, or also non-arithmetic domains (like relational databases). Essentially, in this view, a DBF will be a concrete domain and where the predicates defined in this domain correspond to the concept names and methods defined in DBF.

While the first method is certainly useful for fast prototyping it certainly suffers from a computational point of view, as anything is performed within logic.

We follow instead the second approach as it delegates the retrieval of complex multimedia objects to an underlying database system implementing a multimedia databases about form.

At first, we define the notion of concrete domain. A *concrete domain* $\mathcal{D}$ consists of a set $dom(\mathcal{D})$, the *domain* of $\mathcal{D}$, and a set $pred(\mathcal{D})$, the *predicate names* of $\mathcal{D}$. Each predicate name P is associated with an arity n, and an n-ary predicate $P^{\mathcal{D}} \subseteq dom(\mathcal{D})^n$.

An example of concrete domain is the following: assume that $\mathcal{DB}$ is an arbitrary relational database equipped with an appropriate query language. Then $\mathcal{DB}$ can be seen as a concrete domain where $dom(\mathcal{DB})$ is the set of atomic values in the database. The predicates of $\mathcal{DB}$ are the relations which can be defined over $\mathcal{DB}$ with the help of the query language.

Moreover, let $P_1, \ldots, P_k$ be k (not necessarily different) predicate names in $pred(\mathcal{D})$ of arities $n_1, \ldots, n_k$. Consider the conjunction

$$\bigwedge_{i=1}^{k} P_i(\vec{x}^i).$$

Here $\vec{x}^i$ stands for an n_i-tuple $(x_1^i, \ldots, x_{n_i}^i)$ of variables. Such a conjunction is said to be *satisfiable* iff there is an assignment of elements of $dom(\mathcal{D})$ to the variables such that the conjunction becomes true in $\mathcal{D}$. We will assume that the satisfiability problem for finite conjunctions of the above mentioned form is decidable.

Now, let

$$\mathsf{DBF} \quad = \quad (\mathsf{CH}, \mathsf{TY}, \mathsf{MET}, \pi, \mathsf{OID})$$

be a multimedia database about form, where $\mathsf{CH} = (\mathsf{CN}, =, \prec)$ is a class hierarchy, CN is a set of class names, TY is the set of types over CN, MET is a set of methods m of classes in CN, π is an OID assignment over CN and OID is a set of objects (o, v) instances of classes in CN.

The *concrete domain* DBF is defined as follows.

1. The domain $dom(\mathsf{DBF})$ is defined as follows. Let V be

$$V = \textsc{Value}_A \cup \textsc{ObjectId} \cup \{nil\}. \tag{10.1}$$

   Then

$$dom(\mathsf{DBF}) = V \cup 2^V. \tag{10.2}$$

   Note that the domain is the set of basic values.

2. The predicate names in $pred(\mathsf{DBF})$ are defined as follows.

   (a) For each class name $\mathsf{C} \in \mathsf{CN}$ there is an unary predicate $\mathsf{c} \in pred(\mathsf{DBF})$ such that

$$\mathsf{c}^{\mathsf{DBF}} = \pi^*(\mathsf{C}). \tag{10.3}$$

   Note that the predicate $\mathsf{c}^{\mathsf{DBF}}$ includes all the instance of the class C, according to the class hierarchy CH.

   (b) For each method $m \in \mathsf{MET}$ (including implicit methods), where $m\colon C \times T_1 \times \ldots \times T_{n-1} \to T_n$, there is a $n+1$-ary predicate $\mathsf{m} \in pred(\mathsf{DBF})$ such that

$$\mathsf{m}^{\mathsf{DBF}} = \{(o, v_1, \ldots, v_{n-1}, v_n) : m(o, v_1, \ldots, v_{n-1}) = v_n\}. \tag{10.4}$$

We will extend fuzzy Horn-$\mathcal{ALC}$ as follows.

Syntax From a syntax point of view, consider a new alphabet $\mathcal{OC}$ of *concrete individuals* and *concrete variables*. A *concrete object* (denoted by w) is either a concrete individual, a concrete variable or a set of concrete individuals and variables. We will assume that there is a fixed bijection δ between $V \subset dom(\mathsf{DBF})$ and concrete individuals. For ease of notation we will assume that for all $v \in V$, $\delta(v) = v$. An element $\{v_1, \ldots, v_n\} \in 2^V \subset dom(\mathsf{DBF})$ will be represented as a prolog term $[\delta(v_1), \ldots, \delta(v_n)]$, where $[\cdot]$ is the usual list operator. Note that the $w_i = \delta(v_i)$ are concrete objects. The empty set is denoted by $[]$ or also with nil. A *concrete atom* is an expression of the form $P(w_1, \ldots, w_n)$, where $P \in pred(\mathsf{DBF})$ is an n-ary predicate and w_i are concrete objects. Any concrete atom may appear in the body of a fuzzy Horn-$\mathcal{ALC}$ rule or in a fuzzy Horn-$\mathcal{ALC}$ query.

Semantics From a semantics point of view we extend interpretations $\mathcal{I}$ w.r.t. DBF for fuzzy HORN-$\mathcal{ALC}$ as follows. The interpretation $\mathcal{I}$ w.r.t. DBF has to be such that

1. $\Delta^{\mathcal{I}} \cap dom(\mathsf{DBF}) = \emptyset$;
2. $w^{\mathcal{I}} \in dom(\mathsf{DBF})$, if w is a concrete variable;
3. $w^{\mathcal{I}} = \delta^{-1}(w) \in dom(\mathsf{DBF})$, if w is a concrete individual, where δ is a fixed bijection between V and $\mathcal{OC}$;
4. $[w_1, \ldots, w_n]^{\mathcal{I}} = \{{w_1}^{\mathcal{I}}, \ldots, {w_n}^{\mathcal{I}}\} \in dom(\mathsf{DBF})$;
5. $t \in P(w_1, \ldots, w_n)^{\mathcal{I}}$ iff $({w_1}^{\mathcal{I}}, \ldots, {w_n}^{\mathcal{I}}) \in P^{\mathsf{DBF}}$;
6. $f \in P(w_1, \ldots, w_n)^{\mathcal{I}}$ iff $({w_1}^{\mathcal{I}}, \ldots, {w_n}^{\mathcal{I}}) \notin P^{\mathsf{DBF}}$.

All the results described in Section 8.3.2 are easily extendible to the case of concrete atoms too.

At last, querying DBF by means of a fuzzy HORN-$\mathcal{ALC}$ query Q is defined as follows. A multimedia database about form DBF *entails* a query Q (denoted by $\mathsf{DBF} \models Q$) iff all interpretations $\mathcal{I}$ w.r.t. DBF satisfy Q. Of course the notions of *correct answer* θ ($\mathsf{DBF} \models Q\theta$) and *answer set* (denoted by $AnswerSet(\mathsf{DBF}, Q)$) are as usual.

We extend all the above definitions in case a fuzzy HORN-$\mathcal{ALC}$ KB Σ is taken into account too. A fuzzy HORN-$\mathcal{ALC}$ KB Σ *entails* a query Q w.r.t. a multimedia database about form DBF (denoted by $(\Sigma, \mathsf{DBF}) \models Q$) iff all interpretations $\mathcal{I}$ w.r.t. DBF satisfying Σ satisfy Q too. The notions of *correct answer* θ and *answer set* are as above.

In summary, the above definitions specify in a formal way (*i*) what a multimedia database about form is; (*ii*) what a query is; and (*iii*) what the answer set w.r.t. a multimedia database about form and a query is. Of course, given a multimedia database about form DBF, a KB Σ and a query Q, there exists a simple method in order to determine whether $(\Sigma, \mathsf{DBF}) \models Q$.

- We may use the methods described in Section D.5.2, based on SLD-refutation. In particular, Method 4.2. is quite interesting as it relies on standard SLD-refutation for first-order horn logic, *i.e.* a standard prolog system is sufficient together with external procedure calls for concrete atoms.

10.2 Retrieval examples

In this section we present a list of examples showing the retrieval capabilities, along the form dimension of multimedia data, allowed in our framework. We will concentrate our attention to those cases which highlight the main features, by showing different types of queries allowed. In the following let DBF be a multimedia database about form.

10.2.1 About text ...

The first case concerns typical text retrieval.

> **"Find sections in a text document relevant to the query expressed through the text T"**.

The formalisation of the above query is as follows: we build a new CSMO which corresponds to text T. So, let (o, v) be a new object, instance of class CTO, *i.e.* the class of complex text objects[1], such that

1. $\mathsf{o.MediaDataObj}$ is a MDO of type $\mathsf{TextObject}$ representing text T,
2. $\mathsf{o.ComposedOf} = nil$
3. $\mathsf{o.Region}$ refers to the whole text T.

Let us assume that a class $\mathsf{Section}$ exists and a similarity function between complex text objects is given (*e.g.* [173]) with signature

$$\mathsf{simtxt}\colon \mathsf{CTO} \times \mathsf{CTO} \to [0,1].$$

We consider the KB Σ with the following rule

$$(\mathsf{SimilarText}(X,Y) \geq V) \leftarrow \mathsf{cto}(X), \mathsf{cto}(Y), \mathsf{simtxt}(X,Y,Z), \langle V, Z\rangle$$

which defines the predicate $\mathsf{SimilarText}$ as a similarity predicate: in $(\mathsf{SimilarText}(X,Y) \geq V)$ the truth degree V indicates the similarity between X and Y. As a consequence, the fuzzy HORN-$\mathcal{ALC}$ query Q_T is thus simply

$$Q_T \quad = \quad \exists X \exists V.\mathsf{section}(X) \wedge (\mathsf{SimilarText}(X, \mathsf{o}) \geq V). \tag{10.5}$$

Of course, given the set of correct answers,

$$AnswerSet((\Sigma, \mathsf{DBF}), Q_T) = \{\{X/\mathsf{o}_i, V/n_i\}\colon i \geq 0\},$$

the output is a list of tuples (o_i, n_i) ranked in descending order according to the value n_i.

Retrieval relying on attributes is also supported.

> **"Find articles written by Smith relevant to the query expressed through the text T".**

Assuming that $\mathsf{Article} \prec \mathsf{CTO}$ and Author is an attribute of $\mathsf{Article}$, then the above query can be formulated as

$$Q_T = \exists X \exists V.\mathsf{article}(X) \wedge \mathsf{author}(X, \text{“Smith”}) \wedge (\mathsf{SimilarText}(X, \mathsf{o}) \geq V). \tag{10.6}$$

Notice that in the examples above we assume that a similarity function like $\mathsf{SimilarText}$ is given. Of course, if there is no such method, a similarity predicate $\mathsf{SimilarText}$, or any other type of similarity predicate, can directly be implemented in prolog. This is done *e.g.* in [124]. This is particularly interesting during the prototyping phase, as it enables to test different similarity functions for their effectiveness. We do not investigate them here.

[1] Remember that $\mathsf{CTO} \prec \mathsf{CSMO}$. See Example 6.

10.2.2 About image ...

An usual query allowed in image retrieval systems is like the following

> **"Find images similar to a given image I"**.

The query formulation is similar to the text case. We build a new CSMO which corresponds to image I. So, let (o, v) be a new object, instance of class CIO, *i.e.* the class of complex image objects[2], such that

1. $\mathsf{o.MediaDataObj}$ is a MDO of type $\mathsf{ImageObject}$ representing image I,
2. $\mathsf{o.ComposedOf} = nil$
3. $\mathsf{o.Region}$ refers to the whole image I.

Let us assume that a similarity function between complex image objects is given with signature

$$\mathsf{simimag}\colon \mathsf{CIO} \times \mathsf{CIO} \to [0,1].$$

Furthermore, we consider the KB Σ with the following rule

$$(\mathsf{SimilarImage}(X,Y) \geq V) \leftarrow \mathsf{cio}(X), \mathsf{cio}(Y), \mathsf{simimag}(X,Y,Z), \langle V, Z\rangle$$

which defines the predicate $\mathsf{SimilarImage}$ as a similarity predicate: in $(\mathsf{SimilarImage}(X,Y) \geq V)$ the truth degree V indicates the similarity between the complex image objects X and Y. As a consequence, the fuzzy HORN-$\mathcal{ALC}$ query Q_I is thus simply

$$Q_I \quad = \quad \exists X \exists V.\mathsf{cio}(X) \wedge (\mathsf{SimilarImage}(X, \mathsf{o}) \geq V). \tag{10.7}$$

As for the text case, attributes can be used in retrieving images.

> **"Find images containing an image region whose shape is similar to a given shape S_1 and which has to its right hand side another region whose shape is similar to S_2"**.

Again, we build a new object $(\mathsf{o1}, \mathsf{v1})$, instance of class CIO, such that the feature $\mathsf{o1.HasShape}$ represents S_1 and a new object $(\mathsf{o2}, \mathsf{v2})$, instance of class CIO, such that the feature $\mathsf{o2.HasShape}$ represents S_2. The fuzzy HORN-$\mathcal{ALC}$ query Q_I is

$$Q_I \quad = \quad \exists X \exists V.(\mathsf{Retrieve}(X, \mathsf{o1}, \mathsf{o2}) \geq V) \tag{10.8}$$

where

$$\begin{array}{lcl}
(\mathsf{SimilarShape}(X,Y) \geq V) & \leftarrow & \mathsf{shape}(X), \mathsf{shape}(Y), \mathsf{simshape}(X,Y,Z), \langle V, Z\rangle \\
(\mathsf{IsLeftOf}(X,Y) \geq V) & \leftarrow & \mathsf{cio}(X), \mathsf{cio}(Y), \mathsf{isleftof}(X,Y,Z), \langle V, Z\rangle \\
(\mathsf{Retrieve}(X, O1, O2) \geq V) & \leftarrow & \mathsf{cio}(X), \mathsf{composedof}(X,Y), \\
 & & \mathsf{hasshape}(Y, Y1), \mathsf{hasshape}(O1, Y2), (\mathsf{SimilarShape}(Y1, Y2) \geq V1), \\
 & & \mathsf{composedof}(X,Z), \mathsf{hasshape}(Z, Z1), \mathsf{hasshape}(O2, Z2), \\
 & & (\mathsf{IsLeftOf}(Z,Y) \geq V2), (\mathsf{SimilarShape}(Z1, Z2) \geq V3), \\
 & & \langle V, V1 \cdot V2 \cdot V3\rangle
\end{array}$$

[2]Remember that $\mathsf{CIO} \prec \mathsf{CSMO}$. See Example 5.

where similarshape is the similarity function for the shape attribute (see Section 4.3.4) and isleftof is a topological operator (see Section 4.3.3). Notice that the resulting retrieval degree is the product of the two similarity degrees for shape similarity with the degree of being left. Of course, the retrieval by relying on color and/or texture attributes is similar.

Often it is useful to retrieve annotated images through a text query, like in [137].

"Find images relevant to the query expressed through the text T".

We build a new object (o, v), instance of class CTO representing text T. The fuzzy HORN-$\mathcal{ALC}$ query Q_I is

$$\begin{aligned} Q_I \quad = \quad & \exists X1, X2 \exists V. \quad \mathsf{cio}(X1) \wedge \mathsf{composedof}(X1, X2) \wedge \\ & \qquad\qquad \mathsf{caption}(X2) \wedge (\mathsf{SimilarText}(X2, \mathsf{o}) \geq V). \end{aligned} \tag{10.9}$$

Just notice that a huge variety of queries may be formulated in our framework, so as new similarity functions can be defined. For instance, consider the following set of rules defining similarity predicate MySimImage as follows. Suppose the user is looking for images similar to a given one I. Moreover, we suppose that the user is allowed to input priority degrees $t, s, c \in [0, 1]$ for texture, shape and color, respectively. For instance, if $s > c$ then the user considers shape similarity more important than color similarity in image retrieval. The following set of rules, defining similarity predicate MySimImage, takes this priorities into account.

$$\begin{aligned} (\mathsf{MySimImage}(X, Y) \geq V) \quad & \leftarrow \quad \mathsf{cio}(X), \mathsf{cio}(Y), \\ & \qquad (\mathsf{SimValT}(X, Y) \geq V1), \\ & \qquad (\mathsf{SimValS}(X, Y) \geq V2), \\ & \qquad (\mathsf{SimValC}(X, Y) \geq V3), \\ & \qquad \langle V, \tfrac{t \cdot V1 + s \cdot V2 + c \cdot V3}{t+s+c} \rangle \\ (\mathsf{SimValT}(X, Y) \geq V) \quad & \leftarrow \quad \mathsf{hastexture}(X, Z1), \\ & \qquad \mathsf{hastexture}(Y, Z2), \\ & \qquad (\mathsf{SimilarTexture}(Z1, Z2) \geq V1), \langle V, V1 \rangle \\ (\mathsf{SimValS}(X, Y) \geq V) \quad & \leftarrow \quad \mathsf{hasshape}(X, Z1), \\ & \qquad \mathsf{hasshape}(Y, Z2), \\ & \qquad (\mathsf{SimilarShape}(Z1, Z2) \geq V1), \langle V, V1 \rangle \\ (\mathsf{SimValC}(X, Y) \geq V) \quad & \leftarrow \quad \mathsf{hascolor}(X, Z1), \\ & \qquad \mathsf{hascolor}(Y, Z2), \\ & \qquad (\mathsf{SimilarColor}(Z1, Z2) \geq V1), \langle V, V1 \rangle \end{aligned} \tag{10.10}$$

More generally, if $A_1{:}F_1, \ldots, A_n{:}F_n$ are the feature attributes for complex image objects,

$$m_i^{sim}{:}\, F_i \times F_i \rightarrow [0, 1]$$

are the similarity functions for feature type F_i and $p_1, \ldots, p_n$ are the priorities for features $A_1, \ldots, A_n$, respectively, then

$$\begin{aligned} & m{:}\, \mathsf{CIO} \times \mathsf{CIO} \rightarrow [0, 1], \\ & m(o_1, o_2) = \frac{\sum_{i=1}^{n} p_i \cdot m_i^{sim}(o_1.A_i, o_2.A_i)}{\sum_{i=1}^{n} p_i} \end{aligned} \tag{10.11}$$

gives us a simple way to define a similarity function between images by relying on given similarity functions between image features. Note that the assignment

$$\langle V, \tfrac{t \cdot V_3 + s \cdot V_4 + c \cdot V_5}{t+s+c} \rangle$$

in the above rules is a special case of Equation (10.11).

10.2.3 About video and audio ...

Essentially, the examples about text and image can easily be adapted to the case of video or audio. For instance,

> **"Find audio streams regions similar to a given audio stream** A**"**.
>
> **"Find audio streams regions relevant to the query expressed through the text** T**"**.

and

> **"Find video frame sequence relevant to the query expressed through the text** T**"**.

can easily be formulated in our framework. Of course, many other kind of queries may be formulated involving methods and attributes of complex video objects and complex audio objects. It is not our intention to go through this long list.

10.2.4 About multimedia

In all the sections above, we have seen how an user's query involving text, image, video and audio, are easily translated into an fuzzy HORN-$\mathcal{ALC}$ query, as long as the user's query can be expressed as a fuzzy HORN-$\mathcal{ALC}$ query. The richness of the logic fuzzy HORN-$\mathcal{ALC}$ guarantees us that almost all queries of interest to the user can be formulated through it. In particular, this means that an user can express its interest through a multimedia document, *i.e.* a query is a document mixing text, images, video and audio. For instance, suppose the user is looking for documents about animals and specifies her information need through an HTML document H including text and images about these animals, *i.e.* her request is like

> **"Find all complex multimedia objects relevant to the query expressed through the HTML document** H**"**.

The query sounds quite complex as it involves both text and images. Notice that an answer may be a video frame sequence or an audio stream concerning animals: not necessary we are looking for HTML documents. Any kind of object which is about animals could be a candidate to be an answer relevant to the user's information need.

Of course, if a similarity function for complex multimedia objects is given, *i.e.* a function

$$\mathsf{cmosim}\colon \mathsf{CMO} \times \mathsf{CMO} \to [0,1],$$

then we are able to formulate the query in fuzzy HORN-$\mathcal{ALC}$ as follows. Let (o, v) be a new CMO which corresponds to HTML document H. The fuzzy HORN-$\mathcal{ALC}$ query Q_H is simply

$$Q_H = \exists X \exists V.\mathsf{cmo}(X) \wedge (\mathsf{CMOSim}(X, \mathsf{o}) \geq V), \tag{10.12}$$

where the rule

$$(\mathsf{CMOSim}(X, Y) \geq V) \leftarrow \mathsf{cmo}(X), \mathsf{cmo}(Y), \mathsf{cmosim}(X, Y, Z), \langle V, Z \rangle$$

has been considered. Unfortunately, a similarity function like cmosim for CMO is *rarely* given. So, how can we proceed? Here we show how a similarity method

$$\mathsf{cmosim}\colon \mathsf{CMO} \times \mathsf{CMO} \rightarrow [0, 1]$$

may be easily defined. It is reasonable to assume that similarity functions for both text and images are given, so as for other media as video and audio. Formally, let us consider the four classes CTO, CIO, CVO and CAO, representing complex text objects, complex image objects, complex video objects and complex audio objects, respectively. Let us assume that there are similarity functions for each of these classes with signature

$$\mathsf{simX}\colon \mathsf{X} \times \mathsf{X} \rightarrow [0, 1], \tag{10.13}$$

where $\mathsf{X} \in \{\mathsf{CTO}, \mathsf{CIO}, \mathsf{CVO}, \mathsf{CAO}\}$ and simX(o1, o2) is defined for *atomic* CSMOs only: o1.ComposedOf $= nil$, o2.ComposedOf $= nil$. It is not hard to see that it is reasonable to assume that such similarity measures already exists. We extend now each $\mathsf{simX}(\cdot, \cdot)$ to the general case where o.ComposedOf $= \{\mathsf{o1}, \ldots, \mathsf{om}\}$ (*e.g.* where a video frame sequence is composed of a set of video frame sequences, a text is composed of a set of sections, etc.).

We first determine all the components of (o, v), comp(o), through the method comp defined inductive as

$$\mathsf{comp}(\mathsf{o}) \quad = \quad \begin{cases} \{\mathsf{o}\} & \text{if } \mathsf{o.ComposedOf} = nil \\ \bigcup_{\mathsf{o}' \in \mathsf{o.ComposedOf}} \mathsf{comp}(\mathsf{o}') & \text{otherwise.} \end{cases} \tag{10.14}$$

Now, let $(\mathsf{o1}, \mathsf{v1})$ and $(\mathsf{o2}, \mathsf{v2})$ be two CSMO of the same media type. We define the following similarity functions for each of the classes CTO, CIO, CVO and CAO, with signature

$$\mathsf{simgenX}\colon \mathsf{X} \times \mathsf{X} \rightarrow [0, 1], \tag{10.15}$$

where $\mathsf{X} \in \{\mathsf{CTO}, \mathsf{CIO}, \mathsf{CVO}, \mathsf{CAO}\}$ such that

$$\mathsf{simgenX}(\mathsf{o1}, \mathsf{o2}) = \frac{\sum_{\mathsf{o}' \in \mathsf{comp(o1)}, \mathsf{o}'' \in \mathsf{comp(o1)}} \mathsf{simX}(\mathsf{o}', \mathsf{o}'')}{|\mathsf{comp(o1)}| \cdot |\mathsf{comp(o1)}|}. \tag{10.16}$$

simgenX(o1,o2) is in principle quite simple: it sums up the similarity between the components of o1 and o2 and then normalises the sum with $|\mathsf{comp(o1)}| \cdot |\mathsf{comp(o1)}|$. Just notice that more sophisticated similarity functions can be computed as well. For instance one could rely on the work described in [173]. Of course, the simgenX similarity functions may be implemented directly in prolog too.

So far, we have now similarity functions simgenCTO, simgenCIO, simgenCVO and simgenCAO measuring similarity between CSMOs. We are going now to define a similarity function cmosim measuring similarity between CMOs. Let (o, v) be CMO. Consider comp(o) the set

of components of (o, v). For each $\mathsf{X} \in \{\mathsf{CTO}, \mathsf{CIO}, \mathsf{CVO}, \mathsf{CAO}\}$, let $\mathsf{comp}(\mathsf{o})_{\mathsf{X}}$ be the subset of all those CSMOs in $\mathsf{comp}(\mathsf{o})$ of media type X. Note that $\mathsf{comp}(\mathsf{o})_{\mathsf{CTO}}$, $\mathsf{comp}(\mathsf{o})_{\mathsf{CIO}}$, $\mathsf{comp}(\mathsf{o})_{\mathsf{CVO}}$ and $\mathsf{comp}(\mathsf{o})_{\mathsf{CAO}}$ constitutes a partition of the set $\mathsf{comp}(\mathsf{o})$. Let α_{X}, with $\mathsf{X} \in \{\mathsf{CTO}, \mathsf{CIO}, \mathsf{CVO}, \mathsf{CAO}\}$ be priorities factors such that

$$\sum_{\mathsf{X} \in \{\mathsf{CTO},\mathsf{CIO},\mathsf{CVO},\mathsf{CAO}\}} \alpha_{\mathsf{X}} \leq 1.$$

Then we may define cmosim measuring the similarity between two CMOs, with signature

$$\mathsf{cmosim}\colon \mathsf{CMO} \times \mathsf{CMO} \to [0, 1],$$

as

$$\mathsf{cmosim}(\mathsf{o1}, \mathsf{o2}) = \sum_{\mathsf{X} \in \{\mathsf{CTO},\mathsf{CIO},\mathsf{CVO},\mathsf{CAO}\}} (\alpha_{\mathsf{X}} \cdot \frac{\sum_{o' \in \mathsf{comp}(\mathsf{o1})_{\mathsf{X}}, o'' \in \mathsf{comp}(\mathsf{o2})_{\mathsf{X}}} \mathsf{simX}(\mathsf{o}', \mathsf{o}'')}{|\mathsf{comp}(\mathsf{o1})_{\mathsf{X}}| \cdot |\mathsf{comp}(\mathsf{o2})_{\mathsf{X}}|}). \quad (10.17)$$

Now, the user's query specified at the beginning of this section can be answered through the fuzzy HORN-$\mathcal{ALC}$ query (10.12).

In conclusion, we have seen that it quite easy not only to formalise user's queries in terms of fuzzy HORN-$\mathcal{ALC}$ queries, but also to define new similarity measures. This is especially important for the development of a prototypical system and for those cases in which the declarative character of this measure plays an important role, *i.e.* if the system should allow the definition of user specific retrieval criteria.

Chapter 11

Reasoning about form and semantics

11.1 Formalisation

We have seen that the form dimension concerns media dependent properties of multimedia data, represented in terms of complex multimedia objects (CMOs). Essentially, these properties are of three different types: the first type concerns the aggregation structure of a CMO, the second type concerns the feature attributes of a each CMO and the third type concerns methods, *i.e.* relations among CMO. The structure concerns how the multimedia data is organised (if it has an organisation). For instance, an article has a title, an author, an abstract, an introduction, other sections, a conclusion and a bibliography reference list (see also Figure 4.5). Concerning images, regions of an image can be aggregated, like in Figure 4.4. The aggregation structure can typically be extracted automatically from text documents, whereas it is mostly manual in cases of image, video and audio data. Each CMO has a list of features attributes whose type is media depended. For instance, typically a text document has a term/weight vector, whereas an image has color distribution, texture and shape attributes. Finally, methods allow us to computes some relationships between CMO. For instance, given two image regions of the same image, the relation isLeftof determines whether the first is on the left of the second region. As a consequence, *queries about form* concern nothing else than these three types of properties, as we have seen in the previous chapter.

In this chapter we address the *semantics*, or *meaning*, dimension of multimedia data, *i.e.* properties of them which are medium independent. In particular, we will formally specify

- how the semantics, or meaning, of a CMO will be described;
- what a multimedia database about form and semantics is;
- what a query about form and semantics is; and
- what querying a multimedia database about form and semantics means.

So, let

$$\mathsf{DBF} \quad = \quad (\mathsf{CH}, \mathsf{TY}, \mathsf{MET}, \pi, \mathsf{OID})$$

be a multimedia database about form and consider an object (o,v), instance of class CMO, *i.e.* (o,v) is a complex media object. For the sake of explanation purposes, suppose that (o,v) corresponds to an image region depicting a person, say Tom. Now, in order to say that (o,v) is *about* Tom, an (image) interpretation (or understanding) process happened. Essentially, this process *maps* object (o,v) representing a region in the image into an entity of the real world, called Tom, *i.e.* Tom is what we call the semantics of region (o,v). We will represent the semantic entities (or also events) we are talking about with individuals. As a consequence, assigning meaning to a generic CMO (o, v) can be defined by mapping it into an individual a. In our case, the object (o,v) is mapped into the individual tom representing a person. As this mapping is almost *imprecise*, a degree in $[0, 1]$ is given too.

Formally, let $\mathcal{O}$ be the set of individuals. Let $\mathsf{CMO}_1, \ldots, \mathsf{CMO}_k$, $k \geq 1$, be k subclasses of CMO, not necessarily be pairwise distinct. Remember that $\pi(\mathsf{CMO}_i)$ is the set of instances of CMO_i and, thus, $\delta(\pi(\mathsf{CMO}_i))$ is the set of concrete individuals denoting the CMOs in $\pi(\mathsf{CMO}_i)$. A *meaning interpretation function for class* CMO_i is a degree function

$$int\_as_i\colon \delta(\pi(\mathsf{CMO}_i)) \times \mathcal{O} \to [0, 1], \tag{11.1}$$

mapping a concrete individual denoting a complex multimedia object OID o, belonging to class CMO_i, into an individual a, with a degree $n \in [0, 1]$. Here, each int_as_i plays the role of a meaning interpretation function which is specialised in determining the meaning of the objects of class CMO_i. Each meaning interpretation function specifies a connection between the form dimension and the semantics dimension of multimedia data. $int\_as_i(o, a)$ determines to which extent object (o, v) can be interpreted as individual a. For instance, in our example we have that $\mathsf{CIO} \prec \mathsf{CMO}$ and int_as_{CIO} is specialised in image interpretation. Therefore, *e.g.* $int\_as_{\mathsf{CIO}}(\mathsf{o},\mathsf{tom}) = .8$ exploits the fact that the interpretation of the complex image object (o,v) is the entity represented by the individual tom, and this relation holds with degree .8 (see Figure 11.1).

Certainly, the definition of the int_as_i functions is rather a hard problem, as this means that our understanding process has been completely understood. Of course, as int_as_i is an interpretation function, it is a subjective matter: *e.g.* it is questionable whether $int\_as_{\mathsf{CIO}}(\mathsf{o},\mathsf{tom}) = .8$ or $int\_as_{\mathsf{CIO}}(\mathsf{o},\mathsf{tom}) = .6$, or any other value $n \in [0, 1]$. This kind of situation can be modelled by observing that there could be i, j such that $i \neq j$, $\mathsf{CMO}_i = \mathsf{CMO}_j$, $int\_as_i(o) \neq int\_as_j(o)$ for some OID o, *i.e.* int_as_i and int_as_j represent two different interpretation functions for the same class CMO_i. A simple consequence is that our model allows *multiple interpretations* for the same object.

Moreover, mostly the interpretation process is done manually by indexers, *i.e.* int_as_i is defined as a a finite set of tuples (o, a, n) which in turn means that int_as_i is defined through its graph. This process is called usually *manual indexing*.

Now we are ready to define what we mean with a database involving both the form dimension and semantics dimension of multimedia data. At first, we extend fuzzy HORN-$\mathcal{ALC}$ as follows.

Syntax From a syntax point of view, int_as_i is a degree function and, thus, as such may be used in fuzzy HORN-$\mathcal{ALC}$. Furthermore, we assume that there are ordinary predicates $\mathsf{Int\_As}_i$ such that expressions of the form $(\mathsf{Int\_As}_i(w_1, w_2) \geq T)$ may appear as fact, in a rule or in a query, where w_1 is a concrete object, w_2 is an object and T is a fuzzy value.

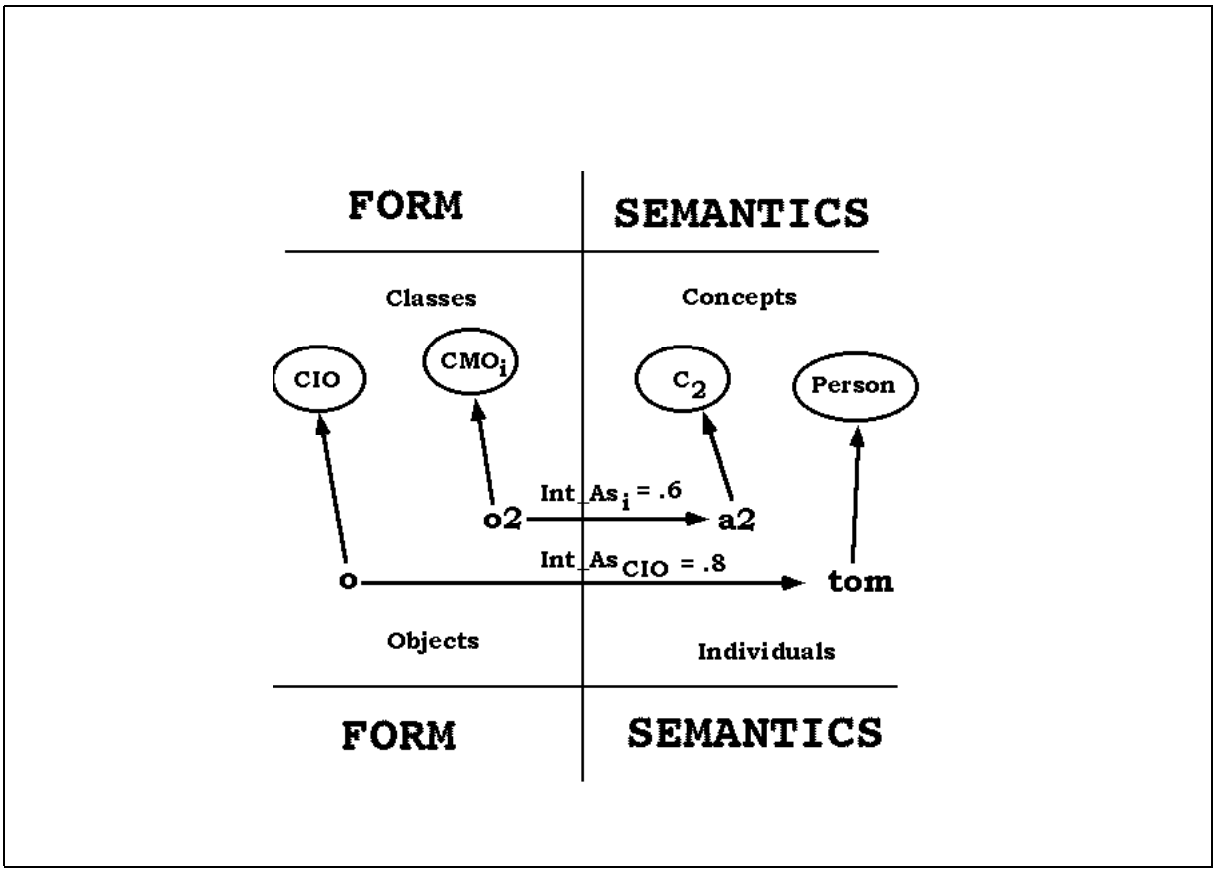


Figure 11.1: Meaning interpretation of CMOs.

Semantics From a semantics point of view, an interpretation $\mathcal{I}$ has to be such that

$$t \in (\mathsf{Int\_As}_i(w_1, w_2) \geq T)^{\mathcal{I}} \text{ if } int\_as_i({w_1}^{\mathcal{I}}, {w_2}^{\mathcal{I}}) \geq T^{\mathcal{I}}. \tag{11.2}$$

Note that it might be the case that $(\mathsf{Int\_As}_i(w_1, w_2) \geq T)^{\mathcal{I}} = \emptyset$, describing the state of total ignorance about the meaning of w_1.

Given a multimedia database about form, DBF, let INT be a set of meaning interpretation functions int_as_i. Let Σ be a fuzzy HORN-$\mathcal{ALC}$ KB. Σ specifies our background knowledge about the entities/events of the considered slice of real world. For instance, $(\mathsf{Person}(\mathsf{tom}) \geq 1) \in \Sigma$, *i.e.* tom is a Person. Without loss of generality we will assume that each primitive concept appearing in a fuzzy assertion in Σ_F is a concept name defined in the terminology Σ_T. Then a *multimedia database about form and semantics*, DBFC, is defined as a triple

$$\mathsf{DBFC} \quad = \quad (\mathsf{DBF}, \mathsf{INT}, \Sigma). \tag{11.3}$$

Here, DBF concerns the form dimension of multimedia documents, Σ concerns the semantics dimension o multimedia documents, whereas INT exploits the link between these two dimensions. Moreover, Σ has to be such that Σ_R includes the following rules:

$$
\begin{array}{l}
(\mathsf{Int\_As}(X,Y) \geq V) \leftarrow \mathsf{CMO}_1(X), (\mathsf{Int\_As}_1(X,Y) \geq V_1), \langle V, V_1 \rangle \\
\ldots \\
(\mathsf{Int\_As}(X,Y) \geq V) \leftarrow \mathsf{CMO}_k(X), (\mathsf{Int\_As}_k(X,Y) \geq V_1), \langle V, V_1 \rangle \\
(\mathsf{Int\_As}_1(X,Y) \geq V) \leftarrow \mathsf{CMO}_1(X), \langle V, int\_as_1(X,Y) \rangle \\
\ldots \\
(\mathsf{Int\_As}_k(X,Y) \geq V) \leftarrow \mathsf{CMO}_k(X), \langle V, int\_as_k(X,Y) \rangle
\end{array}
\tag{11.4}
$$

Of course, other functions may be defined through rules of the above type. The ordinary predicate Int_As is called *meaning interpretation function of complex multimedia objects*.

As for the form dimension, the query language is nothing else than fuzzy HORN-$\mathcal{ALC}$ and querying a multimedia database about form and semantics is defined in terms of entailment. A a multimedia databases about form and semantics DBFC *entails* a query Q (denoted by $\mathsf{DBFC} \models Q$) iff all interpretations w.r.t. DBF satisfying Σ satisfy Q too. Of course the notions of *correct answer* θ ($\mathsf{DBFC} \models Q\theta$) and *answer set* (denoted by $AnswerSet(\mathsf{DBFC}, Q)$) are as usual.

Example 26 Let us consider Example 5 (see Figure 4.4). We have seen that the OIDs $\mathsf{o}_{\mathsf{reg}i}$, $i \geq 3$, correspond to the regions depicted in GetData(o2.Data) in Figure 4.4 which are *interpreted* as the left eye, the mouth, the right eye, the nose and the face, respectively.

Now, let DBF be a multimedia database about form containing all the objects identified in Figure 4.4, *i.e.* (o2, v), (o3, v3), (o4, v4), (o5, v5), (o6, v6), (o7, v7). Let Σ be a fuzzy HORN-$\mathcal{ALC}$ KB containing the following facts:

$$
\begin{array}{l}
(\mathsf{Int\_As}_{\mathsf{CIO}}(\mathsf{o3}, \mathsf{leye}) \geq .8) \\
(\mathsf{Int\_As}_{\mathsf{CIO}}(\mathsf{o4}, \mathsf{mouth}) \geq .7) \\
(\mathsf{Int\_As}_{\mathsf{CIO}}(\mathsf{o5}, \mathsf{reye}) \geq .6) \\
(\mathsf{Int\_As}_{\mathsf{CIO}}(\mathsf{o6}, \mathsf{nose}) \geq .4) \\
\\
(\mathsf{Eye}(\mathsf{leye}) \geq 1) \\
(\mathsf{Eye}(\mathsf{reye}) \geq 1) \\
(\mathsf{Mouth}(\mathsf{mouth}) \geq 1) \\
(\mathsf{Nose}(\mathsf{nose}) \geq 1)
\end{array}
$$

which correspond to the meaning interpretation, determined either manually or automatically, of the CIOs objects. Just note that o7 has not been interpreted. Furthermore assume that in Σ_R there is a rule like

$$
\begin{array}{rcl}
(\mathsf{Int\_As}_{\mathsf{CIO}}(O, \mathsf{face}) \geq V) & \leftarrow & \mathsf{cio}(O), \mathsf{composedof}(O, [Y1, Y2, Y3|Z]), \\
& & (\mathsf{Int\_As}_{\mathsf{CIO}}(Y1, X1) \geq V1), (\mathsf{Eye}(X1) \geq V2), \\
& & (\mathsf{Int\_As}_{\mathsf{CIO}}(Y2, X2) \geq V3), (\mathsf{Mouth}(X2) \geq V4), \\
& & (\mathsf{Int\_As}_{\mathsf{CIO}}(Y3, X3) \geq V5), (\mathsf{Nose}(X3) \geq V6), \\
& & \langle V, p \cdot \frac{V1 \cdot V2 + V3 \cdot V4 + V5 \cdot V6}{3} \rangle
\end{array}
$$

where p is computed as

$$p = \max\{\frac{p_1}{p_2} + .5, 1\}$$
$$p_1 = \sum_{i=1}^{3} \mathsf{Dimension}(Yi.\mathsf{Region})$$
$$p_2 = p_1 + \sum_{Y \in Z} \mathsf{Dimension}(Y.\mathsf{Region})$$

where $\mathsf{Dimension}(Y.\mathsf{Region})$ determines the dimension of the area covered by region $Y.\mathsf{Region}$.

The above rule for $\mathsf{Int\_As_{CIO}}$ may be interpreted as follows: a complex image object O is interpreted as being a face, with degree V, if

1. O is composed of at least three objects $Y1, Y2, Y3$;
2. each $Y1, Y2, Y3$ is interpreted as Eye, Mouth, Nose, respectively;
3. the degree V is obtained as follows:
   (a) $V1 \cdot V2$ is the degree of being $Y1$ an Eye;
   (b) $V3 \cdot V4$ is the degree of being $Y2$ a Mouth;
   (c) $V5 \cdot V6$ is the degree of being $Y3$ a Nose;
   (d) $\frac{V1 \cdot V2 + V3 \cdot V4 + V5 \cdot V6}{3}$ is the degree that O is composed of an Eye, a Mouth and Nose;
   (e) the factor p is determined in terms of the fraction of area covered by the regions addressed by $Y1, Y2, Y3$, p_1, w.r.t. the total area covered by object O, p_2.

■

11.2 Retrieval examples

In this section we present a list of examples showing the retrieval capabilities, along the both the form and semantics dimension of multimedia data. We will concentrate our attention to those cases which highlight the main features, and in particular, the cases involving images. The case were queries involve other types of media are similar.

"Find images about faces".

The query is straightforward.

$$Q_I \quad = \quad \exists X \exists V.\mathsf{cio}(X) \wedge (\mathsf{Int\_As}(X, \mathsf{face}) \geq V). \tag{11.5}$$

i.e. find all complex image objects X which are interpreted as face. By relying on the multimedia database about form DBF and Σ as in Example 26, it is easily verified that for $\mathsf{DBFC} = (\mathsf{DBF}, \mathsf{INT}, \Sigma)$, $\mathsf{DBFC} \models Q_I$ holds. Just notice that a correct answer θ may be such that $\theta = \{V/n\}$ where

$$\begin{aligned} n &= p \cdot (.8 \cdot 1 + .7 \cdot 1 + .4 \cdot 1) \\ &= p \cdot \tfrac{1.9}{3} \\ &= .9 \cdot \tfrac{1.9}{3} \\ &= .57. \end{aligned}$$

The above value n is obtained by the application of the $\mathsf{Int\_As_{CIO}}$ rule, where p has been approximated to .9 (*i.e.* $p = \max\{\frac{2}{5} + .5, 1\}$, 5 = total area of o7, 2 = area of o7's components).

The combination of semantics and form is quite straightforward.

"Find images about red cars".

Here "car" concerns semantics, whereas "red" concerns the form dimension. The query is

$$\begin{aligned} Q_I \;=\; & \exists X1, X2 \exists V1, V2, V3.\mathsf{cio}(X1) \wedge \\ & (\mathsf{Int\_As}(X1, X2) \geq V1) \wedge \\ & (\mathsf{Car}(X2) \geq V2) \wedge \\ & \mathsf{hascolor}(X1, C) \wedge \\ & (\mathsf{SimilarColor}(C, \mathsf{red}) \geq V3) \end{aligned} \tag{11.6}$$

There query

"Find images about Tom and Tim, where Tom is on the right of Tim".

involves semantics and form in terms of topological operators. The query is simply

$$\begin{aligned} Q_I \;=\; & \exists X1, X2, X3, X4, X5 \exists V1, V2, V3.\mathsf{cio}(X1) \wedge \mathsf{cio}(X2) \wedge \\ & \mathsf{mediadataobj}(X1, X3) \wedge \mathsf{mediadataobj}(X2, X3) \wedge \\ & \mathsf{region}(X1, X4) \wedge \mathsf{region}(X2, X5) \wedge \\ & (\mathsf{Int\_As}(X1, \mathsf{tom}) \geq V1) \wedge (\mathsf{Int\_As}(X2, \mathsf{tim}) \geq V2) \wedge \\ & (\mathsf{IsRightOf}(X5, X4, X3) \geq V3) \end{aligned} \tag{11.7}$$

where isRightof is defined in Section 4.3.3.

We conclude this section by proposing a simple query mixing different media types together.

"Find HTML documents in which there is an image about a car and a text relevant to the given text T".

As usual, let (o, v) be a new object, instance of class CTO, representing text T. The query can be formulated as follows:

$$\begin{aligned} Q_I \;=\; & \exists X1, X2, X3, X4 \exists V1, V2, V3.\mathsf{cmo}(X1) \wedge \\ & \mathsf{composedof}(X1, X2) \wedge \mathsf{composedof}(X1, X3) \wedge \\ & \mathsf{cto}(X2) \wedge \mathsf{cio}(X3) \wedge \\ & (\mathsf{Int\_As}(X3, X4) \geq V1) \wedge (\mathsf{Car}(X4) \geq V2) \wedge (\mathsf{SimilarText}(X2, \mathsf{o}) \geq V3). \end{aligned} \tag{11.8}$$

Of course, a huge variety of combinations involving the form and semantics dimension can easily be formulated within our framework.

A final note concerns the distinction between similarity at the form level and similarity at the semantics level. We have seen that similarity measures are essentially defined on CMOs, *e.g.* a method

$$\mathsf{simimag}\colon \mathsf{CIO} \times \mathsf{CIO} \rightarrow [0, 1]$$

measure the similarity between two images. A characteristic of these functions is that the similarity degree depends on several feature attributes of the form level. We talk about *form similarities*, as they are media dependent. Of course, if we switch at the semantics level, a notion of *semantics similarity* may be introduced. Here, similarity at the semantics

level concerns (subjective) similarity between concepts, and thus, is media independent. For instance, we can decide that similarity between two people is determined by their character's similarity, *i.e.* through a degree function

$$\mathsf{similarcharacter}\colon \mathsf{Person} \times \mathsf{Person} \rightarrow [0,1].$$

It is quite obvious that similarcharacter is at a different level than simimag: the first is at the semantics level, whereas the second is at the form level. If an ordinary predicate SimilarCharacter has been defined through fuzzy HORN-$\mathcal{ALC}$ rules, then a complex query like

> **"Find all images about a person whose character is similar to the person depicted in image I and which are physically similar"**.

can be formulated in fuzzy HORN-$\mathcal{ALC}$ as

$$\begin{aligned} Q_I \;=\; & \exists X1, X2, X3 \exists V1, V2, V3, V4, V5, V6.\mathsf{cio}(X1) \wedge \\ & (\mathsf{Int\_As}(X1, X2) \geq V1) \wedge \\ & (\mathsf{Person}(X2) \geq V2) \wedge \\ & (\mathsf{Int\_As}(\mathsf{o}, X3) \geq V3) \wedge \\ & (\mathsf{Person}(X3) \geq V4) \wedge \\ & (\mathsf{SimilarCharacter}(X2, X3) \geq V5) \wedge \\ & (\mathsf{SimilarImage}(X1, \mathsf{o}) \geq V6) \end{aligned} \tag{11.9}$$

where the new object (o, v) represents image I. These kind of queries are clearly behind the capabilities of traditional retrieval systems, and constitutes an interesting starting point in building intelligent retrieval systems. Note that recently some work has been done on semantics similarity. For instance, in [268, 269, 270] notions of similarity between "fuzzy" $\mathcal{ALC}$ concepts and "fuzzy" $\mathcal{ALC}$ roles have been introduced to the end of semantic based retrieval purposes.

In summary, we have a retrieval system which (i) has an appropriate language for representing documents at the form level; (ii) has an appropriate language for representing documents at the semantics level; (iii) allows the retrieval about form by relying on form similarity; (iv) allows the retrieval about semantics by relying on semantics similarity; and (v) allows the retrieval about form and semantics in a unified and principled way.

11.3 Relevance feedback

Most retrieval systems are iterative search systems. Such systems are typically implemented by initially submitting a tentative query, and then using system facilities to improve the query and the resulting set of retrieved objects.

A method which has widely been used to construct improved query formulations fall into the class of *relevance feedback* (see *e.g.* [242]): improved query formulation depends on the prior retrieval of some retrieved objects. The query alteration process is based on the execution of an initial search operation and an initial retrieval of certain objects. In particular, in case of text retrieval systems, the display of information relating retrieved documents, *e.g.* titles or abstracts, is then used to modify the query, normally by adding terms that appear relevant to the user and by deleting terms from the documents which appear useless. In practice these terms are identified after the user selects those objects which are relevant

and those objects which are nonrelevant to her information need. This produces a new query whose resemblance to the relevant documents is greater than before while their resemblance to the nonrelevant documents is smaller. It has been shown experimentally that the relevance feedback process can account for improvements in retrieval effectiveness of up to 50 percent in precision for high-recall searches, and of approximately 20 percent in precision for low-recall searches.

11.3.1 Relevance feedback in text case

In order to be more concrete, we present here a quite popular relevance feedback method, used *e.g.* in the SMART system [242]. See [5, 50, 121, 135, 164, 256, 276] for other techniques.

Following [242], a text document D_i is represented through a vector $\vec{d_i}$ of weighted terms. Let w_{ik} be the value, or weight, of term k in document i. Therefore, $\vec{d_i} = (w_{i1}, \ldots, w_{in})$, where n is the number of all terms considered in the document collection. A query is simply a text document Q and, thus is represented through a vector $\vec{q}$ of weighted terms too. The similarity between two documents $\vec{d_i}$ and $\vec{d_j}$ is defined in terms of the cosine function, *i.e.*

$$sim(\vec{d_i}, \vec{d_j}) = \frac{\sum_{k=1}^{n} w_{ik} \cdot w_{jk}}{\sqrt{\sum_{k=1}^{n} w_{ik}^2} \cdot \sqrt{\sum_{k=1}^{n} w_{jk}^2}}.$$

The interaction of an user with a text information retrieval system is usually as follows:

1. the user submits a query Q to the system and Q's representation $\vec{q}$ is determined;
2. the system returns the top k documents, $D_1, \ldots, D_k$, ordered according the similarity values $sim(\vec{q}, \vec{d_1}), \ldots, sim(\vec{q}, \vec{d_k})$. If the user is satisfied then exit. Otherwise
3. from the k documents, the user selects a set R of relevant documents and a set N of nonrelevant documents;
4. the system refines the query $\vec{q}$ according to the following formula

$$\vec{q'} = \alpha \cdot \vec{q} + \beta \cdot \vec{d}_R - \gamma \cdot \vec{d}_N \tag{11.10}$$

   where α, β and γ are suitable constants such that $\alpha + \beta - \gamma \leq 1$ and

$$\vec{d}_R = \frac{1}{|R|} \cdot \sum_{D_i \in R} \vec{d_i}$$
$$\vec{d}_N = \frac{1}{|N|} \cdot \sum_{D_i \in N} \vec{d_i};$$

5. go to Step 2 determining the top k documents according to the new query $\vec{q'}$.

Note that Equation (11.10) specifies a new query as the vector sum of the old query plus the weighted difference between the average of the known relevant and the average of the known nonrelevant documents. The experimental evidence available for relevance feedback indicates that one or two feedback operations are quite effective in raising retrieval performance. Following the second query reformulation a state of diminishing returns sets in and not much further improvement can be expected.

11.3.2 Relevance feedback in the multimedia case

In this section we will show how relevance feedback may be performed within our framework. At first, we extend our simple MIR system, described in Figure 1.1, in order to take relevance feedback into account (see Figure 11.2).

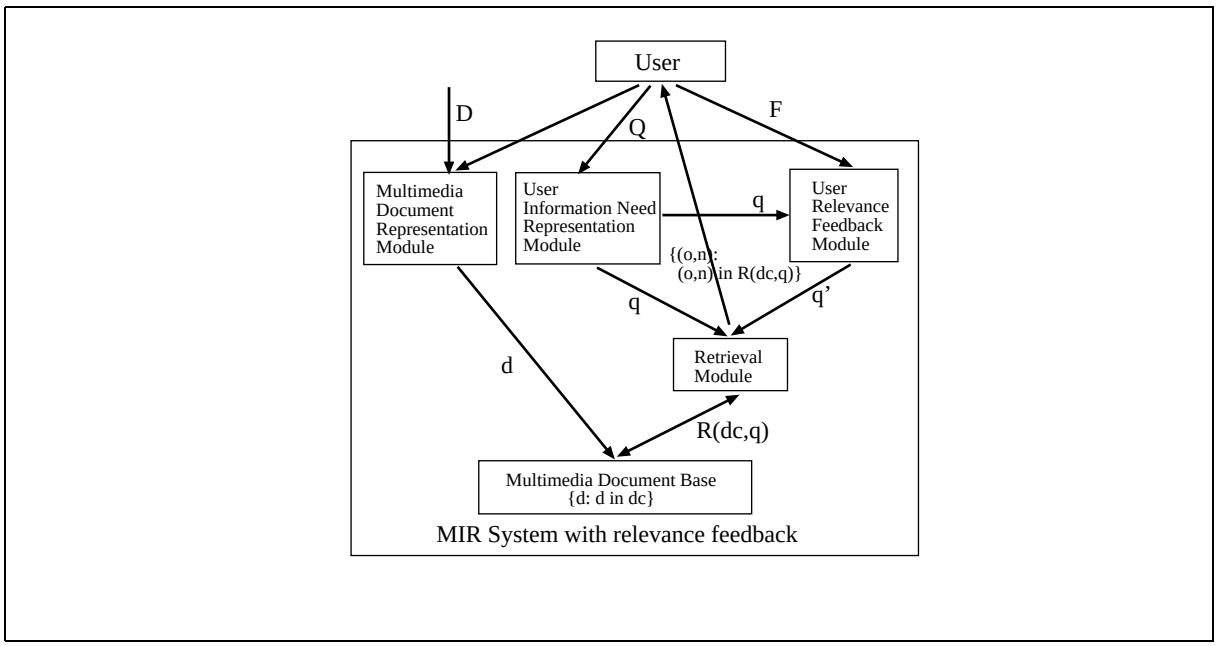


Figure 11.2: A simple MIR system with relevance feedback.

Roughly, the interaction of an user with the system is as follows.

- Once the MIR system presents to the user the result $R(dc, q) = \{(o, n) : o \text{ CMO}, n = RSV(o, q)\}$ in terms of retrieved CMOs, she may submit her relevance feedback F to the User Relevance Feedback Module (URFM) by selecting those CMOs which are *relevant* and selecting those CMOs which are *nonrelevant* to her information need;
- the URFM, given the query representation q and feedback F, determines a new query q' to be submitted to the Retrieval Module;
- a new set of CMOs ($R(dc, q')$) will be retrieved and shown to the user.

Formally, consider a multimedia database about form and semantics, $\mathsf{DBFC} = (\mathsf{DBF}, \mathsf{INT}, \Sigma)$. It is reasonable to assume that the user is interested in the retrieval of CMOs satisfying certain conditions, *i.e.* we assume that user's query is of the form

$$Q = \exists X \exists V.\mathsf{cmo}(X) \wedge (\mathsf{RSV}(\mathsf{X}) \geq V) \tag{11.11}$$

and that Σ contains the rule, defining $\mathsf{RSV}(\mathsf{X})$,

$$(\mathsf{RSV}(\mathsf{X}) \geq V) \leftarrow F(X, \vec{Y}, \vec{V'}), \langle V, f(\vec{V'}, \vec{Y})\rangle. \tag{11.12}$$

Query (11.11) specifies that the user is interested in the retrieval of a CMO X such that its retrieval status value is V. Rule (11.12) specifies both (i) the condition $F(X, \vec{Y}, \vec{V'})$ that the CMO X to be retrieved has to satisfy; and (ii) the value $f(\vec{V'}, \vec{Y})$ of the retrieval status value V of the retrieved object X. An instance of Rule (11.12) is *e.g.*

$$(\mathsf{RSV(X)} \geq V) \leftarrow (\mathsf{Int\_As}(X, X1) \geq V1), (\mathsf{Father}(X1) \geq V2)\langle V, V1 \cdot V2\rangle. \tag{11.13}$$

Query (11.11) together with rule (11.13) specify that the user is interested in the retrieval of those CMOs which are about father and the retrieval status value of the retrieved objects is determined by the product. A final assumption that we make is the following: we assume that a similarity function $\mathsf{cmosim}\colon \mathsf{CMO} \times \mathsf{CMO} \rightarrow [0,1]$ between CMOs is given. An example of it may be Equation (10.17).

Our relevance feedback algorithm closely relates to the text case and is given in in Table 11.1.

Algorithm 1 (*RelFeedback*)
Let $\mathsf{DBFC} = (\mathsf{DBF}, \mathsf{INT}, \Sigma)$ be a multimedia database about form and semantics, assume that the initial query is $Q = \exists X \exists V.\mathsf{cmo}(X) \wedge (\mathsf{RSV(X)} \geq V)$, that Σ contains the rule, $(\mathsf{RSV(X)} \geq V) \leftarrow F(X, \vec{Y}, \vec{V'}), \langle V, f(\vec{V'}, \vec{Y})\rangle$, that $\mathsf{cmosim}\colon \mathsf{CMO} \times \mathsf{CMO} \rightarrow [0,1]$ is a similarity function between CMOs and that the rule $(\mathsf{CMOSim}(X, Y) \geq V) \leftarrow \mathsf{cmo}(X), \mathsf{cmo}(Y), \mathsf{cmosim}(X, Y, Z), \langle V, Z\rangle$ belongs to Σ. The following steps are executed.

1. Return the top k answers, $\theta_1, \ldots, \theta_k$, where $\theta_i = \{X/o_i, V/n_i\}$, of query Q w.r.t. DBFC, ordered according the retrieval status value V. If the user is satisfied then exit. Otherwise, let $Ret = \{\delta^{-1}(o_1), \ldots, \delta^{-1}(o_k)\}$ be the k corresponding retrieved CMOs.
2. From the k CMOs, the user selects a set $R \subseteq Ret$ of *relevant* CMOs and a set $N \subseteq Ret$ of *nonrelevant* CMOs.
3. Build a new CMO, $\mathsf{o_R}$, such that $\mathsf{o_R.composedof} = R$ and build a new CMO, $\mathsf{o_N}$, such that $\mathsf{o_N.composedof} = N$.
4. Refine the retrieval query as follows. The new query Q' is

$$Q' = \exists X \exists V.\mathsf{cmo}(X) \wedge (\mathsf{RSV'(X)} \geq V), \tag{11.14}$$

and then add the following rule to Σ:

$$\begin{aligned}(\mathsf{RSV'(X)} \geq V) \leftarrow\ & (\mathsf{RSV(X)} \geq V_Q),\\ & (\mathsf{CMOSim}(X, \mathsf{o_R}) \geq V_R), (\mathsf{CMOSim}(X, \mathsf{o_N}) \geq V_N),\\ & \langle V, g(V_Q, V_R, V_N)\rangle\end{aligned} \tag{11.15}$$

where, given constants α, β and γ with $\alpha + \beta - \gamma \leq 1$,

$$g(V_Q, V_R, V_N) = \alpha \cdot V_Q + \beta \cdot V_R - \gamma \cdot V_N. \tag{11.16}$$

5. Go to Step 1 and compute the top k answers according to the new query Q' and KB Σ. ■

Table 11.1: Algorithm *RelFeedback*.

Note: It seems that g is not monotone not decreasing, but indeed, g may be rewritten as $g(V_Q, X) = \alpha \cdot V_Q + \beta \cdot \mathsf{cmosim}(X, \mathsf{o_R}) - \gamma \cdot \mathsf{cmosim}(X, \mathsf{o_N})$ and, thus, g is monotone not

decreasing w.r.t. V_Q.

We conclude by noticing that the *RelFeedback* algorithm behaves in a similar way as in the text case. Essentially, once the user selected those CMOs which are relevant (R) and nonrelevant (N), we build two new CMOs $\mathsf{o_R}$ and $\mathsf{o_N}$. The intended meaning of the rule (11.15) defining ($\mathsf{RSV'(X)} \geq V$) closely relates to Equation (11.10):

1. determine the retrieval status value V_Q of a retrieved CMO X w.r.t. the original query Q;
2. determine the similarity value V_R between a retrieved CMO X and the relevant CMOs in R;
3. determine the similarity value V_N between a retrieved CMO X and the nonrelevant CMOs in N; and
4. determine the retrieval status value V of a retrieved CMO X w.r.t. the new query Q', as specified in Equation (11.16) (compare with Equation (11.10)).

11.4 Implementation issues

Finally, some (very short) considerations about implementation issues. It follows clearly from the chapters presented until now that a simple prototype of a system dealing with retrieval about form and semantics may be realised through a standard prolog system which is connected to a database system with multimedia capabilities. In fact, the multimedia system should provide storage, form similarity functions and retrieval features at the form level, *i.e.* it deals with the objects of the multimedia model and the storage/retrieval of fuzzy HORN-$\mathcal{ALC}$ facts, whereas the prolog system is responsible for user querying answering through the rules in the KB. Of course, in realising such a system, several engineering details should be addressed which go behind the scope of this thesis.

Anyway, theorem provers for $\mathcal{L}$, $\mathcal{L}_+$, $\mathcal{ALC}$, $\mathcal{L}^f$, $\mathcal{L}^f_+$, HORN-$\mathcal{L}^f$, fuzzy $\mathcal{ALC}$ and horn fuzzy HORN-$\mathcal{ALC}$ have been implemented in standard Common Lisp.

11.5 Summary

The identification of the two orthogonal dimensions of multimedia data, the form dimension and the semantics dimension, respectively, leads to three type of retrieval: (*i*) the retrieval of objects along the form dimension; (*ii*) the retrieval of objects along the semantics dimension; and (*iii*) the retrieval of objects along both the form and semantics dimension.

Point 1. has been addressed in Chapter 10, whereas Point 2. and 3. have been addressed in this chapter. Both show clearly the capabilities of our overall model (in particular of the developed logic fuzzy HORN-$\mathcal{ALC}$) which go clearly beyond those of existing systems.

An additional topic, namely relevance feedback, has been addressed in Section 11.3 which, besides confirming the adequacy of our retrieval model, is a contribution to a relatively unexplored field of relevance feedback in the context of multimedia data representation and retrieval.

Part IV

Conclusions

Chapter 12

Conclusions

12.1 Contributions

We have presented a model for MIR. In a overall view, the model makes the following important contribution. It makes full and proper use of the semantics and knowledge in dealing with the retrieval of *e.g.* text, image, video and audio (and their composition), while offering, at the same time, the similarity-based kind of retrieval that is undoubtedly the most significant contribution of the research carried out in this area in the last decade. More importantly, all forms of retrieval coexist in a well-founded framework, which combines in a neat way the different techniques, notably digital signal processing and semantic information processing, required to deal with the various aspects of the model.

The main features of the model are that

- it is object-oriented, which makes it possible to use an object-oriented representation of multimedia data both at the form dimension and at the semantics dimension, by taking into account physical features as well as semantical features;
- it allows to represent the structure of multimedia objects, making the composition of objects explicit in terms of other objects;
- both kind of features, physical and semantical, are not predefined, so that new one can be created according to the application needs;
- the query language allows to address all kinds of retrieval on multimedia data that have been deemed as useful.

In a more detail, for each part of this thesis the contributions worked out in it can be summarised as follows.

Part I It presents our quite general and powerful object-oriented model for representing all relevant aspects of multimedia data at the form level. To the various objects a meaning may be associated through the logic fuzzy HORN-$\mathcal{ALC}$ developed in Part II;

Part II In this part undoubtedly the main effort has been done. It presents formally the logic fuzzy HORN-$\mathcal{ALC}$ for (i) representing multimedia data at the semantics level; (ii) integrating the form dimension and semantics dimension of multimedia data in a principled

way; and (*iii*) querying multimedia databases about form and semantics. In particular, the contributions in this part (together with Appendices C and D) involve:

- *at the propositional level*, (*i*) a new decision algorithm as well as computational complexity results w.r.t. four-valued semantics; (*ii*) a new fuzzy semantics, sound and complete reasoning algorithms and computational complexity results, as well as the specialisation to the horn case;

- *at the first-order level*, (*i*) a new four-valued semantics for DLs, sound and complete reasoning algorithms and computational complexity results. An extension to horn logic has been worked out, from a syntax, semantics and algorithms point of view; (*ii*) a new fuzzy extension for DLs has been worked out, both for classical semantics as well as for four-valued semantics: we have specified, syntax, semantics and reasoning algorithms. The extension to the horn case has been considered too, yielding our final logic fuzzy HORN-$\mathcal{ALC}$, and a new method for reasoning in it, which can be implemented through a standard prolog system, has been determined.

Part III Presents an integration of the form dimension and the semantics dimension of multimedia data within our logic fuzzy HORN-$\mathcal{ALC}$, as well as shows with concrete examples that fuzzy HORN-$\mathcal{ALC}$ allows to address all kinds of retrieval on multimedia data that have been deemed as useful. Finally, a relevance feedback algorithm within the multimedia context has been presented.

12.2 Future work

We believe that the presented model can open the way to a novel approach to the modeling of multimedia information and its retrieval, leading to the development of retrieval systems able to cope in a formally neat and practically adequate way with documents including any kind of media.

A primary key for further work concerns the development of a prototype implementing our model. A promising approach is to integrate a multimedia database, which provides all the necessary functionalities at the form dimension (like storage and similarity measures), with a prolog system, a DL system and an user interface. Besides the engineering difficulties of such an integration several critical points are involved:

1. the chosen logic should be a good compromise between computational complexity and expressive power. In particular, all DLs for which an unique completion can be computed are certainly preferable, *e.g.* $\mathcal{PL}_1$ and $\mathcal{PL}_2$ [95]. More probably, new and MIR specific DLs have to be developed. Additionally, some features like negation as failure [194] have to be worked out w.r.t. the fuzzy horn part;

2. another key issue concerns decision algorithms for DLs. A prototypical implementation should consider all possible enhancements developed in current research. A good reference system is [151, 152]. The extension of these methods to the fuzzy case seems far from being simple and should be optimised to the problem of determining the maximal degree;

3. research on how to combine similarity degrees deriving from different features of different media, *i.e.* the specification of similarity measures for data involving different media;

4. investigating on the implications of our model w.r.t. the storage and access of complex multimedia objects. In fact, real applications will require the storage of huge quantities of data. This implies that data placement is crucial for effective manipulation of the data and efficient retrieval;

5. perhaps the most exciting research topic relates to the definition of interpretation functions which automatically enables us to map complex multimedia objects into a set of fuzzy HORN-$\mathcal{ALC}$ facts. We are aware that this is a quite difficult job, but some results have been reached in cases in which the application domain is very restricted (see *e.g.* [4, 214]). This topic may involve certainly research in the area of machine learning;

6. last but not least, several aspects about relevance feedback have to be worked out. For instance, (*i*) notwithstanding a close resemblance of our algorithm to the simple text case, which has been shown to be effective, there is no guarantee that it will work until some experiments has been taken into consideration; and (*ii*) we confined our algorithm to the case in which the refinement process takes the form dimension into account only, *i.e.* the user's feedback involves data belonging to the form dimension only (relevant CMOs and nonrelevant CMOs). It may be of interest to extent the algorithm by considering semantical aspects (data concerning semantics) too.

Part V

Appendices

Appendix A

About Extensions to DLs

In the following we list some extensions, from an expressive power point of view, developed in the context of DLs (the list is not exhaustive).

Probabilistic extension : Probabilistic versions of DLs [149, 157, 166, 252, 295] may be investigated as a means of making explicit various sources of uncertainty, such as uncertainty related to domain knowledge and uncertainty related to automatic document representation, which is typical in IR;

Concrete domain extension : Ability to refer to concrete domains and predicates on these domains [21, 31], allowing to deal with data types like "string", "integer", "link" (link to the position of a keyword in a document, link to another related document, etc.), etc.;

Rule language extension : Rules, as those appearing in the context of frame-based systems (procedural rules), has been shown to be very useful in real applications as they helps to describe knowledge about the domain [97, 186, 187, 224];

Closed World Assumption, Closed Domain Reasoning : Close world reasoning and closed domain reasoning seem to be suitable for IR purposes, as they are close to usual databases reasoning [96, 231, 232, 233, 283];

Temporal extension : Integrating time into DLs using temporal logics and interval calculus, yielding a temporal DL which combines structural with temporal abstraction [11, 12, 13, 250];

N-ary terms extension : Usually, DLs allows the representation of at most two place relation. N-ary terms allows the representation of relation whose arity exceeds two [251];

OODBMS extension : Extension about the integration of DLs and Object Oriented Database Systems seems to be very useful for both [49];

Relational database operators extension : The operators of relational databases are integrated into DLs and are shown to be very useful [88];

Modal extension : Modal operators are integrated into DLs, yielding a modal DL which handles notions as belief, intention and time, which are essential for the representation of multi-agent environments [37];

Default extension : Default inheritance reasoning, a kind of default reasoning that is specifically oriented to reasoning on taxonomies (typical of frame-based systems) is included into DLs [33, 34, 264].

Appendix B

About connectives in DLs

Some typical connectives considered in the context of DLs (the list is far from being exhaustive).
Concept connectives syntax:

$$
\begin{array}{rcll}
C, D & \longrightarrow & A \mid & \text{(primitive concept)} \\
 & & C \sqcap D \mid & \text{(concept conjunction)} \\
 & & C \sqcup D \mid & \text{(concept disjunction)} \\
 & & \neg C \mid & \text{(concept negation)} \\
 & & \forall R.C \mid & \text{(universal quantification)} \\
 & & \exists R.C \mid & \text{(qualified existential quantification)} \\
 & & \exists R \mid & \text{(existential quantification)} \\
 & & \geq nR \mid & \text{(at least number restriction)} \\
 & & \leq nR \mid & \text{(at most number restriction)} \\
 & & \geq nRC \mid & \text{(qualified at least number restriction)} \\
 & & \leq nRC \mid & \text{(qualified at most number restriction)} \\
 & & \{a_1, \ldots, a_n\} & \text{(one-of)}
\end{array}
\tag{B.1}
$$

Role connectives syntax:

$$
\begin{array}{rcll}
R, Q & \longrightarrow & P \mid & \text{(primitive role)} \\
 & & R \sqcap Q \mid & \text{(role conjunction)} \\
 & & R \sqcup Q \mid & \text{(role disjunction)} \\
 & & \neg R \mid & \text{(role negation)} \\
 & & R^{-1} \mid & \text{(inverse role)} \\
 & & R|C \mid & \text{(role restriction)} \\
 & & C|R \mid & \text{(role domain restriction)} \\
 & & R \circ Q \mid & \text{(role composition)} \\
 & & C \times D \mid & \text{(domainrange role)} \\
 & & f & \text{(functional role)}
\end{array}
\tag{B.2}
$$

Concept connectives two-valued semantics[1]:

[1] $|S|$ is the cardinality of set S.

$$
\begin{array}{rll}
(C \sqcap D)^{\mathcal{I}}(d) = t & \text{iff} & C^{\mathcal{I}}(d) = t \text{ and } D^{\mathcal{I}}(d) = t \\
(C \sqcup D)^{\mathcal{I}}(d) = t & \text{iff} & C^{\mathcal{I}}(d) = t \text{ or } D^{\mathcal{I}}(d) = t \\
(\neg C)^{\mathcal{I}}(d) = t & \text{iff} & C^{\mathcal{I}}(d) = f \\
(\forall R.C)^{\mathcal{I}}(d) = t & \text{iff} & \text{for all } d' \in \Delta^{\mathcal{I}}, \text{ if } R^{\mathcal{I}}(d,d') = t \text{ then } C^{\mathcal{I}}(d') = t \\
(\exists R.C)^{\mathcal{I}}(d) = t & \text{iff} & \text{for some } d' \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(d,d') = t \text{ and } C^{\mathcal{I}}(d') = t \\
(\exists R)^{\mathcal{I}}(d) = t & \text{iff} & \text{for some } d' \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(d,d') = t \\
(\geq nR)^{\mathcal{I}}(d) = t & \text{iff} & |\{d' \in \Delta^{\mathcal{I}} \colon R^{\mathcal{I}}(d,d') = t\}| \geq n \\
(\leq nR)^{\mathcal{I}}(d) = t & \text{iff} & |\{d' \in \Delta^{\mathcal{I}} \colon R^{\mathcal{I}}(d,d') = t\}| \leq n \\
\{a_1, \ldots, a_n\}^{\mathcal{I}}(d) = t & \text{iff} & \text{for some } 1 \leq i \leq n, d = a_i{}^{\mathcal{I}}.
\end{array}
\tag{B.3}
$$

Role connectives semantics:

$$
\begin{array}{lll}
(R \sqcap Q)^{\mathcal{I}}(d,d') = t & \text{iff} & R^{\mathcal{I}}(d,d') = t \text{ and } Q^{\mathcal{I}}(d,d') = t \\
(R \sqcup Q)^{\mathcal{I}}(d,d') = t & \text{iff} & R^{\mathcal{I}}(d,d') = t \text{ or } Q^{\mathcal{I}}(d,d') = t \\
(\neg R)^{\mathcal{I}}(d,d') = t & \text{iff} & R^{\mathcal{I}}(d,d') = f \\
(R^{-1})^{\mathcal{I}}(d,d') = t & \text{iff} & R^{\mathcal{I}}(d',d) = t \\
(R|C)^{\mathcal{I}}(d,d') = t & \text{iff} & R^{\mathcal{I}}(d,d') = t \text{ and } C^{\mathcal{I}}(d') = t \\
(C|R)^{\mathcal{I}}(d,d') = t & \text{iff} & R^{\mathcal{I}}(d,d') = t \text{ and } C^{\mathcal{I}}(d) = t \\
(R \circ Q)^{\mathcal{I}}(d,d') = t & \text{iff} & \text{for some } d'' \in \Delta^{\mathcal{I}}, R^{\mathcal{I}}(d,d'') = t \text{ and } Q^{\mathcal{I}}(d'',d') = t \\
(C \times D)^{\mathcal{I}}(d,d') = t & \text{iff} & C^{\mathcal{I}}(d') = t \text{ and } D^{\mathcal{I}}(d') = t \\
f^{\mathcal{I}}(d,d') = t & \text{iff} & f \text{ functional.}
\end{array}
\tag{B.4}
$$

Appendix C

Crisp decision algorithms

C.1 Deciding entailment in $\mathcal{L}$

Effectively deciding whether $\Sigma \models_4 A$ requires a calculus. A well known algorithm for deciding entailment in $\mathcal{L}$ is Levesque's algorithm [182] which is shown in Table C.1 below.

Algorithm 2 ($Lev(\Sigma, A)$)
In $\mathcal{L}$, let Σ be a KB and let A be a proposition. In order to check whether $\Sigma \models_4 A$,

1. put Σ and A into equivalent CNFs[a]. Call the results of this transformation Σ_{CNF} and A_{CNF}, respectively;
2. verify whether for each conjunct A^j_{CNF} of A_{CNF} there is a proposition $B_{CNF} \in \Sigma_{CNF}$ and a conjunct B^i_{CNF} of B_{CNF} such that $B^i_{CNF} \subseteq A^j_{CNF}$, where B^i_{CNF} and A^j_{CNF} are seen as clauses. ■

[a] A set is in CNF iff each element of it is.

Table C.1: Algorithm $Lev(\Sigma, A)$.

Hence, entailment between Σ and A in CNF can be verified in time $O(|\Sigma||A|)$, whereas checking whether $\Sigma \models_4 A$ is a coNP-complete problem in the general case [218]. Just notice that this algorithm has ben extended to the DL case in [217, 220, 222]. Unfortunately, it does not work within the four-valued semantics we discussed in this thesis.

A final remark on the above algorithm is that it gives us a simple way in order to proof Proposition 1. In fact, assume $A \models_4 B$ which is equivalent to assume $A_{CNF} \models_4 B_{CNF}$, where A_{CNF} is $A_1 \wedge A_2 \ldots \wedge A_n$ and B_{CNF} is $B_1 \wedge B_2 \ldots \wedge B_m$. Now, $\neg A_{CNF}$ can be transformed into an equivalent *Disjunctive Normal Form* (DNF) A', similarly for $\neg B_{CNF}$ for which there is an equivalent DNF B', where A' is $\neg A_1 \vee \neg A_2 \ldots \vee \neg A_n$ and B' is $\neg B_1 \vee \neg B_2 \ldots \vee \neg B_m$. Now, let $\mathcal{I}$ be a model of $\neg B$. Hence $\mathcal{I}$ is a model of $\neg B_{CNF}$ and, thus, of B'. Therefore, for some j, $\mathcal{I}$ is a model of $\neg B_j$, *i.e.* $f \in {B_j}^{\mathcal{I}}$. From Levesque's algorithm we have that there is an i such that $A_i \subseteq B_j$. Since both A_i and B_j are disjunctions of literals, $f \in {B_j}^{\mathcal{I}}$ implies $f \in {A_i}^{\mathcal{I}}$, which in turn implies $t \in \neg {A_i}^{\mathcal{I}}$. Therefore, $t \in {A'}^{\mathcal{I}}$ and, thus, $t \in \neg A^{\mathcal{I}}$.

From an algorithmic point of view, we propose an alternative calculus which (*i*) does not require any transformation into CNF, which besides can be exponential in length with respect to the original formula, (*ii*) has the same polynomial complexity in the CNF case and (*iii*) is easy extensible to the DL case.

The calculus we will develop is one inspired on the calculus **KE** [85] for determining entailment with respect to two-valued semantics. **KE** has been introduced in order to solve the anomalies of *cut-free* (like analytic tableaux) calculi. It is not our aim to discuss these anomalies here. The interested reader may consult [85, 132, 133]. We mention only that analytic tableaux like calculi have the characteristic that a proof of a proposition A may be exponential with respect to the length of A rather than with respect to the number of different letters occurring in A. In fact, consider the formula $C = (A \vee B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B)$. The number of different letters is 2. Hence, there are exactly 2^2 possible two-valued interpretations. As a consequence, a semantic based decision procedure like the Davis-Longemann-Loveland procedure [87] stops after enumerating at most 4 interpretations. On the other hand an analytic tableaux calculus [119] using the two rules

$$\wedge\text{-rule } \frac{A \wedge B}{A, B} \quad \vee\text{-rule } \frac{A \vee B}{A \mid B} \tag{C.1}$$

generates $O(2^3)$ leafs as shown in Figure C.1. Analytic tableaux perform a *syntactic branching*,

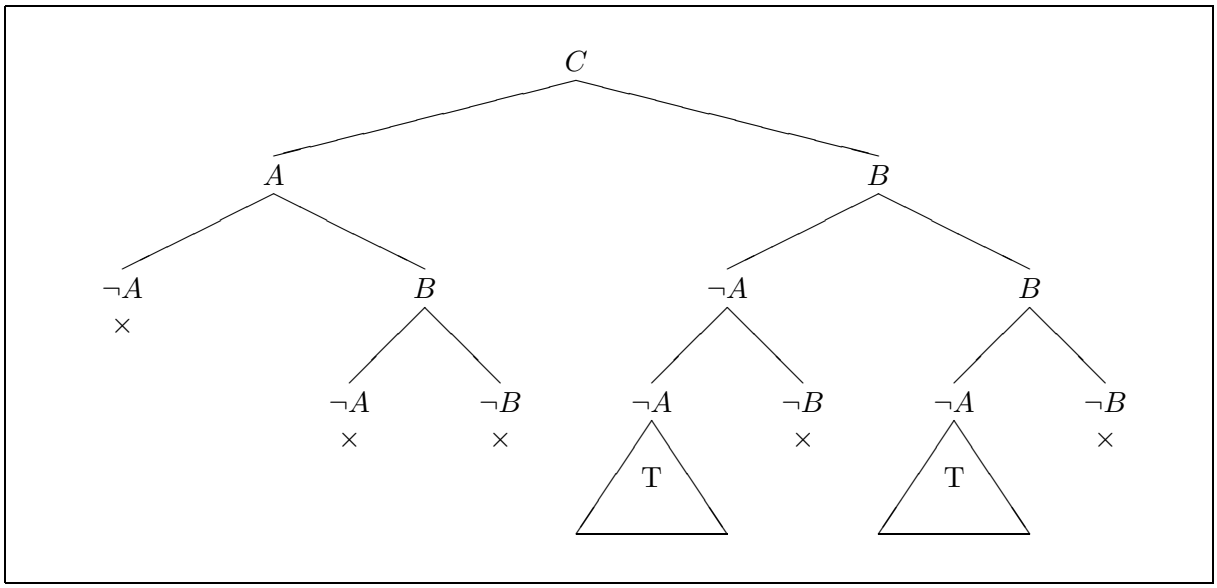


Figure C.1: Tableaux for $C = (A \vee B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B)$.

that is, a branching on the syntactic structure of proposition C. As discussed in [85], any application of the $\vee - rule$ generates two subtrees which are *not mutually inconsistent*, that is two subtrees which may share propositional models (like tree T in Figure C.1). The set of interpretations enumerated by analytic tableaux procedures is intrinsically redundant. As a consequence, the number of interpretations generated grows exponentially with the number of disjunctions occurring (positively) in C, although the number of non-redundant interpretations satisfying C is much smaller. This redundancy is a source of a high degree of

inefficiency when using analytic tableaux for propositional satisfiability.

The calculus **KE** does not have this problem [85]. The calculus, a *semantic tableaux*, is based on *signed propositions of type* α (called *of conjunctive type*) and *type* β (called *of disjunctive type*) and on their *components* which are defined as usual [260]. In the table below (Table C.2) T and NT play the role of "True" and "Not True", respectively.:

α	α_1	α_2
$\mathsf{T}A \wedge B$	$\mathsf{T}A$	$\mathsf{T}B$
$\mathsf{NT}A \vee B$	$\mathsf{NT}A$	$\mathsf{NT}B$

β	β_1	β_2
$\mathsf{T}A \vee B$	$\mathsf{T}A$	$\mathsf{T}B$
$\mathsf{NT}A \wedge B$	$\mathsf{NT}A$	$\mathsf{NT}B$

Table C.2: α and β table for $\mathcal{L}$.

For instance, in Table C.2, the signed proposition of type α, $\mathsf{T}A \wedge B$, may be interpreted: "if $A \wedge B$ is true then A *and* B are true". Whereas, the signed propositions of type β, $\mathsf{NT}A \vee B$, may be interpreted as follows: "if $A \vee B$ is not true then neither A is true *nor* B is not true".

In the following we will use σ as metavariable for signed expressions. $\mathsf{T}A$ and $\mathsf{NT}A$ are called *conjugated signed propositions*. They represent an inconsistent situation. With β_i^c we indicate the *conjugate* of β_i. For instance, the conjugate of $\mathsf{T}A$ is $\mathsf{NT}A$ and vice-versa. We extend the definition of satisfiability to signed propositions as follows. An interpretation $\mathcal{I}$ *satisfies* $\mathsf{T}A$ iff $\mathcal{I}$ satisfies A, whereas $\mathcal{I}$ *satisfies* $\mathsf{NT}A$ iff $\mathcal{I}$ does not satisfy A. $\mathcal{I}$ *satisfies* a set of signed propositions iff $\mathcal{I}$ *satisfies* each element of it. As a consequence,

$$\Sigma \models_4 A \text{ iff } \mathsf{T}\Sigma \cup \{\mathsf{NT}A\} \text{ is not satisfiable} \tag{C.2}$$

where $\mathsf{T}\Sigma = \{\mathsf{T}B : B \in \Sigma\}$.

The calculus is based on the following four rules described in Table C.3 below.

$$(A)\ \frac{\alpha}{\alpha_1, \alpha_2} \qquad (B1)\ \frac{\beta, \beta_1^c}{\beta_2} \qquad (B2)\ \frac{\beta, \beta_2^c}{\beta_1}$$

$$(PB)\ \frac{\beta}{\beta_1 \mid \beta_1^c, \beta_2}$$

Table C.3: Semantic tableaux inference rules.

It is worth noting that the only branching rule is the rule (PB) (called *Principle of Bivalence*). We have restricted the proof procedure to the so-called *canonical* form [85, p. 299], rather using the general form

$$(PB - general)\ \frac{}{\mathsf{T}A \mid \mathsf{NT}A} \tag{C.3}$$

Just notice that the (PB) rule is a shorthand for the application of the general $(PB - general)$ rule to β and successively applying to the right hand side the $(B1)$ rule with arguments β and β_1^c.

As usual, a deduction is represented as a tree, called *deduction tree*. A branch ϕ in a deduction tree is *closed* iff for some proposition A, both $\mathsf{T}A$ and $\mathsf{NT}A$ are in ϕ. A *closed deduction tree* is a deduction tree in which all branches are closed. With S^ϕ we indicate the set of signed propositions occurring in ϕ. A set of signed propositions S has a *refutation* iff in each deduction tree all branches ϕ are closed.

For instance, Figure C.2 is a closed deduction tree for $A \wedge (B \vee C) \models_4 (A \vee C) \wedge (B \vee C \vee D)$.

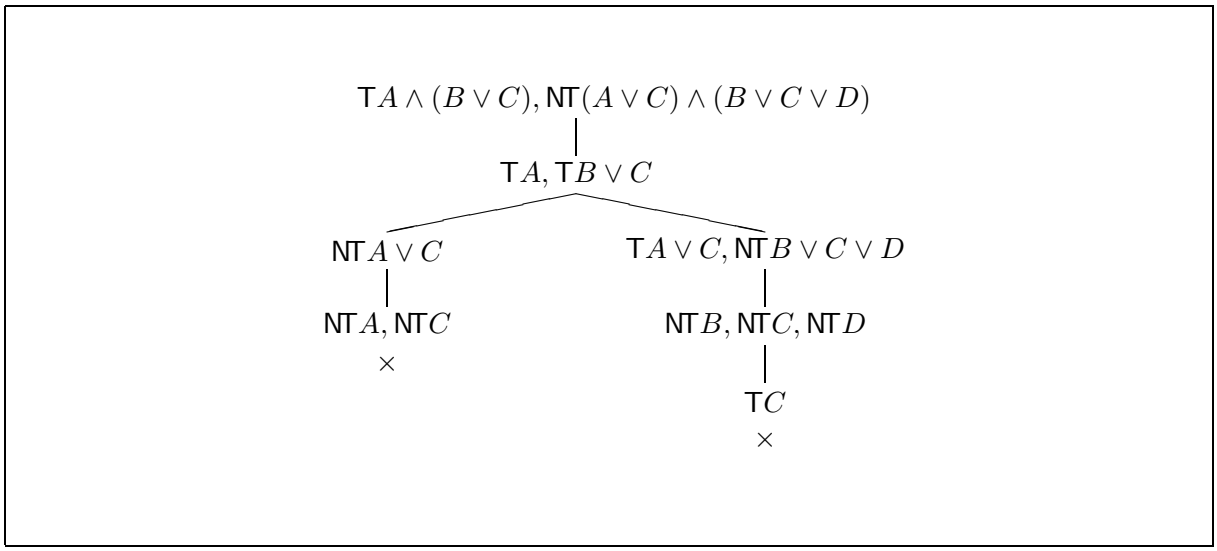


Figure C.2: Deduction tree for $A \wedge (B \vee C) \models_4 (A \vee C) \wedge (B \vee C \vee D)$.

A signed proposition is *AB-analysed* in a branch ϕ if either (i) it is of type α and both α_1 and α_2 occur in ϕ; or (ii) it is of type β and (iia) if β_1^c occurs in ϕ then β_2 occurs in ϕ, (iib) if β_2^c occurs in ϕ then β_1 occurs in ϕ. A branch is *AB-completed* if all the signed propositions in it are AB-analysed. A signed proposition of type β is *fulfilled* in a branch ϕ if either β_1 or β_2 occurs in ϕ. We say that a branch ϕ is *completed* if it is AB-completed and, every signed proposition of type β occurring in ϕ is fulfilled. A deduction tree is *completed* if all its branches are completed.

The procedure $Sat(S)$ below determines whether S is satisfiable or not.
The following proposition can easily be shown.

Proposition 18 *If S is a set of signed propositions in $\mathcal{L}$ then $Sat(S)$ iff S is satisfiable.* $\dashv$

Proof: It can be easily verified that the rules (A), $(B1)$, $(B2)$ and (PB) are correct, *i.e.* ϕ is a branch and S^ϕ is satisfiable iff there is a branch ϕ' as the result of the application of a rule to ϕ such that $S^{\phi'}$ satisfiable.
$\Rightarrow$.) Suppose $Sat(S)$. Let T be the generated deduction tree and let ϕ be a not closed branch from S to a leaf in T. Such a branch has to exist, otherwise $Sat(S) = false$. Let

$$S^{\mathsf{T}} = \{\mathsf{T}A \in S^\phi\}, \tag{C.4}$$

$$S^{\mathsf{NT}} = \{\mathsf{NT}A \in S^\phi\}. \tag{C.5}$$

Algorithm 3 ($Sat(S)$)
$Sat(S)$ starts from the root labelled S and applies the rules until the resulting tree is either closed or completed. If the tree is closed, $Sat(S)$ returns false, otherwise true. At each step of the construction the following steps are performed:

1. select a branch ϕ which is not yet completed;
2. expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed;
3. if the resulting branch ϕ' is neither closed nor completed then
   (a) select a signed proposition of type β which is not yet fulfilled in the branch;
   (b) apply rule (PB) and go to Step 1.

   otherwise, go to Step 1. ■

Table C.4: Algorithm $Sat(S)$ for $\mathcal{L}$.

Of course, $S^\phi = S^{\mathsf{T}} \cup S^{\mathsf{NT}}$. Let $\mathcal{I}$ be a relation such that $t \in A^{\mathcal{I}}$ if $\mathsf{T}A \in S^{\mathsf{T}}$, $A^{\mathcal{I}} = \emptyset$ otherwise. More precisely, for each propositional letter p, set $p^{\mathcal{I}} = \emptyset$ and

1. for each $\mathsf{T}p \in S^{\mathsf{T}}$, assign $p^{\mathcal{I}} := p^{\mathcal{I}} \cup \{t\}$;
2. for each $\mathsf{T}\neg p \in S^{\mathsf{T}}$, assign $p^{\mathcal{I}} := p^{\mathcal{I}} \cup \{f\}$.

Since, ϕ is completed and not closed, $\mathcal{I}$ is a four-valued interpretation satisfying S^{T}, S^{NT} and, thus, S^ϕ. As a consequence, $S \subseteq S^\phi$ is satisfiable.
$\Leftarrow$.) Suppose S is satisfiable. Let T be the generated completed tree. From the correctness of the rules it follows that there is a completed branch ϕ in T such that S^ϕ is satisfiable. Therefore, $Sat(S)$. Q.E.D.

As example, in Figure C.3 a deduction tree of $\mathsf{T}(A \vee B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B)$ is shown.

If we switch to the classical two-valued setting, soundness and completeness is obtained by extending signed propositions as usual [119] by adding the following signed propositions of type α too.

α	α_1	α_2
$\mathsf{T}\neg A$	$\mathsf{NT}A$	$\mathsf{NT}A$
$\mathsf{NT}\neg A$	$\mathsf{T}A$	$\mathsf{T}A$

Hence, the resulting α and β tables for two-valued $\mathcal{L}$ are described in Table C.5 below.
Just notice that in this case $Sat(S)$ is exactly the canonical procedure for **KE** [85]. Therefore, in the general case the only difference between four-valued and two-valued semantics relies on the negation connective. This is not a surprise as we already said that the semantics for the negation is constructive, *i.e.* expressed in terms of $\in$ rather than on $\notin$. Figure C.4 shows the proof of $(A \vee B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B) \models_2 \neg A \wedge B$.

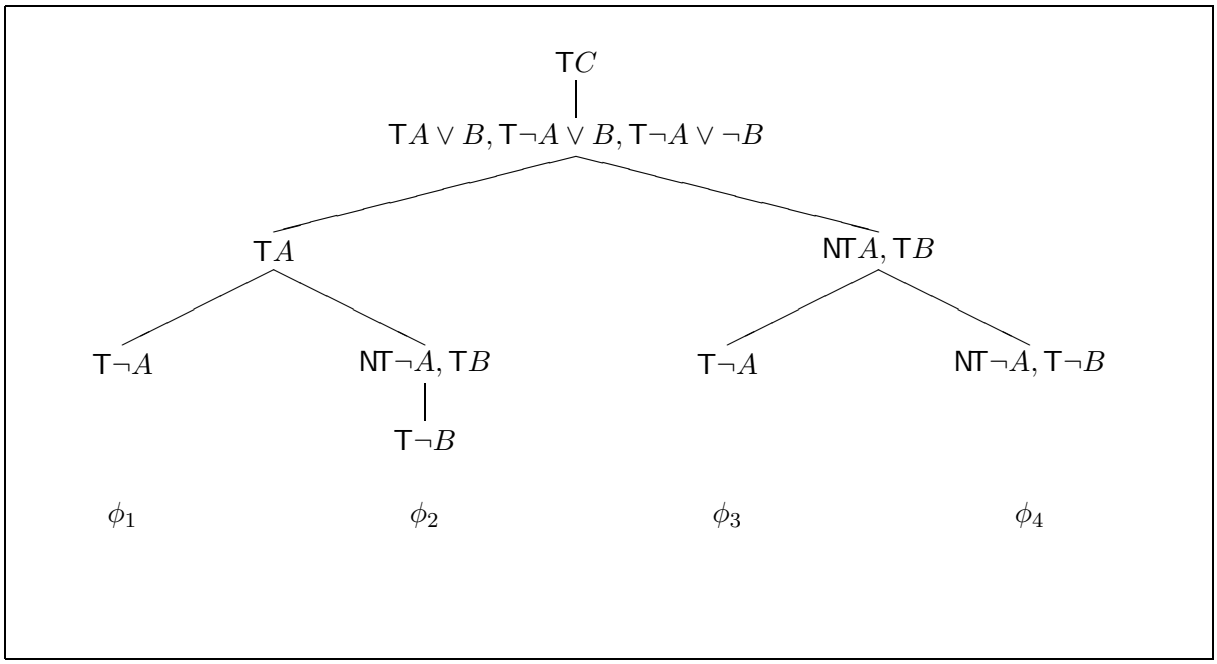


Figure C.3: Deduction tree for $\mathsf{T}C = \mathsf{T}(A \vee B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B)$.

α	α_1	α_2
$\mathsf{T}A \wedge B$	$\mathsf{T}A$	$\mathsf{T}B$
$\mathsf{NT}A \vee B$	$\mathsf{NT}A$	$\mathsf{NT}B$
$\mathsf{T}\neg A$	$\mathsf{NT}A$	$\mathsf{NT}A$
$\mathsf{NT}\neg A$	$\mathsf{T}A$	$\mathsf{T}A$

β	β_1	β_2
$\mathsf{T}A \vee B$	$\mathsf{T}A$	$\mathsf{T}B$
$\mathsf{NT}A \wedge B$	$\mathsf{NT}A$	$\mathsf{NT}B$

Table C.5: α and β table for two-valued $\mathcal{L}$.

Consider the proof of Proposition 18. Consider the sets S^{T} and S^{NT} build during the proof (see Equation $(C.4)$ and Equation $(C.5)$), as the result of running $Sat(S)$. The set

$$\begin{aligned} \tilde{\mathcal{S}} \;=\; & \{\mathsf{T}p \in S^{\mathsf{T}} : p \text{ letter}\} \cup \\ & \{\mathsf{T}\neg p \in S^{\mathsf{T}} : p \text{ letter}\} \cup \\ & \{\mathsf{NT}p \in S^{\mathsf{NT}} : p \text{ letter}\} \cup \\ & \{\mathsf{NT}\neg p \in S^{\mathsf{NT}} : p \text{ letter}\} \end{aligned} \tag{C.6}$$

is called a *four-valued completion* of S. A *two-valued completion* of S is the set

$$\begin{aligned} \tilde{\mathcal{S}} \;=\; & \{\mathsf{T}p \in S^{\mathsf{T}} : p \text{ letter}\} \cup \\ & \{\mathsf{NT}p \in S^{\mathsf{NT}} : p \text{ letter}\}. \end{aligned} \tag{C.7}$$

Moreover, we define

$$\tilde{\mathcal{S}}^{\mathsf{T}} = \{\mathsf{T}A \in \tilde{\mathcal{S}}\}, \tag{C.8}$$

$$\tilde{\mathcal{S}}^{\mathsf{NT}} = \{\mathsf{NT}A \in \tilde{\mathcal{S}}\}, \tag{C.9}$$

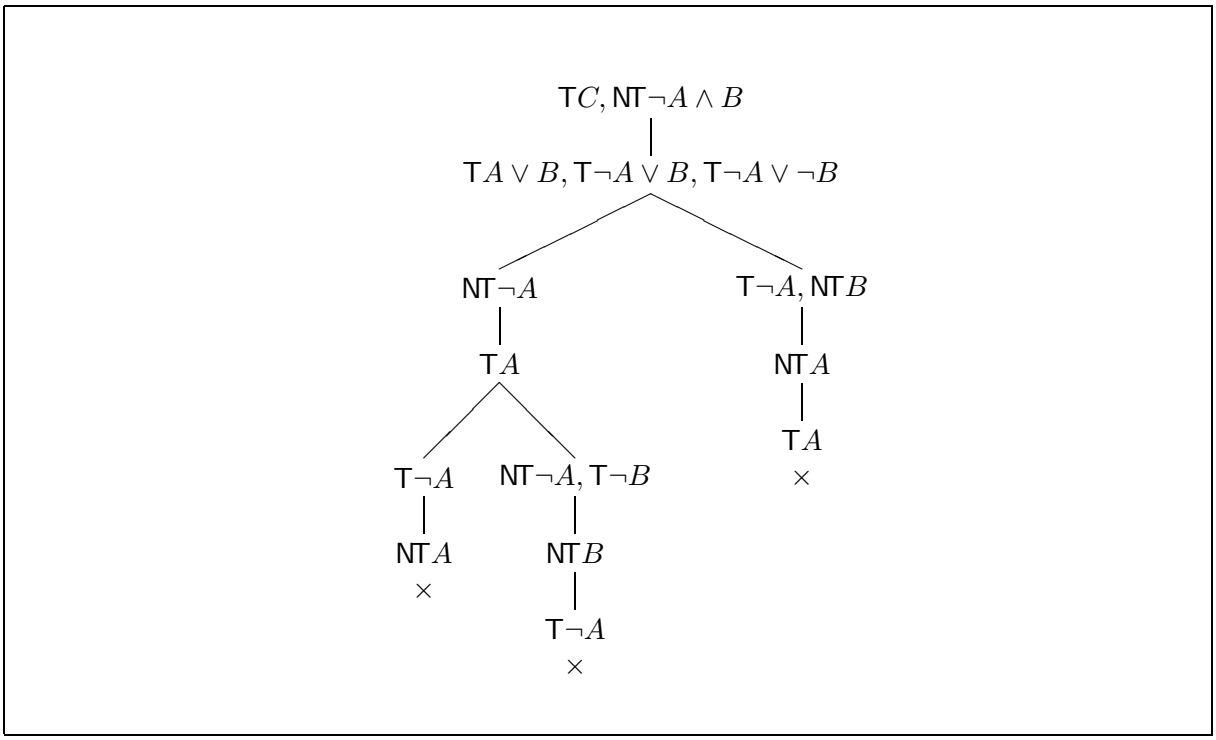


Figure C.4: Deduction tree for $(A \vee B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B) \models_2 \neg A \wedge B$.

$$\tilde{\mathcal{S}}^{+} = \{\mathsf{T}p \in \tilde{\mathcal{S}} : p \text{ letter}\}. \tag{C.10}$$

Given a four-valued completion $\tilde{\mathcal{S}}$ of S, we define the following *four-valued completion KBs* of $\tilde{\mathcal{S}}$:

$$\Sigma_{\tilde{\mathcal{S}}} = \{A : \mathsf{T}A \in \tilde{\mathcal{S}}^{\mathsf{T}}\}, \tag{C.11}$$
$$\Sigma^{+}_{\tilde{\mathcal{S}}} = \{A : \mathsf{T}A \in \tilde{\mathcal{S}}^{+}\}. \tag{C.12}$$

For the two-valued case, given a two-valued completion $\tilde{\mathcal{S}}$ of S, we define the following *two-valued completion KBs* of $\tilde{\mathcal{S}}$:

$$\Sigma_{\tilde{\mathcal{S}}} = \{p : \mathsf{T}p \in \tilde{\mathcal{S}}^{\mathsf{T}}\} \cup \{\neg p : \mathsf{NT}p \in \tilde{\mathcal{S}}^{\mathsf{NT}}\}, \tag{C.13}$$
$$\Sigma^{+}_{\tilde{\mathcal{S}}} = \{p : \mathsf{T}p \in \tilde{\mathcal{S}}^{+}\}. \tag{C.14}$$

Given a four-valued completion $\tilde{\mathcal{S}}$ of S, then a *four-valued canonical model* $\mathcal{I}$ of $\tilde{\mathcal{S}}$ is obtained as in proof of Proposition 18: for each propositional letter p, set $p^{\mathcal{I}} := \emptyset$ and

1. for each $\mathsf{T}p \in \tilde{\mathcal{S}}$, assign $p^{\mathcal{I}} := p^{\mathcal{I}} \cup \{t\}$;
2. for each $\mathsf{T}\neg p \in \tilde{\mathcal{S}}$, assign $p^{\mathcal{I}} := p^{\mathcal{I}} \cup \{f\}$.

On the other hand, given a two-valued completion $\tilde{\mathcal{S}}$ of S, then a *two-valued canonical model* of $\tilde{\mathcal{S}}$ is obtained as follows. Let $\mathcal{I}$ be an arbitrary two-valued interpretation. For each propositional letter p, redefine $\mathcal{I}$ as follows:

1. for each $\mathsf{T}p \in \tilde{\mathcal{S}}$, assign $p^{\mathcal{I}} := \{t\}$;
2. for each $\mathsf{NT}p \in \tilde{\mathcal{S}}$, assign $p^{\mathcal{I}} := \{f\}$.

It is easily verified that a canonical model of $\tilde{\mathcal{S}}$ is also a model of S.

It is worth noticing that in the four-valued case, given $\tilde{\mathcal{S}}$, the canonical model of $\tilde{\mathcal{S}}$ is unique, whereas in the two-valued case there are 2^{l-s} models, where l is the number of propositional letters appearing in $\mathcal{L}$ and s is the number of propositional letters appearing in $\tilde{\mathcal{S}}$. Essentially, in this last case, $\tilde{\mathcal{S}}$ allows us to define the truth of the s letters appearing in $\tilde{\mathcal{S}}$, whereas for all the others the choice of the truth value is free.

The following Algorithm 4 allows us to build all completions $\tilde{\mathcal{S}}$ of a set of signed propositions S. It is a simple extension of the procedure $Sat(S)$, according to the proof of Proposition 18.

Algorithm 4 (*Completions*(S))
Essentially, *Completions*(S) proceeds in a similar way as $Sat(S)$:

1. select a branch ϕ which is not yet completed;
2. expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed;
3. if the resulting branch ϕ' is neither closed nor completed then
   (a) select a signed proposition of type β which is not yet fulfilled in the branch;
   (b) apply rule $(PB)$and go to Step 1.

   otherwise, go to Step 1.
4. let T be the generated deduction tree. If all branches are closed, then set *Completions*(S):$=\emptyset$ and exit. Otherwise,
5. for all not closed branches ϕ from S to a leaf in T do
   (a) let $S^{\mathsf{T}} = \{\mathsf{T}A \in S^{\phi}\}$ and $S^{\mathsf{NT}} = \{\mathsf{NT}A \in S^{\phi}\}$;
   (b) define $\tilde{\mathcal{S}}$ according to $(C.6)$ or $(C.7)$, in case four-valued or two-valued semantics, respectively;
   (c) *Completions*(S):$=$*Completions*(S) $\cup \{\tilde{\mathcal{S}}\}$. ■

Table C.6: Algorithm *Completions*(S) for $\mathcal{L}$.

The following proposition follows immediately.

Proposition 19 *In $\mathcal{L}$, let Σ be a KB and let A be a proposition. Then*

1. $\Sigma \models_4 A$ *iff for all four-valued canonical models* $\mathcal{I}$ *of four-valued completions* $\tilde{\mathcal{S}}$ *in* $Completions(\mathsf{T}\Sigma)$, $\mathcal{I}$ *satisfies* A;

2. $\Sigma \models_2 A$ *iff for all two-valued canonical models* $\mathcal{I}$ *of two-valued completions* $\tilde{\mathcal{S}}$ *in* $Completions(\mathsf{T}\Sigma)$, $\mathcal{I}$ *satisfies* A. ⊣

Just notice here that there are finitely many four-valued canonical models, whereas there could be an infinite number of two-valued canonical models.

A consequence of Proposition 19 is

Proposition 20 *In* $\mathcal{L}$, *let* Σ *be a KB and let* A *be a proposition. Then*

1. $\Sigma \models_4 A$ *iff for all four-valued completions* $\tilde{\mathcal{S}} \in Completions(\mathsf{T}\Sigma)$, $\tilde{\mathcal{S}}^{\mathsf{T}} \cup \{\mathsf{NT}A\}$ *is not satisfiable;*

2. $\Sigma \models_4 A$ *iff for four-valued completions* $\tilde{\mathcal{S}} \in Completions(\mathsf{T}\Sigma)$, $\Sigma_{\tilde{\mathcal{S}}} \models_4 A$, *where* $\Sigma_{\tilde{\mathcal{S}}}$ *is the four-valued completion KB of* $\tilde{\mathcal{S}}$;

3. $\Sigma \models_2 A$ *iff for all two-valued completions* $\tilde{\mathcal{S}} \in Completions(\mathsf{T}\Sigma)$, $\tilde{\mathcal{S}} \cup \{\mathsf{NT}A\}$ *is not satisfiable;*

4. $\Sigma \models_2 A$ *iff for two-valued completions* $\tilde{\mathcal{S}} \in Completions(\mathsf{T}\Sigma)$, $\Sigma_{\tilde{\mathcal{S}}} \models_2 A$, *where* $\Sigma_{\tilde{\mathcal{S}}}$ *is the two-valued completion KB of* $\tilde{\mathcal{S}}$. ⊣

Let us consider the following example illustrating the properties above.

Example 27 Consider $C = (A \vee B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B)$. As we already noticed (see Example 13), in a four-valued setting there are more models than two-valued models w.r.t. proposition C. In fact, the following table enumerates the possible models of C.

Models of C	A	B
$\mathcal{I}_1$	$\{t, f\}$	$\emptyset$
$\mathcal{I}_2$	$\{t\}$	$\{t, f\}$
$\mathcal{I}_3$	$\{f\}$	$\{t\}$
$\mathcal{I}_4$	$\emptyset$	$\{t, f\}$
$\mathcal{I}_5$	$\{t, f\}$	$\{t\}$
$\mathcal{I}_6$	$\{t, f\}$	$\{f\}$
$\mathcal{I}_7$	$\{t, f\}$	$\{t, f\}$
$\mathcal{I}_8$	$\{f\}$	$\{t, f\}$

and only $\mathcal{I}_3$ is a two-valued model of C.

A deduction tree of $\mathsf{T}(A \vee B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B)$ is shown in Figure C.3. The set of four-valued completions, $Completions(\mathsf{T}C)$, is

$$\begin{aligned}
\tilde{\mathcal{S}}_1 &= \{\mathsf{T}A, \mathsf{T}\neg A\} \\
\tilde{\mathcal{S}}_2 &= \{\mathsf{T}A, \mathsf{T}B, \mathsf{T}\neg B, \mathsf{NT}\neg A\} \\
\tilde{\mathcal{S}}_3 &= \{\mathsf{T}\neg A, \mathsf{T}B, \mathsf{NT}A\} \\
\tilde{\mathcal{S}}_4 &= \{\mathsf{T}B, \mathsf{T}\neg B, \mathsf{NT}A, \mathsf{NT}\neg A\},
\end{aligned}$$

and their four-valued completion KBs are

$$\begin{array}{lcl}\Sigma_{\tilde{\mathcal{S}}_1} & = & \{A, \neg A\} \\ \Sigma_{\tilde{\mathcal{S}}_2} & = & \{A, B, \neg B\} \\ \Sigma_{\tilde{\mathcal{S}}_3} & = & \{\neg A, B\} \\ \Sigma_{\tilde{\mathcal{S}}_4} & = & \{B, \neg B\}.\end{array}$$

With respect to two-valued semantics, the only two-valued completion corresponds to branch ϕ_3:

$$\tilde{\mathcal{S}}_5 \quad = \quad \{\mathsf{T}B, \mathsf{NT}A\}$$

and its two-valued completion KB is

$$\Sigma_{\tilde{\mathcal{S}}_5} \quad = \quad \{\neg A, B\}.$$

Moreover, the set of four-valued canonical models of four-valued completions is M_4, where

$$M_4 = \{\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3, \mathcal{I}_4\}.$$

according to the paths $\phi_1, \ldots \phi_4$ in Figure C.3. The set of two-valued canonical models of two-valued completions is $M_2 \subseteq M_4$, where

$$M_2 = \{\mathcal{I}_3\}.$$

Now it can be verified that $C \not\models_4 \neg A \wedge B$, whereas $C \models_2 \neg A \wedge B$: in fact,

1. $\mathcal{I}_1$ does not satisfy $\neg A \wedge B$ and, thus, $C \not\models_4 \neg A \wedge B$ by Proposition 19. Similarly, from Proposition 20, $\tilde{\mathcal{S}}_1 \cup \{\mathsf{NT}\neg A \wedge B\}$ is satisfiable and $\Sigma_{\tilde{\mathcal{S}}_1} \not\models_4 \neg A \wedge B$ follows;
2. $\mathcal{I}_3$ does satisfy $\neg A \wedge B$ and, thus, $C \models_2 \neg A \wedge B$ by Proposition 19. Similarly, from Proposition 20, $\tilde{\mathcal{S}}_5 \cup \{\mathsf{NT}\neg A \wedge B\}$ is not two-valued satisfiable and $\Sigma_{\tilde{\mathcal{S}}_5} \models_2 \neg A \wedge B$ follows. ■

As we have seen, Levesque's algorithm $Lev(\Sigma, A)$ determines whether $\Sigma \models_4 A$ in time $O(|\Sigma||A|)$ whenever Σ and A are in CNF. We now present a specialised version of the *Sat* procedure: the following algorithm $EasyEntail(\Sigma, A)$ in Table C.7 determines whether $\Sigma \models_4 A$, where Σ is a KB in CNF and A is a proposition in CNF and runs in polynomial. It is worth noting that the important point in algorithm $EasyEntail(\Sigma, A)$ is at Step 3a: the (PB) rule is applied only on signed propositions of type β of the form $\mathsf{NT}B$, *i.e.* of the form $\mathsf{NT}B_1 \wedge \ldots \wedge B_n$ and *not* to signed propositions of type $\mathsf{T}B_1 \vee \ldots \vee B_n$. In particular, *the* (PB) *rule is applied at most* n *times* during the execution of $EasyEntail(\Sigma, A_1 \wedge \ldots \wedge B_n)$. Therefore, every deduction tree build by *EasyEntail* has at most n branches $\phi_1, \ldots, \phi_n$. Moreover, we will show that the depth of each branch ϕ_i is $O(|\Sigma||A_i|)$ and, thus, $EasyEntail(\Sigma, A)$ runs in time $O(|\Sigma||A|)$ and is correct and complete whenever Σ and A are in CNF.

Example 28 Figure C.2 is a closed deduction tree build by the application of $EasyEntail(\Sigma, F)$, where Σ is $\{A \wedge (B \vee C)\}$ and F is $(A \vee C) \wedge (B \vee C \vee D)$, confirming that $\Sigma \models_4 F$. Moreover, the depth of the left branch is bounded by $|\Sigma||A \vee C|$, whereas the depth of the right branch is bounded by $|\Sigma||B \vee C \vee D|$. Hence, $EasyEntail(\Sigma, F)$ runs in time $O(|\Sigma||F|)$. ■

Algorithm 5 ($EasyEntail(\Sigma, A)$)
$EasyEntail$ takes as input a KB Σ in CNF and a proposition A in CNF. $EasyEntail(\Sigma, A) = true$ iff $\Sigma \models_4 A$. Let S be $\mathsf{T}\Sigma \cup \{\mathsf{NT}A\}$. $EasyEntail$ starts from the root labelled S and applies the rules until the resulting tree is either closed or completed. If the tree is closed, $EasyEntail$ returns true, otherwise false. At each step of the construction the following steps are performed:

1. select a branch ϕ which is not yet completed;
2. expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed;
3. if the resulting branch ϕ' is neither closed nor completed then
   (a) select a signed proposition of type β of the form $\mathsf{NT}B$ which is not yet fulfilled in the branch;
   (b) apply rule (PB) and go to Step 1.

   otherwise, go to Step 1. ■

Table C.7: Algorithm $EasyEntail(\Sigma, A)$ in $\mathcal{L}$.

The following proposition holds.

Proposition 21 *In $\mathcal{L}$, let both Σ and A be in CNF.*

1. *$EasyEntail(\Sigma, A) = true$ iff $\Sigma \models_4 A$.*
2. *$EasyEntail(\Sigma, A)$ runs in time $O(|\Sigma||A|)$.* ⊣

Proof: Let Σ and A be in CNF. Let A be $A_1 \wedge \ldots A_n$, where each A_i is a disjunction of literals (letters or the negation of letters).

1. Since $EasyEntail$ is a special form of Sat, from Proposition 18 if $EasyEntail(\Sigma, A) = true$ then $\Sigma \models_4 A$ follows. Therefore, $EasyEntail$ is correct. We show now completeness, *i.e.* if $\Sigma \models_4 A$ then $EasyEntail(\Sigma, A) = true$.

   Suppose $\Sigma \models_4 A$. Now, we already know from Levesque' complete algorithm Lev that $\Sigma \models_4 A$ holds iff for all A_i there is a proposition $B \in \Sigma$ in CNF and a conjunct B_j of B such that $B_j \subseteq A_i$, where B_j and A_i are seen as clauses. Let S be $\mathsf{T}\Sigma \cup \{\mathsf{NT}A\}$.

   Case $n = 1$**:** Hence, A is A_1, A_1 is $A_1^1 \vee A_1^2 \vee \ldots \vee A_1^{k_1}$. Apply Steps 1. and 2. of $EasyEntail$ until the *unique* branch becomes AB-completed. Let ϕ' be the resulting branch. We know that there is $B \in \Sigma$ in CNF and a conjunct B_j of B such that $B_j \subseteq A_1$. Now, $\mathsf{T}B_j, \mathsf{NT}A_1 \in S^{\phi'}$. Since $B_j \subseteq A_1$, B_j is equivalent to $\bigvee_{l \in I_1} A_1^l$, where $I_1 \subseteq \{1, \ldots, k_1\}$, *i.e.* all literals occurring in B_j occur in A_1. By applying repeatedly rule (A) to $\mathsf{NT}A_i = \mathsf{NT}A_i^1 \vee A_i^2 \vee \ldots \vee A_i^{k_i}$ and rule $(B1)$ (or $(B2)$) to $\mathsf{T}B_j$, a conjugated pair $\mathsf{NT}A_1^h$ and $\mathsf{T}A_1^h$ should be obtained. Therefore, we obtain a closed branch and, thus $EasyEntail$ returns true and an unique branch has been generated.

Case $n = 2$**:** Therefore, A is $A_1 \wedge A_2$, A_i is $A_i^1 \vee A_i^2 \vee \ldots \vee A_i^{k_i}$. Apply Steps 1. and 2. of *EasyEntail* until the *unique* branch becomes AB-completed. Let ϕ' be the resulting branch. We known that $\mathsf{NT}A_1 \wedge A_2 \in S^{\phi'}$. By Step 3. of *EasyEntail*, rule (PB) is applied to $\mathsf{NT}A$ generating two branches ϕ_1' and ϕ_2' such that $\mathsf{NT}A_1 \in S^{\phi_1'}$ and $\mathsf{T}A_1, \mathsf{NT}A_2 \in S^{\phi_2'}$. Consider ϕ_1'. By arguments as for case $n = 1$, by applying repeatedly rule (A) and rule $(B1)$ (or $(B2)$), the branch ϕ_1' will be extended to a closed branch. Similarly for branch ϕ_2'. Therefore, a closed deduction tree with two closed branches will be obtained. Hence, *EasyEntail* returns true.

The above arguments can be iterated for any $n > 0$: *EasyEntail* returns true and it builds a closed deduction tree with at most n branches.

2. We show that $EasyEntail(\Sigma, A)$ runs in time $O(|\Sigma||A|)$. We have already seen that $EasyEntail(\Sigma, A)$ builds a deduction tree with at most n closed branches $\phi_1, \ldots, \phi_n$, as *the* (PB) *rule is applied at most* n *times*. Moreover, we have seen that each branch ϕ_i is closed iff there is $B \in \Sigma$ and a conjunct B_j of B such that $B_j \subseteq A_i$. Therefore, the depth of ϕ_i is $O(|\Sigma||A_i|)$. Just note that if $B_j \subseteq A_i$, then $EasyEntail(B_j, A_i)$ requires $O(|B_j||A_i|)$ deterministic rule applications. As a consequence, $EasyEntail(\Sigma, A)$ runs in time

$$\begin{array}{l} O(\sum_{i=1}^{n} |\Sigma||A_i|) \\ = O(|\Sigma| \sum_{i=1}^{n} |A_i|) \\ = O(|\Sigma||A|) \end{array}$$

Q.E.D.

C.2 Deciding entailment in $\mathcal{L}_+$

A complete calculus with respect to $\mathcal{L}_+$ is obtained by extending the definition of signed propositions of type α and type β to the cases $\mathsf{NT}A \rightarrow B$ and $\mathsf{T}A \rightarrow B$, respectively, in a similar way as in [85]:

α	α_1	α_2
$\mathsf{NT}A \rightarrow B$	$\mathsf{T}A$	$\mathsf{NT}B$

β	β_1	β_2
$\mathsf{T}A \rightarrow B$	$\mathsf{NT}A$	$\mathsf{T}B$

Hence, the α and β tables for $\mathcal{L}_+$ are described in Table C.8.

α	α_1	α_2
$\mathsf{T}A \wedge B$	$\mathsf{T}A$	$\mathsf{T}B$
$\mathsf{NT}A \vee B$	$\mathsf{NT}A$	$\mathsf{NT}B$
$\mathsf{NT}A \rightarrow B$	$\mathsf{T}A$	$\mathsf{NT}B$

β	β_1	β_2
$\mathsf{T}A \vee B$	$\mathsf{T}A$	$\mathsf{T}B$
$\mathsf{NT}A \wedge B$	$\mathsf{NT}A$	$\mathsf{NT}B$
$\mathsf{T}A \rightarrow B$	$\mathsf{NT}A$	$\mathsf{T}B$

Table C.8: α and β tables for $\mathcal{L}_+$.

The following proposition is a straightforward extension of Proposition 18.

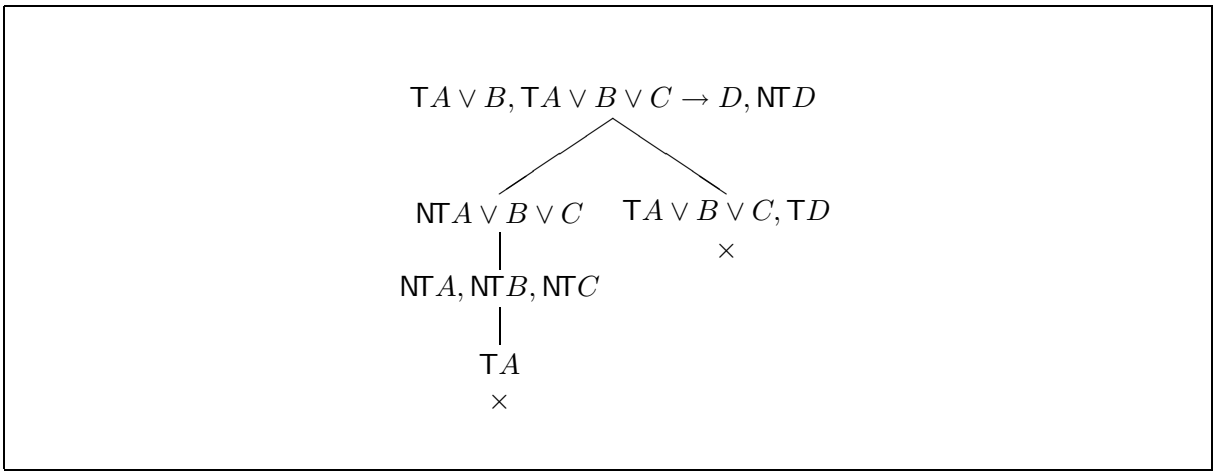


Figure C.5: Deduction tree for $A \vee B, A \vee B \vee C \rightarrow D \models_4 D$.

Proposition 22 *Let S be a set of signed propositions in $\mathcal{L}_+$. The following holds: $Sat(S)$ iff S is satisfiable.* ⊣

Figure C.5 shows a trivial proof of $A \vee B, A \vee B \vee C \rightarrow D \models_4 D$.

Certainly, Algorithm *Completions*(S) works in the case $\mathcal{L}^+$ too and, thus, Proposition 19 and Proposition 20 hold in $\mathcal{L}^+$.

Checking whether $\Sigma \models_4 A$ is a coNP-complete problem in $\mathcal{L}_+$. But, if we restrict $\mathcal{L}_+$ to $\overline{\mathcal{L}}_+$ we obtain a tractable logic. $\overline{\mathcal{L}}_+$ is defined inductively as follows: $\overline{\mathcal{L}}_+$ is the minimal set such that

1. every proposition in $\mathcal{L}$ in CNF is in $\overline{\mathcal{L}}_+$;
2. if $A, A_1, \ldots, A_n$ and $B, B_1, \ldots, B_m$ are literals in $\mathcal{L}$, then both $A_1 \wedge \ldots \wedge A_n \rightarrow B$ and $A \rightarrow B_1 \vee \ldots \vee B_m$ are in $\overline{\mathcal{L}}_+$.

Essentially, $\overline{\mathcal{L}}_+$ constrains the syntax of propositions involving the $\rightarrow$ connective. Just note that Horn propositions are in $\overline{\mathcal{L}}_+$. Moreover, we have the following equivalences:

$$((A_1 \wedge A_2) \rightarrow C) \wedge ((B_1 \wedge B_2) \rightarrow C) \equiv_4 ((A_1 \wedge A_2) \vee (B_1 \wedge B_2)) \rightarrow C \tag{C.15}$$

$$(A \rightarrow (B_1 \vee B_2)) \wedge (A \rightarrow (C_1 \vee C_2)) \equiv_4 A \rightarrow ((B_1 \vee B_2) \wedge (C_1 \vee C_2)) \tag{C.16}$$

From the above relations it follows immediately that $\overline{\mathcal{L}}_+$ allows the expression of implications which are equivalent to

$$A_{DNF} \rightarrow B_{CNF} \tag{C.17}$$

which are quite expressive. In particular, $A_{DNF} \rightarrow B_{CNF}$ can be seen as a macro of a formula $F \in \overline{\mathcal{L}}_+$. F has the property that $|A_{DNF} \rightarrow B_{CNF}| \leq |F|$.

We adapt algorithm *EasyEntail* to take as input an $\overline{\mathcal{L}}_+$ KB Σ and an $\overline{\mathcal{L}}_+$ proposition A in such a way that it remains complete and runs in polynomial time. We slightly have to modify the $(B1)$ and the $(B2)$ rules when applied to $A \rightarrow B \in \overline{\mathcal{L}}_+$, as shown in Table C.9.

$$(\overline{B1})\ \frac{\mathsf{T}A_1 \wedge \ldots \wedge A_n \rightarrow B, \mathsf{T}A_1, \ldots \mathsf{T}A_n}{\mathsf{T}B}$$
$$(\overline{B2})\ \frac{\mathsf{T}A \rightarrow B_1 \vee \ldots \vee B_m, \mathsf{NT}B_1, \ldots \mathsf{NT}B_m}{\mathsf{NT}A} \tag{C.18}$$

Table C.9: Modified $(B1)$ and $(B2)$ rules for $A \rightarrow B \in \overline{L}_+$.

It is straightforward to see that these rules preserve correctness and completeness of the *Sat* algorithm.

Algorithm $EasyEntail_+(\Sigma, A)$ is as $EasyEntail(\Sigma, A)$, except that rather applying $(B1)$ or $(B2)$ rules to $A \rightarrow B \in \overline{L}_+$, it applies $(\overline{B1})$ or $(\overline{B2})$.

Algorithm 6 ($EasyEntail_+(\Sigma, A)$)
$EasyEntail_+$ takes as input $\Sigma \subseteq \overline{L}_+$ and a proposition $A \in \overline{L}_+$. $EasyEntail_+(\Sigma, A) = true$ iff $\Sigma \models_4 A$. Let S be $\mathsf{T}\Sigma \cup \{\mathsf{NT}A\}$. $EasyEntail_+$ starts from the root labelled S and applies the rules until the resulting tree is either closed or completed. If the tree is closed, $EasyEntail_+$ returns true, otherwise false. At each step of the construction the following steps are performed:

1. select a branch ϕ which is not yet completed;
2. expand ϕ by means of the rules (A), $(B1)$ and $(B2)$(not applied to $\mathsf{T}A \rightarrow B$) and $(\overline{B1})$ and $(\overline{B2})$(for $\mathsf{T}A \rightarrow B$) until it becomes AB-completed;
3. if the resulting branch ϕ' is neither closed nor completed then
   (a) select a signed proposition of type β of the form $\mathsf{NT}B$ which is not yet fulfilled in the branch;
   (b) apply rule (PB) and go to Step 1.

   otherwise, go to Step 1. ■

Table C.10: Algorithm $EasyEntail_+(\Sigma, A)$ in $\mathcal{L}_+$.

Example 29 Let Σ be the KB

$$\Sigma = \{ \begin{array}{l} \mathsf{Gil} \rightarrow \mathsf{Adult}, \\ \mathsf{Adult} \rightarrow \mathsf{Tall}, \\ \mathsf{Karl} \rightarrow \mathsf{Child}, \\ \mathsf{Child} \rightarrow \mathsf{Tall} \\ \mathsf{Gil} \vee \mathsf{Karl} \ \} \end{array}$$

Certainly, $\Sigma \subset \overline{\mathcal{L}}_+$. Moreover, $\mathsf{Tall} \in \overline{\mathcal{L}}_+$.

It can easily be verified that,

$$\Sigma \models_4 \mathsf{Tall}.$$

Figure C.6 shows a closed deduction tree build by $EasyEntail_+(\Sigma, \mathsf{Tall})$

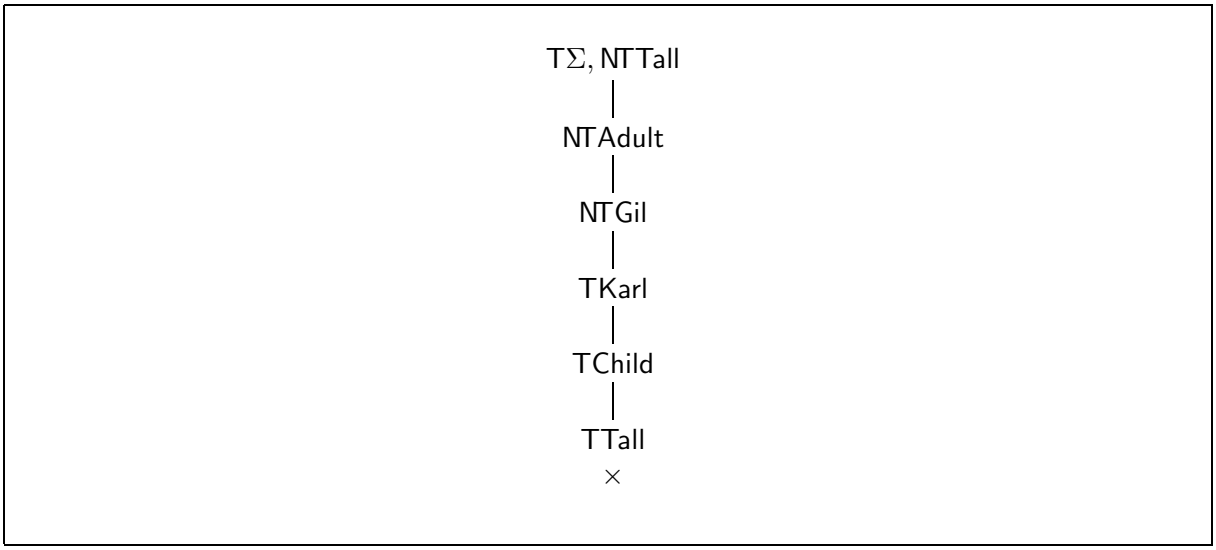


Figure C.6: A deduction tree for $\Sigma \models_4 \mathsf{Tall}$.

■

It follows that

Proposition 23 *Let Σ be a $\overline{\mathcal{L}}_+$ KB and let A be in $\overline{\mathcal{L}}_+$.*

1. *$EasyEntail_+(\Sigma, A) = true$ iff $\Sigma \models_4 A$.*
2. *$EasyEntail_+(\Sigma, A)$ runs in time $O(|\Sigma||A|)$.* ⊣

Proof: Correctness and completeness follows easily from Proposition 22 and Proposition 21, as $(\overline{B1})$ and $(\overline{B2})$ rules a both a specialisation of the $(B1)$ and $(B2)$ rules to the case $\mathsf{T}A_1 \wedge \ldots \wedge A_n \rightarrow B$ and $\mathsf{T}A \rightarrow B_1 \vee \ldots \vee B_m$, respectively. From a complexity point of view, note that no rule involving $A \rightarrow B$ generates a branch and the result of their application yield signed propositions of type α. Therefore, as for $EasyEntail(\Sigma, A)|$, the complexity of $EasyEntail_+(\Sigma, A)$ is $O(|\Sigma||A|)$. Q.E.D.

C.3 Deciding entailment in HORN-$\mathcal{L}$

The key point of this section is to show that it is possible to develop a decision procedure for determining $\Sigma \models_4 A_1 \wedge \ldots \wedge A_n$ which, instead to be based on our *Sat* procedure for $\mathcal{L}_+$, is a combination of traditional SLD-derivation in logic programming [194] and our decision

procedure for $\mathcal{L}$. The method is a simple adaption of the one described in *e.g.* [29, 72, 73, 97, 186, 187] and makes use of Proposition 19 and Proposition 20.

For the sake of readability, we briefly describe the notions of SLD-derivation and SLD-refutation (see *e.g.* [194]).

Let G be a goal of the form $\leftarrow A_1, \ldots, A_n$. Let E be a horn rule R of the form $A \leftarrow B_1, \ldots, B_k$ or a horn fact A. Let A_i be a letter in G. If $A_i = A$ then

1. the *resolvent* of the goal G and the horn rule R is the goal

$$\leftarrow A_1, \ldots, A_{i-1}, B_1, \ldots, B_k, A_{i+1}, \ldots, A_n;$$

2. the *resolvent* of the goal G and the horn fact A is the goal

$$\leftarrow A_1, \ldots, A_{i-1}, A_{i+1}, \ldots, A_n.$$

A *SLD-derivation* for a goal G_0 in a HORN-$\mathcal{L}$ KB Σ is a derivation constituted by:

1. a sequence of horn rules and horn facts $E_1, \ldots, E_n$ in Σ;
2. a sequence of goals $G_0, \ldots, G_n$ such that for each $i \in \{0, \ldots, n-1\}$, G_{i+1} is the resolvent of G_i and E_{i+1}.

A SLD-derivation may terminate with an empty goal in which case the derivation is a *SLD-refutation*. It is well known that given a horn KB Σ and a query $A_1 \wedge \ldots \wedge A_n$, then $\Sigma \models_2 A_1 \wedge \ldots \wedge A_n$ iff there is a SLD-refutation for goal $\leftarrow A_1, \ldots, A_n$ in Σ. From Proposition 4 it follows immediately that

Proposition 24 *In* HORN-$\mathcal{L}$*, let Σ be a horn KB and let Q be a query. Then $\Sigma \models_4 Q$ iff there is a SLD-refutation for goal G_Q in Σ.* ⊣

What about the case in which Σ is a HORN-$\mathcal{L}$ KB? Of course, by (7.47) it follows that if there is a SLD-refutation for goal $\leftarrow A_1, \ldots, A_n$ in Σ then $\Sigma \models_4 A_1 \wedge \ldots \wedge A_n$. Unfortunately, the converse is not true as shown in the following example.

Example 30 Consider the following safe and non-recursive HORN-$\mathcal{L}$ KB:

$$\Sigma \;=\; \{C \leftarrow A, C \leftarrow B, A \vee B\}$$

and the query C. It is quite easy to see that $\Sigma \models_4 C$. But, there is no SLD refutation for goal $\leftarrow C$ in Σ. The only two SLD derivations end with goal G_1 and G_2, which are $\leftarrow A$ and $\leftarrow B$, respectively. ■

Concerning Example 30, what is actually inferable from the derivation ending with the goal G_1 is that C is satisfied in all those models of Σ satisfying A. Similarly, for goal G_2: C is satisfied in all those models of Σ satisfying B. Therefore, in order to prove the query C, each model of Σ should satisfy either A or B. By relying on Proposition 19 it follows that each model of Σ satisfies either A or B iff both canonical models $\mathcal{I}_1, \mathcal{I}_2$ of completions $\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2 \in \mathit{Completions}(\mathsf{T}\Sigma)$, where $A^{\mathcal{I}_1} = \{t\}, B^{\mathcal{I}_1} = \emptyset, C^{\mathcal{I}_1} = \{t\}$ and $B^{\mathcal{I}_2} = \{t\}, A^{\mathcal{I}_2} = \emptyset, C^{\mathcal{I}_1} = \{t\}$, are such that they satisfy either A or B. Finally, from the safeness of Σ neither A nor B

appears in any head of rule in Σ. Therefore, we can restrict our attention to the canonical models of completions in $Completions(\mathsf{T}\Sigma_F)$ rather than those of Σ: *i.e.* given $\Sigma_F = \{A \vee B\}$, $\Sigma \models_4 C$ iff both canonical models $\mathcal{I}_3, \mathcal{I}_4$ of completions $\tilde{\mathcal{S}}_3, \tilde{\mathcal{S}}_4 \in Completions(\mathsf{T}\Sigma_F)$, where $A^{\mathcal{I}_3} = \{t\}, B^{\mathcal{I}_3} = \emptyset$ and $B^{\mathcal{I}_4} = \{t\}, A^{\mathcal{I}_4} = \emptyset$, are such that they satisfy either A or B.

It is worth noting that this last step is not allowed in case of non safe KBs. For example, consider the non safe KB $\Sigma = \{A \vee B, A \leftarrow B\}$. Let Σ_F be $\{A \vee B\}$. It is easily verified that $\Sigma \models_4 A$. Consider the last goal of a derivation of $\leftarrow A$ in Σ, *i.e.* $\leftarrow B$. The canonical models of completions in $Completions(\mathsf{T}\Sigma_F)$, are $\mathcal{I}_3, \mathcal{I}_4$ above. But, $\mathcal{I}_3$ does not satisfy B.

The above arguments can be generalized, yielding the following proposition. We will assume that each query Q_i is of the form $A_{i_1} \wedge \ldots \wedge A_{i_{k_i}}$ and each associated goal G_{Q_i} is of the form $\leftarrow A_{i_1}, \ldots, A_{i_{k_i}}$.

Proposition 25 *Let Σ be a safe and non recursive* HORN-$\mathcal{L}$ *KB and let Q be a query. Then $\Sigma \models_4 Q$ iff either*

1. *there is a SLD-refutation for goal G_Q in Σ; or*

2. *there are $n \geq 1$ SLD-derivations for goal G_Q in Σ ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, such that for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in Completions($ $\mathsf{T}\Sigma_F)$, $\mathcal{I}$ does satisfy $Q_1 \vee \ldots \vee Q_n$,* i.e. *there is some $1 \leq i \leq n$ such that $\mathcal{I}$ does satisfy Q_i.* ⊣

Similarly for the two-valued case. Note that if $\mathcal{I}$ has to satisfy $A_{i_1} \wedge \ldots \wedge A_{i_{k_i}}$ (Point 2. in Proposition 25), where each $A_{i_{k_h}}$ is a propositional letter, then we can restrict our attention to those letters p such that $t \in p^{\mathcal{I}}$: *i.e.* let $\mathcal{I}_1$ be an interpretation such that for all letters p, $p^{\mathcal{I}_1} = \{t\}$ if $t \in p^{\mathcal{I}}$ and $p^{\mathcal{I}_1} = \emptyset$ otherwise, then $\mathcal{I}$ satisfies $A_{i_1} \wedge \ldots \wedge A_{i_{k_i}}$ iff $\mathcal{I}_1$ satisfies $A_{i_1} \wedge \ldots \wedge A_{i_{k_i}}$. This means that we can restrict our attention to $\tilde{\mathcal{S}}^+$ or, equivalently, to $\Sigma^+_{\tilde{\mathcal{S}}}$.

From the above observation and from Proposition 20 it follows immediately that

Proposition 26 *Let Σ be a safe and non recursive* HORN-$\mathcal{L}$ *KB and let Q be a query. Then $\Sigma \models_4 Q$ iff either*

1. *there is a SLD-refutation for goal G_Q in Σ; or*

2. *there are $n \geq 1$ SLD-derivations for goal G_Q in Σ ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, such that for all four-valued completions $\tilde{\mathcal{S}} \in Completion(\mathsf{T}\Sigma_F)$, $\tilde{\mathcal{S}}^+ \cup \{\mathsf{NT}Q_1 \vee \ldots \vee Q_n\}$ is not satisfiable, or equivalently, $\Sigma^+_{\tilde{\mathcal{S}}} \models_4 Q_1 \vee \ldots \vee Q_n$.* ⊣

Similarly for the two-valued case.

Proposition 25 and Proposition 26 give us several solutions in order to determine whether $\Sigma \models_4 Q$, where Σ is a safe and non recursive HORN-$\mathcal{L}$ KB.

Method 1: Collect $n \geq 1$ SLD-derivations of G_Q in Σ, ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, until (*i*) there is an empty goal; or, (*ii*) $\Sigma_F \models_4 Q_1 \vee \ldots \vee Q_n$, by relying on procedure *Sat* for $\mathcal{L}$. Note that $\Sigma_F \subseteq \mathcal{L}$ and $Q_i \in \mathcal{L}$. In the worst case we have to compute all SLD-derivations.

Method 2: Collect $n \geq 1$ SLD-derivations of G_Q in Σ, ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, until (i) there is an empty goal; or, (ii) for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in \mathit{Completions}(\mathsf{T}\Sigma_F)$, $\mathcal{I}$ does satisfy $Q_1 \vee \ldots \vee Q_n$. In the worst case we have to compute all SLD-derivations.

Method 3: Collect $n \geq 1$ SLD-derivations of G_Q in Σ, ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, until (i) there is an empty goal; or, (ii) for all four-valued completions $\tilde{\mathcal{S}} \in \mathit{Completions}(\mathsf{T}\Sigma_F)$, $\tilde{\mathcal{S}}^+ \cup \{\mathsf{NT}Q_1 \vee \ldots \vee Q_n\}$ is not satisfiable, or equivalently, $\Sigma^+_{\tilde{\mathcal{S}}} \models_4 Q_1 \vee \ldots \vee Q_n$. In the worst case we have to compute all SLD-derivations.

Method 4: Compute $\mathit{Completions}(\mathsf{T}\Sigma_F)$. Determine whether for all $\tilde{\mathcal{S}} \in \mathit{Completions}(\mathsf{T}\Sigma_F)$, G_Q has a SLD-refutation in $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$, *i.e.* whether $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R \models_4 A_1 \wedge \ldots \wedge A_n$. Note that $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$ is a horn KB.

We will not discuss which of the above methods is the better one, nor we will explore other methods. Just let us mention that certainly Method 4 is the simplest to implement. In fact, since $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$ is a horn KB, from Proposition 24 it follows that what we need is an implementation of procedure *Completions*. Then we can determine whether $\Sigma \models_4 Q$, where Σ is a safe and non recursive HORN-$\mathcal{L}$ KB, by relying on a simple prolog system which checks whether $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R \models_4 Q$, for all four-valued completions $\tilde{\mathcal{S}} \in \mathit{Completions}(\mathsf{T}\Sigma_F)$. We conclude this section with the following example.

Example 31 Consider Example 30, *i.e.*

$$\Sigma = \{C \leftarrow A, C \leftarrow B, A \vee B\}$$

Σ_F is $\{A \vee B\}$ and Σ_R is $\{C \leftarrow A, C \leftarrow B\}$. We check that $\Sigma \models_4 C$ holds, by applying Method 4. $\mathit{Completions}(\mathsf{T}\Sigma_F)$ contains two completions such that

$$\begin{array}{rcl} \tilde{\mathcal{S}}_1 & = & \{\mathsf{T}A\} \\ \tilde{\mathcal{S}}_2 & = & \{\mathsf{NT}A, \mathsf{T}B\}, \end{array}$$

and, thus,

$$\begin{array}{rcl} \Sigma_{\tilde{\mathcal{S}}_1} & = & \{A\} \\ \Sigma_{\tilde{\mathcal{S}}_2} & = & \{B\}. \end{array}$$

Therefore, $\Sigma^+_{\tilde{\mathcal{S}}_1} = \Sigma_{\tilde{\mathcal{S}}_1}$ and $\Sigma^+_{\tilde{\mathcal{S}}_2} = \Sigma_{\tilde{\mathcal{S}}_2}$. Let $\Sigma_1 = \Sigma^+_{\tilde{\mathcal{S}}_1} \cup \Sigma_R$ and $\Sigma_2 = \Sigma^+_{\tilde{\mathcal{S}}_2} \cup \Sigma_R$, *i.e.*

$$\begin{array}{rcl} \Sigma_1 & = & \{C \leftarrow A, C \leftarrow B, A\} \\ \Sigma_2 & = & \{C \leftarrow A, C \leftarrow B, B\}. \end{array}$$

It is easily verified that there is a SLD-refutation of $\leftarrow C$ in Σ_i, for $i = 1, 2$. Therefore, $\Sigma \models_4 C$ hold, as expected. ■

Just note that the computational complexity of *e.g.* Method 4 depends on the number of completions computed as on their dimension. In fact, the complexity is bounded by

$$\sum_i |\Sigma^+_{\tilde{\mathcal{S}}_i} \cup \Sigma_R||Q|, \tag{C.19}$$

where $\tilde{\mathcal{S}}_i$ is a completion of $\mathsf{T}\Sigma_F$, *i.e.* $\tilde{\mathcal{S}}_i \in \mathit{Completions}(\mathsf{T}\Sigma_F)$. Of course, in case of horn KBs, $\mathit{Completions}(\mathsf{T}\Sigma_F) = \{\mathsf{T}\Sigma_F\}$, and thus, Method 4, runs in polynomial time.

C.4 Deciding entailment in $\mathcal{ALC}$

With respect to classical two-valued semantics, DLs has been largely investigated from a computability point of view. There exists both a well known algorithm based on constraint propagation [249] (which is essentially an analytic tableaux based decision procedure[1]) for deciding KB satisfiability and a lot of results from a computational complexity point of view (see *e.g.* [69, 71, 94, 95, 99, 150, 206, 208, 247])[2].

As we have seen, all decision problems, usually considered important in classical DLs, can be reduced to the KB satisfiability test (see Equations (6.21) - (6.30). If we switch to a four-valued setting, similarly to what happens for the propositional case $\mathcal{L}$ and $\mathcal{L}^+$, we need an alternative proof procedure. There exists two proof methods for four-valued DLs:

1. there is a well known subsumption testing procedure, which is an DL adaption of Levesque's algorithm for entailment (see Algorithm 2) [54, 57, 217, 220, 221, 222]. The algorithm performs in a efficient way structural subsumption [54]. A drawback of this method is that: (*i*) it works only within type B semantics; (*ii*) it can not be used for the instance checking test as this latter problem is in a higher complexity class than the subsumption problem, and (*iii*) it is rather difficult to adapt the algorithm in the case we change the DL[3].

2. there is a sequent calculus based proof procedure for instance checking [266]. This method has the advantage to solve the subsumption problem too. It works within type A semantics and within type B semantics. In this latter case it can be modified in such a way that it runs the subsumption test in the same time order as for the above mentioned structural subsumption algorithm. The method is easily adaptable to the different DLs described in the literature. But, the main drawback of the sequent calculus based method is that is suffers of the same problems of the analytic tableaux based methods.

In order to overcome to the above listed problems, we will present here a semantic tableaux for instance checking in four-valued DLs. This gives us immediately a decision procedure for the subsumption problem. Thereafter we will extend it to the two-valued case.

The calculus is essentially an adaption of the calculus for deciding entailment in $\mathcal{L}^+$ to the DL case. We will give decision procedures for both type A and type B semantics. In the following, if not stated otherwise, we will assume type A semantics.

The major difficulties arises by the treatment of the $\forall$ and $\exists$ connectives. We take inspiration on the work [132, 133]. In [133] it has been experimentally shown that analytic tableaux methods for two-valued DLs are quite inefficient with respect to semantic based methods. Moreover, they have pointed out that the following holds:

[1] Hence, inherits the same problem of it.

[2] It is not our intention to give an exhaustive list of all results. The interested reader can consult the DL WWW home page at http://www.dl.kr.org/dl.

[3] This is the price we must pay if we want fast special purpose algorithms.

$$\begin{array}{c}(\exists R.C) \sqcap (\forall R.D_1) \ldots \sqcap (\forall R.D_m)\\ \text{is coerent iff}\\ C \sqcap D_1 \ldots \sqcap D_m\\ \text{is coerent}\end{array} \tag{C.20}$$

In general,

$$\begin{array}{l}(\exists R_1.C_1) \sqcap \sqcap_{j=1}^{m_1}(\forall R_1.D_{1_j}) \sqcap\\ (\exists R_2.C_2) \sqcap \sqcap_{j=1}^{m_2}(\forall R_2.D_{2_j}) \sqcap\\ \ldots \sqcap\\ (\exists R_k.C_k) \sqcap \sqcap_{j=1}^{m_k}(\forall R_k.D_{k_j})\\ \\ \text{is coerent iff for each } 1 \leq i \leq k\\ \\ C_i \sqcap \sqcap_{j=1}^{m_i} D_{i_j}\\ \\ \text{is coerent}\end{array} \tag{C.21}$$

with $k \geq 1$, and $m_i \geq 0$ (for $1 \leq i \leq k$). For instance, in order to determine the coherence of

$$\exists R_1.C_1 \sqcap \forall R_1.D_{1_1} \sqcap \forall R_1.D_{1_2} \sqcap \exists R_2.C_2 \tag{C.22}$$

it is sufficient to check out the coherence of both $C_1 \sqcap D_{1_1} \sqcap D_{1_2}$ and C_2. We will adapt this property to our decision procedure.

The above property follows from the well-known relationship between $\mathcal{ALC}$ and modal logic $\mathbf{K}_m$, *i.e.* $\mathbf{K}$ with m modal relations (see *e.g.* [155]). In fact, according to [248], consider the following function K mapping $\mathcal{ALC}$ concepts into modal formulae:

$$\begin{array}{lcl}K(A) & = & A\\ K(\neg A) & = & \neg A\\ K(C \sqcap D) & = & K(C) \wedge K(D)\\ K(C \sqcup D) & = & K(C) \vee K(D)\\ K(\forall R.C) & = & \mathsf{L}_R K(C)\\ K(\exists R.C) & = & \mathsf{M}_R K(C).\end{array} \tag{C.23}$$

where, as usual, L is the necessity modal operator and M is the possibility modal operator. In [248] it is shown that a concept C is satisfiable iff $K(C)$ is, *i.e.* $K(C)$ ha a Kripke model. Just notice that the well known equivalence between M and $\neg\mathsf{L}\neg$ confirms the equivalence between $\exists$ and $\neg\forall\neg$. Now, the relation (C.21) is a direct consequence of the fact that $(\mathsf{M}_R A) \wedge (\mathsf{L}_R B)$ is satisfiable iff $A \wedge B$ is satisfiable.

Using the above property, our semantic tableaux is defined as follows. Consider a new alphabet $\mathcal{V}_{\mathtt{ALC}}$ of *variables* (denoted by x, y). An *object* (denoted by v, w) is either an individual or a variable. Interpretations are extended to a variable x by mapping it into an element of $\mathcal{I}$'s domain, $x^{\mathcal{I}} \in \Delta^{\mathcal{I}}$.

Signed formulae (denoted by σ) are expressions of type $\mathsf{T}A$, $\mathsf{NT}A$, $\mathsf{T}C \Rightarrow D$ or of type $\mathsf{NT}C \Rightarrow D$, where A is an assertion, in which variables can occur, and $C \Rightarrow D$ is a specialization. For example, $\mathsf{T}(A \sqcap B)(a), \mathsf{NT}(A \sqcup B)(x)$ are signed formulae.

We extend the definition of satisfiability to signed formulae as follows. An interpretation $\mathcal{I}$ *satisfies* $\mathsf{T}A$ iff $\mathcal{I}$ satisfies A; $\mathcal{I}$ *satisfies* $\mathsf{NT}A$ iff $\mathcal{I}$ does not satisfy A; $\mathcal{I}$ *satisfies* $\mathsf{T}C \Rightarrow D$ iff $\mathcal{I}$ satisfies $C \Rightarrow D$, and $\mathcal{I}$ *satisfies* $\mathsf{NT}C \Rightarrow D$ iff $\mathcal{I}$ does not satisfy $C \Rightarrow D$.

An interpretation $\mathcal{I}$ *satisfies* set of signed formulae iff $\mathcal{I}$ satisfies each element of it. As a consequence, as for the propositional case,

$$\Sigma \models_4^A A \text{ iff } \mathsf{T}\Sigma \cup \{\mathsf{NT}A\} \text{ is not satisfiable} \tag{C.24}$$

where $\mathsf{T}\Sigma = \{\mathsf{T}A : A \in \Sigma\}$ and

$$\Sigma \models_4^A C \Rightarrow D \text{ iff } \mathsf{T}\Sigma \cup \{\mathsf{NT}C \Rightarrow D\} \text{ is not satisfiable.} \tag{C.25}$$

Signed formulae of type α (of conjunctive type) and β (of disjunctive type) and on their *components* are defined as follows:

1. With respect to the connectives $\sqcap$ and $\sqcup$ we have (recall Table C.2)

α	α_1	α_2
$\mathsf{T}(C \sqcap D)(w)$	$\mathsf{T}C(w)$	$\mathsf{T}D(w)$
$\mathsf{NT}(C \sqcup D)(w)$	$\mathsf{NT}C(w)$	$\mathsf{NT}D(w)$

β	β_1	β_2
$\mathsf{T}(C \sqcup D)(w)$	$\mathsf{T}C(w)$	$\mathsf{T}D(w)$
$\mathsf{NT}(C \sqcap D)(w)$	$\mathsf{NT}C(w)$	$\mathsf{NT}D(w)$

These definitions reflects entirely the definitions about signed proposition seen in Table C.2.

2. With respect to specializations we have:

α	α_1	α_2	Condition
$\mathsf{NT}C \Rightarrow D$	$\mathsf{T}C(a)$	$\mathsf{NT}D(a)$	for a new individual a

Signed formulae of this type are indicated also with $\alpha^{\Rightarrow}$ and their components with $\alpha_1^{\Rightarrow}$ and $\alpha_2^{\Rightarrow}$.

β	β_1	β_2	Condition
$\mathsf{T}C \Rightarrow D$	$\mathsf{NT}C(w)$	$\mathsf{T}D(w)$	for all objects w

Signed formulae of this type are indicated also with $\beta^{\Rightarrow}$ and their components with $\beta_1^{\Rightarrow}$ and $\beta_2^{\Rightarrow}$. We will say that the specialisation $\beta^{\Rightarrow}$ has been *instantiated* with w, if w is the object involved in $\beta^{\Rightarrow}$'s components. Just notice here that a specialization $C \Rightarrow D$ is viewed as $\forall x.C(x) \rightarrow D(x)$. Hence, the table above is nothing else as a generalization of the table for the $\rightarrow$ connective seen for $\mathcal{L}+$. For instance, given $\mathsf{NT}C \Rightarrow D$, the application of a (A) rule will be of type "let a be a new individual not appearing in any branch, then split $\mathsf{NT}C \Rightarrow D$ into $\mathsf{T}C(a)$ and $\mathsf{NT}D(a)$. It is worth noting that now, from $(C.24)$ it follows that

$$\Sigma \models_4^A C \Rightarrow D \text{ iff} \tag{C.26}$$
$$\mathsf{T}\Sigma \cup \{\mathsf{NT}C \Rightarrow D\} \text{ is not satisfiable, iff} \tag{C.27}$$
$$\mathsf{T}\Sigma \cup \{\mathsf{T}C(a), \mathsf{NT}D(a)\} \text{ is not satisfiable, iff} \tag{C.28}$$
$$\Sigma \cup \{C(a)\} \models_4^A D(a). \tag{C.29}$$

which are perfectly compatible with the relations (6.25), (6.26) (7.67), (7.68) and are similar to (6.27) and (6.28).

3. With respect to the $\forall$ and $\exists$ connectives we have:

α	α_1	α_2	Condition
$\mathsf{NT}(\forall R.C)(w)$	$\mathsf{T}R(w,x)$	$\mathsf{NT}C(x)$	for a new variable x
$\mathsf{T}(\exists R.C)(w)$	$\mathsf{T}R(w,x)$	$\mathsf{T}C(x)$	for a new variable x

Signed formulae of this type are indicated also with $\alpha^{\exists}$ and their components with $\alpha_1^{\exists}$ and $\alpha_2^{\exists}$, respectively. Let $Ind(\alpha^{\exists}) = w$, $Ind(\alpha_i^{\exists}) = x$ and $Role(\alpha^{\exists}) = R$.

β	β_1	β_2	Condition
$\mathsf{T}(\forall R.C)(w)$	$\mathsf{NT}R(w,v)$	$\mathsf{T}C(v)$	for all objects v
$\mathsf{NT}(\exists R.C)(w)$	$\mathsf{NT}R(w,v)$	$\mathsf{NT}C(v)$	for all objects v

Signed formulae of this type are indicated also with $\beta^{\forall}$ and their components with $\beta_1^{\forall}$ and $\beta_2^{\forall}$, respectively. Let $Ind(\beta^{\forall}) = w$, $Ind(\beta_i^{\forall}) = v$ and $Role(\beta^{\forall}) = R$.

As already said, the above table is an immediate consequence of the fact that *e.g.* $(\forall R.C)(w)$ is viewed as $\forall x.R(w,x) \rightarrow C(x)$.

Moreover, we will say that the above signed formulae σ of type $\alpha^{\exists}$ ($\beta^{\forall}$) has been *instantiated* with x (v), if x is the new variable (object v is) involved in σ's components and given by the condition. For instance, if $\beta^{\forall} = \mathsf{T}(\forall R.C)(a)$ has been instantiated with c then $\beta_1^{\forall}$ is $\mathsf{NT}R(a,c)$, whereas $\beta_2^{\forall}$ is $\mathsf{T}C(c)$.

In one shot, the α and β tables are shown in Table C.11 below.

α	α_1	α_2	Condition
$\mathsf{T}(C \sqcap D)(w)$	$\mathsf{T}C(w)$	$\mathsf{T}D(w)$	
$\mathsf{NT}(C \sqcup D)(w)$	$\mathsf{NT}C(w)$	$\mathsf{NT}D(w)$	
$\mathsf{NT}C \Rightarrow D$	$\mathsf{T}C(a)$	$\mathsf{NT}D(a)$	for a new individual a
$\mathsf{NT}(\forall R.C)(w)$	$\mathsf{T}R(w,x)$	$\mathsf{NT}C(x)$	for a new variable x
$\mathsf{T}(\exists R.C)(w)$	$\mathsf{T}R(w,x)$	$\mathsf{T}C(x)$	for a new variable x

β	β_1	β_2	Condition
$\mathsf{T}(C \sqcup D)(w)$	$\mathsf{T}C(w)$	$\mathsf{T}D(w)$	
$\mathsf{NT}(C \sqcap D)(w)$	$\mathsf{NT}C(w)$	$\mathsf{NT}D(w)$	
$\mathsf{T}C \Rightarrow D$	$\mathsf{NT}C(w)$	$\mathsf{T}D(w)$	for all objects w
$\mathsf{T}(\forall R.C)(w)$	$\mathsf{NT}R(w,v)$	$\mathsf{T}C(v)$	for all objects v
$\mathsf{NT}(\exists R.C)(w)$	$\mathsf{NT}R(w,v)$	$\mathsf{NT}C(v)$	for all objects v

Table C.11: α and β table for $\mathcal{ALC}$.

As usual, $\mathsf{T}A$ and $\mathsf{NT}A$ are called *conjugated signed formulae*. With β_i^c we indicate the *conjugate* of β_i.

The calculus is based on similar rules as for the propositional case (see Table C.12 below). Just notice that rule (A) is *not applicable* for signed formulae of type $\alpha^{\exists}$. The reason relies on the fact that we will treat $\alpha^{\exists}$ expressions accordingly to $(C.21)$. Similarly, rules $(B2)$ and (PB) are not applicable to signed formulae of type $\beta^{\forall}$. Finally, in applying the rule (PB) to a signed formula of the form $\mathsf{T}C \Rightarrow D$, we are not allowed to instantiate it with a new object.

$$(A)\ \frac{\alpha}{\alpha_1, \alpha_2} \qquad \text{if } \alpha \text{ is not of type } \alpha^{\exists}$$

$$(B1)\ \frac{\beta, \beta_1^c}{\beta_2}$$

$$(B2)\ \frac{\beta, \beta_2^c}{\beta_1} \qquad \text{if } \beta \text{ is not of type } \beta^{\forall}$$

$$(PB)\ \frac{\beta}{\beta_1 \mid \beta_1^c, \beta_2} \qquad \text{if } \beta \text{ is not of type } \beta^{\forall}$$

Table C.12: Semantic tableaux inference rules in $\mathcal{ALC}$.

In order to obtain a complete calculus, it is necessary to handle the $\alpha^{\exists}$ expressions too, as we will see in the procedure $Sat_D L$ below.

The definitions of (in the context of DLs) *deduction tree*, *closed branch* and *refutation* are the natural adaption of the $\mathcal{L}$ case to the DL case.

We will say that a signed formula is *AB-analysed* in a branch ϕ if either (i) it is of type α (and not of type $\alpha^{\exists}$) and both α_1 and α_2 occur in ϕ; or (ii) it is of type β and (iia) if β_1^c occurs in ϕ then β_2 occurs in ϕ, (iib) if β_2^c occurs in ϕ then β_1 occurs in ϕ (case β is not of type $\beta^{\forall}$). A branch is *AB-completed* if all the propositions in it are AB-analysed. A signed formula of type β, not being of type $\beta^{\forall}$, is *fulfilled* in a branch ϕ if either β_1 or β_2 occurs in ϕ.

Let S be a set of signed formulae. With $S(\Rightarrow)$ we indicate the set

$$S(\Rightarrow) \quad = \quad \{\mathsf{T}C \Rightarrow D \in S\}. \tag{C.30}$$

Moreover, if there is an $\alpha^{\exists} \in S$ then let $S(\alpha^{\exists})$ be such that

$$\begin{aligned} S(\mathsf{T}(\exists R.C)(w)) \quad = \quad & \{\mathsf{T}C(x)\} \cup \\ & \{\mathsf{T}D(x) : \mathsf{T}(\forall R.D)(w) \in S\} \cup \\ & \{\mathsf{NT}D(x) : \mathsf{NT}(\exists R.D)(w) \in S\} \end{aligned} \tag{C.31}$$

$$\begin{aligned} S(\mathsf{NT}(\forall R.C)(w)) \quad = \quad & \{\mathsf{NT}C(x)\} \cup \\ & \{\mathsf{T}D(x) : \mathsf{T}(\forall R.D)(w) \in S\} \cup \\ & \{\mathsf{NT}D(x) : \mathsf{NT}(\exists R.D)(w) \in S\} \end{aligned} \tag{C.32}$$

where x is a new variable. The above two equations can be rewritten in the following compact way:

$$
\begin{array}{rl}
S(\alpha^{\exists}) = & \{\alpha_2^{\exists}\} \cup \\
& \{\beta_2^{\forall} : \ \beta^{\forall} \in S, \\
& \qquad Ind(\alpha^{\exists}) = Ind(\beta^{\forall}), \\
& \qquad Role(\alpha^{\exists}) = Role(\beta^{\forall})\}
\end{array}
\tag{C.33}
$$

where both $\alpha^{\exists}$ and $\beta^{\forall}$ have been instantiated with the same new variable x ($Ind(\alpha_2^{\exists}) = Ind(\beta_2^{\forall} = x)$. For instance, if S is

$$\{\mathsf{T}(\exists R_1.C_1)(a), \mathsf{T}(\forall R_1.D_{1_1})(a), \mathsf{T}(\forall R_1.D_{1_2})(a), \mathsf{T}(\exists R_2.C_2)(b)\}$$

then $S(\mathsf{T}(\exists R_1.C_1)(a))$ is $\{\mathsf{T}C_1(x), \mathsf{T}D_{1_1}(x), \mathsf{T}D_{1_2}(x)\}$ whereas $S(\mathsf{T}(\exists R_2.C_2)(a))$ is $\{\mathsf{T}C_2(y)\}$, reflecting the case ($C.22$).

Suppose that S is a set of signed formulae such that $S(\Rightarrow) = \emptyset$, *i.e.* no specializations occur in S. The procedure $Sat_{DL}(S)$ in Table C.13 determines whether a set of signed formulae S is satisfiable or not, whenever $Sat_{DL}(S)$ relies on the inference rules describe in Table C.12. The procedure is a modification of the Sat procedure for the propositional case. Essentially, in order to determine whether S is satisfiable or not, $Sat_{DL}(S)$ proceeds as follows: the procedure starts with root labelled S. Each not closed branch, not being AB-completed, will be expanded until it becomes AB-completed or closed. If it becomes closed then $Sat_{DL}(S)$ fails. Otherwise, if a signed formula of type $\alpha^{\exists}$ occurs in a branch ϕ, *i.e.* $\alpha^{\exists} \in S^{\phi}$, and $S^{\phi}(\alpha^{\exists})$ has not yet been considered with respect to ϕ, *i.e.* $Sat_{DL}(S^{\phi}(\alpha^{\exists}))$ has not yet been tested, then we test $Sat_{DL}(S^{\phi}(\alpha^{\exists}))$. If for some of these $\alpha^{\exists}$ the test fails then, accordingly to ($C.21$), $Sat_{DL}(S)$ fails. Otherwise, we proceed (similarly to the propositional case) by applying the principle of bivalence to a not fulfilled β (not being of type $\beta^{\forall}$).

The case in which $S(\Rightarrow) \neq \emptyset$ is more complicated. The problem arises from the fact that there could be infinite applications of rules, *e.g.* rule ($B1$), and thus, termination of the $Sat_{DL}(S)$ procedure is not guaranteed. In fact, consider the following example.

Example 32 Consider the following set of signed formulae,

$$S = \{\mathsf{T}A(a), \mathsf{T}A \Rightarrow \exists R.A\}.$$

S is obviously satisfiable: for instance, the interpretation $\mathcal{I}$ such that $t \in A^{\mathcal{I}}(a^{\mathcal{I}})$, $t \in A^{\mathcal{I}}(d)$, $t \in R^{\mathcal{I}}(a^{\mathcal{I}}, d)$, $t \in R^{\mathcal{I}}(d, d)$ satisfies S.

By running $Sat_{DL}(S)$, we can apply rule ($B1$) to $\mathsf{T}A(a)$ and $\mathsf{T}A \Rightarrow \exists R.A$, yielding $\mathsf{T}(\exists R.A)(a)$. Now, we have to test whether $S_1 = S(\alpha^{\exists}) \cup S(\Rightarrow)$ is satisfiable, *i.e.* whether $S_1 = \{\mathsf{T}A(x_1), \mathsf{T}A \Rightarrow \exists R.A\}$, for a new x_1, is satisfiable.

Again, we can apply rule ($B1$) to $\mathsf{T}A(x_1)$ and $\mathsf{T}A \Rightarrow \exists R.A$, yielding a new set $S_2 := S_1 \cup \{\mathsf{T}(\exists R.A)(x_1)\}$, and so on. For each $i = 1, 2, \ldots$, we have to check whether $S_i = \{\mathsf{T}A(x_i), \mathsf{T}(\exists R.A)(x_i), \mathsf{T}A \Rightarrow \exists R.A\}$, for a new variable x_i, is satisfiable. Hence, an infinite loop of $Sat_{DL}(S)$ may arise, unless a termination condition is specified. ■

The solution we will adopt to this problem is based on the technique described in [68], which guarantees termination.

We assume that there is a well-founded total ordering $\prec_{\mathcal{V}_{\mathtt{ALC}}}$ on $\mathcal{V}_{\mathtt{ALC}}$. We assume that variables are introduced, through a signed formula of type $\alpha^{\exists}$, according to the ordering $\prec_{\mathcal{V}_{\mathtt{ALC}}}$: *i.e.* if y has been introduced during a proof then $x \prec_{\mathcal{V}_{\mathtt{ALC}}} y$ for all variables x that

already has been introduced. For instance, in Example 32 we assume that $x_1 \prec_{\mathcal{V}_{\mathtt{ALC}}} x_2 \prec_{\mathcal{V}_{\mathtt{ALC}}} \ldots \prec_{\mathcal{V}_{\mathtt{ALC}}} x_i \prec_{\mathcal{V}_{\mathtt{ALC}}} \ldots$.

Given a set of signed formulae S and a variable x, we define the functions

$$\sigma_{\mathsf{T}}(S, x) = \{C\colon \mathsf{T}C(x) \in S\} \quad \text{(C.34)}$$

$$\sigma_{\mathsf{NT}}(S, x) = \{C\colon \mathsf{NT}C(x) \in S\}. \quad \text{(C.35)}$$

We will say that two variables $x, y \in S$ are *concept equivalent* in S (denoted by $x \equiv_S y$) if $\sigma_{\mathsf{T}}(S, x) = \sigma_{\mathsf{T}}(S, y)$ and $\sigma_{\mathsf{NT}}(S, x) = \sigma_{\mathsf{NT}}(S, y)$. Intuitively, two variables are concept equivalent in S if, as far the signed formulae in S are concerned, they have the same properties. For instance, w.r.t. Example 32 above, let $S = \bigcup_{i=1,2,\ldots} S_i$. It is easily verified that for all $i = 1, 2, \ldots$, $\sigma_{\mathsf{T}}(S, x_i) = \{A, \exists R.A\}$ whereas, $\sigma_{\mathsf{NT}}(S, x_i) = \emptyset$. Therefore, *e.g.* x_1 and x_2 have the same properties in the sense that they are concept equivalent in S.

The functions $\sigma_{\mathsf{T}}(\cdot, \cdot)$ and $\sigma_{\mathsf{NT}}(\cdot, \cdot)$ will be used for the termination condition. Intuitively, w.r.t. Example 32 above, given $\mathsf{T}(\exists R.A)(x_2) \in S_2$, we do not perform the recursive call on $S_2(\mathsf{T}(\exists R.A)(x_2))$ since, there is an already introduced variable x_1, where $x_1 \prec_{\mathcal{V}_{\mathtt{ALC}}} x_2$, such that x_1 and x_2 are concept equivalent in $S = S_1 \cup S_2$.

Formally, let $S_1, S_2, \ldots, S_n$ be a sequence of sets of signed formulae. Consider,

$$S = \bigcup_{i=1,2,\ldots,n} S_i,$$

and $\alpha^{\exists} \in S_n$. Suppose that $Ind(\alpha^{\exists}) = y$. We will say that $\alpha^{\exists}$ *is blocked* at S_n (also, y *is a blocked variable* in S) if

1. there is another variable x such that $x \prec_{\mathcal{V}_{\mathtt{ALC}}} y$; and
2. x, y are concept equivalent in S, *i.e.* $x \equiv_S y$.

Moreover, we will say that y has *witness* x in S if x is the least variable w.r.t. $\prec_{\mathcal{V}_{\mathtt{ALC}}}$ satisfying conditions 1. and 2.

The sequence $S_1, S_2, \ldots, S_n$ plays the role of the the sets of signed formulae considered by recursive calls of Sat_{DL} during the proof: *i.e.* it is quite obvious that the recursive calls induce a tree where (i) each node is a set of signed formulae; (ii) the root is the set to the "AB-completion" of $S^{\phi}(\alpha^{\exists})$ of the first recursive call; and (iii) a node S has as children node the "AB-completion" $S(\alpha^{\exists})$ if w.r.t. S the procedure Sat_{DL} is recursively called on $S^{\phi}(\alpha^{\exists})$ (see Step 3a). The sets in the sequence $S_1, S_2, \ldots, S_n$ correspond to the sets along a path from root S_1 to node S_n.

The definition of blocked signed formula of type $\alpha^{\exists}$ expresses the condition under which no recursive call of the Sat_{DL} procedure should be made w.r.t. $\alpha^{\exists}$. For instance, w.r.t. Example 32, no recursive call will be made by considering $\mathsf{T}(\exists R.A)(x_2) \in S_2$. In fact, by considering the sequence S_1, S_2 in Example 32, it is easily verified that $\mathsf{T}(\exists R.A)(x_2)$ is blocked in S_2.

The procedure Sat_{DL} is defined in Table C.13 below.

C.4.1 The case without specialisations

We first discuss examples, correctness and completeness in the case without specialisations.

Algorithm 7 ($Sat_{DL}(S)$)
$Sat_{DL}(S)$ starts from the root labelled S and applies the following steps:

1. Select a not closed branch ϕ. If there is no such branch then return $false$ and exit.
2. If ϕ is not yet AB-completed then expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed. Update ϕ to the resulting branch;
3. If ϕ is AB-completed then
   (a) if for *some* signed formula $\alpha^{\exists}$ in S^{ϕ}, $Sat_{DL}(S^{\phi}(\alpha^{\exists}) \cup S^{\phi}(\Rightarrow)) = false$, where
      i. $S^{\phi}(\alpha^{\exists})$ has not yet been tested with respect to ϕ;
      ii. $\alpha^{\exists}$ is not blocked in S^{ϕ},

   then close ϕ and go to Step 1. Otherwise,
   (b) select a signed formula of type β, not being of type $\beta^{\forall}$, which is not yet fulfilled in the branch;
   (c) apply rule (PB) and go to Step 1. ■

Table C.13: Algorithm $Sat_{DL}(S)$ for $\mathcal{ALC}$.

Example 33 Let Σ be the set

$$\begin{aligned}\Sigma \;=\; & \{(\exists R_1.C_1)(a), (\forall R_1.D_{1_1})(a), (\forall R_1.D_{1_2})(a),\\ & (\exists R_2.C_2)(b), R_1(a,a)\}.\end{aligned}$$

Let A be the assertion

$$A = (D_{1_2} \sqcap \forall R_1.(D_{1_1} \sqcap D_{1_2}))(a).$$

It can easily be verified that $\Sigma \models_4 A$ holds. We will show that effectively $Sat_{DL}(\mathsf{T}\Sigma \cup \{\mathsf{NT}A\}) = false$. We will present the proof by means of the deduction tree in Figure C.7.

The first three nodes after the first one are obtained by a straightforward application of the $(B1)$ rule. Thereafter, Step 3a of Sat_{DL} has been applied. It is easily verified that

$$\begin{aligned}Sat_{DL}(S(\mathsf{T}(\exists R_1.C_1)(a))) &= true,\\ Sat_{DL}(S(\mathsf{T}(\exists R_2.C_2)(a))) &= true,\\ Sat_{DL}(S(\mathsf{NT}(\forall R_1.(D_{1_1} \sqcap D_{1_2}))(a))) &= false.\end{aligned}$$

In fact, just observe that

$$\begin{aligned}S(\mathsf{T}(\exists R_1.C_1)(a)) &= \{\mathsf{T}C_1(x_1), \mathsf{T}D_{1_1}(x_1), \mathsf{T}D_{1_2}(x_1)\},\\ S(\mathsf{T}(\exists R_2.C_2)(a)) &= \{\mathsf{T}C_2(x_2)\},\\ S(\mathsf{NT}(\forall R_1.D_{1_1} \sqcap D_{1_2})(a)) &= \{\mathsf{NT}(D_{1_1} \sqcap D_{1_2})(x_3), \mathsf{T}D_{1_1}(x_3), \mathsf{T}D_{1_2}(x_3)\}.\end{aligned}$$

Therefore, the conjunction of the three recursive calls is $false$, and, thus, according to $(C.21)$ Sat_{DL} returns $false$ as expected. ■

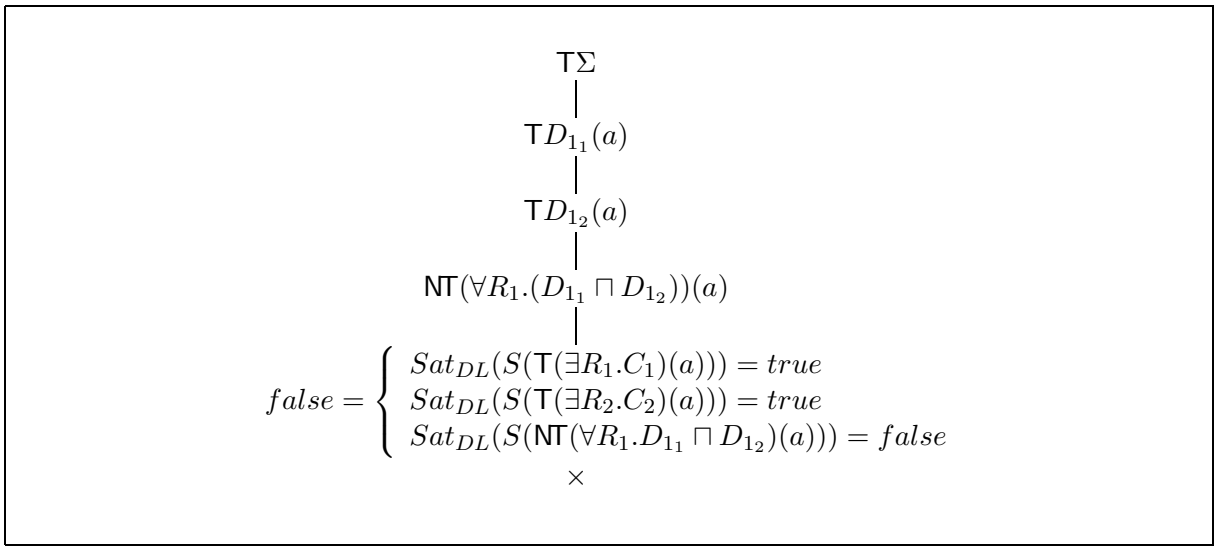


Figure C.7: Proof with recursive call to Sat_{DL}.

We show that Algorithm 7 is sound and complete with respect to type A semantics whenever there are no specialisations.

Proposition 27 *Let S be a set of signed formulae in $\mathcal{ALC}$ without specialisations,* i.e. $S(\Rightarrow) = \emptyset$. *Consider the set of rules in Table C.12. Then $Sat_{DL}(S)$ iff S is satisfiable with respect to type A semantics.* ⊣

Proof: The proof is an extension of the proof of Proposition 18 for the propositional case $\mathcal{L}$.

At first, it can be easily verified that the rules (A), $(B1)$, $(B2)$ and (PB) in Table C.12 are correct, *i.e.* ϕ is a branch and S^ϕ is satisfiable iff there is a branch ϕ' as the result of the application of a rule to ϕ such that $S^{\phi'}$ satisfiable. Furthermore, it can be verified that the blocking condition is never satisfied.

$\Rightarrow$.) Suppose $Sat(S)$. We show by induction on the number n of occurrences of signed formulae of type $\alpha^\exists \in S$ that there is (i) a finite deduction tree T build by $Sat(S)$; (ii) a not closed branch ϕ from S to a leaf in T; and a set $S(\phi)$ such that $S(\phi)$ is satisfiable and $S^\phi \subseteq S(\phi)$. As a consequence, $S \subseteq S^\phi$ is satisfiable.

Case $n = 0$ **:** In this case, similarly to the proof of Proposition 18, an interpretation $\mathcal{I}$ satisfying S can be build. Let T be the generated deduction tree and let ϕ be a not closed branch from S to a leaf in T. It is easily verified that T is finite. Such a branch has to exist, otherwise $Sat_{DL}(S) = false$. Let

$$S^\mathsf{T} = \{\mathsf{T}A \in S^\phi\}$$
$$S^\mathsf{NT} = \{\mathsf{NT}A \in S^\phi\}.$$

Let $S(\phi) = S^\mathsf{T} \cup S^\mathsf{NT}$. Of course, $S^\phi = S^\mathsf{T} \cup S^\mathsf{NT} \subseteq S(\phi)$. Let $\Delta^\mathcal{I}$ be the set of objects appearing in $S(\phi)$ and $w^\mathcal{I} = w$ for all $w \in \Delta^\mathcal{I}$. Let $\mathcal{I}$ be a relation such that $t \in A^\mathcal{I}$

if $\mathsf{T}A \in S^{\mathsf{T}}$, $A^{\mathcal{I}} = \emptyset$ otherwise, where A is an assertion of the form $B(w)$, $\neg B(w)$ or $R(w, v)$ (B is a primitive concept). More precisely, for each primitive concept A, for each role R, for all objects $w, v \in \Delta^{\mathcal{I}}$, set $A^{\mathcal{I}}(w) = \emptyset$, $R^{\mathcal{I}}(w, v) = \emptyset$ and

1. for each $\mathsf{T}A(w) \in S^{\mathsf{T}}$, assign $A^{\mathcal{I}}(w) := A^{\mathcal{I}}(w) \cup \{t\}$;
2. for each $\mathsf{T}\neg A^{\mathcal{I}}(w) \in S^{\mathsf{T}}$, assign $A^{\mathcal{I}}(w) := A^{\mathcal{I}}(w) \cup \{f\}$;
3. for each $\mathsf{T}R(w, v) \in S^{\mathsf{T}}$, assign $R^{\mathcal{I}}(w, v) := R^{\mathcal{I}}(w, v) \cup \{t\}$.

Since, ϕ is completed and not closed, $\mathcal{I}$ with domain $\Delta^{\mathcal{I}}$ is a four-valued interpretation satisfying S^{T}, S^{NT} and, thus, $S(\phi)$. As a consequence, $S \subseteq S^{\phi}$ is satisfiable.

Case $n > 0$ **:** Let T be the generated finite deduction tree and let ϕ be a not closed and completed branch from S to a leaf in T. Such a branch has to exist, otherwise $Sat_{DL}(S) = false$. Now, Step 3. of the algorithm is applied. Suppose $\alpha_1^{\exists}, \ldots, \alpha_m^{\exists} \in S^{\phi}$ are the top level $\alpha_i^{\exists}$ for which $S^{\phi}(\alpha_i^{\exists})$ has been checked. Let x_i be the new variables introduced. Certainly, $m \leq n$ holds. Since $Sat_{DL}(S) = true$, it follows that $\bigwedge_{1 \leq i \leq m} Sat(S^{\phi}(\alpha_i^{\exists}))$ holds. Since, for all $1 \leq i \leq m$ the number of occurrences of signed formulae of type $\alpha^{\exists}$ in $S^{\phi}(\alpha_i^{\exists})$ is less than n, by induction there are (i) finite deduction trees T_i; (ii) not closed and completed branches ϕ_i from $S^{\phi}(\alpha_i^{\exists})$ to a leaf in T_i; and (iii) sets $S(\phi_i)$ such that $S(\phi_i)$ is satisfiable and $S_i^{\phi} \subseteq S(\phi_i)$.

Let

$$S(\phi) = S^{\phi} \cup \bigcup_{1 \leq i \leq m} S(\phi_i)$$

and

$$S^{\mathsf{T}} = \{\mathsf{T}A \in S(\phi)\} \cup \{\mathsf{T}R_i(w_i, x_i) : w_i = Ind(\alpha_i^{\exists}), R_i = Role(\alpha_i^{\exists})\}$$

and

$$S^{\mathsf{NT}} = \{\mathsf{NT}A \in S(\phi)\}.$$

By definition, $S^{\phi} \subseteq S(\phi)$. Let $\Delta^{\mathcal{I}}$ be the set of objects appearing in $S(\phi)$ and let $w^{\mathcal{I}} = w$, for all $w \in \Delta^{\mathcal{I}}$. Let $\mathcal{I}$ be a relation such that $t \in A^{\mathcal{I}}$ if $\mathsf{T}A \in S^{\mathsf{T}}$, $A^{\mathcal{I}} = \emptyset$ otherwise, where A is an assertion of the form $B(w)$, $\neg B(w)$ or $R(w, v)$ (B is a primitive concept). More precisely, for each primitive concept A, for each role R, for all objects $w, v \in \Delta^{\mathcal{I}}$, set $A^{\mathcal{I}}(w) = \emptyset$, $R^{\mathcal{I}}(w, v) = \emptyset$ and

1. for each $\mathsf{T}A(w) \in S^{\mathsf{T}}$, assign $A^{\mathcal{I}}(w) := A^{\mathcal{I}}(w) \cup \{t\}$;
2. for each $\mathsf{T}\neg A(w) \in S^{\mathsf{T}}$, assign $A^{\mathcal{I}}(w) := A^{\mathcal{I}}(w) \cup \{f\}$;
3. for each $\mathsf{T}R(w, v) \in S^{\mathsf{T}}$, assign $R^{\mathcal{I}}(w, v) := R^{\mathcal{I}}(w, v) \cup \{t\}$.

Since, ϕ is completed and not closed, $\mathcal{I}$ with domain $\Delta^{\mathcal{I}}$ is a four-valued interpretation satisfying S^{T}, S^{NT}, and thus, $S(\phi)$. As a consequence, $S \subseteq S^{\phi}$ is satisfiable.

$\Leftarrow$.) Suppose S is satisfiable. Let T be the generated completed tree. It is easily verified that T is finite. Therefore, from the correctness of the rules it follows that there is a finite completed branch ϕ in T such that S^{ϕ} is satisfiable. Therefore, $Sat(S)$. Q.E.D.

Example 34 Let us see how the interpretation of the above proof is build. Let Σ be the set

$$\Sigma = \{R(a,a), (\forall R.D_{1_1})(a), (\forall R.D_{1_2})(a)\}$$

Let A be the assertion

$$A = (D_{1_2} \sqcap \forall R.(D_{1_1} \sqcap D_{1_3}))(a).$$

It is easily verified that $\Sigma \not\models_4 A$ holds. In fact, $Sat_{DL}(\mathsf{T}\Sigma \cup \{\mathsf{NT}A\}) = true$, as shown in the not closed and completed deduction tree in Figure C.8.

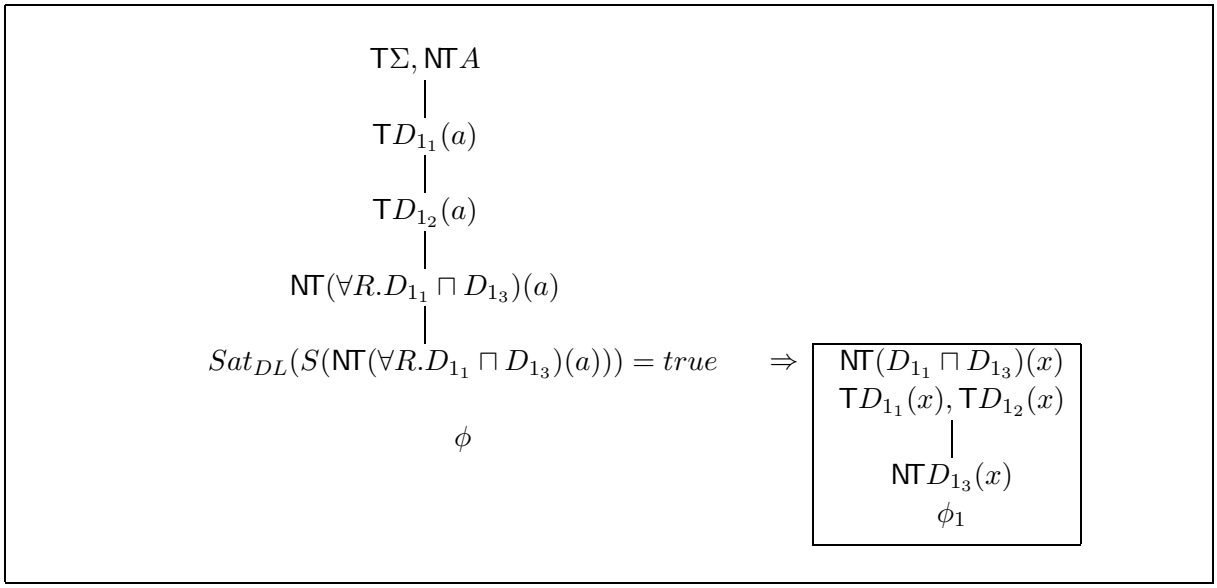


Figure C.8: Not closed deduction tree with recursive call to Sat_{DL}.

Let $\alpha_1^{\exists}$ be $\mathsf{NT}(\forall R_1.D_{1_1} \sqcap D_{1_3})(a))$. Just notice that $S(\alpha_1^{\exists})$ is

$$S(\alpha_1^{\exists}) = \{\mathsf{NT}(D_{1_1} \sqcap D_{1_3})(x), \mathsf{T}D_{1_1}(x), \mathsf{T}D_{1_2}(x)\}$$

As shown in Figure C.8, $Sat_{DL}(\alpha_1^{\exists}) = true$. According to the proof of Proposition 28 above, it follows that $S(\phi_1)$ is

$$S(\phi_1) = \{\mathsf{NT}(D_{1_1} \sqcap D_{1_3})(x), \mathsf{T}D_{1_1}(x), \mathsf{T}D_{1_2}(x), \mathsf{NT}D_{1_3}(x)\}$$

which is satisfiable. Finally, let $S(\phi) = S^{\phi} \cup S(\phi_1)$, where

$$S^{\phi} = \mathsf{T}\Sigma \cup \{\mathsf{NT}A, \mathsf{T}D_{1_1}(a), \mathsf{T}D_{1_2}(a), \mathsf{NT}(\forall R_1.D_{1_1} \sqcap D_{1_3})(a))\}$$

and let

$$S^{\mathsf{T}} = \{\mathsf{T}A \in S(\phi)\} \cup \{\mathsf{T}R(a,x)\},$$
$$S^{\mathsf{NT}} = \{\mathsf{NT}A \in S(\phi)\}.$$

$\mathcal{I}$ is build as follows. The domain of $\mathcal{I}$ is

$$\Delta^{\mathcal{I}} = \{a, x\}$$

Of course, $a^{\mathcal{I}} = a$ and $x^{\mathcal{I}} = x$. Define $\mathcal{I}$ such that $t \in A^{\mathcal{I}}(a)$ if $\mathsf{T}A(a) \in S^{\mathsf{T}}$ ($A^{\mathcal{I}}(a) = \emptyset$, otherwise) and A primitive concept, and $t \in R^{\mathcal{I}}(a,x)$ if $\mathsf{T}R(a,x) \in S^{\mathsf{T}}$ ($R^{\mathcal{I}}(a,x) = \emptyset$, otherwise): *i.e.*

$$t \in {D_{1_1}}^{\mathcal{I}}(a),$$
$$t \in {D_{1_2}}^{\mathcal{I}}(a),$$
$$t \in R^{\mathcal{I}}(a,a),$$
$$t \in R^{\mathcal{I}}(a,x),$$
$$t \in {D_{1_1}}^{\mathcal{I}}(x),$$
$$t \in {D_{1_2}}^{\mathcal{I}}(x),$$
$${D_{1_3}}^{\mathcal{I}}(a) = \emptyset,$$
$${D_{1_3}}^{\mathcal{I}}(x) = \emptyset,$$
$$R^{\mathcal{I}}(x,a) = \emptyset, \text{ and}$$
$$R^{\mathcal{I}}(x,x) = \emptyset.$$

It follows that $\mathcal{I}$ satisfies $\mathsf{T}\Sigma \cup \{\mathsf{NT}A\}$. ■

C.4.2 The case with specialisations

Let us consider Example 32. We have already pointed out that without any blocking condition, our Sat_{DL} procedure could never stop. The following example instead shows how the blocking condition prevents Sat_{DL} to loop infinitely.

Example 35 Consider Example 32, *i.e.* consider the set of signed formulae,

$$S = \{\mathsf{T}A(a), \mathsf{T}A \Rightarrow \exists R.A\}.$$

We have already seen that S is satisfiable. Figure C.9 below shows the deduction tree build by $Sat_{DL}(S)$, which returns *true*.

Note that from

$$\begin{array}{lcl} S_0 & = & S_1 \cup S_2, \\ \sigma_{\mathsf{T}}(x_1, S_0) & = & \{A, \exists R.A\}, \\ \sigma_{\mathsf{T}}(x_2, S_0) & = & \{A, \exists R.A\}, \\ \sigma_{\mathsf{NT}}(x_1, S_0) & = & \sigma_{\mathsf{NT}}(x_1, S_0) = \emptyset, \\ x_1 \prec_{\mathcal{V}_{\mathtt{ALC}}} x_2, & & \end{array}$$

it follows that $x_1 \equiv_{S_0} x_2$ and, thus, $\mathsf{T}(\exists R.A)(x_2)$ is blocked in S_2.

How can a model $\mathcal{I}$ be build from the deduction tree in Figure C.9? Essentially we proceed as for the case without specialisations (see proof of Proposition 27), except that we add some additional signed formulae to S^{T}. So, let ϕ be the top branch. Hence, S^{ϕ} is $S \cup \{\mathsf{T}(\exists R.A)(a)\}$. As in proof of Proposition 27, case $n > 0$, let $S(\phi) = S^{\phi} \cup S_1 \cup S_2$, *i.e.*

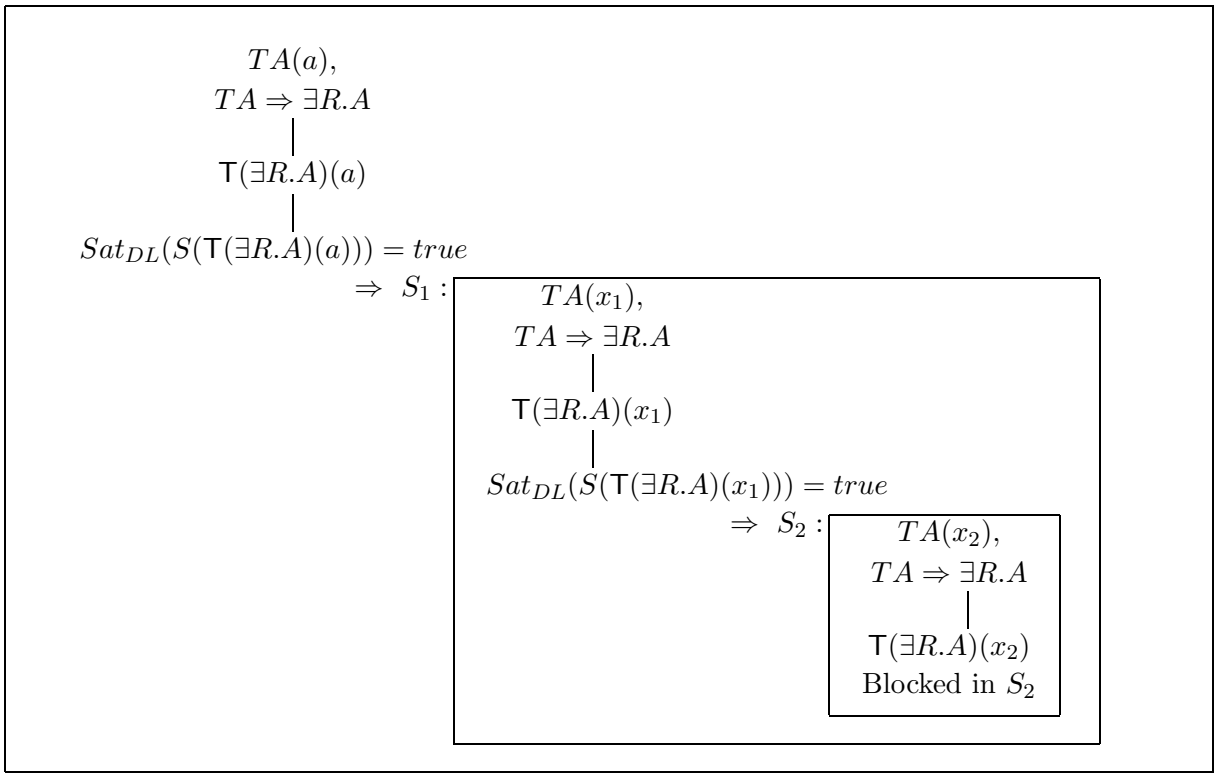


Figure C.9: Deduction tree with loop block condition in $\mathcal{ALC}$.

$$S(\phi) = \{\mathsf{T}A(a), \mathsf{T}(\exists R.A)(a), \mathsf{T}A(x_1), \mathsf{T}(\exists R.A)(x_1), \mathsf{T}A(x_2), \mathsf{T}(\exists R.A)(x_2)\}.$$

It follows that S^{T} is

$$S^{\mathsf{T}} = S(\phi) \cup \{\mathsf{T}R(a, x_1), \mathsf{T}R(x_1, x_2)\}.$$

The key difference to the case without specialisations is that we add additionally $\mathsf{T}R(x_2, x_2)$ to S^{T}, as specified in [68], *i.e.*

$$S^{\mathsf{T}} := S^{\mathsf{T}} \cup \{\mathsf{T}R(x_2, x_2)\}.$$

Now, from S^{T} a model of S can be build in the usual way, yielding an model $\mathcal{I}$ with domain $\Delta^{\mathcal{I}} = \{a, x_1, x_2\}$ such that (i) $w^{\mathcal{I}} = w$, for all $w \in \Delta^{\mathcal{I}}$; and (ii)

$$A^{\mathcal{I}}(a) = A^{\mathcal{I}}(x_1) = A^{\mathcal{I}}(x_2) = \{t\},$$
$$R^{\mathcal{I}}(a, x_1) = R^{\mathcal{I}}(x_1, x_2) = R^{\mathcal{I}}(x_2, x_2) = \{t\}.$$

■

Sat_{DL} is sound and complete with respect to type A semantics in case there are specialisations.

Proposition 28 *Let S be a set of signed formulae in $\mathcal{ALC}$. Consider the set of rules in Table C.12. Then $Sat_{DL}(S)$ iff S is satisfiable with respect to type A semantics.* ⊣

Proof: The proof is an extension of the proof of Proposition 27 for the case without specialisations. We do not give here the complete proof, as it is an adaption of the proof described in [68] to the four-valued case. What we do here is to show how a model $\mathcal{I}$ for S can be build from the deduction tree of $Sat_{DL}(S)$ (according to [68]), whenever S is satisfiable.

As for case $n > 0$ in proof of Proposition 27, consider

$$S(\phi) = S^{\phi} \cup \bigcup_{1 \leq i \leq m} S(\phi_i)$$

and

$$S_0^{\mathsf{T}} = \{\mathsf{T}A \in S(\phi)\} \cup \{\mathsf{T}R_i(w_i, x_i) : w_i = Ind(\alpha_i^{\exists}), R_i = Role(\alpha_i^{\exists})\}.$$

By definition $S^{\phi} \subseteq S(\phi)$. Finally, define

$$S^{\mathsf{T}} = S_0^{\mathsf{T}} \cup \{\mathsf{T}R(x,y) \colon x \text{ blocked in } S(\phi), x \text{ has witness } z, \mathsf{T}R(z,y) \in S_0^{\mathsf{T}}\}, \tag{C.36}$$

and

$$S^{\mathsf{NT}} = \{\mathsf{NT}A \in S(\phi)\}. \tag{C.37}$$

Let $\Delta^{\mathcal{I}}$ be the set of objects appearing in $S(\phi)$ and let $w^{\mathcal{I}} = w$, for all $w \in \Delta^{\mathcal{I}}$. Let $\mathcal{I}$ be a relation such that $t \in A^{\mathcal{I}}$ if $\mathsf{T}A \in S^{\mathsf{T}}$, $A^{\mathcal{I}} = \emptyset$ otherwise, where A is an assertion of the form $B(w)$, $\neg B(w)$ or $R(w,v)$ (B is a primitive concept). More precisely, for each primitive concept A, for each role R, for all objects $w, v \in \Delta^{\mathcal{I}}$, set $A^{\mathcal{I}}(w) = \emptyset$, $R^{\mathcal{I}}(w,v) = \emptyset$ and

1. for each $\mathsf{T}A(w) \in S^{\mathsf{T}}$, assign $A^{\mathcal{I}}(w) \colon = A^{\mathcal{I}}(w) \cup \{t\}$;
2. for each $\mathsf{T}\neg A(w) \in S^{\mathsf{T}}$, assign $A^{\mathcal{I}}(w) \colon = A^{\mathcal{I}}(w) \cup \{f\}$;
3. for each $\mathsf{T}R(w,v) \in S^{\mathsf{T}}$, assign $R^{\mathcal{I}}(w,v) \colon = R^{\mathcal{I}}(w,v) \cup \{t\}$.

Since, ϕ is completed and not closed, $\mathcal{I}$ with domain $\Delta^{\mathcal{I}}$ is a four-valued interpretation satisfying S^{T}, S^{NT} and, thus, $S(\phi)$ is satisfiable. Therefore, $S \subseteq S^{\phi}$ is satisfiable. Q.E.D.

Similarly to propositional case, Algorithm 8 shown in Table C.14, allows us to build all completions of a set of signed formulae S.

Consider proof of Proposition 28. Consider the sets S^{T} and S^{NT} (see Equation (C.36) and Equation (C.37)). The set

$$\begin{array}{rcl} \tilde{\mathcal{S}} & = & \{\mathsf{T}A \in S^{\mathsf{T}} : A \text{ atomic assertion}\} \cup \\ & & \{\mathsf{NT}A \in S^{\mathsf{NT}} : A \text{ atomic assertion}\} \end{array} \tag{C.38}$$

is called a *four-valued completion* of S. A *two-valued completion* of S is the set

$$\begin{array}{rcl} \tilde{\mathcal{S}} & = & \{\mathsf{T}A(w) \in S^{\mathsf{T}} : A \text{ primitive concept}\} \cup \\ & & \{\mathsf{NT}A(w) \in S^{\mathsf{NT}} : A \text{ primitive concept}\} \cup \\ & & \{\mathsf{T}R(w,v) \in S^{\mathsf{T}}\}. \end{array} \tag{C.39}$$

Moreover, we define

$$\tilde{\mathcal{S}}^{\mathsf{T}} = \{\mathsf{T}A \in \tilde{\mathcal{S}}\}, \tag{C.40}$$
$$\tilde{\mathcal{S}}^{\mathsf{NT}} = \{\mathsf{NT}A \in \tilde{\mathcal{S}}\}, \tag{C.41}$$
$$\tilde{\mathcal{S}}^{+} = \{\mathsf{T}A(w) \in S^{\mathsf{T}} : A \text{ primitive concept}\} \cup \{\mathsf{T}R(w,v) \in S^{\mathsf{T}}\}. \tag{C.42}$$

Given a four-valued completion $\tilde{\mathcal{S}}$ of S, we define the following *four-valued completion KBs* of $\tilde{\mathcal{S}}$:

$$\Sigma_{\tilde{\mathcal{S}}} = \{A : \mathsf{T}A \in \tilde{\mathcal{S}}^{\mathsf{T}}\}, \tag{C.43}$$
$$\Sigma^{+}_{\tilde{\mathcal{S}}} = \{A : \mathsf{T}A \in \tilde{\mathcal{S}}^{+}\}. \tag{C.44}$$

For the two-valued case, given a two-valued completion $\tilde{\mathcal{S}}$ of S, we define the following *two-valued completion KBs* of $\tilde{\mathcal{S}}$:

$$\Sigma_{\tilde{\mathcal{S}}} = \{A : \mathsf{T}A \in \tilde{\mathcal{S}}^{\mathsf{T}}\} \cup \{\neg A(w) : \mathsf{NT}A(w) \in \tilde{\mathcal{S}}^{\mathsf{NT}}\}, \tag{C.45}$$
$$\Sigma^{+}_{\tilde{\mathcal{S}}} = \{A : \mathsf{T}A \in \tilde{\mathcal{S}}^{+}\}. \tag{C.46}$$

Given a four-valued completion $\tilde{\mathcal{S}}$ of S, then a *four-valued canonical model* of $\tilde{\mathcal{S}}$ is obtained as in proof of Proposition 28. Let $\Delta^{\mathcal{I}}$ be the set of objects appearing in $\tilde{\mathcal{S}}$ and let $w^{\mathcal{I}} = w$, for all $w \in \Delta^{\mathcal{I}}$. For each primitive concept A, for each role R, for all objects $w, v \in \Delta^{\mathcal{I}}$, set $A^{\mathcal{I}}(w) = \emptyset$, $R^{\mathcal{I}}(w,v) = \emptyset$ and

1. for each $\mathsf{T}A(w) \in \tilde{\mathcal{S}}$, assign $A^{\mathcal{I}}(w) := A^{\mathcal{I}}(w) \cup \{t\}$;
2. for each $\mathsf{T}\neg A(w) \in \tilde{\mathcal{S}}$, assign $A^{\mathcal{I}}(w) := A^{\mathcal{I}}(w) \cup \{f\}$;
3. for each $\mathsf{T}R(w,v) \in \tilde{\mathcal{S}}$, assign $R^{\mathcal{I}}(w,v) := R^{\mathcal{I}}(w,v) \cup \{t\}$.

On the other hand, given a two-valued completion $\tilde{\mathcal{S}}$ of S, then a *two-valued canonical model* of $\tilde{\mathcal{S}}$ is obtained as follows. Let $\mathcal{I}$ be an arbitrary two-valued interpretation. For each primitive concept A, for each role R, for all objects $w, v \in \Delta^{\mathcal{I}}$, redefine $\mathcal{I}$ as follows:

1. for each $\mathsf{T}A(w) \in \tilde{\mathcal{S}}$, assign $A^{\mathcal{I}}(w) := \{t\}$;
2. for each $\mathsf{T}R(w,v) \in \tilde{\mathcal{S}}$, assign $R^{\mathcal{I}}(w,v) := \{t\}$;
3. for each $\mathsf{NT}A(w) \in \tilde{\mathcal{S}}$, assign $A^{\mathcal{I}}(w) := \{f\}$.

It is easily verified that a canonical model of $\tilde{\mathcal{S}}$ is also a model of S. It is worth noticing that in the four-valued case, given $\tilde{\mathcal{S}}$, the canonical model of $\tilde{\mathcal{S}}$ is unique, whereas in the two-valued case there are infinitely many. Just notice that in Step 5. $\tilde{\mathcal{S}}(\alpha^{\exists})$ is the set of all completions of $S^{\phi}(\alpha^{\exists}) \cup S^{\phi}(\Rightarrow)$. Therefore, given $\Psi = \{\alpha^{\exists}_1, \ldots, \alpha^{\exists}_m\}$, in order to build a completion of S we have to consider m completions $\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_m$, each of them being a completion of $S^{\phi}(\alpha^{\exists}_1) \cup S^{\phi}(\Rightarrow)$, $\ldots$, $S^{\phi}(\alpha^{\exists}_m) \cup S^{\phi}(\Rightarrow)$, respectively (Step 5b.).

We address now the problem of determining whether the analogue of Proposition 19 and Proposition 20 hold in the context of $\mathcal{ALC}$. The following proposition is immediate and is the $\mathcal{ALC}$ equivalent of Proposition 19.

Algorithm 8 ($Compl_{DL}(S)$)
Essentially, the procedure proceeds in a similar way as Sat_{DL}.

1. Select a not closed branch ϕ. If there is no such ϕ then $Compl_{DL}(S) := \emptyset$ and exit.
2. If ϕ is not yet AB-completed then expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed. Update ϕ to the resulting branch;
3. If ϕ is AB-completed then
   (a) for *all* $\alpha^{\exists} \in S^{\phi}$, let $\tilde{\mathcal{S}}(\alpha^{\exists}) = Compl_{DL}(S^{\phi}(\alpha^{\exists}) \cup S^{\phi}(\Rightarrow))$, where $S^{\phi}(\alpha^{\exists})$ has not yet been considered with respect to ϕ and $\alpha^{\exists}$ is not blocked in S^{ϕ}. If for *some* $\alpha^{\exists} \in S^{\phi}$, $\tilde{\mathcal{S}}(\alpha^{\exists}) = \emptyset$ then close ϕ and go to Step 1. Otherwise,
   (b) select a signed formula of type β, not of type $\beta^{\forall}$ and not yet fulfilled in the branch;
   (c) apply rule (PB) and go to Step 1.
4. Let T be the generated deduction tree. Note that T at this point is not closed.
5. For all not closed branches ϕ from S to a leaf in T do:
   (a) let $\Psi = \{\alpha_1^{\exists}, \ldots, \alpha_m^{\exists}\}$ be the set of top level $\alpha_i^{\exists} \in S^{\phi}$ for which $S^{\phi}(\alpha_i^{\exists})$ has been considered and let x_i be the new variables introduced.
   (b) for each tuple $(\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_m)$ such that $\tilde{\mathcal{S}}_j \in \tilde{\mathcal{S}}(\alpha_j^{\exists})$, let $S(\phi) = S^{\phi} \cup \bigcup_{1 \leq j \leq m} \tilde{\mathcal{S}}_j$. Let

$$S_0^{\mathsf{T}} = \{\mathsf{T}A \in S(\phi)\} \cup \{\mathsf{T}R_i(w_i, x_i) : w_i = Ind(\alpha_i^{\exists}), R_i = Role(\alpha_i^{\exists})\},$$
$$S^{\mathsf{T}} = S_0^{\mathsf{T}} \cup \{\mathsf{T}R(x,y) : x \text{ blocked in } S(\phi), x \text{ has witness } z, \mathsf{T}R(z,y) \in S_0^{\mathsf{T}}\},$$

   and let

$$S^{\mathsf{NT}} = \{\mathsf{NT}A \in S(\phi)\};$$

   (c) define the $\tilde{\mathcal{S}}$ according to $(C.38)$ or $(C.39)$, in case four-valued or two-valued semantics;
   (d) set $Compl_{DL}(S) := Compl_{DL}(S) \cup \{\tilde{\mathcal{S}}\}$. ■

Table C.14: Algorithm $Compl_{DL}(S)$ for $\mathcal{ALC}$.

Proposition 29 *In $\mathcal{ALC}$, let Σ be a KB and let $A(a)$ be an assertion such that A is a primitive concept. Then*

1. $\Sigma \models_4 A(a)$ *iff a occurs in Σ_F and for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma)$, $\mathcal{I}$ satisfies $A(a)$;*
2. $\Sigma \models_2 A(a)$ *iff for all two-valued canonical models $\mathcal{I}$ of two-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma)$, $\mathcal{I}$ satisfies $A(a)$.* ⊣

Just notice that the conditions that A should be primitive and a occurs in Σ_F are essential. First, since there is no four-valued top concept, then from $\Sigma \models_4 A(a)$, a occurs in Σ_F follows. Second, suppose that Σ is

$$\Sigma = \{(\forall R.(C \wedge D))(a)\}$$

It follows that $Compl_{DL}(\mathsf{T}\Sigma) = \{\tilde{\mathcal{S}}\}$, where $\tilde{\mathcal{S}} = \emptyset$. Hence, the four-valued canonical model $\mathcal{I}$ of $\tilde{\mathcal{S}}$ is such that *e.g.* the positive extension of R is empty. Therefore, $\mathcal{I}$ satisfies $(\forall R.E)(a)$,

but $\Sigma \not\models_4 (\forall R.E)(a)$. On the other hand, suppose we want apply Proposition 29 in order to verify that $\Sigma \models_4 (\forall R.C)(a)$. How do we proceed? Well, first we define $A := (\forall R.C)(a)$. Then we check whether for $\Sigma' = \Sigma \cup \{A := \forall R.C\}$, $\Sigma' \models_4 A(a)$ holds. Now, it can be verified that $Compl_{DL}(\mathsf{T}\Sigma') = \{\tilde{\mathcal{S}}\}$, where $\tilde{\mathcal{S}} = \{\mathsf{T}A(a)\}$.

In the two-valued case, the condition "a occurs in Σ_F" is wrong as there are two-valued top concepts: for instance, $\emptyset \models_2 (A \sqcup \neg A)(a)$. The following proposition is a consequence of Proposition 29 above.

Proposition 30 *In $\mathcal{ALC}$, let Σ be a KB and let $A(a)$ be an assertion such that A is a primitive concept. Then*

1. *$\Sigma \models_4 A(a)$ iff a occurs in Σ_F and for all four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma)$, $\tilde{\mathcal{S}}^{\mathsf{T}} \cup \{\mathsf{NT}A(a)\}$ is not satisfiable;*
2. *$\Sigma \models_4 A(a)$ iff a occurs in Σ_F and for four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma)$, $\Sigma_{\tilde{\mathcal{S}}} \models_4 A(a)$, where $\Sigma_{\tilde{\mathcal{S}}}$ is the four-valued completion KB of $\tilde{\mathcal{S}}$;*
3. *$\Sigma \models_2 A(a)$ iff for all two-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma)$, $\tilde{\mathcal{S}} \cup \mathsf{T}\Sigma_T \cup \{\mathsf{NT}A(a)\}$ is not satisfiable;*
4. *$\Sigma \models_2 A(a)$ iff for two-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma)$, $\Sigma_{\tilde{\mathcal{S}}} \cup \mathsf{T}\Sigma_T \models_2 A(a)$, where $\Sigma_{\tilde{\mathcal{S}}}$ is the two-valued completion KB of $\tilde{\mathcal{S}}$.* ⊣

Example 36 For instance, consider the KB

$$\Sigma \;=\; \{B(b), \exists R.E \Rightarrow A, \forall R.\neg E \Rightarrow A\}$$

It is easily verified that $\Sigma \models_2 A(a)$, whereas $\Sigma \not\models_4 A(a)$. The reason relies behind the fact that if $C = \exists R.E$ then $\neg C$ is equivalent (two-valued and four-valued) to $\forall R.\neg E$. Therefore, Σ can be rewritten in the form

$$\Sigma_1 \;=\; \{B(b), C \Rightarrow A, \neg C \Rightarrow A\}$$

which is equivalent to

$$\Sigma_2 \;=\; \{B(b), C \sqcup \neg C \Rightarrow A\}$$

In two-valued semantics, $C \sqcup \neg C$ is equivalent to the top concept and, thus, the specialisation dictates that every individual is an A, *i.e.* $\Sigma_2 \models_2 A(a)$. On the other hand, we have already seen that in our four-valued semantics there is no top concept, *e.g.* $\not\models_4 (C \sqcup \neg C)(a)$. Therefore, $\Sigma_2 \not\models_4 A(a)$. This is the reason why in Proposition 30, in case of two-valued semantics, we have to consider the set $\mathsf{T}\Sigma_T$ too. In fact, it is easily verified that the set of four-valued completions is $Compl_{DL}(\mathsf{T}\Sigma) = \{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2, \tilde{\mathcal{S}}_3, \tilde{\mathcal{S}}_4\}$, where

$$\begin{array}{rcl}
\tilde{\mathcal{S}}_1 & = & \{\mathsf{T}B(b), \mathsf{T}R(b,x), \mathsf{NT}E(x), \mathsf{NT}\neg E(x)\}, \\
\tilde{\mathcal{S}}_2 & = & \{\mathsf{T}B(b), \mathsf{T}A(b)\}, \\
\tilde{\mathcal{S}}_3 & = & \{\mathsf{T}B(b), \mathsf{T}A(b), \mathsf{T}R(b,x_1), \mathsf{T}R(b,x_2), \mathsf{T}E(x_1), \mathsf{NT}\neg E(x_2)\}, \\
\tilde{\mathcal{S}}_4 & = & \{\mathsf{T}B(b), \mathsf{T}A(b), \mathsf{T}R(b,x), \mathsf{T}E(x), \mathsf{T}\neg E(x)\}.
\end{array}$$

Now, *e.g.* $\tilde{\mathcal{S}}_2 \cup \{\mathsf{NT}A(a)\}$ is four-valued satisfiable, according to Point 3. of Proposition 30. On the other hand, the set of two-valued completions is $Compl_{DL}(\mathsf{T}\Sigma) = \{\tilde{\mathcal{S}}_5, \tilde{\mathcal{S}}_6\}$, where

$$\begin{aligned}\tilde{\mathcal{S}}_5 &= \{\mathsf{T}B(b), \mathsf{T}A(b)\},\\ \tilde{\mathcal{S}}_6 &= \{\mathsf{T}B(b), \mathsf{T}A(b), \mathsf{T}R(b,x_1), \mathsf{T}R(b,x_2), \mathsf{T}E(x_1), \mathsf{T}E(x_2)\}.\end{aligned}$$

Note that $\mathsf{T}\Sigma_T = \{\mathsf{T}\exists R.E \Rightarrow A, \mathsf{T}\forall R.\neg E \Rightarrow A\}$. It is easily verified that both $\tilde{\mathcal{S}}_5 \cup \mathsf{T}\Sigma_T \cup \{\mathsf{NT}A(a)\}$ and $\tilde{\mathcal{S}}_6 \cup \mathsf{T}\Sigma_T \cup \{\mathsf{NT}A(a)\}$ are not two-valued satisfiable and, thus, $\Sigma \models_2 A(a)$, according Point 4. of Proposition 30. The above example shows that it is necessary to consider $\mathsf{T}\Sigma_T$. Otherwise both $\tilde{\mathcal{S}}_5 \cup \{\mathsf{NT}A(a)\}$ and $\tilde{\mathcal{S}}_6 \cup \{\mathsf{NT}A(a)\}$ are two-valued satisfiable. ■

C.4.3 The case with acyclic specialisations

We conclude by considering the case of well formed KBs, *i.e.* KBs Σ in which Σ_T contains only specialisations of the form $A \Rightarrow C$ and concept definitions of the form $A := C$ (see Section 6.2). Remember that the form of this kind of specialisations are the usual one present in real systems and their limited expressive power has little impact in the case of multimedia document retrieval. Moreover, their limited form considerably simplifies the computation of completions in the four-valued case. The benefits will be seen in Section C.5 later on.

The extension of the definition of well formed KB Σ to the case of *signed formulae* S is straightforward: S is well formed iff the set of definitions and specialisations in $S(\Rightarrow)$ is well formed. For instance, $S = \{\mathsf{T}A(a), \mathsf{T}A \Rightarrow \exists R.A\}$ is not well formed, as $S(\Rightarrow) = \{\mathsf{T}A \Rightarrow \exists R.A\}$ is cyclic.

Now, suppose that a set of signed formulae S is well formed. Hence, $S(\Rightarrow)$ contains only expressions of the form $TA := C$ and $\mathsf{T}A \Rightarrow C$. But, $TA := C$ is a macro for $\mathsf{T}A \Rightarrow C$ and $\mathsf{T}C \Rightarrow A$. Therefore, we can assume that $S(\Rightarrow)$ contains only specialisations of the form $\mathsf{T}A \Rightarrow C$ and of the form $\mathsf{T}C \Rightarrow A$. Given these restrictions on the form of specialisations, we can tailor the inference rule as follows: (*i*) rule $(B1)$ should not be applied to $\beta^{\Rightarrow}$ of the form $\mathsf{T}C \Rightarrow A$; (*ii*) rule $(B2)$ should not be applied to $\beta^{\Rightarrow}$ of the form $\mathsf{T}A \Rightarrow C$; and (*iii*) rule (PB) should not be applied to $\beta^{\Rightarrow}$. Note that, according to (*iii*), the condition "(PB) should not be applied to $\beta^{\Rightarrow}$" is necessary. Otherwise, for $S = \{\mathsf{T}C(a), \mathsf{T}A \Rightarrow \exists R.B\}$ we have infinite applications of the (PB) rule.

The key note on well formed sets S is that no infinite loop arise through these rules, and thus, no blocking condition is needed. This is a well known property of well formed KBs.

For well formed sets S of signed formulae, the procedure $AcyclicSat_{DL}(S)$ described in Table C.16 terminates and determines whether S is satisfiable or not. $AcyclicSat_{DL}(S)$ relies on the inference rules described in Table C.15 below.

In [148] it is shown that Proposition 31 below holds for the the two-valued case. The proof easily applies to the four-valued case too.

Proposition 31 *Let S be a well formed set of signed formulae in $\mathcal{ALC}$. Then $AcyclicSat_{DL}(S)$ iff S is satisfiable with respect to type A semantics.* ⊣

Moreover, suppose that Σ is a well formed KB. We have seen that there is no top concept in four-valued $\mathcal{ALC}$, *i.e.* there is no concept C and individual a such that $\emptyset \models_4 C(a)$. Of course, this property does not hold for $\models_2$, *e.g.* $\emptyset \models_2 (\neg A \sqcup A)(a)$ (but $\emptyset \not\models_4 (\neg A \sqcup A)(a)$). Now, consider Σ_T. Since there is no top concept, there is no concept C and individual a such that $\Sigma_T \models_4 C(a)$. More generally, this property holds for generic KBs too. Again, observe that this is not true for the two-valued case. For example, just consider $\Sigma_T = \{A := E, B := \neg E\}$.

$$(A)\ \frac{\alpha}{\alpha_1, \alpha_2} \qquad \text{if } \alpha \text{ is not of type } \alpha^{\exists}$$

$$(B1)\ \frac{\beta, \beta_1^c}{\beta_2} \qquad \text{if } \beta \text{ is not of type } \mathsf{T}C \Rightarrow A$$

$$(B2)\ \frac{\beta, \beta_2^c}{\beta_1} \qquad \text{if } \beta \text{ is neither of type } \beta^{\forall} \text{ nor of type } \mathsf{T}A \Rightarrow C$$

$$(PB)\ \frac{\beta}{\beta_1 \mid \beta_1^c, \beta_2} \qquad \text{if } \beta \text{ is neither of type } \beta^{\forall} \text{ nor of type } \beta^{\Rightarrow}$$

Table C.15: Specialized semantic tableaux for well formed $\mathcal{ALC}$ KBs.

Then $\Sigma \models_2 (A \sqcup B)(a)$, but $\Sigma \not\models_4 (A \sqcup B)(a)$. As a consequence, together with Proposition 31 we have

Proposition 32 *In $\mathcal{ALC}$, let Σ is a well formed KB. Then for all concept names A in Σ_T and for all individuals a occurring in Σ_F, $\Sigma \models_4 A(a)$ iff not $AcyclicSat_{DL}(\mathsf{T}\Sigma \cup \{\mathsf{NT}A(a)\})$.* $\dashv$

Now, given a well formed KB Σ, define

$$Ext(\Sigma) \quad = \quad \{A(a) \colon A := C \in \Sigma_T, a \text{ occurs in } \Sigma_F, \Sigma \models_4 A(a)\}. \tag{C.47}$$

$Ext(\Sigma)$ can be determined through Proposition 32. We extend the definition to sets S of signed formulae as follows:

$$Ext(S) \quad = \quad \{\mathsf{T}A(a) \colon \mathsf{T}A := C \in S, a \text{ occurs in } S, S \cup \{\mathsf{NT}A(a)\} \text{ not satisfiable}\}. \tag{C.48}$$

$Ext(S)$ can be determined through Proposition 32 as well. The following algorithm in Table C.17 computes the four-valued completions of a well formed set S of signed formulae.

The analogue of Proposition 29 holds.

Proposition 33 *In $\mathcal{ALC}$, let Σ be a well formed KB and let $A(a)$ be an assertion such that A is a primitive concept. Then $\Sigma \models_4 A(a)$ iff a occurs in Σ_F and for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in AcyclicCompl_{DL}(\mathsf{T}\Sigma)$, $\mathcal{I}$ satisfies $A(a)$.* $\dashv$

Just notice that the set of four-valued canonical models build of four-valued completions $\tilde{\mathcal{S}} \in AcyclicCompl_{DL}(S)$ is a subset of the set of four-valued canonical models build on four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(S)$, *i.e.*

Proposition 34 *In $\mathcal{ALC}$, let S be a well formed set of signed formulae. Then for all $\tilde{\mathcal{S}}_1 \in AcyclicCompl_{DL}(S)$ there is $\tilde{\mathcal{S}}_2 \in Compl_{DL}(S)$ such that $\tilde{\mathcal{S}}_1^{\mathsf{T}} = \tilde{\mathcal{S}}_2^{\mathsf{T}}$.* $\dashv$

As a consequence, a minor set of completions are generated.

Algorithm 9 ($AcyclicSat_{DL}(S)$)
Consider the set of rules in Table C.15. $AcyclicSat_{DL}(S)$ starts from the root labelled S and applies the following steps:

1. Select a not closed branch ϕ. If there is no such branch then return $false$ and exit.
2. If ϕ is not yet AB-completed then expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed. Update ϕ to the resulting branch;
3. If ϕ is AB-completed then
   (a) if for *some* signed formula $\alpha^{\exists}$ in S^{ϕ}, $Sat_{DL}(S^{\phi}(\alpha^{\exists}) \cup S^{\phi}(\Rightarrow)) = false$ holds, where $S^{\phi}(\alpha^{\exists})$ has not yet been tested with respect to ϕ, then close ϕ and go to Step 1. Otherwise,
   (b) select a signed formula of type β, neither being of type $\beta^{\forall}$ nor of type $\beta^{\Rightarrow}$, which is not yet fulfilled in the branch;
   (c) apply rule (PB) and go to Step 1. ■

Table C.16: Algorithm $AcyclicSat_{DL}(S)$ for $\mathcal{ALC}$.

Example 37 Consider $S = \{\mathsf{T}A \Rightarrow C, \mathsf{T}C \Rightarrow D, \mathsf{T}A(a), \mathsf{T}B(b)\}$. Then $Ext(S) = \emptyset$. Moreover, $AcyclicCompl_{DL}(S) = \{\tilde{\mathcal{S}}\}$, where $\tilde{\mathcal{S}}^{\mathsf{T}} = \{\mathsf{T}A(a), \mathsf{T}B(b), \mathsf{T}C(a), \mathsf{T}D(a)\}$. On the other hand, $Compl_{DL}(S) = \{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2, \tilde{\mathcal{S}}_3\}$, where $\tilde{\mathcal{S}}_1^{\mathsf{T}} = \tilde{\mathcal{S}}^{\mathsf{T}}$, $\tilde{\mathcal{S}}_2^{\mathsf{T}} = \{\mathsf{T}A(a),\ \mathsf{T}B(b), \mathsf{T}C(a), \mathsf{T}D(a), \mathsf{T}C(b),\ \mathsf{T}D(b)\}$, $\tilde{\mathcal{S}}_3^{\mathsf{T}} = \{\mathsf{T}A(a), \mathsf{T}B(b), \mathsf{T}C(a), \mathsf{T}D(a), \mathsf{T}A(b), \mathsf{T}C(b), \mathsf{T}D(b)\}$. ■

Example 38 Consider Σ defined as

$$\Sigma \ = \ \{A{:=}C, B{:=}D, (C \sqcup D)(a)\}.$$

Let $S = \mathsf{T}\Sigma$. It follows that $Ext(S) = \emptyset$. Now, $AcyclicCompl_{DL}(S)$ applies rule (PB) to $(C \sqcup D)(a)$ creating

$$\begin{array}{rcl} S_1 & = & S \cup \{\mathsf{T}C(a)\}, \\ S_2 & = & S \cup \{\mathsf{NT}C(a), \mathsf{T}D(a)\}. \end{array}$$

Now, $Ext(S_1) = \{\mathsf{T}A(a)\}$ and $Ext(S_2) = \{\mathsf{T}B(a)\}$, respectively. As a consequence, we have $AcyclicCompl_{DL}(S) = \{\tilde{\mathcal{S}}', \tilde{\mathcal{S}}''\}$, where

$$\begin{array}{rcl} \tilde{\mathcal{S}}' & = & \{\mathsf{T}A(a), \mathsf{T}C(a)\}, \\ \tilde{\mathcal{S}}'' & = & \{\mathsf{NT}C(a), \mathsf{T}B(a), \mathsf{T}D(a)\}. \end{array}$$

On the other hand, $Compl_{DL}(S) = \{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2, \tilde{\mathcal{S}}_3\}$, where

$$\begin{array}{rcl} \tilde{\mathcal{S}}_1 & = & \{\mathsf{T}A(a), \mathsf{T}C(a), \mathsf{NT}B(a), \mathsf{NT}D(a)\}, \\ \tilde{\mathcal{S}}_2 & = & \{\mathsf{T}A(a), \mathsf{T}C(a), \mathsf{T}B(a), \mathsf{T}D(a)\}, \\ \tilde{\mathcal{S}}_3 & = & \{\mathsf{NT}A(a), \mathsf{NT}C(a), \mathsf{T}B(a), \mathsf{T}D(a)\}. \end{array}$$

Note that $\tilde{\mathcal{S}}'^{\mathsf{T}} = \tilde{\mathcal{S}}_1^{\mathsf{T}}$. ■

Algorithm 10 ($AcyclicCompl_{DL}(S)$)
Essentially, the procedure proceeds in a similar way as $AcyclicSat_{DL}$.

1. Select a not closed branch ϕ. If there is no such ϕ then $AcyclicCompl_{DL}(S) := \emptyset$ and exit.
2. Let $Ext(S^\phi)$ as in Equation (C.48). Assign $S^\phi := S^\phi \cup Ext(S^\phi)$.
3. If ϕ is not yet AB-completed then expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed. Update ϕ to the resulting branch;
4. If ϕ is AB-completed then
   (a) for *all* $\alpha^\exists \in S^\phi$, let $\tilde{\mathcal{S}}(\alpha^\exists) = AcyclicCompl_{DL}(S^\phi(\alpha^\exists) \cup S^\phi(\Rightarrow))$, where $S^\phi(\alpha^\exists)$ has not yet been considered with respect to ϕ. If for *some* $\alpha^\exists \in S^\phi$, $\tilde{\mathcal{S}}(\alpha^\exists) = \emptyset$ then close ϕ and go to Step 1. Otherwise,
   (b) select a signed formula of type β, neither of type $\beta^\forall$ nor of type $\beta^\Rightarrow$ and not yet fulfilled in the branch;
   (c) apply rule (PB) and go to Step 1.
5. Let T be the generated deduction tree. Note that T at this point is not closed.
6. For all not closed branches ϕ from S to a leaf in T do:
   (a) let $\Psi = \{\alpha_1^\exists, \ldots, \alpha_m^\exists\}$ be the set of top level $\alpha_i^\exists \in S^\phi$ for which $S^\phi(\alpha_i^\exists)$ has been considered and let x_i be the new variables introduced.
   (b) for each tuple $(\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_m)$ such that $\tilde{\mathcal{S}}_j \in \tilde{\mathcal{S}}(\alpha_j^\exists)$, let $S(\phi) = S^\phi \cup \bigcup_{1 \le j \le m} \tilde{\mathcal{S}}_j$. Let
   $$S^{\mathsf{T}} = \{\mathsf{T}A \in S(\phi)\} \cup \{\mathsf{T}R_i(w_i, x_i) : w_i = Ind(\alpha_i^\exists), R_i = Role(\alpha_i^\exists)\},$$
   and let $S^{\mathsf{NT}} = \{\mathsf{NT}A \in S(\phi)\}$;
   (c) define the $\tilde{\mathcal{S}}$ according to (C.38);
   (d) set $AcyclicCompl_{DL}(S) := AcyclicCompl_{DL}(S) \cup \{\tilde{\mathcal{S}}\}$. ■

Table C.17: Algorithm $AcyclicCompl_{DL}(S)$ for $\mathcal{ALC}$.

C.4.4 About type B and two-valued semantics

Just notice that $Sat_{DL}(S)$ can easily adapted to the case of type B semantics. In fact, by observing that

$$(\exists R.C)(a), (\forall R.D)(a) \not\models_4^B (\exists R.(C \sqcap D))(a) \tag{C.49}$$
$$(\forall R.C)(a), (\forall R.D)(a) \models_4^B (\forall R.(C \sqcap D))(a) \tag{C.50}$$
$$(\exists R.(C \sqcap D))(a) \models_4^B (\exists R.C)(a) \tag{C.51}$$

it follows that the definition od the set $S(\alpha^\exists)$ is no longer valid as *e.g.* if S is $\{\mathsf{T}(\exists R.C)(a), \mathsf{T}(\forall R.D)(a)\}$ then $S(\mathsf{T}(\exists R.C)(a))$ is $\{\mathsf{T}C(b), \mathsf{T}D(b)\}$ which is incorrect in the light of (C.49). Relations (C.49), (C.50) and (C.51) tell us that in the case of type B semantics, we should consider another definition for $S(\alpha^\exists)$. The definition in this case is quite simple. In fact, given a set S of signed assertions, let $S_B(\alpha^\exists)$ be such that

$$\begin{array}{rcl} S_B(\mathsf{T}(\exists R.C)(a)) & = & \{\mathsf{T}C(x)\} \cup \\ & & \{\mathsf{NT}D(x) : \mathsf{NT}(\exists R.D)(a) \in S\} \end{array} \tag{C.52}$$

$$S_B(\mathsf{NT}(\forall R.C)(a)) = \{\mathsf{NT}C(x)\} \cup \{\mathsf{T}D(x) : \mathsf{T}(\forall R.D)(a) \in S\} \quad \text{(C.53)}$$

It is worth nothing that (*C*.52) and (*C*.53) reflect the properties (*C*.50) and (*C*.51), respectively. Moreover, $S_B(\alpha^{\exists}) \subseteq S(\alpha^{\exists})$. By considering the same algorithm $Sat_{DL}(S)$ where at Step 3*a* the set $S(\alpha^{\exists})$ is replaced by $S_B(\alpha^{\exists})$, we obtain straightforwardly

Proposition 35 *Let S be a set of signed propositions in $\mathcal{ALC}$ and Sat_{DL} modified in such a way that at Step* 3*a the set $S(\alpha^{\exists})$ is replaced by $S_B(\alpha^{\exists})$. Then $Sat_{DL}(S)$ iff S is satisfiable with respect to type B semantics.* ⊣

Since $S_B(\alpha^{\exists}) \subseteq S(\alpha^{\exists})$ holds, $\preceq_4^B \subseteq \preceq_4^A$ and $\models_4^B \subseteq \models_4^A$ follows, *i.e.* type B semantics is weaker than type A semantics.

Finally, Sat_{DL} can be extended to two-valued reasoning as well. Simply add a table for signed type α formulae as for the propositional case $\mathcal{L}$:

α	α_1	α_2
$\mathsf{T}\neg C(a)$	$\mathsf{NT}C(a)$	$\mathsf{NT}C(a)$
$\mathsf{NT}\neg C(a)$	$\mathsf{T}C(a)$	$\mathsf{T}C(a)$

Practical consideration: Just notice that by case analysis it can be verified that any set of signed assertion of type $\mathsf{T}A$ is always four-valued satisfiable. Similarly, any set of signed assertion of type $\mathsf{NT}A$ is four-valued satisfiable[4]. Now, consider the case where $S(\mathsf{T}(\exists R.C)(a))$ has no signed assertion of type $\mathsf{NT}A$ (the case where $S(\mathsf{NT}(\forall R.C)(a))$ has no signed assertion of type $\mathsf{T}A$ is analogous). Therefore, $S(\mathsf{T}(\exists R.C)(a))$ is trivially satisfiable. This means that (in the four-valued case) in Step 3*a* of Sat_{DL} we can leave out those cases, and thus, improving the decision procedure. For instance, if S is (*e.g.* $n = 10.000$).

$$\bigcup_{i=1}^{n} \{\mathsf{T}(\exists R_i.C_i)(a)\}$$

then the general $Sat_{DL}(S)$ procedure performs n unnecessary recursive calls $Sat_{DL}(\{\mathsf{T}C_i(b)\})$ which could be very time consuming (depending on $|C_i|$), which can be avoided.

More generally, since any set S which contains only signed assertions of type $\mathsf{T}A$ (or only of type $\mathsf{NT}A$) is four-valued satisfiable, we can improve the Sat_{DL} procedure by adding the following first performed step to it:

- if we are considering four-valued satisfiability, then if S does not contain a pair of assertions of type $\mathsf{T}A_1$ and $\mathsf{NT}A_2$, then return *true* and exit.

C.4.5 *Remarks on computational complexity

In this section we address some computational complexity issues of four-valued reasoning in DLs. It is not the scope of this thesis to give an exhaustive presentation.

At first we will *not* consider specialisations.

[4]Obviously, this property does not hold for the two-valued case, *e.g.* $\{\mathsf{T}(A \sqcap \neg A)(a)\}$ is not two-valued satisfiable.

C.4.5.1 The case without specialisations

In the following, $\mathcal{AL}$ is the DL with syntax rule[5]

$$C, D \rightarrow \top | \bot | A | \neg A | C \sqcap D | \forall R.C | \exists R$$

$\mathcal{ALE}$ is $\mathcal{AL}$ plus qualified existential quantification $\exists R.C$. $\mathcal{ALE}_1^-$ is the DL with syntax rule

$$C, D \rightarrow A | \neg A | C \sqcap D | \forall R.C | \exists R.C$$

and $\mathcal{ALE}_2^-$ is $\mathcal{ALE}_1^-$ plus unqualified existential quantification $\exists R$.

It is well known that the (in)coherence problem is PSPACE-complete for two-valued $\mathcal{ALC}$ [249].

The main characteristic behaviour can be shown by means of the following example. Consider *e.g.* the concept C:

$$\begin{array}{l} (\exists R_1.A_1^1) \sqcap (\exists R_1.A_2^1) \sqcap \\ (\forall R_1.((\exists R_2.A_1^2) \sqcap (\exists R_2.A_2^2) \sqcap \\ (\forall R_2.((\exists R_3.A_1^3) \sqcap (\exists R_3.A_2^3) \sqcap \\ \vdots \\ (\forall R_n.((\exists R_n.A_1^n) \sqcap (\exists R_n.A_2^n))))) \end{array} \tag{C.54}$$

The dimension of C is bounded by n. Unfortunately, it can be verified that the execution of $Sat_{DL}(\{\mathsf{T}C(a)\})$ generates 2^n recursive calls to Sat_{DL}. There are two calls at each level $1 \leq i \leq n$ of the role R_i.

By adopting the same reduction technique as for two-valued $\mathcal{ALC}$, the following proposition can be shown. We recall that the PSPACE-hardness of the (two-valued) coherence problem has been shown by means of a reduction of the validity problem for quantified boolean formulae [249] and briefly presented here.

A *literal* is a nonzero integer. A *clause* is a nonempty finite set c of literals such that $l \in c$ implies $-l \notin c$. A *prefix* from m to n, where m and n are positive integers such that $m \leq n$, is a sequence $(Q_m)m)(Q_{m+1})m+1)\ldots(Q_n)n)$, where each Q_i is $\forall$ or $\exists$. A *quantified boolean formula* is a pair $P.M$, where, for some n, P is a prefix from 1 to n and M is a finite nonempty set of clauses containing only literals between n and n.

Let P be a prefix from m to n. A *P-assignment* is a mapping $\{m, m+1, \ldots, n\} \rightarrow \{t, f\}$. An assignment α *satisfies* a literal l if $\alpha(l) = t$ if l is positive and $\alpha(-l) = f$ if l is negative. An assignment *satisfies* a clause if it satisfies at least one literal of the clause.

Let P be a prefix from m to n. A set $\mathbf{A}$ of P-assignment is *canonical* for P if it satisfies the following conditions:

1. $\mathbf{A}$ is nonempty;
2. if $P = (\exists m)P'$, then all assignments of $\mathbf{A}$ agree on m and, if P' is nonempty, $\{\alpha_{|\{m+1,\ldots,n\}} : \alpha \in \mathbf{A}\}$ is canonical for P';
3. if $P = (\forall m)P'$, then
   (a) $\mathbf{A}$ contains an assignment that satisfies m and, if P' is nonempty, $\{\alpha_{|\{m+1,\ldots,n\}} : \alpha \in \mathbf{A}, \alpha(m) = t\}$ is canonical for P';

[5] The 2-valued extension of $\exists R$ is $\{d \in \Delta^{\mathcal{I}} : \exists d' \in \Delta^{\mathcal{I}}$ such that $(d, d') \in R^{\mathcal{I}}\}$. We use $\bot$ as a macro for $\neg\top$.

(b) $\mathbf{A}$ contains an assignment that satisfies $-m$ and, if P' is nonempty, $\{\alpha_{|\{m+1,\ldots,n\}} : \alpha \in \mathbf{A}, \alpha(m) = f\}$ is canonical for P'.

A quantified boolean formula $P.M$ is *valid* if there exists a set $\mathbf{A}$ of P-assignment that is canonical for P such that every assignment in $\mathbf{A}$ satisfies every clause of M. For instance, $\forall x \exists y.(x \vee \neg y) \wedge \neg y$ is a valid boolean formula, where $P = \forall x \exists y$ and $M = (x \vee \neg y) \wedge \neg y$[6].

It is well known that deciding the validity of quantified boolean formulae is a PSPACE-complete problem [129].

Proposition 36 *The instance checking problem and the subsumption checking problem with respect to type A semantics are PSPACE-complete for $\mathcal{ALC}$.* ⊣

Proof: Let $P.M$ be a quantified boolean formula. Consider the reduction of $P.M$ into the $\mathcal{ALC}$ concept $[P.M] = [P]^0 \sqcap [C_1]^0 \sqcap \ldots \sqcap [C_n]^0$, as in [249]:

- $[P]$ is defined inductively by the equations

$$\begin{array}{l} [(\exists m)P] = ((\exists R.A) \sqcup (\exists R.\neg A)) \sqcap \forall R.[P] \\ [(\forall m)P] = ((\exists R.A) \sqcap (\exists R.\neg A)) \sqcap \forall R.[P] \\ [(\exists m)] = (\exists R.A) \sqcup (\exists R.\neg A) \\ [(\forall m)] = (\exists R.A) \sqcap (\exists R.\neg A) \end{array} \tag{C.55}$$

- $[C]^m$ is defined inductively by the equations

$$\begin{array}{l} [lC]^m = \forall R.[lC]^{m+1} \text{ if } |l| > m \\ [mC]^m = A \sqcup [lC]^m \\ [-mC]^m = \neg A \sqcup [lC]^m \\ [l]^m = \forall R.[l]^{m+1} \text{ if } |l| > m \\ [m]^m = A \\ [-m]^m = \neg A \end{array} \tag{C.56}$$

It has been shown that $P.M$ is a valid boolean formula iff $[P.M]$ is coherent. Now, consider $[P] \sqcap \neg D$, where $D = D_1 \sqcup \ldots \sqcup D_1$ and D_i is the NNF of $\neg[C_i]^0$. It follows that $[P.M]$ is incoherent iff $[P] \sqcap \neg D$ is incoherent iff $\{[P](a)\} \models_2 D(a)$, where a is an individual. Therefore, by completeness it follows that $\{[P](a)\} \models_2 D(a)$ iff not $Sat_{DL}(\{\mathsf{T}[P](a), \mathsf{NT}D(a)\})$, two-valued. It can be verified that any two-valued deduction $Sat_{DL}(\{\mathsf{T}[P](a), \mathsf{NT}D(a)\})$ does not rely on signed formulae of type $\mathsf{T}(\neg A)(a)$ or $\mathsf{NT}(\neg A)(a)$. Hence, two-valued $Sat_{DL}(\{\mathsf{T}[P](a), \mathsf{NT}D(a)\})$ iff four-valued $Sat_{DL}(\{\mathsf{T}[P](a), \mathsf{NT}D(a)\})$. Therefore, $[P.M]$ is incoherent iff $\{[P](a)\} \models_4^A D(a)$. As a consequence, instance checking is a PSPACE-hard problem. Analogously, $[P.M]$ is incoherent iff $[P] \preceq_4^A D$. Hence, subsumption checking is a PSPACE-hard problem too.

Furthermore, it is easily verified that Sat_{DL} runs in polynomial space (by induction on the number n of occurrences of conditioned signed fuzzy formulae of type $\alpha^{\exists} \in S$). Therefore, instance checking is a PSPACE-complete problem with respect to type A semantics. Since subsumption can be reduced to instance checking, it follows that the subsumption problem is PSPACE-complete too. Q.E.D.

This result shows that modus ponens on roles is effectively sufficient to get PSPACE-hardness. Moreover, from the proof it follows that the proposition holds for $\mathcal{ALC}$ without the negation constructor ($\neg$) too.

[6]For readability, we write $(x \vee \neg y) \wedge \neg y$ in place of $\{\{x, \neg y\}, \{\neg y\}\}$.

By using the same reduction as described in [98], where PSPACE-completeness of the instance checking problem with respect to two-valued $\mathcal{ALE}$ is shown, it can be verified that Proposition 36 holds for the language $\mathcal{ALE}_2^-$ too.

Proposition 37 *The instance checking problem with respect to type A semantics is PSPACE-complete for $\mathcal{ALE}_2^-$.* ⊣

Similarly, by proceeding as in [92], where NP-completeness of the two-valued subsumption problem is shown for $\mathcal{ALE}$ and $\mathcal{FLE}^-$, it can be proven that

Proposition 38 *Subsumption checking with respect to type A semantics is a NP-complete problem for $\mathcal{ALE}_2^-$.* ⊣

Hence, with respect to type A semantics instance checking is strictly more difficult than subsumption for $\mathcal{ALE}_2^-$. As a consequence, the instance checking problem in $\mathcal{ALE}_2^-$ cannot be reduced in polynomial time into an subsumption problem in $\mathcal{ALE}_2^-$ and, thus, we cannot use any subsumption algorithm for $\mathcal{ALE}_2^-$ in order to solve the instance checking problem. This fact give us evidence that structural algorithms cannot work for instance checking in $\mathcal{ALE}_2^-$.

By using type B semantics, modus ponens on roles is not a valid inference rule. As a consequence it is easily verified that the satisfiability of S with respect to type B semantics can be done in non-deterministic polynomial time. In fact, the only branches in $Sat_{DL}(S)$ arises from the $\sqcup$ connectives. Hence, just non-deterministically choose the right one. It is worth noting that this does not work in the context of type A semantics (see (*C*.54). Since propositional logic is a sub language of $\mathcal{ALC}$ and since it is well known that propositional tautological entailment is a coNP-complete problem, it follows that:

Proposition 39 *The instance checking problem and the subsumption problem with respect to type B semantics are coNP-complete for $\mathcal{ALC}$.* ⊣

As noted in Section 7.4.1.2, type B semantics has a weaker entailment relation than type A semantics. On the other hand, from a computational point of view the computational complexity switches from coNP to PSPACE.

Since in certain circumstances reasoning by cases does not hold, the instance checking problem can be even in P. Suppose that Σ is formed out by $\mathcal{AL}$ assertions, and consider an assertion $C(a)$ which is an $\mathcal{ALE}_1^-$ assertion.

As shown in [98], the instance checking problem with respect to two-valued semantics is coNP-hard in this case. Whereas, it can be shown that

Proposition 40 *Let Σ be formed out by $\mathcal{AL}$ assertions and let C be an $\mathcal{ALE}_1^-$ concept. Then checking whether $\Sigma \models_4^A C(a)$ and checking whether $\Sigma \models_4^B C(a)$ can be done in polynomial time.* ⊣

Proof: Consider the algorithm Sat_{DL} where $S(\alpha^\exists)$ is replaced with $S^-(\alpha^\exists)$. The set $S^-(\alpha^\exists)$ is defined to be empty for all $\alpha^\exists$ except

$$S^-(\mathsf{NT}(\forall R.C)(a)) = S_B(\mathsf{NT}(\forall R.C)(a)) \tag{C.57}$$

Consider $S_{\Sigma,C(a)} = \mathsf{T}\Sigma \cup \{\mathsf{NT}C(a)\}$ and run $Sat_{DL}(S_{\Sigma,C(a)})$. By simple case analysis on the connective, it can be shown that each generated set S' during the computation has exactly one signed assertion of the form $\mathsf{NT}A$, *i.e.* each S' is of the form $\{\mathsf{T}A_1, \mathsf{T}A_2, \ldots, \mathsf{T}A_n, \mathsf{NT}A_{n+1}\}$. Moreover, n, and thus $|S'|$ is polynomially bounded by $|S_{\Sigma,C(a)}|$. Finally, the number of generated sets S' is polynomially bounded by $|S_{\Sigma,C(a)}|$. It is worth noting that this property is based on the connectives which could appear in S' and on the form of S'. For instance, in a "or" situation like $S = \{\mathsf{NT}((A_1 \sqcap A_2) \sqcup (A_3 \sqcap A_4) \sqcup (A_4 \sqcap A_5))(a)\}$, we get $S' = \{\mathsf{NT}(A_1 \sqcap A_2)(a), \mathsf{NT}(A_3 \sqcap A_4)(a), \mathsf{NT}(A_4 \sqcap A_5)(a)\}$. Thereafter, by branching (PB application) we obtain 2^3 sets, where $3 =$ " number of signed assertions of type $\mathsf{NT}A$", which is exponential with respect to the size of S.

From the polynomially bounding of set dimensions and number of generated sets, it follows that $Sat_{DL}(S_{\Sigma,C(a)})$ runs in polynomial time with respect to $|S_{\Sigma,C(a)}|$, and thus with respect to $|\Sigma| + |C(a)|$. Q.E.D.

It is worth noting that, from [98] it follows that whenever unqualified existential concepts of type $\exists R$ are allowed to occur in C, the proposition does not hold for type A semantics (again, we rely on the relation $\exists R \sqcup \forall R.A \equiv_4^A \top$).

Proposition 41 *Checking whether $\Sigma \models_4^A C(a)$, where Σ is formed out by $\mathcal{AL}$ assertions and C is an $\mathcal{ALE}_2^-$ concept, is a coNP-hard problem.* ⊣

Proposition 40 is certainly a positive result, as the $\exists R.C$ construct is very useful in the query language. This result suggests a detailed investigation about the computational complexity of the instance checking problem in those cases where the knowledge base language and the query language are different. Just notice that the idea of distinguishing between the knowledge base language and the query language is certainly not new in the DL area [71, 180] and has given positive results.

C.4.5.2 The case with specialisations

Let us briefly recall the complexity of reasoning in $\mathcal{ALC}$ in presence of specialisations of type $C \Rightarrow D$.

Most results about reasoning in presence of specialisations can be found in [69, 74]. The complexity depends on the form of specialisations and of the form of the set of he specialisations considered. In the case of $\mathcal{ALC}$ specialisations, just notice that [69]

Proposition 42 (Buchheit *et al.* 1993) *Let Σ be a set of $\mathcal{ALC}$ assertions and specialisations of type $C \Rightarrow D$, and let A be an assertion and let $C \Rightarrow D$ be a specialisation. Then determining whether $\Sigma \models_4 A$ and checking whether $\Sigma \models_4 C \Rightarrow D$ are EXPTIME-hard problems.* ⊣

C.5 Deciding entailment in HORN-$\mathcal{ALC}$

As we did in Section C.3, our aim is to investigate decision procedures which are a combination of SLD-derivation and and our decision procedure for $\mathcal{ALC}$, Sat_{DL}. Since, w.r.t. two-valued semantics, HORN-$\mathcal{ALC}$ is a subset of CARIN [186, 187, 188], we will concentrate our attention to the four-valued case only. The reader may consult [186, 187, 188] for the two-valued case.

For the sake of readability, we briefly describe the notions of SLD-derivation and SLD-refutation (see *e.g.* [194]) in HORN-$\mathcal{ALC}$.

Let G be a goal of the form $\leftarrow P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$. Let E be a horn rule R of the form $P'(\vec{Y}) \leftarrow P'_1(\vec{Y}_1), \ldots, P'_m(\vec{Y}_m)$ or a horn fact $P'(\vec{a})$. Let $P_i(\vec{X}_i)$ the *selected atom* in G.

1. If there is a *most general unifier* (mgu) θ of $P_i(\vec{X}_i)$ and $P'(\vec{Y})$ then the *resolvent* of the goal G and the horn rule R using θ is the goal

$$\leftarrow (P_1(\vec{X}_1), \ldots, P_{i-1}(\vec{X}_{i-1}), P'_1(\vec{Y}_1), \ldots, P'_m(\vec{Y}_m), P_{i+1}(\vec{X}_{i+1}), \ldots, P_n(\vec{X}_n))\theta;$$

2. If there is a most general unifier θ of $P_i(\vec{X}_i)$ and $P'(\vec{a})$ then the *resolvent* of the goal G and the horn fact $P'(\vec{a})$ using θ is the goal

$$\leftarrow (P_1(\vec{X}_1), \ldots, P_{i-1}(\vec{X}_{i-1}), P_{i+1}(\vec{X}_{i+1}), \ldots, P_n(\vec{X}_n))\theta.$$

A *SLD-derivation* for a goal G_0 in a HORN-$\mathcal{ALC}$ KB Σ is a derivation constituted by:

1. a sequence of horn rules and horn facts $E_1, \ldots, E_n$ in Σ;
2. a sequence of mgu's $\theta_1, \ldots, \theta_n$;
3. a sequence of goals $G_0, \ldots, G_n$ such that for each $i \in \{0, \ldots, n-1\}$, G_{i+1} is the resolvent of G_i and E_{i+1} using θ_{i+1}.

A SLD-derivation may terminate with an empty goal in which case the derivation is a *SLD-refutation*.

Following the standard terminology, an answer θ to a query Q w.r.t. a HORN-$\mathcal{ALC}$ KB Σ is called a *computed answer* if the goal associated with $Q\theta$ has a SLD-refutation in Σ, *i.e.* if θ is the restriction to the variables in Q of the composition $\theta_1\theta_2 \ldots \theta_n$, where $\theta_1, \ldots, \theta_n$ are the mgu's used in the SLD-refutation. The *success set* of Q w.r.t. Σ is the set

$$SuccessSet(\Sigma, Q) \quad = \quad \{\theta : \theta \text{ computed answer of } Q \text{ w.r.t. } \Sigma\}. \tag{C.58}$$

Let Σ be horn KB and let Q be a query. It is well known that in two valued-semantics, every computed answer θ of Q w.r.t. Σ is a correct answer (correctness), and for every correct answer θ of Q w.r.t. Σ there is a computed answer θ_1 of Q w.r.t. Σ and a substitution θ_2 such that $\theta = \theta_1\theta_2$ (completeness).

From Proposition 5 it follows immediately correctness and completeness in case of four-valued semantics.

Proposition 43 *Let Σ be a horn* HORN-$\mathcal{ALC}$ *KB and let Q be a query. In four-valued semantics,*

1. *every computed answer θ of Q w.r.t. Σ is a correct answer (correctness),* i.e.

$$SuccessSet(\Sigma, Q) \subseteq AnswerSet(\Sigma, Q);$$

2. *for every correct answer θ of Q w.r.t. Σ there is a computed answer θ_1 of Q w.r.t. Σ and a substitution θ_2 such that $\theta = \theta_1\theta_2$ (completeness).* $\dashv$

What about the case in which Σ is a generic HORN-$\mathcal{ALC}$ KB? Of course, by (7.82) it follows that if there is a SLD-refutation for goal $\leftarrow P_1(\vec{X}_1), \ldots, P_n(\vec{X}_n)$ in Σ then $\Sigma \models_4 \exists \vec{X}.P_1(\vec{X}_1) \wedge \ldots \wedge P_n(\vec{X}_n)$. The converse in obviously not true. An example can easily be obtained from Example 30 in case HORN-$\mathcal{L}$.

Example 39 Consider the following safe and non-recursive HORN-$\mathcal{ALC}$ KB:

$$\Sigma \;=\; \{C(X) \leftarrow A(X), C(X) \leftarrow B(X), (A \sqcup B)(a)\}$$

and the query $C(a)$. It is quite easy to see that $\Sigma \models_4 C(a)$. But, there is no SLD refutation for goal $\leftarrow C(a)$ in Σ. The only two SLD derivations end with goal G_1 and G_2, which are $\leftarrow A(a)$ and $\leftarrow B(a)$, respectively. ■

Unfortunately, in HORN-$\mathcal{ALC}$ the situation is more complex that in HORN-$\mathcal{L}$. In particular, the analogues of Proposition 25 and Proposition 26 do not hold in the general case. In the following we will assume that each query Q_i is of the form $\exists \vec{X}.P_{i_1}(\vec{X}_{i_1}) \wedge \ldots \wedge P_{i_{k_i}}(\vec{X}_{i_{k_i}})$ and each associated goal G_{Q_i} is of the form $\leftarrow P_{i_1}(\vec{X}_{i_1}), \ldots, P_{i_{k_i}}(\vec{X}_{i_{k_i}})$.

C.5.1 The case without specialisations

At first, we will consider HORN-$\mathcal{ALC}$ KBs Σ without terminology, *i.e.* $\Sigma_T = \emptyset$. This is the easiest one. In this case, both the analogue of Proposition 25 and Proposition 26 hold in HORN-$\mathcal{ALC}$.

Proposition 44 *Let Σ be a safe and non recursive* HORN-$\mathcal{ALC}$ *KB such that $\Sigma_T = \emptyset$. Let Q be a query. Then $\Sigma \models_4 Q$ iff either*

1. *there is a SLD-refutation for goal G_Q in Σ; or*
2. *there are $n \geq 1$ SLD-derivations for goal G_Q in Σ ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, such that for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}}$ in the set $Compl_{DL}(\mathsf{T}\Sigma_F)$, $\mathcal{I}$ does satisfy some Q_i.* ⊣

Some of the methods described in Page 179 can be applied for determining whether $\Sigma \models_4 Q$, where Σ is a safe and non recursive HORN-$\mathcal{ALC}$ KB with empty terminology.

Method 2: Collect $n \geq 1$ SLD-derivations of G_Q in Σ, ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, until (i) there is an empty goal; or, (ii) for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F)$, $\mathcal{I}$ does satisfy some Q_i. In the worst case we have to compute all SLD-derivations.

Method 4: Compute $Compl_{DL}(\mathsf{T}\Sigma_F)$. Determine whether for all $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F)$, G_Q has a SLD-refutation in $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$, *i.e.* whether $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R \models_4 Q$. Note that $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$ is a horn KB.

It is worth noting that Method 1 and Method 3, described in Page 179 do not work. The reason relies on the fact that the query Q_i should be an $\mathcal{ALC}$ expressions, which in our case is not true.

Example 40 Consider

$$\Sigma \quad = \quad \{(\exists R.A)(a), B(x) \leftarrow R(X,Y), A(Y)\}.$$

Then it follows that $\Sigma \models_4 B(a)$. Now, a SLD-derivation of $\leftarrow B(a)$ ends up with the goal $\leftarrow R(a,Y), A(Y)$, which is the associate of the query $Q = \exists Y.R(a,Y) \wedge A(Y)$. Q is not an $\mathcal{ALC}$ expression, and thus, neither Method 1 nor Method 3 can be applied. Notice that the unique four-valued completion $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F)$ is

$$\tilde{\mathcal{S}} \quad = \quad \{\mathsf{T}R(a,x), \mathsf{T}A(x)\}.$$

Therefore, the four-valued KB completion of $\tilde{\mathcal{S}}$ is

$$\Sigma_{\tilde{\mathcal{S}}} = \Sigma^+_{\tilde{\mathcal{S}}} \quad = \quad \{R(a,x), A(x)\}.$$

Now, let Σ_1 be $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$, *i.e.*

$$\Sigma_1 \quad = \quad \{R(a,x), A(x), B(X) \leftarrow R(X,Y), A(Y)\}.$$

It is easily verified that $\leftarrow B(a)$ has a SLD-refutation in Σ_1, according to Method 4. ■

C.5.2 The case with specialisations

Suppose now that Σ is a HORN-$\mathcal{ALC}$ KB with non empty terminology, *i.e.* $\Sigma_T \neq \emptyset$. The decision procedure we present is simply an adaption of the one presented for CARIN [186, 188]. As pointed out in [186, 188], the notion o blocked variable has to extended, as the following example illustrates.

Example 41 Let Σ be the following HORN-$\mathcal{ALC}$ KB:

$$\Sigma = \{A(a), A \Rightarrow \exists R.A\}.$$

It follows that $\mathsf{T}\Sigma$ is the set of signed formulae S of Example 35.

$$\mathsf{T}\Sigma = \{\mathsf{T}A(a), \mathsf{T}A \Rightarrow \exists R.A\}.$$

For Example 35 it follows that:

1. the unique four-valued completion, $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma)$, is given by

$$\tilde{\mathcal{S}} \quad = \quad \{\mathsf{T}A(a), \mathsf{T}A(x_1), \mathsf{T}A(x_2), TR(a,x_1), \mathsf{T}R(x_1,x_2), \mathsf{T}R(x_2,x_2)\}.$$

2. the four-valued canonical model $\mathcal{I}$ of $\tilde{\mathcal{S}}$ is such that $\Delta^{\mathcal{I}} = \{a, x_1, x_2\}$ with (i) $w^{\mathcal{I}} = w$, for all $w \in \Delta^{\mathcal{I}}$; and (ii)

$$A^{\mathcal{I}}(a) = A^{\mathcal{I}}(x_1) = A^{\mathcal{I}}(x_2) = \{t\}$$
$$R^{\mathcal{I}}(a,x_1) = R^{\mathcal{I}}(x_1,x_2) = R^{\mathcal{I}}(x_2,x_2) = \{t\}.$$

As we already pointed out, $\mathcal{I}$ satisfies $\mathsf{T}\Sigma$, and thus, Σ. The subtle point in the procedure $Compl_{DL}$ is that when building completions, fillers to the variables *have* to be assigned. In our case, a filler for role R has been assigned to x_2. In doing so cycles were introduced in the completion $\tilde{\mathcal{S}}$, *i.e.* in the four-valued canonical model $\mathcal{I}$, which do not exists in every model of Σ. For instance, if the query Q is

$$Q = \exists X.R(X, X),$$

it would be satisfied in the canonical model of the unique completion $\tilde{\mathcal{S}}$, even though it is not entailed by Σ, *i.e.* $\Sigma \not\models_4 Q$.

■

In [186, 187, 188] a solution to the above problem has been given in terms of an extension of the blocking condition that depend on the query, and that guarantees that the resulting canonical models of completions are sufficient for checking the entailment of the query. We now describe the method presented in [186, 187, 188]. The blocking condition will refine the previous one by considering the values of the $\sigma_{\mathsf{T}}, \sigma_{\mathsf{NT}}$ functions not only of the variables itself, but also of its neighbors.

Let S be a set of signed formulae. We say that an object v is an *R-successor* of an object w if $\mathsf{T}R(w, v) \in S$. We will say that an object v is a *direct successor* of an object w if v is an R-successor for some role R. The *successor* relationship denotes the transitive closure of the direct-successor relation.

The *n-tree* of a variable x is the tree that includes the variable x and its successors, whose distance from x is at most n direct successor arcs. We denote the set of variables in the n-tree of x by $V_n(x)$. Two variables x, y appearing in a set of signed formulae S are said to be *n-tree equivalent in S* if there is an isomorphism $\psi: V_n(x) \to V_n(y)$, such that for every $x_1, x_2 \in V_n(x)$, $\mathsf{T}R(x_1, x_2) \in S$ iff $\mathsf{T}R(\psi(x_1), \psi(x_2)) \in S$, and x_1 and $\psi(x_1)$ are concept equivalent in S. Intuitively, two variables are n-tree equivalent if the trees of depth n of which they are roots are isomorphic.

We denote with D_Q the maximum number of literals in a query Q. Let $S_1, S_2, \ldots, S_n$ be a sequence of sets of signed formulae. Consider,

$$S = \bigcup_{i=1,2,\ldots,n} S_i.$$

and $\alpha^{\exists} \in S_n$. Suppose that $Ind(\alpha^{\exists}) = x$. We will say that $\alpha^{\exists}$ is *tree blocked* at S_n (also, x is a *tree blocked variable* in S) if

1. x is the leaf of a D_Q-tree rooted in a variable x_1;
2. there is a variable x_2, such that x_1 is a successor of x_2, and x_1 and x_2 are D_Q-tree equivalent in S, and x_1 is not a node in the D_Q-tree of x_2.

The *tree witness* of the tree blocked variable x is $\psi(x)$, where ψ is the isomorphism between the D_Q-trees of x_1 and x_2.

In order to build completions correctly, we replace in the procedure $Compl_{DL}$ the notions blocked and witness with tree blocked and tree witness, respectively. Moreover, to ensure that we can correctly detect tree blocked variables, we apply the rules in a breadth first fashion. That is, we only apply a rule to a variable whose distance from an individual in S is n direct-successor arcs, if no rule is applicable to a variable with distance less than n.

Example 42 Consider Example 41, *i.e.* let

$$\Sigma = \{A(a), A \Rightarrow \exists R.A\}.$$

and let

$$Q = \exists X.R(X, X).$$

It follows that $D_Q = 1$. The execution of procedure $Compl_{DL}(\mathsf{T}\Sigma)$ generates the following set S^{T}:

$$\begin{array}{rcl} S^{\mathsf{T}} & = & \{\mathsf{T}A(a), \mathsf{T}(\exists R.A)(a), \mathsf{T}R(a, x_1), \\ & & \mathsf{T}A(x_1), \mathsf{T}(\exists R.A)(x_1), \mathsf{T}R(x_1, x_2), \\ & & \mathsf{T}A(x_2), \mathsf{T}(\exists R.A)(x_2), \mathsf{T}R(x_2, x_3), \\ & & \mathsf{T}A(x_3), \mathsf{T}(\exists R.A)(x_3), \mathsf{T}R(x_3, x_4), \\ & & \mathsf{T}A(x_4), \mathsf{T}(\exists R.A)(x_4), \mathsf{T}R(x_4, x_3)\} \end{array}$$

Note that S^{T} contains $\mathsf{T}R(x_4, x_3)$. The motivation is as follows. The generated tree rooted x_1, through the successor relation, is the linear sequence $x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4$. Now, variable x_4 is tree blocked, because x_4 is the leaf of a 1-tree rooted x_3, and the two 1-tree rooted in x_1 and x_3 are tree equivalent. Now, the witness of x_4 is $\psi(x_4) = x_2$. Since, $\mathsf{T}R(x_2, x_3) \in S^{\mathsf{T}}$, $\mathsf{T}R(x_4, x_3)$ has to be in S^{T}, too.

It follows that:

1. the unique four-valued completion, $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma)$, is given by

$$\begin{array}{rcl} \tilde{\mathcal{S}} & = & \{\mathsf{T}A(a), \mathsf{T}A(x_1), \mathsf{T}A(x_2), \mathsf{T}A(x_3), \mathsf{T}A(x_4), \\ & & TR(a, x_1), \mathsf{T}R(x_1, x_2), \mathsf{T}R(x_2, x_3), \mathsf{T}R(x_3, x_4), \mathsf{T}R(x_4, x_3)\}. \end{array}$$

2. the four-valued canonical model $\mathcal{I}$ of $\tilde{\mathcal{S}}$ is such that $\Delta^{\mathcal{I}} = \{a, x_1, x_2, x_3, x_4\}$ with (i) $w^{\mathcal{I}} = w$, for all $w \in \Delta^{\mathcal{I}}$; and (ii)

$$\begin{array}{l} A^{\mathcal{I}}(a) = A^{\mathcal{I}}(x_i) = \{t\}, \text{ for } 1 \leq i \leq 4 \\ R^{\mathcal{I}}(a, x_1) = R^{\mathcal{I}}(x_i, x_{i+1}) = \{t\}, \text{ for } 1 \leq i \leq 3 \\ R^{\mathcal{I}}(x_4, x_3) = \{t\}. \end{array}$$

$\mathcal{I}$ satisfies $\mathsf{T}\Sigma$, and thus, Σ, but $\mathcal{I}$ does not satisfy Q.

■

The following proposition holds.

Proposition 45 *Let Σ be a safe and non recursive* HORN-$\mathcal{ALC}$ *KB. Let Q be a query. Then $\Sigma \models_4 Q$ iff either*

1. *there is a SLD-refutation for goal G_Q in Σ; or*
2. *there are $n \geq 1$ SLD-derivations for goal G_Q in Σ ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, such that for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_{Q_1,\ldots,Q_n})$, $\mathcal{I}$ does satisfy some Q_i, where $\Sigma_{Q_1,\ldots,Q_n}$ is the set of specialisations $\{A \Rightarrow A{:}\, A$ concept appearing in $Q_1, \ldots, Q_n\}$.* ⊣

It is worth noting that a specialisation $\mathsf{T}A \Rightarrow A \in S$ enforces that in every completion of S, either $\mathsf{T}A(w) \in S$ or $\mathsf{NT}A(w) \in S$, for every object w. This is required in order to use our Sat_{DL} based method for determining entailment as opposed to determining satisfiability. Furthermore, notice that completions depend on the query $Q_1, \ldots, Q_n$ since both (i) the set $\Sigma_{Q_1,\ldots,Q_n}$ has to be considered; and (ii) the tree blocking condition depend on the number of literals appearing in $Q_1, \ldots, Q_n$. As a consequence of Proposition 45 above, Method 4 cannot be applied, and thus, the only method for determining whether $\Sigma \models_4 \exists \vec{X}.P_1(\vec{X}_1) \wedge \ldots \wedge P_n(\vec{X}_n)$, where Σ is a safe and non recursive HORN-$\mathcal{ALC}$ KB, is

Method 2: Collect $n \geq 1$ SLD-derivations of G_Q in Σ, ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, until (i) there is an empty goal; or, (ii) for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_{Q_1,\ldots,Q_n})$, $\mathcal{I}$ does satisfy some Q_i. In the worst case we have to compute all SLD-derivations.

Example 43 Consider the following simple HORN-$\mathcal{ALC}$KB (see Example 17).

$$\begin{aligned}\Sigma \;=\; \{&\mathsf{SuperVisor} := \mathsf{Professor} \sqcap \exists \mathsf{Teaching}.\mathsf{AdvancedCourse},\\ &\mathsf{MayDoThesis}(X,Y,Z) \leftarrow \mathsf{Student}(X), \mathsf{SuperVisor}(Y), \mathsf{Expert}(Y,Z),\\ &\mathsf{Professor}(\mathsf{paul}), \mathsf{Teaching}(\mathsf{paul},\mathsf{ai}), \mathsf{AdvancedCourse}(\mathsf{ai}),\\ &\mathsf{Student}(\mathsf{tom}), \mathsf{Expert}(\mathsf{paul},\mathsf{kr})\}\end{aligned}$$

Consider the query

$$Q \;=\; \exists Y, Z.\mathsf{MayDoThesis}(\mathsf{tom}, Y, Z)$$

i.e. we are asking whether tom is doing a thesis.

It is easily verified that there is a SLD-derivation for goal $\leftarrow \mathsf{MayDoThesis}(\mathsf{tom}, Y, Z)$ in Σ, ending up with goal $\leftarrow \mathsf{SuperVisor}(\mathsf{paul})$ and substitution $\theta = \{Y/\mathsf{paul}, Z/\mathsf{kr}\}$.

Let Q_1 be the query $\mathsf{SuperVisor}(\mathsf{paul})$. Hence, $D_{Q_1} = 1$ and $\Sigma_{Q_1} = \{\mathsf{SuperVisor} \Rightarrow \mathsf{SuperVisor}\}$. Now, it can easily be verified that in every completion $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_{Q_1})$, $\mathsf{TSuperVisor}(\mathsf{paul}) \in \tilde{\mathcal{S}}$ hold. Therefore, every four-valued canonical model $\mathcal{I}$ of four-valued completions in $Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_{Q_1})$ satisfies Q_1, and thus,

$$\Sigma \models_4 \exists Y, Z.\mathsf{MayDoThesis}(\mathsf{tom}, Y, Z)$$

and the computed answer is $\theta = \{Y/\mathsf{paul}, Z/\mathsf{kr}\}$.

■

Example 44 Let Σ be the following HORN-$\mathcal{ALC}$ KB:

$$\begin{aligned}\Sigma \;=\; \{&\forall R.\neg A \Rightarrow B,\\ &D(a),\\ &C(X) \leftarrow B(X),\\ &C(X) \leftarrow R(X,Y), A(Y)\}\end{aligned}$$

We show that $\Sigma \models_2 C(a)$. So, consider the goal $\leftarrow C(a)$. It is easily verified that there are two SLD-derivations ending up with goal $\leftarrow B(a)$ and $\leftarrow R(a,Y), A(Y)$, respectively. Let Q_1

and Q_2 be $B(a)$ and $\exists Y.R(a,Y) \wedge A(Y)$, respectively. Let $\tilde{Q}$ be $Q_1 \vee Q_2$. Therefore, $D_{\tilde{Q}} = 3$ and Σ_{Q_1,Q_2} is $\{B \Rightarrow B, A \Rightarrow A\}$. It can be verified that in all two-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_{Q_1,Q_2})$, either $\mathsf{T}B(a) \in \tilde{\mathcal{S}}$ or $\mathsf{T}R(a,x), \mathsf{T}A(x) \in \tilde{\mathcal{S}}$. Therefore, every two-valued canonical model $\mathcal{I}$ of two-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_{Q_1,Q_2})$ satisfies either Q_1 or Q_2, and thus, $\Sigma \models_2 C(a)$.

It is worth noting that $\Sigma \not\models_4 C(a)$. In fact, it can easily be verified that in all four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_{Q_1,Q_2})$, either $\mathsf{T}B(a) \in \tilde{\mathcal{S}}$ or $\mathsf{T}R(a,x), \mathsf{NT}\neg A(x) \in \tilde{\mathcal{S}}$. From the latter case we obtain a four-valued canonical model satisfying neither Q_1 nor Q_2. The reason relies on the fact that no top concept exists in our four-valued semantics. $\dashv$

C.5.3 The case with role-safe rules

We have seen that in determining whether $\Sigma \models_4 Q$, where Q is query and Σ is a HORN-$\mathcal{ALC}$ KB, we have to rely on Method 2: first we collect SLD-derivations, and successively compute canonical models. As pointed out in Section C.3, from an implementation, practical and performance point of view, the possibility of using Method 4 is preferable: we first compute all completions off-line and then test whether there are SLD-derivations.

In this section we show than in certain useful circumstances, Method 4 can be applied. The result reported here is an easy adaption of the one presented in [187, 188].

We show that for an useful sublanguage of HORN-$\mathcal{ALC}$ a sound and complete inference is possible for recursive rules and arbitrary specialisations. This is obtained by restricting the horn rules in the knowledge base to be *role-safe*, as defined below. Role-safe rules restrict the way in which variables can appear in roles in the rules. This restriction is similar in spirit to the *safety* condition on datalog KB's with order constraints (*i.e.* $=, \leq, <, \neq$), which is widely employed in deductive databases [271].

A rule R is said to be *role-safe* if for every role atom of the form $R(X,Y)$ in the body of R, where R is a role and X, Y are horn variables, then either X or Y appear in an atom of an ordinary predicate in the body of R. For instance, $\mathsf{Rich}(X) \leftarrow \mathsf{HasMoney}(X,Y), Y \geq \$100.000.000$ is a role-safe rule, where $\mathsf{HasMoney}(X,Y)$ is the role expression and Y occurs in the ordinary predicate $Y \geq \$100.000.000$. On the other hand, $\mathsf{Poor}(X) \leftarrow \mathsf{HasCar}(X,Y), \mathsf{Small}(Y)$ is not a role-safe rule (Small is concept and HasCar is a role).

A query Q is *role-safe* if for every role atom of the form $R(X,Y)$ in Q, where R is a role and X, Y are horn variables, then either X or Y appear in an atom of an ordinary predicate in Q. For instance, $\exists X, Y.\mathsf{HasMoney}(X,Y), Y \geq \$100.000.000$ is a role-safe query, whereas $\exists X.R(X,X)$ is not a role-safe query.

The idea behind role-safe rules and queries is *to avoid to need* the tuples created through the blocking condition. From [187, 188] and from the fact that no top concept exists in four-valued semantics, it follows that

Proposition 46 *Let Σ be a safe and role-safe* HORN-$\mathcal{ALC}$ *KB. Let Q be a role-safe query. Then $\Sigma \models_4 Q$ iff either*

1. *there is a SLD-refutation for goal G_Q in Σ; or*
2. *there are $n \geq 1$ SLD-derivations for goal G_Q in Σ ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, such that for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\tilde{\Sigma})$, $\mathcal{I}$ does satisfy some Q_i,*

where $\tilde{\Sigma}$ *is the set of specialisations* $\{A \Rightarrow A$: *A concept appearing in* $\Sigma_R\}$ *and* $Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\tilde{\Sigma})$ *is computed according to the standard blocking condition.* ⊣

It is worth noting that Proposition 46 says us that the completions does not depend on the query. Therefore, Method 4 can be applied. The following are the possible methods for determining whether $\Sigma \models_4 Q$, where Σ is a safe and role-safe HORN-$\mathcal{ALC}$ KB and Q is role-safe query.

Method 2: Collect $n \geq 1$ SLD-derivations of G_Q in Σ, ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, until (i) there is an empty goal; or, (ii) for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\tilde{\Sigma})$, $\mathcal{I}$ does satisfy some Q_i. In the worst case we have to compute all SLD-derivations.

Method 4: Compute $Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\tilde{\Sigma})$. Determine whether for all $\tilde{\mathcal{S}} \in Compl_{DL}(\mathsf{T}\Sigma_F \cup \mathsf{T}\Sigma_T \cup \mathsf{T}\tilde{\Sigma})$, G_Q has a SLD-refutation in $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$, *i.e.* whether $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R \models_4 Q$. Note that $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$ is a horn KB.

C.5.4 The case of well formed KBs

We have seen that in determining whether $\Sigma \models_4 Q$, where Q is a role-safe query and Σ is a role-safe HORN-$\mathcal{ALC}$ KB, we can rely on Method 2 or Method 4. Now, in the case of well formed KBs, by considering Method 2 and Method 4, a further improvement can be given: (i) the set $\tilde{\Sigma}$ of specialisations of the form $A \Rightarrow A$ is not needed; and (ii) we can compute the completions by relying on $AcyclicCompl_{DL}$, and thus, a smaller set of completions is generated (see Proposition 34).

From Proposition 33 and Proposition 34 (in particular, from the fact that no top concept exists in four-valued semantics), it follows that

Proposition 47 *Let* Σ *be a well formed* HORN-$\mathcal{ALC}$ *KB. Let* Q *be a query. Then* $\Sigma \models_4 Q$ *iff either*

1. *there is a SLD-refutation for goal* G_Q *in* Σ*; or*
2. *there are* $n \geq 1$ *SLD-derivations of goal* G_Q *in* Σ *ending with goals* $G_{Q_1}, \ldots, G_{Q_n}$*, such that for all four-valued canonical models* $\mathcal{I}$ *of four-valued completions* $\tilde{\mathcal{S}}$ *belonging to* $AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$*,* $\mathcal{I}$ *does satisfy some* Q_i*.* ⊣

As for the role-safe case, Proposition 47 says us that the completions does not depend on the query. Therefore, Method 4 can be applied. Moreover from Proposition 34 it follows that a minor number of completions are needed.

The following are the possible methods for determining whether $\Sigma \models_4 Q$, where Σ is a well formed HORN-$\mathcal{ALC}$ KB and Q is a query.

Method 2: Collect $n \geq 1$ SLD-derivations of G_Q in Σ, ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, until (i) there is an empty goal; or, (ii) for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$, $\mathcal{I}$ does satisfy some Q_i. In the worst case we have to compute all SLD-derivations.

Method 4: Compute $AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$. Determine whether for all $\tilde{\mathcal{S}}$ belonging to $AcyclicCompl_{DL}(\ \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$, G_Q has a SLD-refutation in $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$, *i.e.* whether $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R \models_4 Q$. Note that $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$ is a horn KB.

Example 45 Consider the following HORN-$\mathcal{ALC}$ KB which is an extension of the KB in Example 38.

$$\begin{array}{rcl} \Sigma & = & \{E(X) \leftarrow A(X), \\ & & E(X) \leftarrow B(X), \\ & & A := C, B := D, \\ & & (C \sqcup D)(a)\}. \end{array}$$

Let Q be the query

$$Q = \exists X.E(X).$$

Let us prove that $\Sigma \models_4 Q$.

In Example 38 we have seen that $AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F) = \{\tilde{\mathcal{S}}', \tilde{\mathcal{S}}''\}$, where

$$\begin{array}{rcl} \tilde{\mathcal{S}}' & = & \{\mathsf{T}C(a), \mathsf{T}A(a)\}, \\ \tilde{\mathcal{S}}'' & = & \{\mathsf{NT}C(a), \mathsf{T}D(a), \mathsf{T}B(a)\}. \end{array}$$

It follows that

$$\begin{array}{rcl} \Sigma^+_{\tilde{\mathcal{S}}'} & = & \{C(a), A(a)\}, \\ \Sigma^+_{\tilde{\mathcal{S}}''} & = & \{D(a), B(a)\}. \end{array}$$

Now, it is easily verified that G_Q has a SLD-refutation in $\Sigma^+_{\tilde{\mathcal{S}}'} \cup \Sigma_R$ and in $\Sigma^+_{\tilde{\mathcal{S}}''} \cup \Sigma_R$, confirming $\Sigma \models_4 Q$. ■

Just noticing that, as for HORN-$\mathcal{L}$, the computational complexity of *e.g.* Method 4 depends on the number of completions computed as on their dimension. In fact, the complexity is bounded by

$$\sum_i |\Sigma^+_{\tilde{\mathcal{S}}_i} \cup \Sigma_R||Q|, \tag{C.59}$$

where $\tilde{\mathcal{S}}_i$ is a completion in $AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$. It would be a good choice to restrict the DL to those cases for which $AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$ generates one completion (see *e.g.* [95] for some examples).

Appendix D

Fuzzy decision algorithms

D.1 Deciding entailment in $\mathcal{L}^f$

Note that Proposition 8 give us a simple method for determining whether $(A \geq n) \mid\!\approx_4 (B \geq m)$, by means of an entailment test. In fact,

$$(A \geq n) \mid\!\approx_4 (B \geq m) \text{ iff } A \models_4 B, \text{ and } n \geq m.$$

As a consequence, an algorithm similar to Levesque's $Lev(\Sigma, A)$ can be build, as shown in Table D.1.

Algorithm 11 (*FuzzyLev*$(\Sigma, (A \geq n))$)
In order to check whether $\Sigma \mid\!\approx_4 (A \geq n)$, perform

1. put Σ and A into equivalent CNFs. Call the results of this transformation Σ_{CNF} and A_{CNF}, respectively;
2. verify whether for each conjunct A^j_{CNF} of A_{CNF} there is a fuzzy proposition $(B_{CNF} \geq k)$ in Σ_{CNF} and a conjunct B^i_{CNF} of B_{CNF} such that $k \geq n$ and $B^i_{CNF} \subseteq A^j_{CNF}$, where B^i_{CNF} and A^j_{CNF} are seen as clauses. ■

Table D.1: Algorithm $FuzzyLev(\Sigma, (A \geq n))$.

Therefore,

Proposition 48 *Let $\Sigma \subseteq \mathcal{L}^f$ and $A \in \mathcal{L}$. Fuzzy entailment between Σ and $(A \geq n)$, both in CNF, can be verified in time $O(|\Sigma||A|)$.* ⊣

As we did for $\mathcal{L}$, we present an alternative decision procedure for determining $\Sigma \mid\!\approx_4 (A \geq n)$. The calculus is a straightforward extension of the procedure *Sat* for $\mathcal{L}$ (Algorithm 3). In fact, just consider *signed fuzzy propositions* (denoted by σ) of type $\mathsf{T}(A \geq n)$ and $\mathsf{NT}(A \geq n)$ with the obvious extension of the definition of satisfiability to signed fuzzy propositions. It can be easily verified that (see $(C.2)$ for case $\mathcal{L}$)

$$\Sigma \not\approx_4 (A \geq n) \text{ iff } \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq n)\} \text{ is not satisfiable} \tag{D.1}$$

where

$$\mathsf{T}\Sigma = \{\mathsf{T}(B \geq n) : (B \geq n) \in \Sigma\}$$

We will say that $\mathsf{T}(A \geq n)$ and $\mathsf{NT}(A \geq m)$ are *conjugated* whenever $n \geq m$. Just notice that a conjugate of a signed fuzzy proposition may be not unique, as there could be infinitely many. For instance, given $\mathsf{T}(A \geq n)$, then $\mathsf{NT}(A \geq m)$ is a conjugate, for all $m \leq n$. With σ^c we indicate a conjugate of σ, whereas with $\sigma^{c,max}$ we indicate the conjugate of σ obtained by exchanging the symbols T and NT in σ. Of course $\sigma^{c,max}$ is unique. For instance, if σ is $\mathsf{T}(A \geq .7)$ then $\sigma^c = \mathsf{NT}(A \geq .5)$ is a conjugate of σ, whereas $\sigma^{c,max}$ is then conjugate $\mathsf{NT}(A \geq .7)$.

With respect to the α and β tables for signed fuzzy propositions, we have the following Table D.2.

α	α_1	α_2
$\mathsf{T}(A \wedge B \geq n)$	$\mathsf{T}(A \geq n)$	$\mathsf{T}(B \geq n)$
$\mathsf{NT}(A \vee B \geq n)$	$\mathsf{NT}(A \geq n)$	$\mathsf{NT}(B \geq n)$

β	β_1	β_2
$\mathsf{T}(A \vee B \geq n)$	$\mathsf{T}(A \geq n)$	$\mathsf{T}(B \geq n)$
$\mathsf{NT}(A \wedge B \geq n)$	$\mathsf{NT}(A \geq n)$	$\mathsf{NT}(B \geq n)$

Table D.2: α and β table for $\mathcal{L}^f$.

The rules are quite similar to those described in Table C.3 and are shown in Table D.3.

$$(A)\ \frac{\alpha}{\alpha_1, \alpha_2} \qquad (B1)\ \frac{\beta, \beta_1^c}{\beta_2} \qquad (B2)\ \frac{\beta, \beta_2^c}{\beta_1}$$

$$(PB)\ \frac{\beta}{\beta_1 \mid \beta_1^{c,max}, \beta_2}$$

Table D.3: Semantic tableaux inference rules in $\mathcal{L}^f$.

Just notice that in the branching rule (PB) $\beta^{c,max}$ has been used.

By extending the *Sat* procedure (see Table C.4) to the fuzzy case, we obtain:

Proposition 49 *Let S be a set of signed fuzzy propositions in $\mathcal{L}^f$. Then $Sat(S)$ iff S is satisfiable.* ⊣

Proof: The proof consists in an adaption of proof of Proposition 18.

It can be easily verified that the rules (A), $(B1)$, $(B2)$ and (PB) are correct in the fuzzy case too, *i.e.* if ϕ is a branch then S^ϕ is satisfiable iff there is a branch ϕ' as the result of the application of a rule to ϕ such that $S^{\phi'}$ satisfiable.

$\Rightarrow$.) Suppose $Sat(S)$. Let T be the generated deduction tree and let ϕ be a completed not closed branch from S to a leaf in T. Such a branch has to exist, otherwise $Sat(S) = false$. Let

$$S^{\mathsf{T}} = \{\mathsf{T}(A \geq n) \in S^{\phi}\}, \tag{D.2}$$

$$S^{\mathsf{NT}} = \{\mathsf{NT}(A \geq n) \in S^{\phi}\}. \tag{D.3}$$

Of course, $S^{\phi} = S^{\mathsf{T}} \cup S^{\mathsf{NT}}$. Let $\mathcal{I}$ be a relation such that $|A|^t = \max\{n : \mathsf{T}(A \geq n) \in S^{\mathsf{T}}\}$, and $|A|^f = 0$. More precisely, for each propositional letter p, let

$$|p|^t = \max\{n : \mathsf{T}(p \geq n) \in S^{\mathsf{T}}\}, \text{ and}$$
$$|p|^f = \max\{n : \mathsf{T}(\neg p \geq n) \in S^{\mathsf{T}}\}.$$

Since, ϕ is completed and not closed, $\mathcal{I}$ is a four-valued fuzzy interpretation satisfying all $\sigma \in S^{\mathsf{T}}$, $\sigma \in S^{\mathsf{NT}}$ and, thus, $\mathcal{I}$ satisfies S^{ϕ}. As a consequence, $S \subseteq S^{\phi}$ is satisfiable.
$\Leftarrow$.) Suppose S is satisfiable. Let T be the generated completed tree. From the correctness of the rules it follows that there is a completed branch ϕ in T such that S^{ϕ} is satisfiable. Therefore, $Sat(S)$.

Q.E.D.

Observe that from an implementation point of view, it is convenient to avoid the generation of branches ϕ such that *e.g.* $\mathsf{T}(A \geq n), \mathsf{T}(A \geq m) \in S^{\phi}$ (where $m \leq n$) as $\mathsf{T}(A \geq m)$ can be thrown away.

Example 46 For instance, Figure D.1 is a closed deduction tree for

$$(A \geq .5), (B \vee C \geq .7) \not\approx_4 ((A \vee C) \wedge (B \vee C \vee D) \geq .5)$$

Notice, that the tree is the fuzzy analogue of the closed tree (see Figure C.2) for

$$A \wedge (B \vee C) \models_4 (A \vee C) \wedge (B \vee C \vee D),$$

confirming the close relation between entailment and fuzzy entailment.

■

From a computational point of view, fuzzy entailment and entailment are in the same complexity class.

Proposition 50 *Let $\Sigma \subseteq \mathcal{L}^f$, $A \in \mathcal{L}$ and $n > 0$. Then checking $\Sigma \not\approx_4 (A \geq n)$ is a coNP-complete problem.* ⊣

Proof: Since $A \models_4 B$ iff $(A \geq n) \not\approx_4 (B \geq n)$ (see Proposition 6 and Proposition 7), for all n, and deciding $A \models_4 B$ is a coNP-complete problem, coNP-hardness of the $\Sigma \not\approx_4 (A \geq n)$ decision problem follows. The following NP algorithm determines whether $\Sigma \not\approx_4 (A \geq n)$. Non-deterministically generate a deduction tree T for $\mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq n)\}$: $\Sigma \not\approx_4 (A \geq n)$ iff some branch in T is not closed. The depth of the branch is polynomially bounded by the input. Hence, deciding $\Sigma \not\approx_4 (A \geq n)$ is in NP. Therefore, deciding $\Sigma \not\approx_4 (A \geq n)$ is in coNP. Q.E.D.

It is quite obvious that algorithm *EasyEntail* (Algorithm 5) works for the fuzzy case (with obvious changes), too. As a consequence,

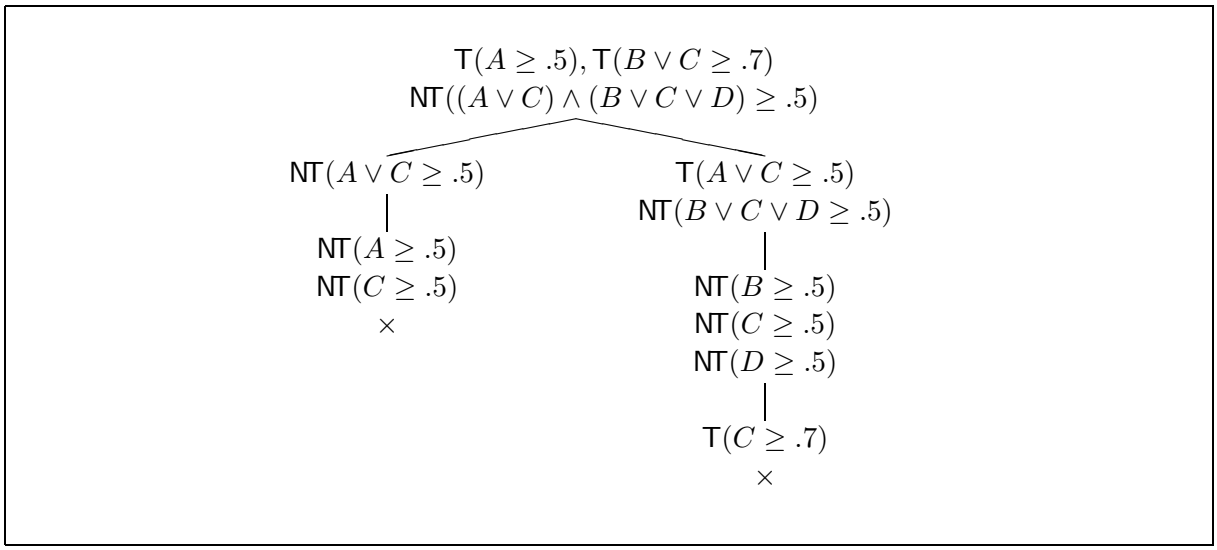


Figure D.1: Deduction tree for $(A \geq .5), (B \vee C \geq .7) \mid\!\approx_4 ((A \vee C) \wedge (B \vee C \vee D) \geq .5)$.

Proposition 51 *Let $\Sigma \subseteq \mathcal{L}^f$, $A \in \mathcal{L}$ and $n > 0$. If Σ and A are in CNF then checking whether $\Sigma \mid\!\approx_4 (A \geq n)$ requires time $O(|\Sigma||A|)$ by means of the procedure EasyEntail.* ⊣

Concerning the problem of determining $\Sigma \mid\!\approx_2 (A \geq n)$, just use the decision procedure described in [78]. The natural extension of our method to the two-valued fuzzy case leads to the same decision algorithm devised in [78].

In the following we extend the notions about "completions in $\mathcal{L}$" to the fuzzy case. Notice that we will restrict our attention to the four-valued case only.

Consider the proof of Proposition 49. Consider the sets S^{T} and S^{NT} build during the proof (see Equation $(D.2)$ and Equation $(D.3)$), as the result of running $Sat(S)$. The set

$$\begin{array}{rcl} \tilde{\mathcal{S}} & = & \{\mathsf{T}(p \geq n) \in S^{\mathsf{T}} : p \text{ letter}\} \cup \\ & & \{\mathsf{T}(\neg p \geq n) \in S^{\mathsf{T}} : p \text{ letter}\} \cup \\ & & \{\mathsf{NT}(p \geq n) \in S^{\mathsf{NT}} : p \text{ letter}\} \cup \\ & & \{\mathsf{NT}(\neg p \geq n) \in S^{\mathsf{NT}} : p \text{ letter}\} \end{array} \tag{D.4}$$

is called a *four-valued completion* of S. Moreover, we define

$$\tilde{\mathcal{S}}^{\mathsf{T}} = \{\mathsf{T}A \in \tilde{\mathcal{S}}\}, \tag{D.5}$$

$$\tilde{\mathcal{S}}^{\mathsf{NT}} = \{\mathsf{NT}A \in \tilde{\mathcal{S}}\}, \tag{D.6}$$

$$\tilde{\mathcal{S}}^{+} = \{\mathsf{T}(p \geq n) \in \tilde{\mathcal{S}} : p \text{ letter}\}. \tag{D.7}$$

Given a four-valued completion $\tilde{\mathcal{S}}$ of S, we define the following *four-valued completion KBs* of $\tilde{\mathcal{S}}$:

$$\Sigma_{\tilde{\mathcal{S}}} = \{A : \mathsf{T}A \in \tilde{\mathcal{S}}^{\mathsf{T}}\}, \tag{D.8}$$

$$\Sigma^{+}_{\tilde{\mathcal{S}}} = \{A : \mathsf{T}A \in \tilde{\mathcal{S}}^{+}\}. \tag{D.9}$$

Given a four-valued completion $\tilde{\mathcal{S}}$ of S, then a *four-valued canonical model* $\mathcal{I}$ of $\tilde{\mathcal{S}}$ is obtained as in proof of Proposition 49: for each propositional letter p, let

$$|p|^t = \max\{n : \mathsf{T}(p \geq n) \in \tilde{\mathcal{S}}^{\mathsf{T}}\}, \text{ and}$$
$$|p|^f = \max\{n : \mathsf{T}(\neg p \geq n) \in \tilde{\mathcal{S}}^{\mathsf{T}}\}.$$

It is easily verified that a canonical model of $\tilde{\mathcal{S}}$ is also a model of S. It is worth noticeable that given $\tilde{\mathcal{S}}$, the four-valued canonical model of $\tilde{\mathcal{S}}$ is unique.

The Algorithm 4 can easily be extended, allowing us to build all completions $\tilde{\mathcal{S}}$ of set of signed propositions S. In fact, just change Point 5b. with

1. 5b. define $\tilde{\mathcal{S}}$ according to $(D.4)$.

The algorithm is shown in Table D.4.

Algorithm 12 ($Completions_f(S)$)
Essentially, $Completions_f(S)$ proceeds in a similar way as $Sat(S)$ for $\mathcal{L}^f$.

1. select a branch ϕ which is not yet completed;
2. expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed;
3. if the resulting branch ϕ' is neither closed nor completed then
   (a) select a signed fuzzy proposition of type β which is not yet fulfilled in the branch;
   (b) apply rule $(PB)$and go to Step 1.

   otherwise, go to Step 1.
4. let T be the generated deduction tree. If all branches are closed, then set $Completions_f(S) := \emptyset$ and exit. Otherwise,
5. for all not closed branches ϕ from S to a leaf in T do
   (a) let $S^{\mathsf{T}} = \{\mathsf{T}A \in S^\phi\}$ and $S^{\mathsf{NT}} = \{\mathsf{NT}A \in S^\phi\}$;
   (b) define $\tilde{\mathcal{S}}$ according to $(D.4)$;
   (c) $Completions_f(S) := Completions_f(S) \cup \{\tilde{\mathcal{S}}\}$. ■

Table D.4: Algorithm $Completions_f(S)$ for $\mathcal{L}^f$.

The following proposition follows immediately.

Proposition 52 *In $\mathcal{L}^f$, let Σ be a KB and let A be a fuzzy proposition. Then $\Sigma \mid\!\approx_4 A$ iff for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in Completions_f(\mathsf{T}\Sigma)$, $\mathcal{I}$ satisfies A.* ⊣

Just notice here that there are finitely many four-valued canonical models.

A consequence of Proposition 52 is

Proposition 53 *In $\mathcal{L}^f$, let Σ be a KB and let A be a fuzzy proposition. Then*

1. *$\Sigma \not\approx_4 A$ iff for all four-valued completions $\tilde{\mathcal{S}} \in Completions_f(\mathsf{T}\Sigma)$, $\tilde{\mathcal{S}}^{\mathsf{T}} \cup \{\mathsf{NT}A\}$ is not satisfiable;*
2. *$\Sigma \not\approx_4 A$ iff for four-valued completions $\tilde{\mathcal{S}} \in Completions_f(\mathsf{T}\Sigma)$, $\Sigma_{\tilde{\mathcal{S}}} \not\approx_4 A$, where $\Sigma_{\tilde{\mathcal{S}}}$ is the four-valued completion KB of $\tilde{\mathcal{S}}$.* ⊣

Let us consider the following example illustrating the above properties.

Example 47 Let Σ be the following $\mathcal{L}^f$ KB.

$$\Sigma \;=\; \{(A \geq .5), (B \vee C \geq .7)\}.$$

It is easily verified that $Completions_f(\mathsf{T}\Sigma) = \{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2\}$, where

$$\begin{array}{rcl} \tilde{\mathcal{S}}_1 & = & \{\mathsf{T}(A \geq .5), \mathsf{T}(B \geq .7)\} \\ \tilde{\mathcal{S}}_2 & = & \{\mathsf{T}(A \geq .5), \mathsf{NT}(B \geq .7), \mathsf{T}(C \geq .7)\}. \end{array}$$

Their four-valued completion KBs are

$$\begin{array}{rcl} \Sigma_{\tilde{\mathcal{S}}_1} & = & \{(A \geq .5), (B \geq .7)\} \\ \Sigma_{\tilde{\mathcal{S}}_2} & = & \{(A \geq .5), (C \geq .7)\}. \end{array}$$

The four-valued canonical models of $\tilde{\mathcal{S}}_1$ and $\tilde{\mathcal{S}}_2$ are $\mathcal{I}_1$ and $\mathcal{I}_2$, respectively, where

1. $\mathcal{I}_1$ is such that $|p|^t = |p|^f = 0$, for all letter p, except that

$$\begin{array}{rcl} |A|^t & = & .5, \\ |B|^t & = & .7; \end{array}$$

2. $\mathcal{I}_2$ is such that $|p|^t = |p|^f = 0$, for all letter p, except that

$$\begin{array}{rcl} |A|^t & = & .5, \\ |C|^t & = & .7. \end{array}$$

Consider $Q = ((A \vee C) \wedge (B \vee C \vee D) \geq .5)$. We have seen in Example 46 that $\Sigma \not\approx_4 Q$. Now, it is easily verified that

1. both $\mathcal{I}_1$ and $\mathcal{I}_2$ satisfy Q, confirming Proposition 52;
2. both $\tilde{\mathcal{S}}_1 \cup \{\mathsf{NT}Q\}$ and $\tilde{\mathcal{S}}_2 \cup \{\mathsf{NT}Q\}$ are not satisfiable, confirming Proposition 53, Point 1.;
3. both $\Sigma_{\tilde{\mathcal{S}}_1} \not\approx_4 Q$ and $\Sigma_{\tilde{\mathcal{S}}_2} \not\approx_4 Q$ hold, confirming Proposition 53, Point 2.

■

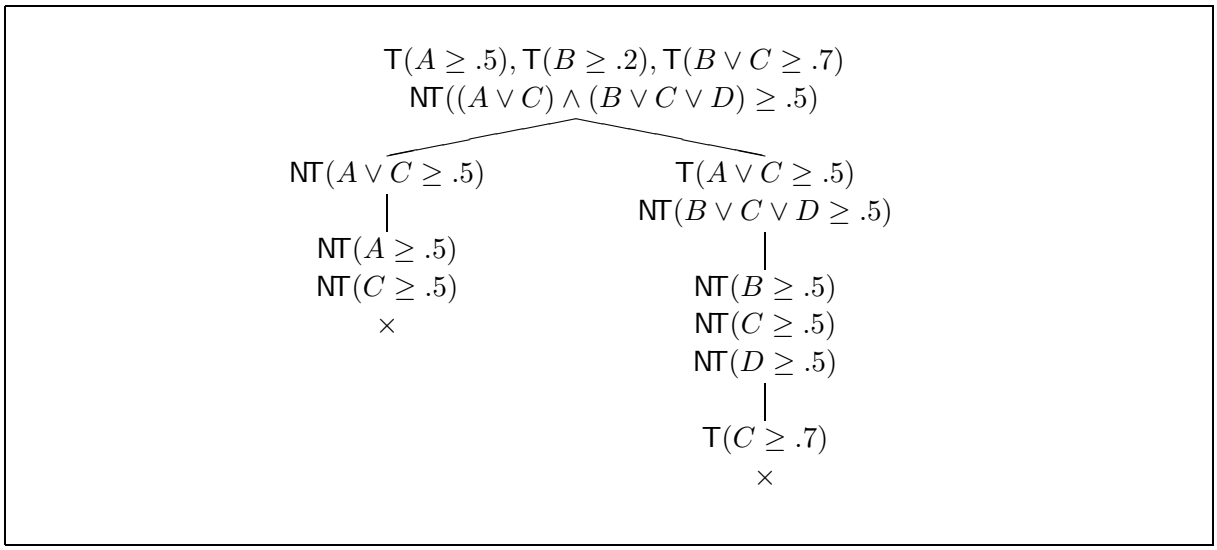


Figure D.2: Deduction tree for $(A \geq .5), (B \geq .2), (B \vee C \geq .7) \not\approx_4 ((A \vee C) \wedge (B \vee C \vee D) \geq .5)$.

Let us turn back to the case of determining $\Sigma \not\approx_4 (A \geq n)$. One can notice that, any successful refutation of $\mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq n)\}$ does not rely on those $(B \geq m) \in \Sigma$ such that $m < n$. For instance, consider a proof of

$$(A \geq .5), (B \geq .2), (B \vee C \geq .7) \not\approx_4 ((A \vee C) \wedge (B \vee C \vee D) \geq .5)$$

as shown in Figure D.2. The signed fuzzy proposition $\mathsf{T}(B \geq .2)$ is not needed (and will be not needed in any refutation).

Hence, given a KB Σ, consider the set

$$\Sigma^n = \{(B \geq m) \in \Sigma : m \geq n\}, \tag{D.10}$$

then the following properties can be proven. From the consideration above, it is easily verified that $\Sigma \not\approx_4 (A \geq n)$ implies $\Sigma^n \not\approx_4 (A \geq n)$. The other direction is obvious. Therefore,

Proposition 54 *Let $\Sigma \subseteq \mathcal{L}^f$, $A \in \mathcal{L}$ and $n > 0$. Then $\Sigma \not\approx_4 (A \geq n)$ iff $\Sigma^n \not\approx_4 (A \geq n)$.* ⊣

Proposition 55 below is the natural generalization of Proposition 6 and Proposition 7.

Proposition 55 *Let $\Sigma \subseteq \mathcal{L}^f$, $A \in \mathcal{L}$ and $n > 0$. Then $\Sigma^n \not\approx_4 (A \geq n)$ iff $\overline{\Sigma^n} \models_4 A$.* ⊣

Proof:

$\Rightarrow$.) Assume $\Sigma^n \not\approx_4 (A \geq n)$ and suppose to the contrary that $\overline{\Sigma^n} \not\models_4 A$. By proceeding as for Proposition 6, it can be shown that there is a four-valued interpretation $\tilde{\mathcal{I}}$ satisfying Σ^n, but not satisfying $(A \geq n)$, which is contrary to our assumption $\Sigma^n \not\approx_4 (A \geq n)$.

$\Leftarrow$.) Assume $\overline{\Sigma^n} \models_4 A$ and suppose to the contrary that $\Sigma^n \not\not\approx_4 (A \geq n)$. Hence, there is a fuzzy interpretation $\mathcal{I}$ satisfying Σ^n, but not satisfying $(A \geq n)$. Define $\tilde{\mathcal{I}}$ as in Proposition 7. For $(B \geq m) \in \Sigma^n$, $\mathcal{I}$ satisfies $(B \geq m)$ and, thus, $|B|^t \geq m \geq n$.

Therefore, $\tilde{\mathcal{I}}$ satisfies B, for all $B \in \overline{\Sigma^n}$. Moreover, from $|A|^t < n$, $t \notin A^{\tilde{\mathcal{I}}}$ follows, contrary to the assumption $\overline{\Sigma^n} \models_4 A$. Q.E.D.

Combining Proposition 54 and Proposition 55 we obtain

Proposition 56 *Let $\Sigma \subseteq \mathcal{L}^f$, $A \in \mathcal{L}$ and $n > 0$. Then $\Sigma \mid\!\approx_4 (A \geq n)$ iff $\overline{\Sigma^n} \models_4 A$.* ⊣

confirming that *fuzzy entailment inherits all the properties of entailment.*

By observing that $Maxdeg(\Sigma, A) \in \{0\} \cup N_\Sigma$, where

$$N_\Sigma = \{n : (B \geq n) \in \Sigma\}, \tag{D.11}$$

a simple consequence of Proposition 56 is Proposition 57 below.

Proposition 57 *Consider $\Sigma \subseteq \mathcal{L}^f$ and $A \in \mathcal{L}$. Then there is a $n > 0$ such that $\Sigma \mid\!\approx_4 (A \geq n)$ iff $\overline{\Sigma} \models_4 A$.* ⊣

Hence,

Proposition 58 *Let $\Sigma \subseteq \mathcal{L}^f$ and $A \in \mathcal{L}$. There is $n > 0$ such that $Maxdeg(\Sigma^n, A) = n$ iff $\overline{\Sigma} \models_4 A$.* ⊣

which generalises Proposition 8. Just note that Proposition 56 does not hold for $\mid\!\approx_2$. In fact, consider

$$\Sigma_1 = \{(p \geq .2), (\neg p \geq .3)\}, and$$
$$\Sigma_2 = \{(p \geq .2), (\neg p \geq .9)\}.$$

Hence,

$$\overline{\Sigma_1^{.1}} = \{p, \neg p\}, \text{ and}$$
$$\overline{\Sigma_2^{.3}} = \{\neg p\}.$$

It can easily be verified that

$$\overline{\Sigma_1^{.1}} \models_2 q, \text{ and}$$
$$\Sigma_1 \not\mid\!\approx_2 (q \geq .1),$$

whereas

$$\Sigma_2 \mid\!\approx_2 (q \geq .3), \text{ and}$$
$$\overline{\Sigma_2^{.3}} \not\models_2 q.$$

Proposition 56 gives us a way for computing $Maxdeg(\Sigma, A)$ in the style of the method proposed in [149] (see Table D.5). This is important, as computing $Maxdeg(\Sigma, A)$, is in fact the way to answer a query of type "to which degree is A (at least) true, given the facts in Σ ?" The method, which requires an algorithm for computing (crisp) entailment (*e.g. Sat* or *Lev*), is based on the observation that

$$Maxdeg(\Sigma, A) \in \{0\} \cup N_\Sigma, \text{ and that } \Sigma^m \supseteq \Sigma^n \text{ if } n \geq m.$$

Algorithm 13 ($Max(\Sigma, A)$)
Let Σ be a KB and A a proposition. Set $Min = 0$, $Max = 2$.

1. Pick $n \in N_\Sigma$ such that $Min < n < Max$. If there is no such n, then set $Maxdeg(\Sigma, A) := Min$ and exit.
2. Check if $\overline{\Sigma^n} \models_4 A$. If so, then set $Min = n$ and go to Step 1. If not so, then set $Max = n$ and go to Step 1.

■

Table D.5: Algorithm $Max(\Sigma, A)$.

Example 48 Consider Example 18, where Σ is

$$\Sigma = \{(p \geq .1), (p \wedge q \geq .5), (q \vee r \geq .6)\}$$

and A is $p \vee r$. Therefore,

$$N_\Sigma = \{.1, .5, .6\}$$

By binary search, let $n := .5$.

$$\overline{\Sigma^{.5}} = \{p \wedge q, q \vee r\} \models_4 A$$

holds. Thus, $Min := .5$; pick $n := .6$. Now,

$$\overline{\Sigma^{.6}} = \{q \vee r\} \not\models_4 A$$

holds. Thus, $Max := .6$. Since there is no $Min < n < Max$ such that $n \in N_\Sigma$, the procedure stops. Hence, $Maxdeg(\Sigma, A) = .5$ as expected. ■

By a binary search on N_Σ the value of $Maxdeg(\Sigma, A)$ can be determined in $\log |N_\Sigma|$ entailment tests. Since N_Σ is $O(\Sigma)$, if Σ and A are in CNF, the complexity of determining $Maxdeg(\Sigma, A)$ is $O(|A||\Sigma| \log |\Sigma|)$.

A drawback, which Algorithm 13 inherits is that checking entailment several times is generally not feasible from a practical point of view, as it could be exponential in time. In Section D.2.2 we will present a method where computing $Maxdeg(\Sigma, A)$ "corresponds" to performing the entailment test only *once*.

Finally, another immediate method for determining $Maxdeg(\Sigma, A)$ relies on Proposition 52 and is described below through Algorithm 14.

Example 49 Consider Example 48. Σ is

$$\Sigma = \{(p \geq .1), (p \wedge q \geq .5), (q \vee r \geq .6)\}$$

and A is $p \vee r$. $Completions_f(\mathsf{T}\Sigma) = \{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2\}$, where

Algorithm 14 ($MaxByModels(\Sigma, A)$)
Let Σ be a KB and A a proposition. Compute $Completions_f(\mathsf{T}\Sigma) = \{\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_k\}$. Set $Min = 0$.

1. For all $1 \leq i \leq k$, consider the four-valued canonical model $\mathcal{I}_i$ of $\tilde{\mathcal{S}}_i$, evaluate $|A|^t$ in $\mathcal{I}_i$ and let n_i be the value;
2. Let $MaxByModels(\Sigma, A) := \min\{n_1, \ldots, n_k\}$. ■

Table D.6: Algorithm $MaxByModels(\Sigma, A)$.

$$\begin{array}{rcl} \tilde{\mathcal{S}}_1 & = & \{\mathsf{T}(p \geq .5), \mathsf{T}(q \geq .6)\} \\ \tilde{\mathcal{S}}_2 & = & \{\mathsf{T}(p \geq .5), \mathsf{T}(q \geq .5), \mathsf{NT}(q \geq .6), \mathsf{T}(r \geq .6)\}. \end{array}$$

The four-valued canonical models of $\tilde{\mathcal{S}}_1$ and $\tilde{\mathcal{S}}_2$ are $\mathcal{I}_1$ and $\mathcal{I}_2$, respectively, where

1. $\mathcal{I}_1$ is such that $|s|^t = |s|^f = 0$, for all letter s, except that

$$\begin{array}{rcl} |p|^t & = & .5, \\ |q|^t & = & .6; \end{array}$$

2. $\mathcal{I}_2$ is such that $|s|^t = |ts|^f = 0$, for all letter s, except that

$$\begin{array}{rcl} |p|^t & = & .5, \\ |q|^t & = & .5, \\ |r|^t & = & .6. \end{array}$$

Therefore, $n_1 = .5$, whereas $n_2 = .6$, and thus

$$MaxByModels(\Sigma, A) = \min\{n_1, n_2\} = .5.$$

Hence, $Maxdeg(\Sigma, A) = .5$ as expected. ■

The trouble with the above approach is that we have to compute all completions which could be exponential in the size of the KB. But, the advantage of it is that we can compute the canonical models off-line once at all. This is especially useful in those cases in which we are performing a huge number of queries to the KB.

D.1.1 **Relations to possibilistic logic in $\mathcal{L}^f$

We have seen that fuzzy propositions deal with vague concepts, *i.e.* concepts *without* precise definitions (like "hot"). Vague concepts may have a degree of truth in $[0, 1]$. On the other hand, (see Section 8.1) there exists also uncertain propositions, *i.e.* propositions about concepts *with* clear and precise definitions (like "triangle"). Precise concepts have a truth in

$\{t, f\}$, but due to the lack of precision of the available information on these concepts, one can only estimate to what extent it is possible or necessary that they are true. The logics dealing with this kind of uncertainty have been called *Possibilistic Logics* [103, 106, 108, 109, 153, 178].

In this section we will show that there is a strict connection between our four-valued fuzzy logic and (necessity-valued) possibilistic logic. It is worth noting that we will not perform an in-depth study, as this is beyond the scope of this thesis. This can be seen as an interesting topic for future research. Our discussion, is fairly concerned to present a result establishing a connection between possibilistic logics and fuzzy logics.

In possibilistic logics, see [106] for an in-depth presentation, the expressions are of type $(A, \mathsf{P}n)$ and $(A, \mathsf{N}n)$, where A is a proposition. We will use ϕ as a metavariable for possibilistic proposition.

A weight $\mathsf{P}n$ (resp. $\mathsf{N}n$) attached to A models to what extent A is *possible* (resp. *necessarily*) true. For instance, $(\mathsf{Triangle}, \mathsf{P}.7)$ intends to assert that $\mathsf{Triangle}$ is *possibly true* at least with degree .7. On the other hand side, $(\mathsf{Polygon}, \mathsf{N}.4)$ intends to assert that $\mathsf{Polygon}$ is *necessarily true* at least with degree .4.

The semantics is given in terms of fuzzy sets of (two-valued) interpretations, *i.e.* to each propositional interpretation $\mathcal{I}$ a weight $\pi(\mathcal{I}) \in [0, 1]$ is assigned. Essentially, $\pi(\mathcal{I})$ determines the possibility that $\mathcal{I}$ is the real world[1].

Therefore, the possibility of a proposition A is then given by

$$\Pi(A) = \max\{\pi(\mathcal{I}) : \mathcal{I} \text{ satisfies } A\} \tag{D.12}$$

Hence, $\Pi(A) = .7$ means that "the maximal possibility of an interpretation satisfying A being the real world is .7", *i.e.* "A is possible with degree .7".

Complementarily, the necessity of a proposition is then given by

$$N(A) = 1 - \Pi(\neg A). \tag{D.13}$$

which can be rewritten as

$$N(A) = \min\{1 - \pi(\mathcal{I}) : \mathcal{I} \text{ does not satisfy } A\} \tag{D.14}$$

($\min \emptyset = 1$) A fuzzy set of interpretations π *satisfies* (*is a model of*) an expression of type $(A, \mathsf{P}n)$ (respectively $(A, \mathsf{N}n)$) if $\Pi(A) \geq n$ (respectively $N(A) \geq n$). Finally, a set of uncertain propositions Φ *entails* ϕ (denoted by $\Phi \models_2^{pos} \phi$) iff every model π of Φ is a model of ϕ. Just note that the following relations can easily be verified.

$$\begin{array}{lcl} \Pi(A \vee B) & = & \max\{\Pi(A), \Pi(B)\} \\ \Pi(A \wedge B) & \leq & \min\{\Pi(A), \Pi(B)\} \\ N(A \wedge B) & = & \min\{N(A), N(B)\} \\ N(A \vee B) & \geq & \max\{N(A), N(B)\} \end{array}$$

By definition we have

$$\Pi(A \vee \neg A) = \max\{\Pi(A), \Pi(\neg A)\} = 1$$

and, thus,

[1] π has to satisfy the constraint that for some $\mathcal{I}$, $\pi(\mathcal{I}) = 1$. This guarantees that there is at least one world which could be considered the real one.

$$\Pi(A) = 1 \text{ or } \Pi(\neg A) = 1 \tag{D.15}$$

It follows that for $n > 0$ [2]

$$(A, \mathsf{N}n) \models_2^{pos} (A, \mathsf{P}n) \tag{D.16}$$

In fact, suppose to the contrary that $(A, \mathsf{N}m) \not\models_2^{pos} (A, \mathsf{P}n)$. Therefore, for some fuzzy set of interpretations we have

$$\begin{array}{l} 1 - \Pi(\neg A) \geq n, \text{ and} \\ \Pi(A) < n. \end{array}$$

Therefore, $\max\{\Pi(A), \Pi(\neg A)\} < 1$, which is in contradiction with $(D.15)$.

By definition we have,

$$A \models_2 B \text{ implies } \Pi(A) \leq \Pi(B) \tag{D.17}$$

As a consequence, if $A \models_2 B$ then $\neg B \models_2 \neg A$ and, thus,

$$\Pi(\neg B) \leq \Pi(\neg A)$$

which is equivalent to

$$1 - \Pi(\neg B) \geq 1 - \Pi(\neg A)$$

Therefore,

$$A \models_2 B \text{ implies } N(A) \leq N(B) \tag{D.18}$$

Furthermore,

$$\begin{array}{rcl} N(A \wedge (\neg A \vee B)) & = & 1 - \Pi(\neg(A \wedge (\neg A \vee B))) \\ & = & 1 - \Pi(\neg A \vee (A \wedge \neg B)) \\ & = & 1 - \max\{\Pi(\neg A), \Pi(A \wedge \neg B)\} \\ & = & \min\{1 - \Pi(\neg A), 1 - \Pi(A \wedge \neg B)\} \\ & = & \min\{N(A), N(\neg A \vee B)\} \end{array} \tag{D.19}$$

Therefore,

$$(A, \mathsf{N}m), (\neg A \vee B, \mathsf{N}n) \models_2^{pos} (B, \mathsf{N} \min\{m, n\}) \tag{D.20}$$

Similarly,

$$(A, \mathsf{N}n), (\neg A \vee B, \mathsf{P}m) \models_2^{pos} (B, \mathsf{P}m) \text{ if } n > 1 - m \tag{D.21}$$

In fact,

[2] Case $n = 0$ is obvious.

$$\begin{aligned} N(A) \geq n \quad &\text{iff} \quad 1 - \Pi(\neg A) \geq n \\ &\text{iff} \quad \Pi(\neg A) \leq 1 - n \end{aligned} \tag{D.22}$$

$$\Pi(\neg A \vee B) \geq m \quad \text{iff} \quad \max\{\Pi(\neg A), \Pi(B)\} \geq m \tag{D.23}$$

Equations $(D.22)$ and $(D.23)$ imply

$$\Pi(B) \geq m \text{ if } 1 - n < m$$

Consider the following example.

Example 50 Let Φ be the following set of uncertain propositions.

$$\Phi = \{(p, \mathsf{N}.8), (p \rightarrow q, \mathsf{N}.4), (q \rightarrow r, \mathsf{P}.7)\}$$

Then $\Phi \models_2^{pos} (r, \mathsf{P}.7)$. In fact, from $(D.20)$ we have

$$\Phi \models_2^{pos} (q, \mathsf{N}.4)$$

Since $.4 > 1 - .7$, from $(D.21)$, $\Phi \models_2^{pos} (r, \mathsf{P}.7)$ follows. ■

Hollunder's Theorem 3.4 in [149] gives us a method for deciding possibilistic entailment by relying on a propositional decision procedure. In fact, let

$$\Phi_n = \{A : (A, \mathsf{N}m) \in \Phi \text{ and } m \geq n\} \tag{D.24}$$

$$\Phi^n = \{A : (A, \mathsf{N}m) \in \Phi \text{ and } m > n\} \tag{D.25}$$

then

Theorem 1 (Hollunder) *Let Φ be a set of possibilistic propositions and $n > 0$. Then*

1. $\Phi \models_2^{pos} (A, \mathsf{N}n)$ *iff* $\Phi_n \models_2 A$;
2. $\Phi \models_2^{pos} (A, \mathsf{P}n)$ *iff*
   (a) $\Phi^0 \models_2 A$, *or*
   (b) *there is some* $(B, \mathsf{P}m) \in \Phi$ *such that* $m \geq n$ *and* $\Phi^{1-m} \cup \{B\} \models_2 A$. ⊣

1. Point 1 can roughly be justified by observing Equation $(D.20)$, which can be seen as a "weak" form of resolution step involving necessity valued propositions only. Suppose we are trying to prove $(B, \mathsf{N}k)$ from Φ. From $(D.20)$ and $(\min\{m, n\} \geq k \text{ iff } m \geq k, n \geq k)$ it follows that in order to prove $(B, \mathsf{N}k)$, we can restrict our attention to those $(A, \mathsf{N}n) \in \Phi$, such that $n \geq k$.

2. Point 2 can roughly be justified by observing Equation $(D.21)$, which can be seen as a "weak" form of resolution step involving both necessity and possibility valued propositions. Suppose we are trying to prove $(B, \mathsf{P}k)$ from Φ.

(a) Case 2a. If $\Phi \models_2^{pos} (B, \mathsf{N}k)$, then $\Phi \models_2^{pos} (B, \mathsf{P}k)$ follows from $(D.16)$.

(b) Case 2b. Otherwise, by $(D.21)$, there must be a $(C, \mathsf{P}m) \in \Phi$, with $m \geq k$, which interacts with a $(A, \mathsf{N}n) \in \Phi$. But this can be the case only if $n > 1 - m$. Hence, we need to consider only those $(A, \mathsf{N}n) \in \Phi$ such that $n > 1 - m$.

Example 51 Let Φ be the set of uncertain propositions of Example 50, *i.e.*

$$\Phi = \{(p, \mathsf{N}.8), (p \rightarrow q, \mathsf{N}.4), (q \rightarrow r, \mathsf{P}.7)\}$$

we will prove that $\Phi \models_2^{pos} (r, \mathsf{P}.7)$ (as already seen in Example 50 by means of semantics considerations) by applying Theorem 1. By Theorem 1, Point 2b, we have $(q \rightarrow r, \mathsf{P}.7) \in \Phi$, $.7 \geq .7$, $\Phi^{1-.7}$ is

$$\Phi^{1-.7} = \{p, p \rightarrow q\}$$

and

$$\Phi^{1-.7} \cup \{q \rightarrow r\} \models_2 r.$$

Hence, $\Phi \models_2^{pos} (r, \mathsf{P}.7)$. ■

The main point is that a closer look to Proposition 56 reveals that it is similar to Hollunder's Theorem 1, Point 1, *i.e.* whenever we restrict Φ to necessity propositions only. As a consequence, let Σ be a $\mathcal{L}^f$ KB and let

$$\hat{\Sigma} = \{(A, \mathsf{N}n) : (A \geq n) \in \Sigma\}. \tag{D.26}$$

Since $\models_4 \subset \models_2$, from Proposition 56 and Theorem 1 it follows that

Proposition 59 *Let $\Sigma \subseteq \mathcal{L}^f$, $A \in \mathcal{L}$ and $n > 0$. If $\Sigma \mid\!\approx_4 (A \geq n)$ then $\hat{\Sigma} \models_2^{pos} (A, \mathsf{N}n)$.* ⊣

Example 52 Let Σ be

$$\Sigma = \{(A \geq .5), (B \vee C \geq .7)\}$$

we have seen (Figure D.1) that

$$\Sigma \mid\!\approx_4 ((A \vee C) \wedge (B \vee C \vee D) \geq .5)$$

By definition w have

$$\hat{\Sigma} = \{(A, \mathsf{N}.5), (B \vee C, \mathsf{N}.7),$$

It can easily be verified that

$$\hat{\Sigma} \models_2^{pos} ((A \vee C) \wedge (B \vee C \vee D), \mathsf{N}.5)$$

■

It is worth noting that the converse of Theorem 59 is not true, *i.e.*

$$\hat{\Sigma} \models_2^{pos} (A, \mathsf{N}n) \text{ does not imply } \Sigma \not\approx_4 (A \geq n). \tag{D.27}$$

For instance,

$$(p \geq .6), (\neg p \geq .7) \not\approx_4 (q \geq .6),$$

whereas

$$(p, \mathsf{N}.6), (\neg p, \mathsf{N}.7) \models_2^{pos} (q, \mathsf{N}.6)$$

and, thus, $\not\approx_4 \subset \models_2^{pos}$ holds.

Neither Proposition 59 nor the converse of it holds for $\not\approx_2$. We can confirm this by considering Σ_1 and Σ_2 of the previous section:

$$\Sigma_1 = \{(p \geq .2), (\neg p \geq .3)\}, \text{ and}$$
$$\Sigma_2 = \{(p \geq .2), (\neg p \geq .9)\}.$$

Therefore,

$$\Sigma_2 \not\approx_2 (q \geq .3), \text{ and}$$
$$\hat{\Sigma}_2 \not\models_2^{pos} (q, \mathsf{N}.3)$$

whereas

$$\Sigma_1 \not\approx_2 (q \geq .1), \text{ and}$$
$$\hat{\Sigma}_1 \models_2^{pos} (q, \mathsf{N}.1)$$

D.2 Deciding entailment in $\mathcal{L}_+^f$

At first, note that the following proposition holds.

Proposition 60 *For case* $\mathcal{L}_+^f$

1. *Proposition 7 does not hold.*
   $A \models_4 B \to B$, *but* $(A \geq .7) \not\approx_4 (B \to B \geq .7)$.

2. *Proposition 8 does not hold.*
   (Only if) direction holds. For the (if) direction we have: $A \models_4 B \to B$, *but* $(A \geq .7) \not\approx_4 (B \to B \geq .7)$.

3. *Proposition 9 does not hold.*
   $(A \wedge (A \to B) \geq .6) \not\approx_4 (B \geq .6)$, *but* $(\neg B \geq .6) \not\approx_4 (\neg(A \wedge (A \to B)) \geq .6)$ *(*$|B|^t = |B|^f = .6, |A|^t = |A|^f = .4$*).*

4. *Proposition 54 does not hold.*
   (If) direction holds. For the (only if) direction we have: let Σ *be* $\{(A \geq .6), (A \to B \geq .8)\}$. *Then* $\Sigma \not\approx_4 (B \geq .8)$, *but* $\Sigma^{\cdot .8} \not\approx_4 (B \geq .8)$.

5. *Proposition 55 does not hold.*
   (Only if) direction holds. For the (if) direction we have: let Σ *be* $\{(A \geq .3), (A \to B \geq .4)\}$. *Then* $\overline{\Sigma^{.3}} \models_4 B$, *but* $\Sigma \not\approx_4 (B \geq .3)$.

6. *Proposition 56 does not hold.*
   (Only if) direction does not hold as a consequence of Point 4 above. (If) direction does not hold as a consequence of Point 5 above.

7. *Proposition 57 does not hold.*
   (Only if) direction holds. (If) direction does not hold: let Σ *be* $\{(A \geq .3), (A \to B \geq .4)\}$. *Then* $\overline{\Sigma} \models_4 B$ *but* $\Sigma \not\approx_4 (B \geq .3)$.

8. *Proposition 58 does not hold.*
   (Only if) direction holds, but (if) direction does not hold (from Point 6 and Point 7 above. $\dashv$

As the above proposition shows, most of the nice properties of $\mathcal{L}^f$ do not hold in $\mathcal{L}^f_+$. Especially, the fact that Proposition 56 does not hold in $\mathcal{L}^f_+$ has the consequence that

1. we cannot decide fuzzy entailment in terms of entailment, and
2. we are not able to compute the maximal degree of truth by means of successive calls to the entailment procedure,

as we did in the case of $\mathcal{L}^f$. The problem relies on the difference of the axiom schema for modus ponens between the non-fuzzy/fuzzy case: we remind that in $\mathcal{L}_+$ we have

$$A, A \to B \models_4 B, \tag{D.28}$$

whereas in $\mathcal{L}^f_+$ we have

$$(A \geq m), (A \to B \geq n) \not\approx_4 (B \geq n) \text{ if } m > 1 - n \tag{D.29}$$

Therefore, it is immediate to verify that given A and $A \to B$ the modus ponens rule always is applied, yielding B, while this is not true for the fuzzy counterpart, *i.e.* from $(A \geq .2), (A \to B \geq .6)$ the modus ponens rule cannot be applied (see also Point 7 in Proposition 60), invalidating Proposition 57.

We present now a calculus which is an extension of the procedure *Sat* for $\mathcal{L}^f$. As for $\mathcal{L}^f$, we consider signed fuzzy propositions of type $\mathsf{T}(A \geq n)$ and $\mathsf{NT}(A \geq n)$. Moreover, unlike $\mathcal{L}^f$, $\mathsf{T}(A > n)$ and $\mathsf{NT}(A > n)$ are signed proposition in $\mathcal{L}^f_+$ too. We extend interpretation functions in such a way that an interpretation $\mathcal{I}$ *satisfies* $\mathsf{T}(A > n)$ iff $|A|^t > n$, whereas $\mathcal{I}$ satisfies $\mathsf{NT}(A > n)$ whenever $|A|^t \leq n$.

As for $\mathcal{L}^f$

$$\Sigma \not\approx_4 (A \geq n) \text{ iff } \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq n)\} \text{ is not satisfiable} \tag{D.30}$$

holds. With respect to conjugates we have the following Table D.7. Each entry says us under which conditions the row-column pair of signed expressions are conjugated. A $\times$ symbol in an entry means that the pair cannot be a conjugated one.

	$\mathsf{T}(A \geq m)$	$\mathsf{T}(A > m)$	$\mathsf{NT}(A \geq m)$	$\mathsf{NT}(A > m)$
$\mathsf{T}(A \geq n)$	$\times$	$\times$	$n \geq m$	$n > m$
$\mathsf{T}(A > n)$	$\times$	$\times$	$n \geq m$	$n \geq m$
$\mathsf{NT}(A \geq n)$	$n \leq m$	$n \leq m$	$\times$	$\times$
$\mathsf{NT}(A > n)$	$n < m$	$n \leq m$	$\times$	$\times$

Table D.7: Conjugated signed fuzzy propositions in $\mathcal{L}^f_+$.

Given two signed fuzzy propositions σ_1 and σ_2, we define $Cond(\sigma, \sigma_2)$ to be the condition, according to Table D.7, under which σ_1 and σ_2 are conjugated, *e.g.* $Cond(\mathsf{T}(A \geq n), \mathsf{NT}(A \geq m)) = n \geq m$ and $Cond(\mathsf{T}(A \geq n), \mathsf{T}(A \geq m)) = \times$.

Moreover, we extend the definition of closed branch ϕ by saying that a branch ϕ in a deduction tree is *closed* whenever it contains a conjugated pair or it contains $\mathsf{T}(A > 1)$.

By considering the following Table D.8

α	α_1	α_2
$\mathsf{T}(A \wedge B \geq n)$	$\mathsf{T}(A \geq n)$	$\mathsf{T}(B \geq n)$
$\mathsf{T}(A \wedge B > n)$	$\mathsf{T}(A > n)$	$\mathsf{T}(B > n)$
$\mathsf{NT}(A \vee B \geq n)$	$\mathsf{NT}(A \geq n)$	$\mathsf{NT}(B \geq n)$
$\mathsf{NT}(A \vee B > n)$	$\mathsf{NT}(A > n)$	$\mathsf{NT}(B > n)$
$\mathsf{NT}(A \rightarrow B \geq n)$	$\mathsf{T}(A > 1 - n)$	$\mathsf{NT}(B \geq n)$
$\mathsf{NT}(A \rightarrow B > n)$	$\mathsf{T}(A \geq 1 - n)$	$\mathsf{NT}(B > n)$

β	β_1	β_2
$\mathsf{T}(A \vee B \geq n)$	$\mathsf{T}(A \geq n)$	$\mathsf{T}(B \geq n)$
$\mathsf{T}(A \vee B > n)$	$\mathsf{T}(A > n)$	$\mathsf{T}(B > n)$
$\mathsf{NT}(A \wedge B \geq n)$	$\mathsf{NT}(A \geq n)$	$\mathsf{NT}(B \geq n)$
$\mathsf{NT}(A \wedge B > n)$	$\mathsf{NT}(A > n)$	$\mathsf{NT}(B > n)$
$\mathsf{T}(A \rightarrow B \geq n)$	$\mathsf{NT}(A > 1 - n)$	$\mathsf{T}(B \geq n)$
$\mathsf{T}(A \rightarrow B > n)$	$\mathsf{NT}(A \geq 1 - n)$	$\mathsf{T}(B > n)$

Table D.8: α and β table for $\mathcal{L}^f_+$.

and the rules in Table D.3, we obtain

Proposition 61 *Let S be a set of signed fuzzy propositions in $\mathcal{L}^f_+$. Then $Sat(S)$ iff S is satisfiable.* ⊣

Proof: The proof consists in an extension of proof of Proposition 49.

Rules (A), $(B1)$, $(B2)$ and (PB) are correct in the $\mathcal{L}^f_+$ case too, *i.e.* if ϕ is a branch then S^ϕ is satisfiable iff there is a branch ϕ' as the result of the application of a rule to ϕ such that $S^{\phi'}$ satisfiable.

$\Rightarrow$.) Suppose $Sat(S)$. Let T be the generated deduction tree and let ϕ be a completed not closed branch from S to a leaf in T. Such a branch has to exist, otherwise $Sat(S) = false$. Let

$$S^{\mathsf{T}}_{\geq} = \{\mathsf{T}(A \geq n) \in S^\phi\},$$

$$S^{\mathsf{T}}_{>} = \{\mathsf{T}(A > n) \in S^\phi\}, \text{ and}$$

$$S^{\mathsf{NT}} = \{\mathsf{NT}(A \geq n) \in S^\phi\}.$$

Of course, $S^\phi = S^{\mathsf{T}}_{\geq} \cup S^{\mathsf{T}}_{>} \cup S^{\mathsf{NT}}$. Define, for $\epsilon > 0$

$$n^{\geq}_A = \max\{n : \mathsf{T}(A \geq n) \in S^{\mathsf{T}}_{\geq}\}, \text{ and}$$

$$n^{>}_A = \max\{n : \mathsf{T}(A > n) \in S^{\mathsf{T}}_{>}\} + \epsilon.$$

Let $\mathcal{I}$ be a relation such that

$$|p|^t = \max\{n^{\geq}_p, n^{>}_p\}, \text{ and}$$
$$|p|^f = 0.$$

Since, ϕ is completed and not closed, $\mathcal{I}$ is a four-valued fuzzy interpretation and there is an $\epsilon > 0$ such that $\mathcal{I}$ satisfies all $\sigma \in S^{\mathsf{T}}_{\geq}$; $\mathcal{I}$ satisfies all $\sigma \in S^{\mathsf{T}}_{>}$ (note that $n + \epsilon > n$), and $\mathcal{I}$ satisfies all $\sigma \in S^{\mathsf{NT}}$ and, thus, $\mathcal{I}$ satisfies S^ϕ. As a consequence, $S \subseteq S^\phi$ is satisfiable.
$\Leftarrow$.) Suppose S is satisfiable. Let T be the generated completed tree. From the correctness of the rules it follows that there is a completed branch ϕ in T such that S^ϕ is satisfiable. Therefore, $Sat(S)$.

Q.E.D.

Example 53 Let Σ be fuzzy KB of Example 19. The following deduction tree in Figure D.3 shows that $\Sigma \cup \{(\mathsf{Gil} \vee \mathsf{Karl} \geq .4)\} \mid\!\approx_4 (\mathsf{Tall} \geq .2)$.

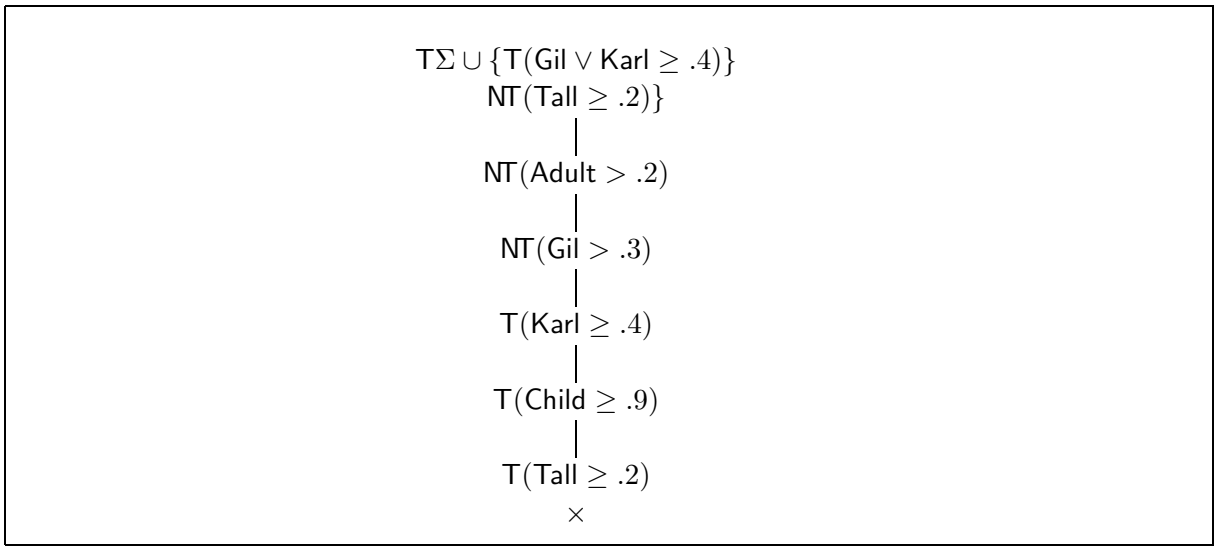


Figure D.3: Example of deduction in $\mathcal{L}^f_+$.

■

From a computational point of view, fuzzy entailment in $\mathcal{L}^f_+$ behaves as fuzzy entailment in $\mathcal{L}^f$.

Proposition 62 *Let $\Sigma \subseteq \mathcal{L}^f_+$ and $A \in \mathcal{L}_+$. Then checking $\Sigma \mid\!\approx_4 (A \geq n)$ is a coNP-complete problem.* ⊣

Proof: Proof is as in Proposition 50. Q.E.D.

Similarly as we did for $\mathcal{L}_+$, if we restrict $\mathcal{L}^f_+$ to $\overline{\mathcal{L}^f_+}$ we obtain a tractable fuzzy logic. $\overline{\mathcal{L}^f_+}$ is defined inductively as follows: $\overline{\mathcal{L}^f_+}$ is the minimal set such that

1. every fuzzy proposition in $\mathcal{L}$ in CNF is in $\overline{\mathcal{L}^f_+}$;
2. if $A, A_1, \ldots, A_n$ and $B, B_1, \ldots, B_m$ are literals in $\mathcal{L}$, then both $(A_1 \wedge \ldots \wedge A_n \rightarrow B \geq k)$ and $(A \rightarrow B_1 \vee \ldots \vee B_m \geq k)$ are in $\overline{\mathcal{L}^f_+}$.

We extend the algorithm $EasyEntail_+$ to take as input a $\overline{\mathcal{L}^f_+}$ fuzzy KB Σ and a $\overline{\mathcal{L}^f_+}$ fuzzy proposition $(A \geq n)$, and adapt rules $(\overline{B1})$ and $(\overline{B2})$ to their fuzzy case, as shown in Table D.9.

$$(\overline{B1})\ \frac{\begin{array}{c}\mathsf{T}(A_1 \wedge \ldots \wedge A_n \rightarrow B \geq k),\\ \mathsf{T}(A_1 \geq k_1), \ldots, \mathsf{T}(A_h \geq k_h),\\ \mathsf{T}(A_1 > k_{h+1}), \ldots, \mathsf{T}(A_n > k_n)\end{array}}{\mathsf{T}(B \geq k)} \quad \text{if} \quad \begin{array}{l}\text{for all } 1 \leq j \leq h,\ k_j > 1-k, \text{ and}\\ \text{for all } h+1 \leq j \leq n,\ k_j \geq 1-k\end{array}$$

$$(\overline{B2})\ \frac{\begin{array}{c}\mathsf{T}(A \rightarrow B_1 \vee \ldots \vee B_m \geq k),\\ \mathsf{NT}(B_1 \geq k_1), \ldots, \mathsf{NT}(B_h \geq k_h),\\ \mathsf{NT}(B_{h+1} > k_{h+1}), \ldots, \mathsf{NT}(B_m > k_m)\end{array}}{\mathsf{NT}(A > 1-k)} \quad \text{if} \quad \begin{array}{l}\text{for all } 1 \leq j \leq h,\ k_j \leq k, \text{ and}\\ \text{for all } h+1 \leq j \leq m,\ k_j < k\end{array}$$

Table D.9: Modified $(B1)$ and $(B2)$ rules for $(A \rightarrow B \geq n) \in \overline{L}^f_+$.

Proposition 63 *Let Σ be a $\overline{\mathcal{L}^f_+}$ KB, let $(A \geq n)$ be in $\overline{\mathcal{L}^f_+}$ and $n > 0$. Checking if $\Sigma \mid\!\approx_4 (A \geq n)$ can be done in time $O(|\Sigma||A|)$ by means of the algorithm $EasyEntail_+$.* ⊣

Proof: As for Proposition 23. Q.E.D.

Consider Example 53. Figure D.3 is a deduction tree build by applying $EasyEntail_+(\Sigma, A)$.

D.2.1 **Relations to possibilistic logic in $\mathcal{L}^f_+$

In Section D.1.1, Proposition 59 shows that there is relation between fuzzy entailment in $\mathcal{L}^f$ and possibilistic implication. Hence, it is natural to ask whether a similar relation continues to hold whenever we switch to $\mathcal{L}^f_+$.

The proof of Proposition 59 relies on the fact that in $\mathcal{L}^f$, $\Sigma \mid\!\approx_4 (A \geq n)$ iff $\overline{\Sigma^n} \models_4 A$ holds. We have already seen in Proposition 60, Point 5. that this is not true $\mathcal{L}^f_+$. But, a weaker relation holds. In fact, it is easily verified that in $\mathcal{L}^f_+$ $\Sigma \mid\!\approx_4 (A \geq n)$ implies $\overline{\Sigma} \models_4 A$ holds. Notwithstanding, a similar result as stated in Proposition 59 can easily be shown. In fact, we already have seen that

$$(A \geq n), (A \to B \geq m) \mid\!\approx_4 (B \geq m) \text{ if } n > 1 - m \tag{D.31}$$
$$(A, \mathsf{N}n), (A \to B, \mathsf{N}m) \mid\!\approx_4 (A, \mathsf{N}\min\{n, m\}). \tag{D.32}$$

Therefore, it could be that $m \neq \min\{n, m\}$. Now, suppose that $\Sigma \mid\!\approx_4 (A \geq n)$. Therefore, $\overline{\Sigma} \models_4 A$. In *e.g.* [106] it has been shown, so as from Theorem 1 it follows, that if $\overline{\Sigma} \models_4 A$ then there is a $k > 0$ such that $\hat{\Sigma} \models_2^{pos} (A, \mathsf{N}k)$, which is very similar to Proposition 57. Therefore,

Proposition 64 *Let $\Sigma \subseteq \mathcal{L}_+^f$, $A \in \mathcal{L}_+$ and $n > 0$. If $\Sigma \mid\!\approx_4 (A \geq n)$ then there is $0 < k \leq n$ such that $\hat{\Sigma} \models_2^{pos} (A, \mathsf{N}k)$.* ⊣

establishing a connection between fuzzy entailment in $\mathcal{L}_+^f$ and possibilistic implication.

Example 54 Let Σ be fuzzy KB

$$\begin{aligned} \Sigma = \ \{ \ & (\mathsf{Gil} \to \mathsf{Adult} \geq .7), \\ & (\mathsf{Adult} \to \mathsf{Tall} \geq .8), \\ & (\mathsf{Karl} \to \mathsf{Child} \geq .9), \\ & (\mathsf{Child} \to \mathsf{Tall} \geq .6), \\ & (\mathsf{Gil} \geq .4), (\mathsf{Karl} \geq .2) \ \} \end{aligned}$$

It follows that

$$\Sigma \mid\!\approx_4 (\mathsf{Tall} \geq .8).$$

By definition, $\hat{\Sigma}$ is

$$\begin{aligned} \hat{\Sigma} = \ \{ \ & (\mathsf{Gil} \to \mathsf{Adult}, \mathsf{N}.7), \\ & (\mathsf{Adult} \to \mathsf{Tall}, \mathsf{N}.8), \\ & (\mathsf{Karl} \to \mathsf{Child}, \mathsf{N}.9), \\ & (\mathsf{Child} \to \mathsf{Tall}, \mathsf{N}.6), \\ & (\mathsf{Gil}, \mathsf{N}.4), (\mathsf{Karl}, \mathsf{N}.2) \ \} \end{aligned}$$

It follows that

$$\hat{\Sigma} \models_2^{pos} (\mathsf{Adult}, \mathsf{N}\min\{.4, .7\}),$$
$$\hat{\Sigma} \models_2^{pos} (\mathsf{Tall}, \mathsf{N}\min\{.4, .8\}),$$

and similarly,

$$\hat{\Sigma} \models_2^{pos} (\mathsf{Child}, \mathsf{N}\min\{.2, .9\}),$$
$$\hat{\Sigma} \models_2^{pos} (\mathsf{Tall}, \mathsf{N}\min\{.2, .6\}).$$

Therefore,

$$\hat{\Sigma} \models_2^{pos} (\mathsf{Tall}, \mathsf{N}.4),$$

confirming Proposition 64. ■

D.2.2 Determining the maximal degree of truth in $\mathcal{L}^f_+$

Concerning the computation of $Maxdeg(\Sigma, A)$ it is worth noticing that we cannot compute $Maxdeg(\Sigma, A)$ by means of a sequence of entailment test (as it happens for KBs in $\mathcal{L}^f$), but rather than by means of a sequence of *fuzzy* entailment test. Moreover, as they are tautologies, for $\mathcal{L}^f_+$

$$Maxdeg(\Sigma, A) \in \{0, .5\} \cup N_\Sigma.$$

holds.

Algorithm 15 ($Max_+(\Sigma, A)$)
Let $\Sigma \subseteq \mathcal{L}^f_+$ and $A \in \mathcal{L}^f_+$. Set $Min = 0$, $Max = 2$.

1. Pick $n \in N_\Sigma \cup \{.5\}$ such that $Min < n < Max$. If there is no such n, then set $Maxdeg(\Sigma, A) := Min$ and exit.
2. Check if $\Sigma \mid\!\approx_4 (A \geq n)$. If so, then set $Min = n$ and go to Step 1. If not so, then set $Max = n$ and go to Step 1.

■

Table D.10: Algorithm $Max_+(\Sigma, A)$ for $\mathcal{L}^f_+$.

By means of Algorithm 15 above, $Maxdeg(\Sigma, A)$ requires $O(|\Sigma|)$ fuzzy entailment tests. As $O(|\Sigma|)$ can be very huge, this may be unfeasible.

We address now this problem by presenting a method for computing $Maxdeg(\Sigma, A)$ which performs the fuzzy entailment test only *once*. At first, we generalise fuzzy propositions to the form $(A \geq \lambda)$, where λ is a fuzzy extended value defined as follows. Let v be a new variable. A *fuzzy extended value* (with metavariable λ) is defined as follows:

1. $n \in [0, 1]$ is a fuzzy extended value;
2. v is a fuzzy extended value;
3. $1 - v$ is a fuzzy extended value.

An interpretation $\mathcal{I}$ is such that $n^\mathcal{I} = n, v^\mathcal{I} \in [0, 1]$ and $(1 - v)^\mathcal{I} = 1 - v^\mathcal{I}$. Note that $(1 - v)^\mathcal{I} \in [0, 1]$. In the following, given $n \in [0, 1]$ and an equation (denoted by eq), $\lambda_1 \geq \lambda_2$ or $\lambda_1 > \lambda_2$, with $eq[v/n]$ we will denote the equation eq' which is the result of replacing all occurrences of v in eq with n. For instance, if eq is $.8 \geq 1 - v$ then $eq[v/.7]$ is $.8 \geq .3$. The following Table D.11, shows the *solution* for v under which an equation $eq := \lambda_1 \geq \lambda_2$ holds. The solution (denoted by $Sol(eq)$) is given in terms of an interval $[m, n] \subseteq [0, 1]$ and has the following property: if $Sol(eq) = [m, n]$ then for all $k \in [m, n]$, $eq[v/k]$ holds. For instance, if eq is $.8 \geq 1 - v$, then $Sol(eq) = [.2, 1]$. In fact, for all $k \in [.2, 1]$, $(.8 \geq 1 - v)[v/k] = .8 \geq 1 - k$ holds.
The solutions for $eq := \lambda_1 > \lambda_2$ are defined similarly. Moreover, we define $Sol(n \geq m) = [0, 1]$ if $n \geq m$ and $Sol(n > m) = [0, 1]$ if $n > m$. $Sol(\cdot)$ is undefined in the other cases.

$eq := \lambda_1 \geq \lambda_2$	$Sol(eq)$
$v \geq v$	$[0, 1]$
$v \geq 1 - v$	$[.5, 1]$
$1 - v \geq v$	$[0, .5]$
$1 - v \geq 1 - v$	$[0, 1]$
$v \geq n$	$[n, 1]$
$1 - v \geq n$	$[0, 1 - n]$
$n \geq v$	$[0, n]$
$n \geq 1 - v$	$[1 - n, 1]$

Table D.11: Solutions for v such that $\lambda_1 \geq \lambda_2$ holds.

Let σ be a signed fuzzy proposition and $n \in (0, 1]$. A *conditioned signed fuzzy proposition* is an expression of the form $\langle \sigma, v \in [0, n] \rangle$. The intended meaning of $\langle \sigma, v \in [0, n] \rangle$ is that σ holds whenever v "ranges" in $[0, n]$. We will write σ as a shortcut of $\langle \sigma, v \in [0, 1] \rangle$. We will use r as a metavariable for an interval $[m, n]$ and ρ as a metavariable for conditioned signed fuzzy proposition.

In the following, let $\sigma, \langle \sigma, v \in r \rangle, S, \phi$ and T be a signed fuzzy propositions, a conditioned signed fuzzy proposition, a sets of conditioned signed fuzzy proposition, a branch and a deduction tree involving conditioned signed fuzzy propositions, respectively. For all $n \in [0, 1]$, we define

1. $\sigma[v/n]$ to be the signed fuzzy proposition σ' which is the result of replacing every occurrence of v in σ with n;
2. $\langle \sigma, v \in r \rangle[v/n]$ as $\sigma[v/n]$ if $n \in r$, otherwise $\langle \sigma, v \in r \rangle[v/n]$ is the empty string;
3. $S[v/n]$ as $\{\rho[v/n] : \rho \in S\}$;
4. $\phi[v/n]$ is the branch ϕ' which is the result of replacing every occurrence of ρ in ϕ with $\rho[v/n]$;
5. $T[v/n]$ as the deduction tree T' which is the result of replacing every branch ϕ in T with $\phi[v/n]$.

Example 55 Let

$$\begin{aligned} &\sigma = \mathsf{T}(A \geq 1 - v), \\ &\rho = \langle \sigma, v \in [0, .7] \rangle, \text{ and} \\ &S = \{\langle \sigma, v \in [0, .7] \rangle, \langle \sigma, v \in [0, .3] \rangle\}. \end{aligned}$$

Then for $n = .6$ we have that

$$\begin{aligned} &\sigma[v/n] = \mathsf{T}(A \geq .4) \\ &\langle \sigma, v \in [0, .7] \rangle[v/n] = \mathsf{T}(A \geq .4), \text{ and} \\ &S[v/n] = \{\mathsf{T}(A \geq .4)\}. \end{aligned}$$

Just note that $\langle \sigma, v \in [0, .3] \rangle[v/n]$ is the empty string, as $n \notin [0, .3]$. ■

Satisfiability of fuzzy propositions is defined as usual: $\mathcal{I}$ satisfies $(A \geq \lambda)$ iff $|A|^t \geq \lambda^{\mathcal{I}}$ and $\mathcal{I}$ satisfies $(A > \lambda)$ iff $|A|^t > \lambda^{\mathcal{I}}$. $\mathcal{I}$ satisfies a signed fuzzy proposition, *e.g.* $\mathsf{T}(A \geq \lambda)$ iff $\mathcal{I}$ satisfies $(A \geq \lambda)$. The other cases are similar.

Finally, $\mathcal{I}$ *satisfies* $\langle \sigma, v \in [0, n] \rangle$ iff if $v^{\mathcal{I}} \in [0, n]$ then $\mathcal{I}$ satisfies $\sigma[v/v^{\mathcal{I}}]$.

Example 56 Let ρ be $\langle \mathsf{T}(A \geq v), v \in r \rangle$, where r is $[0, .6]$. Let $\mathcal{I}$ be an interpretation such that:

$$\begin{array}{l} v^{\mathcal{I}} = .55, \\ |A|^t = .7, \text{ and} \\ |A|^f = .4. \end{array}$$

Therefore, $\mathcal{I}$ satisfies ρ. Consider $n = .3$. Certainly, $n \in r$. Now, $\mathsf{T}(A \geq v)[v/n]$ is $\mathsf{T}(A \geq .3)$. $\mathcal{I}$ satisfies $\mathsf{T}(A \geq .3)$. ■

In what follows, we will use the obvious extension of signed fuzzy propositions of type α and of type β as defined in Table D.8: we just replace n in in Table D.8 with λ with the obvious simplification $1 - (1 - v) = v$. For instance, $\mathsf{T}(A \to B \geq 1 - v)$ is of type β and $\mathsf{NT}(A > v)$ and $\mathsf{T}(B \geq 1 - v)$ are its β_1 and β_2 components, respectively.

With respect to α and β tables in the conditioned case, these looks like Table D.8 and are shown in Table D.12 below, which is its straightforward adaption to the conditioned case.

α	α_1	α_2
$\langle \mathsf{T}(A \wedge B \geq \lambda), v \in r \rangle$	$\langle \mathsf{T}(A \geq \lambda), v \in r \rangle$	$\langle \mathsf{T}(B \geq \lambda), v \in r \rangle$
$\langle \mathsf{T}(A \wedge B > \lambda), v \in r \rangle$	$\langle \mathsf{T}(A > \lambda), v \in r \rangle$	$\langle \mathsf{T}(B > \lambda), v \in r \rangle$
$\langle \mathsf{NT}(A \vee B \geq \lambda), v \in r \rangle$	$\langle \mathsf{NT}(A \geq \lambda), v \in r \rangle$	$\langle \mathsf{NT}(B \geq \lambda), v \in r \rangle$
$\langle \mathsf{NT}(A \vee B > \lambda), v \in r \rangle$	$\langle \mathsf{NT}(A > \lambda), v \in r \rangle$	$\langle \mathsf{NT}(B > \lambda), v \in r \rangle$
$\langle \mathsf{NT}(A \to B \geq \lambda), v \in r \rangle$	$\langle \mathsf{T}(A > 1 - \lambda), v \in r \rangle$	$\langle \mathsf{NT}(B \geq \lambda), v \in r \rangle$
$\langle \mathsf{NT}(A \to B > \lambda), v \in r \rangle$	$\langle \mathsf{T}(A \geq 1 - \lambda), v \in r \rangle$	$\langle \mathsf{NT}(B > \lambda), v \in r \rangle$

β	β_1	β_2
$\langle \mathsf{T}(A \vee B \geq \lambda), v \in r \rangle$	$\langle \mathsf{T}(A \geq \lambda), v \in r \rangle$	$\langle \mathsf{T}(B \geq \lambda), v \in r \rangle$
$\langle \mathsf{T}(A \vee B > \lambda), v \in r \rangle$	$\langle \mathsf{T}(A > \lambda), v \in r \rangle$	$\langle \mathsf{T}(B > \lambda), v \in r \rangle$
$\langle \mathsf{NT}(A \wedge B \geq \lambda), v \in r \rangle$	$\langle \mathsf{NT}(A \geq \lambda), v \in r \rangle$	$\langle \mathsf{NT}(B \geq \lambda), v \in r \rangle$
$\langle \mathsf{NT}(A \wedge B > \lambda), v \in r \rangle$	$\langle \mathsf{NT}(A > \lambda), v \in r \rangle$	$\langle \mathsf{NT}(B > \lambda), v \in r \rangle$
$\langle \mathsf{T}(A \to B \geq \lambda), v \in r \rangle$	$\langle \mathsf{NT}(A > 1 - \lambda), v \in r \rangle$	$\langle \mathsf{T}(B \geq \lambda), v \in r \rangle$
$\langle \mathsf{T}(A \to B > \lambda), v \in r \rangle$	$\langle \mathsf{NT}(A \geq 1 - \lambda), v \in r \rangle$	$\langle \mathsf{T}(B > \lambda), v \in r \rangle$

Table D.12: α and β table for $\mathcal{L}^f_+$ in the conditioned case.

For the definition of conjugated signed fuzzy propositions, we rely on Table D.7, where n and m are replaced with λ_1 and λ_2, respectively. Two signed fuzzy propositions σ_1 and σ_2 are *conditioned conjugated* (*c-conjugated*) iff $Sol(Cond(\sigma_1, \sigma_2)) = [0, k]$, for some $k \in [0, 1]$. Note that if σ_1 and σ_2 are c-conjugated then for all $n \in Sol(Cond(\sigma_1, \sigma_2)) = [0, k]$, $\sigma_1[v/n]$ and $\sigma_2[v/n]$ are conjugated. As usual, a c-conjugate of σ will be identified by σ^c, whereas with $\sigma^{c,max}$ we identify the c-conjugate of σ obtained by exchanging T and NT in σ. Furthermore, if $Sol(Cond(\sigma_1, \sigma_2)) = [0, 1]$ then σ_1 and σ_2 are *conjugated*.

Example 57 Consider the following signed expressions:

$$\begin{array}{l}\sigma_1 = \mathsf{T}(A \geq 1 - v),\\ \sigma_2 = \mathsf{NT}(A \geq v),\\ \sigma_3 = \mathsf{T}(A \geq v), \text{ and}\\ \sigma_4 = \mathsf{NT}(A \geq 1 - v).\end{array}$$

It is easily verified that

$$\begin{array}{l}Sol(Cond(\sigma_1, \sigma_2)) = [0, .5],\\ Sol(Cond(\sigma_1, \sigma_4)) = [0, 1],\\ Sol(Cond(\sigma_3, \sigma_2)) = [0, 1], \text{ and}\\ Sol(Cond(\sigma_3, \sigma_4)) = [.5, 1].\end{array}$$

hold. Thus, σ_1 and σ_2 are c-conjugated, whereas σ_3 and σ_4 are not. Finally, both σ_1 and σ_4 and σ_3 and σ_2 are conjugated. ■

The motivation behind the constraint that $Sol(Cond(\sigma_1, \sigma_2))$ should be of the form $[0, k]$ will be explained later on.

With respect to conditioned signed fuzzy propositions we have: two conditioned signed fuzzy propositions $\langle \sigma_1, v \in r_1 \rangle$ and $\langle \sigma_2, v \in r_2 \rangle$ are *conditioned conjugated* (*c-conjugated*) iff σ_1 and σ_2 are c-conjugated. If $r_1 \cap r_2 \cap Sol(Cond(\sigma_1, \sigma_2)) = [0, 1]$ then $\langle \sigma_1, v \in r_1 \rangle$ and $\langle \sigma_2, v \in r_2 \rangle$ are *conjugated*. Note that if $\langle \sigma_1, v \in r_1 \rangle$ and $\langle \sigma_2, v \in r_2 \rangle$ are c-conjugated then for all $n \in r_1 \cap r_2 \cap Sol(Cond(\sigma_1, \sigma_2))$, $\langle \sigma_1, v \in r_1 \rangle[v/n]$ and $\langle \sigma_2, v \in r_2 \rangle[v/n]$ are conjugated.

The deduction rules are a generalisation of the rules defined in Table D.3 and are an adaption to the case of conditioned signed fuzzy proposition. The rules are shown in Table D.13.

$$(A)\ \frac{\langle \alpha, v \in r \rangle}{\langle \alpha_1, v \in r \rangle, \langle \alpha_2, v \in r \rangle}$$

$$(B1)\ \frac{\langle \beta, v \in r_1 \rangle, \langle \beta_1^c, v \in r_2 \rangle}{\langle \beta_2, v \in [0, k] \rangle} \qquad [0, k] = r_1 \cap r_2 \cap Sol(Cond(\beta_1, \beta_1^c)).$$

$$(B2)\ \frac{\langle \beta, v \in r_1 \rangle, \langle \beta_2^c, v \in r_2 \rangle}{\langle \beta_1, v \in [0, k] \rangle} \qquad [0, k] = r_1 \cap r_2 \cap Sol(Cond(\beta_2, \beta_2^c)).$$

$$(PB)\ \frac{\langle \beta, v \in r \rangle}{\langle \beta_1, v \in r \rangle \mid \langle \beta_1^{c,max}, v \in r \rangle, \langle \beta_2, v \in r \rangle}$$

Table D.13: Inference rules for conditioned singed fuzzy propositions in $\mathcal{L}_+^f$.

The rules are defined in such a way that they are *correct*: *e.g.* consider the $(B1)$ rule. Given an interpretation $\mathcal{I}$, if $\mathcal{I}$ satisfies the premises then $\mathcal{I}$ satisfies the conclusion. Just note that $Sol(Cond(\beta_1, \beta_1^c))$ has the form $[0, l]$, for some $l \in (0, 1]$ and, thus, $r_1 \cap r_2 \cap Sol(Cond(\beta_1, \beta_1^c))$ is of the form $[0, k]$.

Proposition 65 *The rules in Table D.13 are correct.* ⊣

Proof: We show the proof for the $(B1)$ rule. The proof for the others is similar. Let $\mathcal{I}$ be an interpretation. We show that if $\mathcal{I}$ satisfies both $\langle \beta, v \in r_1 \rangle$ and $\langle \beta_1^c, v \in r_2 \rangle$, then $\mathcal{I}$ satisfies $\langle \beta_2, v \in [0,k] \rangle$, where $[0,k] = r_1 \cap r_2 \cap Sol(Cond(\beta_1, \beta_1^c))$. Assume that $\mathcal{I}$ satisfies the premises of the $(B1)$ rule. In order to show that $\mathcal{I}$ satisfies $\langle \beta_2, v \in [0,k] \rangle$ we have to show that if $v^{\mathcal{I}} \in [0,k]$ then $\mathcal{I}$ satisfies $\beta_2[v/v^{\mathcal{I}}]$. So, suppose also that $v^{\mathcal{I}} \in [0,k]$. Hence, $v^{\mathcal{I}} \in r_1 \cap r_2 \cap Sol(Cond(\beta_1, \beta_1^c))$. Therefore,

1. from $v^{\mathcal{I}} \in r_1$ and by hypothesis, $\mathcal{I}$ satisfies $\beta[v/v^{\mathcal{I}}]$;
2. from $v^{\mathcal{I}} \in r_2$ and by hypothesis, $\mathcal{I}$ satisfies $\beta_1^c[v/v^{\mathcal{I}}]$.

Since $v^{\mathcal{I}} \in Sol(Cond(\beta_1, \beta_1^c))$, $\beta_1^c[v/v^{\mathcal{I}}]$ and $\beta_1[v/v^{\mathcal{I}}]$ are conjugated. Therefore, $\mathcal{I}$ cannot satisfy $\beta_1[v/v^{\mathcal{I}}]$. But, $\mathcal{I}$ satisfies $\beta[v/v^{\mathcal{I}}]$ and, thus, $\mathcal{I}$ has to satisfy $\beta_2[v/v^{\mathcal{I}}]$. Q.E.D.

Example 58 Consider the following signed expressions:

$$\begin{array}{l} \sigma_1 = \mathsf{T}(A \vee B \geq .9), \\ \sigma_2 = \mathsf{NT}(A \geq .6), \\ \sigma_3 = \mathsf{T}(A \vee B \geq 1 - v), \\ \sigma_4 = \mathsf{NT}(A \geq v), \text{ and} \\ \sigma_5 = \mathsf{T}(A \vee B \geq .6), \end{array}$$

and the following intervals

$$\begin{array}{l} r_1 = [0, .7], \\ r_2 = [0, .6], \\ r_3 = [0, .7], \\ r_4 = [0, .8], \text{ and} \\ r_5 = [0, .3]. \end{array}$$

Note that σ_1, σ_3 and σ_5 are of type β. Consider σ_1. Its β_1 and β_2 components are $\beta_1 = \mathsf{T}(A \geq .9)$ and $\beta_2 = \mathsf{T}(B \geq .9)$, respectively. It is easily verified that

$$Sol(Cond(\beta_1, \sigma_2)) = [0, 1]$$

hold. Thus, β_1 and σ_2 are c-conjugated, *i.e.* $\beta_1^c = \sigma_2$ is a c-conjugate of β_1. More precisely, they are conjugates. Hence, the following instance of rule $(B1)$ can be applied.

$$(B1) \quad \frac{\langle \sigma_1, v \in r_1 \rangle, \langle \sigma_2, v \in r_2 \rangle}{\langle \mathsf{T}(B \geq .9), v \in [0, .6] \rangle}$$

where $[0, .6] = r_1 \cap r_2 \cap [0, 1]$.

Similarly, it is easily verified that by means of rule $(B1)$ we have:

1. from $\langle \sigma_3, v \in r_3 \rangle$ and $\langle \sigma_4, v \in r_4 \rangle$, infer $\langle \mathsf{T}(B \geq 1 - v), v \in [0, .5] \rangle$;
2. from $\langle \sigma_5, v \in r_5 \rangle$ and $\langle \sigma_4, v \in r_4 \rangle$, infer $\langle \mathsf{T}(B \geq .6), v \in [0, .3] \rangle$. ■

Concerning closed branches we have the following definitions: a branch ϕ *is conditioned closed* (*c-closed*) iff S^ϕ contains $\rho_1 = \langle \sigma_1, v \in r_1 \rangle$ and $\rho_2 = \langle \sigma_2, v \in r_2 \rangle$ such that ρ_1 and ρ_2 are c-conjugated, *i.e.* $Sol(Cond(\sigma_1, \sigma_2)) = [0, k]$; ϕ *is closed* iff $r_1 \cap r_2 \cap Sol(Cond(\sigma_1, \sigma_2)) = [0, 1]$. Just note that by definition, $Sol(Cond(\sigma_1, \sigma_2)) = [0, k]$, for some $k \in (0, 1]$. Hence, for all n in $r_1 \cap r_2 \cap [0, k]$, $\sigma_1[v/n]$ and $\sigma_2[v/n]$ are conjugated, and, thus, $\phi[v/n]$ is closed (the restriction $n \in r_1 \cap r_2 \cap [0, k]$, rather than $n \in [0, k]$, is necessary).

Example 59 If ϕ is a branch such that S^ϕ contains both

$$\langle \mathsf{T}(A \geq 1 - v), v \in [0, 1] \rangle$$
$$\langle \mathsf{NT}(A \geq 1 - v), v \in [0, 1] \rangle,$$

then ϕ is closed. In fact, $\sigma_1 = \mathsf{T}(A \geq 1 - v)$ and $\sigma_2 = \mathsf{NT}(A \geq 1 - v)$ are conjugated, as $Sol(Cond(\sigma_1, \sigma_2)) = [0, 1]$. Therefore for all $n \in [0, 1]$, $\mathsf{T}(A \geq 1 - n)$ and $\mathsf{NT}(A \geq 1 - n)$ are conjugated, and, thus, $\phi[v/n]$ is closed.

This is not the case if *e.g.* $\sigma_1 = \mathsf{T}(A \geq 1 - v)$ and $\sigma_2 = \mathsf{NT}(A \geq v)$. In this case, $Sol(Cond(\sigma_1, \sigma_2)) = [0, .5]$ and, thus, ϕ is c-closed and $\phi[v/n]$ is closed only if $n \in [0, .5]$. ■

Finally, for all branches ϕ we define $SOL(\phi)$ to be

$$SOL(\phi) = \{r_1 \cap r_2 \cap Sol(Cond(\sigma_1, \sigma_2)) : \langle \sigma_i, v \in r_i \rangle \in S^\phi, \sigma_1, \sigma_2 \text{ c-conjugated}\}. \tag{D.33}$$

Roughly, $SOL(\phi)$ gives us the set of solutions $[0, k]$ such that $\phi[v/n]$ is closed for any $n \in [0, k]$.

Given the solutions $SOL(\phi)$ for which the branch ϕ is c-closed, obviously, we are looking for the maximal one. Therefore, we define

$$Max(SOL(\phi)) = \bigcup_{r_i \in SOL(\phi)} r_i, \tag{D.34}$$

where $Max(\emptyset) = [0, 0]$. The interval $[0, k] = Max(SOL(\phi))$ gives us simply the maximal interval such that $\phi[v/n]$ is closed for any $n \in [0, k]$. In other words, among the possible candidates of c-conjugated pairs, which potentially c-close a branch ϕ, we are looking for the pair which gives us the possible highest value n for v, which closes the branch $\phi[v/n]$.

Example 60 Consider the following signed expressions

$$\sigma_1 = \mathsf{T}(A \geq 1 - v),$$
$$\sigma_2 = \mathsf{NT}(A \geq v),$$
$$\sigma_3 = \mathsf{NT}(A \geq .8),$$
$$\sigma_4 = \mathsf{T}(B \geq v),$$
$$\sigma_5 = \mathsf{NT}(B \geq .7), \text{ and}$$
$$\sigma_6 = \mathsf{NT}(B \geq v).$$

It is easily verified that

σ_1, σ_2 are c-conjugated	and $Sol(Cond(\sigma_1, \sigma_2) = [0, .5]$,
σ_1, σ_3 are c-conjugated	and $Sol(Cond(\sigma_1, \sigma_3) = [0, .2]$,
σ_4, σ_5 are not c-conjugated	and $Sol(Cond(\sigma_4, \sigma_5) = [.7, 1]$, and
σ_4, σ_6 are c-conjugated	and $Sol(Cond(\sigma_4, \sigma_6) = [0, 1]$.

Suppose ϕ is a branch such that S^ϕ contains

$$\begin{array}{l}\langle\sigma_1, v \in [0,.4]\rangle,\\ \langle\sigma_2, v \in [0,.1]\rangle,\\ \langle\sigma_3, v \in [0,.6]\rangle,\\ \langle\sigma_4, v \in [0,.5]\rangle,\\ \langle\sigma_5, v \in [0,.3]\rangle, \text{ and}\\ \langle\sigma_6, v \in [0,.15]\rangle.\end{array}$$

Since,

$$\begin{array}{l}[0,.4] \cap [0,.1] \cap [0,.5] = [0,.1],\\ [0,.4] \cap [0,.6] \cap [0,.2] = [0,.2], \text{ and}\\ [0,.5] \cap [0,.15] \cap [0,1] = [0,.15],\end{array}$$

it follows that

$$SOL(\phi) = \{[0,.1], [0,.2], [0,.15]\}.$$

Therefore,

$$Max(SOL(\phi)) = [0,.1] \cup [0,.2] \cup [0,.15] = [0,.2].$$

Note that $n = .2$ closes branch $\phi[v/n]$ and is the result of the c-conjugated pair σ_1, σ_3 and $S^\phi[v/n]$ is $\{\mathsf{T}(A \geq .8), \mathsf{NT}(A \geq .8), \mathsf{T}(B \geq .2), \mathsf{NT}(B \geq .7)\}$.

■

We are ready know to describe our procedure for determining $Maxdeg(\Sigma, A)$. In order to compute $Maxdeg(\Sigma, A)$

1. we start with the set $S_{\Sigma,A} = \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq v)\}$ (we try to find out the maximal value n such that $S_{\Sigma,A}[v/n]$ is not satisfiable);
2. we apply the deduction rules in Table D.13 to $S_{\Sigma,A}$ until the resulting tree is either closed or completed;
3. let $T_{\Sigma,A}$ be the resulting deduction tree, *i.e.* every branch ϕ_i in $T_{\Sigma,A}$ is either closed or completed;
4. for each branch $\phi_i \in T_{\Sigma,A}$, let $r_i = Max(SOL(\phi_i))$, *i.e.* the best solution in order to c-close ϕ_i;
5. as we have to c-close all the branches $\phi_i \in T_{\Sigma,A}$, we have to choose the minimal solution among the r_i, *i.e.* the one which c-closes all the branches: to this purpose, we define

$$\min\max(T_{\Sigma,A}) = \bigcap_{\phi_i \in T_{\Sigma,A}} Max(SOL(\phi_i)). \tag{D.35}$$

6. Let $r_{\Sigma,A} = \min\max(T_{\Sigma,A}) = [0,n]$. n is the maximal degree of truth $Maxdeg(\Sigma, A)$.

Algorithm $MaxVal(\Sigma, A)$ in Table D.14 below describes the above procedure formally.

Algorithm 16 ($MaxVal(\Sigma, A)$)
$MaxVal(\Sigma, A)$ takes as input $\Sigma \subseteq \mathcal{L}^f_+$ and $A \in \mathcal{L}^f_+$. $MaxVal(\Sigma, A)$ returns n iff $Maxdeg(\Sigma, A) = n$. Let the root node be labelled with $S_{\Sigma,A} = \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq v)\}$. The algorithm applies the rules of Table D.13 until each branch in the resulting tree $T_{\Sigma,A}$ is either closed or completed. At each step of the construction of a deduction tree the following steps are performed:

1. select a branch ϕ which is neither completed nor closed. If there is no such branch go to Step 4.;
2. expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed. Let ϕ' be the resulting branch;
3. if ϕ' is neither closed nor completed then
   (a) select a signed expression of type β which is not yet fulfilled in the branch;
   (b) apply rule (PB) and go to Step 1;

   otherwise, go to Step 1.
4. let $T_{\Sigma,A}$ be the resulting tree and let $r_{\Sigma,A} = \min\max(T_{\Sigma,A}) = [0, n]$. Let $MaxVal(\Sigma, A)$ be n and exit.

■

Table D.14: Algorithm $MaxVal(\Sigma, A)$ in $\mathcal{L}^f_+$.

Example 61 Let Σ be fuzzy KB

$$\Sigma = \{ \begin{array}{l} (\mathsf{Gil} \rightarrow \mathsf{Adult} \geq .7), \\ (\mathsf{Adult} \rightarrow \mathsf{Tall} \geq .8), \\ (\mathsf{Karl} \rightarrow \mathsf{Child} \geq .9), \\ (\mathsf{Child} \rightarrow \mathsf{Tall} \geq .2), \\ (\mathsf{Gil} \vee \mathsf{Karl} \geq .4) \ \} \end{array}$$

(see also, Example 53). We are looking for $Maxdeg(\Sigma, \mathsf{Tall})$. Figure D.4 is a deduction tree build during the execution of $MaxVal(\Sigma, A)$, where A is Tall.

Let $T_{S_{\Sigma,A}}$ be the resulting deduction tree and consider the branch ϕ in it. Let σ_1 be $\mathsf{NT}(\mathsf{Tall} \geq v)$, whereas let σ_2 be $\langle \mathsf{T}(\mathsf{Tall} \geq .2), v \in [0, .8] \rangle$. It follows that σ_1 and σ_2 are c-conjugated, as $Sol(Cond(\sigma_1, \sigma_2)) = [0, .2]$. Therefore, $SOL(\phi) = [0, 1] \cap [0, .8] \cap [0, .2] = [0, .2]$. As a consequence, $\min\max(T_{S_{\Sigma,A}}) = [0, .2]$, and $MaxVal(\Sigma, A) := .2$, as expected (in fact, $Maxdeg(\Sigma, A) = .2$). Just note that $T_{S_{\Sigma,A}}[v/.2]$ is a deduction tree for $\mathsf{T}\Sigma \cup \{\mathsf{NT}(\mathsf{Tall} \geq .2)\}$, similar to the one shown in Figure D.3 (Example 53). ■

Example 62 Let Σ be fuzzy KB

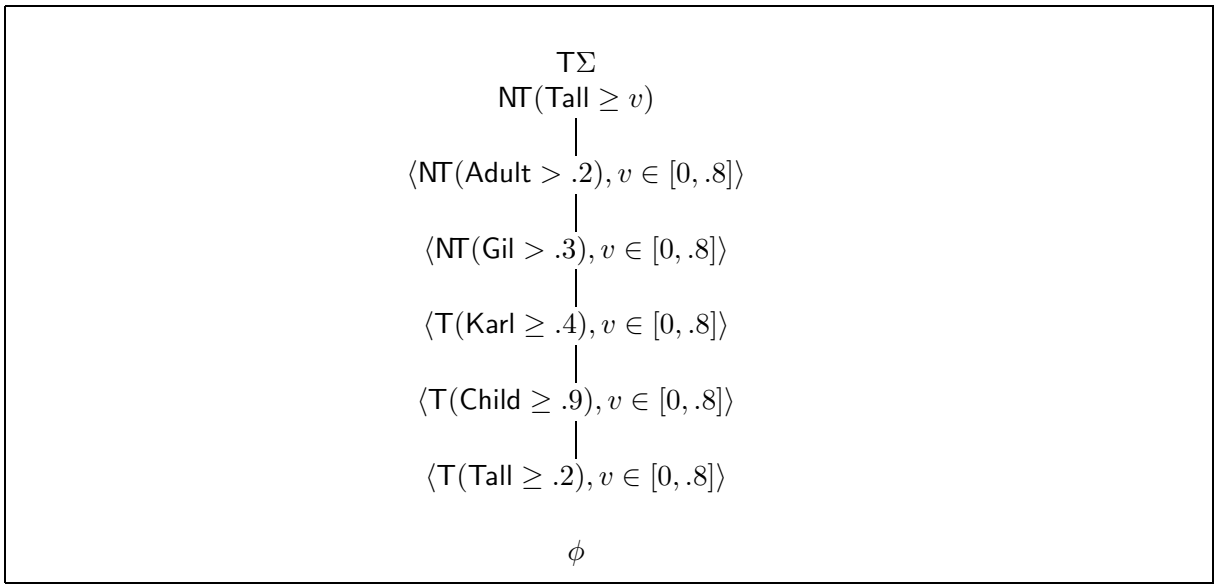


Figure D.4: Example of $MaxVal$ deduction in $\mathcal{L}^f_+$.

$$\begin{aligned}\Sigma = \ \{ \ & (\mathsf{Jon} \rightarrow \mathsf{Student} \wedge \mathsf{Adult} \geq .6),\\ & (\mathsf{Student} \rightarrow \mathsf{Tall} \geq .5),\\ & (\mathsf{Adult} \rightarrow \mathsf{Tall} \geq .8),\\ & (\mathsf{Jon} \geq .3) \ \}\end{aligned}$$

It is easy to see that $\Sigma \not\approx_4 (\mathsf{Tall} \geq n)$, for all $n > 0$. Therefore, $Maxdeg(\Sigma, \mathsf{Tall}) = 0$.

We now show that $MaxVal(\Sigma, \mathsf{Tall})$ returns in fact 0. Consider the deduction tree $T_{S_{\Sigma,\mathsf{Tall}}}$ in Figure D.5, resulting of the application of $MaxVal(\Sigma, \mathsf{Tall})$.
It can be verified that

$$\begin{aligned}&Max(SOL(\phi_1)) = Max(\emptyset) = [0, 0],\\ &Max(SOL(\phi_2)) = [0, .8],\\ &Max(SOL(\phi_3)) = [0, .5],\\ &Max(SOL(\phi_4)) = [0, .8], \text{ and}\\ &Max(SOL(\phi_5)) = [0, .8].\end{aligned}$$

As a consequence,

$$\min \max(T_{\Sigma,A}) = \bigcap_{\phi_i \in T_{\Sigma,A}} Max(SOL(\phi_i)) = [0, 0] \cap [0, .8] \cap [0, .5] \cap [0, .8] \cap [0, .8] = [0, 0].$$

Therefore, $MaxVal(\Sigma, \mathsf{Tall}) = 0$. It is worth noting that, for all $0 < n \leq 1$, by relying on branch ϕ_1, an interpretation $\mathcal{I}$ proving $\Sigma \not\approx_4 (\mathsf{Tall} \geq n)$ can be build. In fact, let $\mathcal{I}$ be such that for all letters A

$$\begin{aligned}&|A|^t = .3 \text{ if } A = \mathsf{Jon},\\ &|A|^t = 0, \text{ for all} A \neq \mathsf{Jon}, \text{ and}\\ &|A|^f = 0.\end{aligned}$$

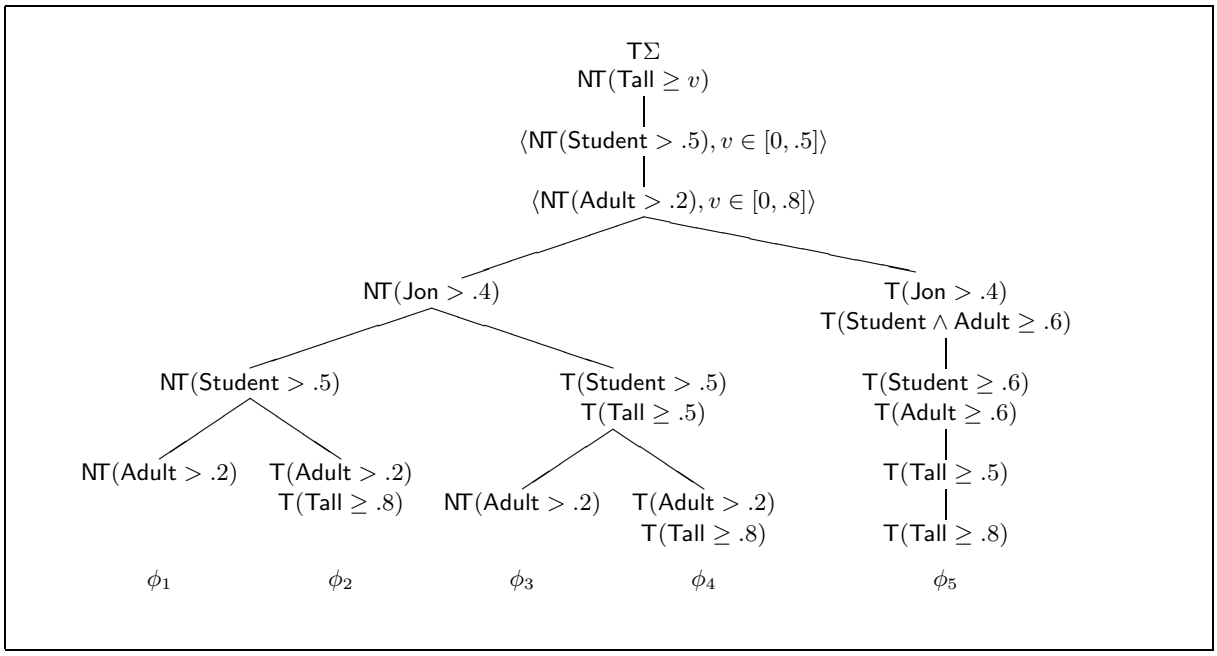


Figure D.5: Example of $MaxVal$ deduction in $\mathcal{L}^f_+$ returning 0.

It is easily verified that $\mathcal{I}$ satisfies $S^{\phi_1}[v/n]$. In particular, $\mathcal{I}$ is a model of both $\mathsf{T}\Sigma \subset S^{\phi_1}[v/n]$, and $\mathsf{NT}(\mathsf{Tall} \geq n) \in S^{\phi_1}[v/n]$. Therefore, $\mathcal{I}$ is a model of Σ not satisfying $(\mathsf{Tall} \geq n)$. ■

Now we are going to proof that $MaxVal(\Sigma, A)$ is working correct, *i.e.* that $MaxVal(\Sigma, A) = Maxdeg(\Sigma, A)$.

Proposition 66 *Let* $\Sigma \subseteq \mathcal{L}^f_+$, $A \in \mathcal{L}_+$ *and* $n \geq 0$. *Then* $Maxdeg(\Sigma, A) = n$ *iff* $MaxVal(\Sigma, A) = n$. ⊣

Proof: The proof is build by means of the following steps. At first we give an alternative and smaller set R_2 of deduction rules which is both correct and complete. We show that $MaxVal(\Sigma, A)$ works well if it relies on R_2. From the observation we will make, it will follows that $MaxVal(\Sigma, A)$ works well also if it relies on (A), $(B1)$, $(B2)$ and (PB).

So, consider our usual set of rules

$$R_0 := \{(A), (B1), (B2), (PB)\}.$$

As we already know, R_0 is equivalent to the set of rules

$$R_1 := \{(A), (PB)\},$$

as the *e.g.* $(B1)$ rule is a shortcut of applying (PB) to β and then closing the left branch. Both R_0 and R_1 are a semantic tableaux. Consider

$$R_2 := \{(A), (PB_\beta)\},$$

where (A) and (PB_β) are

$$(A)\ \frac{\alpha}{\alpha_1, \alpha_2} \qquad (PB_\beta)\ \frac{\beta}{\beta_1 \mid \beta_2}$$

The set of rules R_2 is also known as analytic tableaux.

It is quite easy to see that the *Sat* procedure is correct and complete (in both cases, non fuzzy/fuzzy) whenever it uses sets $R_i, i > 0$ in place of R_0.

In the following we will concentrate our attention to the rules in R_2, applied both to signed fuzzy propositions as to conditioned signed fuzzy propositions. In this latter case, the rules R_2 (derived from Table D.13) are described in Table D.15 below:

$$(A)\ \frac{\langle \alpha, v \in r\rangle}{\langle \alpha_1, v \in r\rangle, \langle \alpha_2, v \in r\rangle}$$

$$(PB_\beta)\ \frac{\langle \beta, v \in r\rangle}{\langle \beta_1, v \in r\rangle \mid \langle \beta_2, v \in r\rangle}$$

Table D.15: Analytic tableaux inference rules for conditioned singed fuzzy propositions.

Now, consider $\Sigma \subseteq \mathcal{L}^f_+$ and $A \in \mathcal{L}_+$ and run $MaxVal(\Sigma, A)$ with the above rule set R_2. Let $T_{\Sigma,A}$ be the resulting deduction tree, obtained from $S_{\Sigma,A} = \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq v)\}$. The following observations can be made:

1. each conditioned signed fuzzy proposition ρ occurring in $T_{\Sigma,A}$ is of the form $\langle \sigma, v \in [0,1]\rangle$. Please note that we use σ as a shortcut for $\langle \sigma, v \in [0,1]\rangle$;

2. as a consequence, for all $n \in (0,1]$, $T_{\Sigma,A}[v/n]$ is a deduction tree for $S_{\Sigma,A}[v/n] = \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq n)\}$;

3. if $\rho = \langle \sigma, v \in [0,1]\rangle$ occurs in $T_{\Sigma,A}$, then σ should have one of the following form ($n \in [0,1]$):

$$\begin{array}{rcl} \sigma_1 & = & \mathsf{T}(A \geq n), \text{ or} \\ \sigma_2 & = & \mathsf{NT}(A > n), \text{ or} \\ \sigma_3 & = & \mathsf{NT}(A \geq v), \text{ or} \\ \sigma_4 & = & \mathsf{T}(A > 1 - v); \end{array}$$

4. combining the σ_i above, it follows that for each branch ϕ in $T_{\Sigma,A}$ and for all $\langle \sigma_i, v \in [0,1]\rangle \in S^\phi$, $\langle \sigma_j, v \in [0,1]\rangle \in S^\phi$, $Sol(Cond(\sigma_i, \sigma_j))$ could be only of the form

$$\begin{array}{rcl} Sol(Cond(\sigma_1, \sigma_2)) & = & [0,1], \text{ or} \\ Sol(Cond(\sigma_1, \sigma_3)) & = & [0,n], \text{ or} \\ Sol(Cond(\sigma_4, \sigma_2)) & = & [0,1-n], \text{ or} \\ Sol(Cond(\sigma_4, \sigma_3)) & = & [0,.5]. \end{array}$$

   Therefore, $Sol(Cond(\sigma_i, \sigma_j))$ can only be of the form $Sol(Cond(\sigma_i, \sigma_j)) = [0,k]$, for some $k \in [0,1]$;

5. for all branches ϕ_i in $T_{\Sigma,A}$, if $Max(SOL(\phi_i)) = [0, k_i]$, where $k_i > 0$ then for all $n \in [0, k_i]$, $\phi_i[v/n]$ is a *closed* branch in $T_{\Sigma,A}[v/n]$. If $k_i = 0$, then $\phi_i[v/n]$ is not closed, for any $n \in [0, 1]$. Moreover,

$$k_i = \max\{n : \phi_i[v/n] \text{ closed}\};$$

6. as a consequence, (i) there is $n > 0$ such that $T_{\Sigma,A}[v/n]$ closed iff

$$\min\max(T_{\Sigma,A}) = [0, k], k > 0$$

and (ii) k is such that

$$k = \max\{n : T_{\Sigma,A}[v/n] \text{ closed}\}.$$

From the last point, $Maxdeg(\Sigma, A) = n$ iff $MaxVal(\Sigma, A) = n$ and $MaxVal$ relies on the set of rules R_2, follows.

Finally, consider the set of rules R_0. By arguments similar to above, it is easy to see that if $T_{\Sigma,A}$ is a deduction tree build by $MaxVal(\Sigma, A)$ by relying on R_0 and $MaxVal(\Sigma, A) = k$, then for all $n \in [0, k]$

- $T_{\Sigma,A}[v/n]$ is a deduction tree for $\mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq n)\}$;
- $T_{\Sigma,A}[v/n]$ is closed iff $k > 0$.

From Point 4. and Point 6. above, we know that $\min\max(T_{\Sigma,A})$ has to be of the form $[0, k]$. Thus any deduction relying on R_0 can be restricted to conditioned signed fuzzy propositions ρ of the form $\langle \sigma, v \in [0, n] \rangle$, so as $Sol(Cond(\sigma_i, \sigma_j))$ can be restricted to the form $[0, n]$. Hence, notwithstanding we to restrict the $(B1)$ rule and the $(B2)$ rule, as described in Table D.13, no solution can be left out. Therefore, $Maxdeg(\Sigma, A) = n$ iff $MaxVal(\Sigma, A) = n$ and $MaxVal$ relies on the set of rules R_0, which concludes the proof. Q.E.D.

From a complexity point of view, the algorithm $MavVal$ behaves as the Sat algorithm.

Proposition 67 *Let $\Sigma \subseteq \mathcal{L}^f_+$, $A \in \mathcal{L}_+$ and $n \in [0, 1]$. Then checking $Maxdeg(\Sigma, A) \geq n$ is a coNP-complete problem.* ⊣

Proof: Let $n \in [0, 1]$. Since $\Sigma \mid\!\!\approx_4 (A \geq n)$ iff $Maxdeg(\Sigma, A) \geq n$, and (see Proposition 62) for all n, deciding $\Sigma \mid\!\!\approx_4 (A \geq n)$ is a coNP-complete problem, coNP-hardness of the $Maxdeg(\Sigma, A) \geq n$ decision problem follows.

The following NP algorithm determines whether $Maxdeg(\Sigma, A) < n$. Non-deterministically generate a deduction tree T for $\mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq v)\}$ by running $MaxVal(\Sigma, A)$: $Maxdeg(\Sigma, A) < n$ iff for some branch ϕ in T, $SOL(\phi) = [0, k]$ and $k < n$. The depth of the branch ϕ is polynomially bounded by the input. Hence, deciding $Maxdeg(\Sigma, A) < n$ is in NP. Therefore, deciding $Maxdeg(\Sigma, A) \geq n$ is in coNP. Q.E.D.

D.2.2.1 The polynomial case $\overline{\mathcal{L}^f_+}$

As we have seen (Proposition 63), given $\Sigma \subseteq \overline{\mathcal{L}^f_+}$, $A \in \overline{\mathcal{L}^f_+}$ and $n > 0$, deciding $\Sigma \approx_4 (A \geq n)$ can be done in time $O(|\Sigma||A|)$. As the algorithm $MaxVal$ behaves as Sat, it is natural to expect that if we restrict our attention to $\overline{\mathcal{L}^f_+}$, and extend $MaxVal$ as we did for $EasyEntail_+$, we obtain an $O(|\Sigma||A|)$ time algorithm too.

Therefore, at first we adapt the $(\overline{B1})$ and $(\overline{B2})$ rules to the case of conditioned signed fuzzy propositions, as shown below in Table D.16. The rules are a generalisation of those described in Table D.9. Just note that *e.g.* with respect to rule $(\overline{B1})$ in Table D.9, the condition

"if for all $1 \leq j \leq h$, $k_j > 1 - k$ ..."

has been replaced with its adaption to the conditioned case

"if for all $1 \leq j \leq h$, $Sol(\lambda_j > 1 - \lambda) = [0, k]$ ..."

Note that $Sol(\lambda_j > 1 - \lambda)$ is the equivalent to $Sol(Cond(\mathsf{NT}(A_j > 1 - \lambda), \mathsf{T}(A_j \geq \lambda_j)))$.

$$(\overline{B1}) \quad \frac{\begin{array}{l}\langle \mathsf{T}(A_1 \wedge \ldots \wedge A_n \rightarrow B \geq \lambda), v \in r_0\rangle, \\ \langle \mathsf{T}(A_1 \geq \lambda_1), v \in r_1\rangle, \ldots, \langle \mathsf{T}(A_h \geq \lambda_h), v \in r_h\rangle, \\ \langle \mathsf{T}(A_1 > \lambda_{h+1}), v \in r_{h+1}\rangle, \ldots, \langle \mathsf{T}(A_n > \lambda_n), v \in r_n\rangle\end{array}}{\langle \mathsf{T}(B \geq \lambda), v \in r\rangle}$$

if for all $1 \leq j \leq h$, $Sol(\lambda_j > 1 - \lambda) = [0, k] = r_{j_k}$, for some $k \in [0, k]$, and
for all $h + 1 \leq j \leq n$, $Sol(\lambda_j \geq 1 - \lambda) = [0, k] = r_{j_k}$, for some $k \in [0, k]$, and
$r = r_0 \cap \bigcap_{1 \leq j \leq n}(r_j \cap r_{j_k})$

$$(\overline{B2}) \quad \frac{\begin{array}{l}\langle \mathsf{T}(A \rightarrow B_1 \vee \ldots \vee B_m \geq \lambda), v \in r_0\rangle, \\ \langle \mathsf{NT}(B_1 \geq \lambda_1), v \in r_1\rangle, \ldots, \langle \mathsf{NT}(B_h \geq \lambda_h), v \in r_h\rangle, \\ \langle \mathsf{NT}(B_{h+1} > \lambda_{h+1}), v \in r_{h+1}\rangle, \ldots, \langle \mathsf{NT}(B_m > \lambda_m), v \in r_m\rangle\end{array}}{\langle \mathsf{NT}(A > 1 - \lambda), v \in r\rangle}$$

if for all $1 \leq j \leq h$, $Sol(\lambda_j \leq \lambda) = [0, k] = r_{j_k}$, for some $k \in [0, k]$, and
for all $h + 1 \leq j \leq m$, $Sol(\lambda_j < \lambda) = [0, k] = r_{j_k}$, for some $k \in [0, k]$, and
$r = r_0 \cap \bigcap_{1 \leq j \leq m}(r_j \cap r_{j_k})$

Table D.16: Modified $(B1)$ and $(B2)$ rules for $EasyMaxVal$ in case $\mathcal{L}^f_+$.

The algorithm $EasyMaxVal$ (see Table D.17) is a combination of algorithm $EasyEntail_+$ and algorithm $MaxVal$ and has the property to run in polynomial time.

Figure D.4 (see Example 61) shows a deduction tree which is the result of an application of the $EasyMaxVal$ algorithm.

Proposition 68 *Let $\Sigma \subseteq \overline{\mathcal{L}^f_+}$, $A \in \overline{\mathcal{L}^f_+}$ and $n \in [0, 1]$. Then checking $Maxdeg(\Sigma, A) \geq n$ can be done in time $O(|\Sigma||A|)$, by running $EasyMaxVal(\Sigma, A)$.* ⊣

D.3 Deciding entailment in HORN-$\mathcal{L}^f$

As we did in Section C.3 and Section C.5 our aim is to investigate decision procedures which are a combination of SLD-derivation and our decision procedure for $\mathcal{L}^f$, Sat.

Algorithm 17 ($EasyMaxVal(\Sigma, A)$)
$EasyMaxVal(\Sigma, A)$ takes as input $\Sigma \subseteq \overline{\mathcal{L}^f_+}$ and $A \in \overline{\mathcal{L}^f_+}$. $EasyMaxVal(\Sigma, A)$ returns n iff $Maxdeg(\Sigma, A) = n$. Let the root node be labelled with $S_{\Sigma,A} = \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq v)\}$. The algorithm applies the rules of Table D.13 until each branch in the resulting tree $T_{\Sigma,A}$ is either closed or completed. At each step of the construction of a deduction tree the following steps are performed:

1. select a branch ϕ which is neither completed nor closed. If there is no such branch go to Step 4.;
2. expand ϕ by means of the rules (A), $(B1)$, $(B2)$(not applied to $\langle \sigma, v \in r \rangle$ if σ is an implication), $(\overline{B1})$ and $(\overline{B2})$ until it becomes AB-completed. Let ϕ' be the resulting branch;
3. if ϕ' is neither closed nor completed then
   (a) select a conditioned signed expression ρ of type β, *where* ρ *is* $\langle \sigma, v \in r \rangle$ *and* σ *involves* NT, which is not yet fulfilled in the branch;
   (b) apply rule (PB) and go to Step 1;

   otherwise, go to Step 1.
4. Let $T_{\Sigma,A}$ be the resulting tree and let $r_{\Sigma,A} = \min\max(T_{\Sigma,A})$. Let $EasyMaxVal(\Sigma, A)$ be n, where $r_{\Sigma,A} = [0, n]$, and exit.

■

Table D.17: Algorithm $EasyMaxVal(\Sigma, A)$.

At first, we define the notions of SLD-derivation and SLD-refutation in HORN-$\mathcal{L}^f$. Let G be a goal of the form

$$\begin{array}{ll} \leftarrow & (A_1 \geq V_1), \ldots, (A_n \geq V_n), \\ & \langle V_{f^1}, f^1(V_{f^1_1}, \ldots, V_{f^1_{n_1}}) \rangle, \ldots, \langle V_{f^k}, f^k(V_{f^k_1}, \ldots, V_{f^k_{n_k}}) \rangle : VR. \end{array}$$

Let E be a horn rule R of the form

$$(A \geq W) \leftarrow (B_1 \geq W_1), \ldots, (B_m \geq W_m), \langle W, f(W_1, \ldots, W_m) \rangle,$$

or a horn fact $(p \geq n)$.

1. If $(A_i \geq V_i)$ is the selected atom in G and if there is a most general unifier (mgu) θ of $(A \geq W)$ and $(A_i \geq V_i)$ (*i.e.* $A = A_i, \theta = \{W/V_i\}$), then the *resolvent* of the goal G and the horn rule R using θ is the goal

$$\begin{array}{ll} \leftarrow & ((A_1 \geq V_1), \ldots, (A_{i-1} \geq V_n), \\ & (B_1 \geq W_1), \ldots, (B_m \geq W_m), \\ & (A_{i+1} \geq V_{i+1}), \ldots, (A_n \geq V_n), \\ & \langle V_{f^1}, f^1(V_{f_1^1}, \ldots, V_{f_{n_1}^1})\rangle, \ldots, \langle V_{f^k}, f^k(V_{f_1^k}, \ldots, V_{f_{n_k}^k})\rangle, \\ & \langle W, f(W_1, \ldots, W_m)\rangle)\theta : VR\theta \cup \{V_i \leq W\}. \end{array}$$

2. If $(A_i \geq V_i)$ is the selected atom in G and if there is a mgu θ of $(p \geq n)$ and $(A_i \geq V_i)$ (*i.e.* $p = A_i, \theta = \{V_i/n\}$), then the *resolvent* of the goal G and the horn fact E is the goal

$$\begin{array}{ll} \leftarrow & ((A_1 \geq V_1), \ldots, (A_{i-1} \geq V_n), \\ & (A_{i+1} \geq V_n), \ldots, (A_n \geq V_n), \\ & \langle V_{f^1}, f^1(V_{f_1^1}, \ldots, V_{f_{n_1}^1})\rangle, \ldots, \langle V_{f^k}, f^k(V_{f_1^k}, \ldots, V_{f_{n_k}^k})\rangle)\theta : VR\theta \cup \{V_i \leq n\}. \end{array}$$

3. If $\langle V_{f^i}, f^i(V_{f_1^i}, \ldots, V_{f_{n_i}^i})\rangle$ is the selected atom in G and, if it is of the form $\langle V_{fi}, n\rangle$, where $n \in [0, 1]$, then for $\theta = \{V_{fi}/n\}$ the *resolvent* of the goal G is the goal

$$\begin{array}{ll} \leftarrow & (A_1 \geq V_1), \ldots, (A_n \geq V_n), \\ & \langle V_{f^1}, f^1(V_{f_1^1}, \ldots, V_{f_{n_1}^1})\rangle, \ldots, \\ & \langle V_{f^{i-1}}, f^{i-1}(V_{f_1^{i-1}}, \ldots, V_{f_{n_{i-1}}^{i-1}})\rangle, \\ & \langle V_{f^{i+1}}, f^{i+1}(V_{f_1^{i+1}}, \ldots, V_{f_{n_{i+1}}^{i+1}})\rangle, \ldots, \\ & \langle V_{f^k}, f^k(V_{f_1^k}, \ldots, V_{f_{n_k}^k})\rangle)\theta : VR\theta \cup \{V_{fi} \leq n\}. \end{array}$$

A *SLD-derivation* for a goal G_0 in a HORN-$\mathcal{L}^f$ KB Σ is a derivation constituted by:

1. a sequence of horn rules and horn facts $E_1, \ldots, E_n$ in Σ;
2. a sequence of mgu's $\theta_1, \ldots, \theta_n$;
3. a sequence of goals $G_0, \ldots, G_n$ such that for each $i \in \{0, \ldots, n-1\}$, G_{i+1} is the resolvent of G_i and E_{i+1} using θ_{i+1}.

A SLD-derivation may terminate with an empty goal in which case the derivation is a *SLD-refutation*.

Let Q be a query $\exists V_1, \ldots, V_n.(A_1 \geq V_1) \wedge \ldots \wedge (A_n \geq V_n)$. An answer θ to a query Q w.r.t. a HORN-$\mathcal{L}^f$ KB Σ is called a *computed answer* if the goal associated with $Q\theta$ has a SLD-refutation in Σ, *i.e.* if θ is the restriction to the variables in Q of the composition $\theta_1\theta_2 \ldots \theta_n$, where $\theta_1, \ldots, \theta_n$ are the mgu's used in the SLD-refutation. The *success set* of Q w.r.t. Σ is defined as

$$SuccessSet(\Sigma, Q) \quad = \quad \{\theta : \theta \text{ computed answer of } Q \text{ w.r.t. } \Sigma\}. \tag{D.36}$$

D.3.1 The case of horn HORN-$\mathcal{L}^f$ KBs

At first, we will concentrate our attention to those HORN-$\mathcal{L}^f$ KBs which are horn and establish correctness and completeness of SLD-refutation.

Let us consider the following examples.

Example 63 Let Σ be the following horn HORN-$\mathcal{L}^f$ KB. With R_i we indicate the ith horn rule and with F_j the jth horn fact.

$$\begin{array}{rcll} \Sigma & = & \{ & R_1: \quad (A_1 \geq V_1) \leftarrow (A_2 \geq V_2), (A_3 \geq V_3), \langle V_1, \min\{V_2, V_3\}\rangle, \\ & & & R_2: \quad (A_2 \geq V_4) \leftarrow (A_4 \geq V_5), (A_5 \geq V_6), \langle V_4, V_5 \cdot V_6\rangle, \\ & & & F_1: \quad (A_4 \geq .5), \\ & & & F_2: \quad (A_5 \geq .8), \\ & & & F_3: \quad (A_3 \geq .7)\} \end{array}$$

Consider the query

$$Q = \exists V.(A_1 \geq V).$$

and the associated goal of Q, G_Q,

$$\leftarrow (A_1 \geq V), \langle V_0, 1\rangle : \emptyset.$$

Below we show that there is a SLD-refutation for goal G_Q in Σ.

(1) $\leftarrow (A_1 \geq V), \langle V_0, 1\rangle : \emptyset$
Associated goal G_Q

(2) $\leftarrow (A_1 \geq V) : \{V_0 \leq 1\}$
$\theta_1 = \{V_0/1\}$

(3) $\leftarrow (A_2 \geq V_2), (A_3 \geq V_3), \langle V, \min\{V_2, V_3\}\rangle : \{V_0 \leq 1, V \leq V_1\}$
$R_1, \theta_2 = \{V_1/V\}$

(4) $\leftarrow (A_4 \geq V_5), (A_5 \geq V_6), (A_3 \geq V_3), \langle V, \min\{V_2, V_3\}\rangle, \langle V_2, V_5 \cdot V_6\rangle : \{V_0 \leq 1, V \leq V_1, V_2 \leq V_4\}$
$R_2, \theta_3 = \{V_4/V_2\}$

(5) $\leftarrow (A_5 \geq V_6), (A_3 \geq V_3), \langle V, \min\{V_2, V_3\}\rangle, \langle V_2, .5 \cdot V_6\rangle : \{V_0 \leq 1, V \leq V_1, V_2 \leq V_4, V_5 \leq .5\}$
$F_1, \theta_4 = \{V_5/.5\}$

(6) $\leftarrow (A_3 \geq V_3), \langle V, \min\{V_2, V_3\}\rangle, \langle V_2, .5 \cdot .8\rangle : \{V_0 \leq 1, V \leq V_1, V_2 \leq V_4, V_5 \leq .5, V_6 \leq .8\}$
$F_2, \theta_5 = \{V_6/.8\}$

(7) $\leftarrow (A_3 \geq V_3), \langle V, \min\{.4, V_3\}\rangle : \{V_0 \leq 1, V \leq V_1, .4 \leq V_4, V_5 \leq .5, V_6 \leq .8, V_2 \leq .4\}$
$\theta_6 = \{V_2/.4\}$

(8) $\leftarrow \langle V, \min\{.4, .7\}\rangle : \{V_0 \leq 1, V \leq V_1, .4 \leq V_4, V_5 \leq .5, V_6 \leq .8, V_2 \leq .4, V_3 \leq .7\}$
$F_3, \theta_7 = \{V_3/.7\}$

(9) $\leftarrow \blacksquare : \{V_0 \leq 1, .4 \leq V_1, .4 \leq V_4, V_5 \leq .5, V_5 \leq .5, V_2 \leq .4, V_3 \leq .7, V \leq .4\}$
$\theta_8 = \{V/.4\}$

The computed answer θ is the restriction to the fuzzy variable V of $\theta_1\theta_2 \cdots \theta_8$, *i.e.* $\theta = \{V/.4\}$. Now, $Q\theta$ is $(A_1 \geq .4)$. It can be verified that $\Sigma \models_4 (A_1 \geq .4)$. It follows that θ is a correct answer. ■

Example 64 Let Σ be the following horn HORN-$\mathcal{L}^f$ KB.

$$\begin{array}{rcll} \Sigma & = & \{ & R_1: \quad (A_1 \geq V_1) \leftarrow (A_2 \geq V_2), \langle V_1, \sqrt{V_2}\rangle, \\ & & & F_1: \quad (A_2 \geq .4), \\ & & & F_2: \quad (A_2 \geq .9)\} \end{array}$$

Consider the query

$$Q = \exists V.(A_1 \geq V).$$

The associated goal of Q is

$$G_Q = \leftarrow (A_1 \geq V), \langle V_0, 1\rangle : \emptyset.$$

Below we show that there are two SLD-refutations for goal G_Q in Σ.

(1)	$\leftarrow (A_1 \geq V), \langle V_0, 1\rangle : \emptyset$	Associated goal G_Q
(2)	$\leftarrow (A_1 \geq V) : \{V_0 \leq 1\}$	$\theta_{1_1} = \{V_0/1\}$
(3)	$\leftarrow (A_2 \geq V_2), \langle V, \sqrt{V_2}\rangle : \{V_0 \leq 1, V \leq V_1\}$	$R_1, \theta_{1_2} = \{V_1/V\}$
(4)	$\leftarrow \langle V, \sqrt{.4}\rangle : \{V_0 \leq 1, V \leq V_1, V_2 \leq .4\}$	$F_1, \theta_{1_3} = \{V_2/.4\}$
(5)	$\leftarrow \blacksquare : \{V_0 \leq 1, V \leq V_1, V_2 \leq .4, V \leq \sqrt{.4}\}$	$\theta_{1_4} = \{V/\sqrt{.4}\}$

(1)	$\leftarrow (A_1 \geq V), \langle V_0, 1\rangle : \emptyset$	Associated goal G_Q
(2)	$\leftarrow (A_1 \geq V) : \{V_0 = 1\}$	$\theta_{2_1} = \{V_0/1\}$
(3)	$\leftarrow (A_2 \geq V_2), \langle V, \sqrt{V_2}\rangle : \{V_0 = 1, V \leq V_1\}$	$R_1, \theta_{2_2} = \{V_1/V\}$
(4)	$\leftarrow \langle V, \sqrt{.9}\rangle : \{V_0 = 1, V \leq V_1, V_2 \leq .9\}$	$F_2, \theta_{2_3} = \{V_2/.9\}$
(5)	$\leftarrow \blacksquare : \{V_0 = 1, V \leq V_1, V_2 \leq .9, V = \sqrt{.9}\}$	$\theta_{2_4} = \{V/\sqrt{.9}\}$

The success set is $SuccessSet(\Sigma, Q) = \{\theta_1, \theta_2\}$, where $\theta_1 = \{V/\sqrt{.4}\}$ and $\theta_2 = \{V/\sqrt{.9}\}$. Moreover, each computed answer θ_i is a correct answer, *i.e.* $\Sigma \models_4 Q\theta_i$ and $Maxdeg(\Sigma, Q) = \theta_2$. Since $\theta_1 < \theta_2$, it follows that $Maxdeg(\Sigma, Q) = \uparrow SuccessSet(\Sigma, Q)$. It follows that for each correct answer θ there is the computed answer $\theta_2 = \uparrow SuccessSet(\Sigma, Q)$ such that $\theta_2 \geq \theta_1$. For instance, for $\theta = \{V/.4\}$ it is easily verified that θ is a correct answer to the query Q w.r.t. Σ, *i.e.* $\Sigma \models_4 (A_1 \geq .4)$ and $\theta_2 \geq \theta$ holds ($\sqrt{.9} \geq .4$). ■

In the following example we show that the condition of being nondecreasing for fuzzy degree functions is necessary in order to preserve correctness of the calculus.

Example 65 Let Σ be the following horn HORN-$\mathcal{L}^f$ KB.

$$\begin{array}{rcll} \Sigma & = & \{ & R_1 : \quad (A_1 \geq V_1) \leftarrow (A_2 \geq V_2), \langle V_1, 1 - V_2\rangle, \\ & & & F_1 : \quad (A_2 \geq .8)\} \end{array}$$

Just notice here that the function $f(V_2) = 1 - V_2$ is a decreasing function. Consider the query

$$Q = \exists V.(A_1 \geq V).$$

The associated goal of Q is

$$G_Q = \leftarrow (A_1 \geq V), \langle V_0, 1\rangle : \emptyset.$$

Below we show that there is a SLD-refutations for goal G_Q in Σ.

(1)	$\leftarrow (A_1 \geq V), \langle V_0, 1\rangle : \emptyset$	Associated goal G_Q
(2)	$\leftarrow (A_1 \geq V) : \{V_0 = 1\}$	$\theta_1 = \{V_0/1\}$
(3)	$\leftarrow (A_2 \geq V_2), \langle V, 1 - V_2\rangle : \{V_0 = 1, V \leq V_1\}$	$R_1, \theta_2 = \{V_1/V\}$
(4)	$\leftarrow \langle V, 1 - .8\rangle : \{V_0 = 1, V \leq V_1, V_2 \leq .8\}$	$F_1, \theta_3 = \{V_2/.8\}$
(5)	$\leftarrow \blacksquare : \{V_0 = 1, V \leq V_1, V_2 \leq .8, V = .2\}$	$\theta_4 = \{V/.2\}$

The computed answer is $\theta = \{V/.2\}$. But, $\Sigma \not\approx_4 (A_1 \geq 1)$. In fact, suppose that $\Sigma \not\approx_4 (A_1 \geq 1)$, *i.e.* there is an interpretation $\mathcal{I}$ satisfying Σ such that $|A_1|^t < 1$. Let

$$V_1^{\mathcal{I}} = \frac{|A_1|^t+1}{2}$$
$$V_2^{\mathcal{I}} = \frac{1-|A_1|^t}{2}.$$

Note that $V_1^{\mathcal{I}} \leq 1 - V_2^{\mathcal{I}}$, $V_1^{\mathcal{I}} > |A_1|^t$ and $V_2^{\mathcal{I}} < .8$. It follows that $\mathcal{I}$ does not satisfy R_1, and thus, $\mathcal{I}$ does not satisfy Σ, contrary to our assumption. Hence, the set of computed answer θ is *not complete*. ■

Finally, the following example shows that some termination problems could arise with recursive horn HORN-$\mathcal{L}^f$ KBs.

Example 66 Let Σ be the following recursive horn HORN-$\mathcal{L}^f$ KB.

$$\begin{array}{rcll} \Sigma & = & \{ \ R_1: & (A_1 \geq V_1) \leftarrow (A_1 \geq V_2), \langle V_1, \frac{V_2+1}{2} \rangle, \\ & & F_1: & (A_1 \geq m)\} \end{array}$$

It can be verified that $\Sigma \not\approx_4 (A_1 \geq n)$, for all $n \geq m$. In particular, $Maxdeg(\Sigma, Q) = 1$. But, consider the query $Q = \exists V.(A_1 \geq V)$. The associated goal of Q is $G_Q = \leftarrow (A_1 \geq V), \langle V_0, 1\rangle : \emptyset$.

Of course, $\theta_0 = \{V/m\}$ is a correct answer. Moreover, it can easily be verified that for each $k \geq 1$, $\theta_k = \{V/\frac{m+2^k-1}{2^k}\}$ is a correct answer too. Therefore, the success set of Q w.r.t. Σ is the infinite set $S = \{\theta_k : k \geq 0\}$. As a consequence, we are unable to compute the success set completely, *i.e.* we cannot compute $Maxdeg(\Sigma, Q)$. Just note that $Maxdeg(\Sigma, Q) = \uparrow S = \uparrow \{\theta_k : k \geq 0\}$. Since θ_k is increasing in k, it follows that

$$Maxdeg(\Sigma, Q) = \lim_{k\to\infty} \frac{m + 2^k - 1}{2^k} = 1.$$

■

The examples above lead us to the convincement that computed answers of a query Q w.r.t. a KB Σ are correct and, if Σ is not recursive, that for each correct answer θ_1 there is a computed answer θ_2 such that $\theta_1 \leq \theta_2$. We can confirm this by means of the following proposition.

Proposition 69 *Let Σ be a horn* HORN-$\mathcal{L}^f$ *KB and let Q be a query:*

1. *every computed answer θ of Q w.r.t. Σ is a correct answer (correctness),* i.e.

$$SuccessSet(\Sigma, Q) \subseteq AnswerSet(\Sigma, Q);$$

2. *if Σ is not recursive then for every correct answer θ_1 of Q w.r.t. Σ there is a computed answer θ_2 of Q w.r.t. Σ such that $\theta_1 \leq \theta_2$ (completeness).* ⊣

Proof: (Sketch) We first consider **Point 1**. Let G be a goal, E be a horn rule or a horn fact. Let G' be the resolvent of the goal G and E by using mgu θ. We show that if $\mathcal{I}$ is an interpretation satisfying both G and E, the $\mathcal{I}$ satisfies G'. In order to ease the proof, we will restrict G and E to a simple form. The generalisation is straightforward. We have 3 cases to consider.

1. Let G be a goal of the form

$$\leftarrow \quad (A_1 \geq V_1), (A_2 \geq V_2), \langle V, f^1(V_1, V_2)\rangle : \emptyset$$

and let E be a horn rule of the form

$$(A_1 \geq V_3) \leftarrow (A_3 \geq V_4), \langle V_3, f^2(V_4)\rangle.$$

Let $(A_1 \geq V_1)$ be the selected atom in G. The mgu is $\theta = \{V_3/V_1\}$. By definition, the resolvent G' of G and E using θ is

$$\leftarrow \quad (A_3 \geq V_4), (A_2 \geq V_2), \langle V, f^1(V_1, V_2)\rangle, \langle V_1, f^2(V_4)\rangle : \{V_1 \leq V_3\}.$$

Let $\mathcal{I}$ be an interpretation satisfying G and E. We show that $\mathcal{I}$ satisfies G'. Let $\vec{m} \in [0,1]^5$ such that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies $\{V_1 \leq V_3\}$. We show that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy some atom of G'.

Assume that $\vec{m} = (m, m_1, m_2, m_3, m_4)$, $\vec{V} = (V, V_1, V_2, V_3, V_4)$. Since $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies $\{V_1 \leq V_3\}$, it follows that $m_1 \leq m_3$. $\mathcal{I}$ satisfies G. Therefore, for $m \leq f^1(m_1, m_2)$ (i) the case where $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_2 \geq V_2)$ is trivial; (ii) if $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_1 \geq V_1)$, from $m_1 \leq m_3$ it follows that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_1 \geq V_3)$. But $\mathcal{I}$ satisfies E. Therefore, for $m_3 \leq f^2(m_4)$, $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_3 \geq V_4)$. But $m_1 \leq m_3$ and, thus, for $m_1 \leq f^2(m_4)$ $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_3 \geq V_4)$ which completes the case.

2. Let G be a goal of the form

$$\leftarrow \quad (A_1 \geq V_1), (A_2 \geq V_2), \langle V, f(V_1, V_2)\rangle : \emptyset$$

and let E be a horn fact of the form

$$(A_1 \geq n).$$

The mgu is $\theta = \{V_1/n\}$. By definition, the resolvent G' of G and E using θ is

$$\leftarrow \quad (A_2 \geq V_2), \langle V, f(n, V_2)\rangle : \{V_1 \leq n\}.$$

Let $\mathcal{I}$ be an interpretation satisfying G and E. We show that $\mathcal{I}$ satisfies G'. Let $\vec{m} \in [0,1]^3$ such that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies $\{V_1 \leq n\}$. We show that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy some atom of G'.

Assume that $\vec{m} = (m, m_1, m_2)$, $\vec{V} = (V, V_1, V_2)$. Since $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies $\{V_1 \leq n\}$, it follows that $m_1 \leq n$. $\mathcal{I}$ satisfies G. Therefore, for $m \leq f(m_1, m_2)$ if $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_1 \geq V_1)$, from $m_1 \leq n$ it follows that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_1 \geq n)$, contrary to the assumption that $\mathcal{I}$ satisfies $(A_1 \geq n)$. Hence, $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_2 \geq V_2)$ or $\langle V, f(V_1, V_2)\rangle$. Therefore, for $m \leq f(m_1, m_2)$ $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_2 \geq V_2)$. Since $m_1 \leq n$ and f is nondecreasing in all its arguments, it follows that $m < f(n, m_2)$. Therefore, for $m \leq f(n, m_2)$ $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_2 \geq V_2)$, which completes the case.

3. Let G be a goal of the form

$$\leftarrow \quad (A_1 \geq V_1), \langle V, f(V_1, V_2)\rangle, \langle V_2, n\rangle : \emptyset$$

The mgu is $\theta = \{V_2/n\}$. By definition, the resolvent G' of G using θ is

$$\leftarrow \quad (A_1 \geq V_1), \langle V, f(V_1, n)\rangle : \{V_2 \leq n\}$$

Let $\mathcal{I}$ be an interpretation satisfying G. We show that $\mathcal{I}$ satisfies G'. Let $\vec{m} \in [0,1]^3$ such that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies $\{V_2 \leq n\}$. We show that $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy some atom of G'.

Assume that $\vec{m} = (m, m_1, m_2)$, $\vec{V} = (V, V_1, V_2)$. Since $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies $\{V_2 \leq n\}$, it follows that $m_2 \leq n$. $\mathcal{I}$ satisfies G. Therefore, $\mathcal{I}^{\vec{m}}_{\vec{V}}$ satisfies $\langle V_2, n\rangle$. Hence, $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_1 \geq V_1)$ or $\langle V, f(V_1, V_2)\rangle$. But, for $m \leq f(m_1, m_2)$ $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_1 \geq V_1)$. But, $m_2 \leq n$, and thus, $m < f(m_1, n)$. As a consequence, for $m < f(m_1, n)$ $\mathcal{I}^{\vec{m}}_{\vec{V}}$ does not satisfy $(A_1 \geq V_1)$, which completes the case.

For **Point 2.**, we will only give a sketch of the proof. Essentially, the proof follows the completeness proof for logic programs (see *e.g.* [194]).

A *fuzzy Herbrand base* w.r.t. Σ is a set of horn facts $(p \geq n)$ such that p occurs in Σ. We will indicate it with B_Σ. A *fuzzy Herbrand interpretation* w.r.t. Σ is a set $M \subseteq B_\Sigma$. We will say that M is *maximal* iff for no $(p \geq n_1) \in M$ there is $(p \geq n_2) \in M$ such that $n_2 > n_1$. A *fuzzy Herbrand model* of horn HORN-$\mathcal{L}^f$ KB Σ is a fuzzy Herbrand interpretation satisfying Σ. Of course, for each fuzzy Herbrand model M of Σ, the maximal fuzzy Herbrand model $MAX(M)$, of Σ is such that $MAX(M) \subseteq M$, where $MAX(M) = \{(p \geq n) \in M : n = \max\{m : (p \geq m) \in M\}\}$. Clearly, if $\{M_i\}_{i\in I}$ is a set of Herbrand models of Σ, then $\bigcap_{i\in I} M_i$ is a fuzzy Herbrand model, called the *least fuzzy Herbrand model*. With M_Σ we indicate the maximal least fuzzy Herbrand model of Σ.

Similarly to the non fuzzy case, it can be shown that

Claim 1 *The maximal least fuzzy Herbrand model of Σ, if Σ is not recursive, is $M_\Sigma = \{(p \geq n) : n = Maxdeg(\Sigma, p)\}$.*

Now, consider the following mapping

$$T_\Sigma : 2^{B_\Sigma} \rightarrow 2^{B_\Sigma}$$

defined as follows. Let M be a fuzzy Herbrand interpretation w.r.t. Σ. Then if $M \neq \emptyset$ then

$$\begin{aligned} T_\Sigma(M) \quad = \quad \{(p \geq k) \in B_\Sigma : (A \geq V) \leftarrow (A_1 \geq V_1), \ldots, (A_n \geq V_n), \langle V, f(V_1, \ldots, V_n)\rangle \in \Sigma_R, \\ (A_1 \geq m_1), \ldots, (A_n \geq m_n) \in M, \\ k = f(m_1, \ldots, m_n)\}. \end{aligned}$$

If $M = \emptyset$ then

$$T_\Sigma(\emptyset) \quad = \quad \{(p \geq n) \in \Sigma_F\}$$

Let T_Σ^k defined as usual:

$$T_\Sigma^1(M) = T_\Sigma(M)$$
$$T_\Sigma^{k+1}(M) = T_\Sigma(T_\Sigma^k(M))$$

Similarly to the non fuzzy case, it can be shown that

Claim 2 *Let Σ be a not recursive horn* HORN-$\mathcal{L}^f$ *KB. Then the maximal least fuzzy Herbrand model of Σ, M_Σ, is such that*

$$M_\Sigma = MAX(lfp(T_\Sigma) = MAX(\lim_{n\to\infty} T_\Sigma^n(\emptyset)),$$

where $lfp(T_\Sigma)$ is the least fixpoint of T_Σ.

This give us a way to prove Point 2. Suppose $\Sigma \mid\!\!\approx_4 (A \geq n)$. Then we know from Claim 1 that there is $(A \geq m) \in M_\Sigma$ such that $m \geq n$. But, from Claim 2 it follows that $(A \geq m) \in MAX(\lim_{n\to\infty} T_\Sigma(\emptyset))$. Hence, there is a $k \geq 1$ such that $(A \geq n) \in MAX(T_\Sigma^k(\emptyset))$. From this it is easily verified that $\leftarrow (A \geq m)$ has a SLD-refutation in Σ, proving Point 2. Q.E.D.

Proposition 70 *Let Σ be a not recursive horn* HORN-$\mathcal{L}^f$ *KB and let Q be a query. Then*

$$Maxdeg(\Sigma, Q) = \uparrow SuccessSet(\Sigma, Q).$$

⊣

Proof: Form Proposition 69 we have that for each correct answer θ_1, there is a computed answer θ_2 such that $\theta_1 \leq \theta_2$. Now consider $\theta_1 = Maxdeg(\Sigma, Q)$. By definition, $\theta_1 \in AnswerSet(\Sigma, Q)$ and $\theta_1 = \uparrow AnswerSet(\Sigma, Q)$. Therefore, there is $\theta_2 \in SuccessSet(\Sigma, Q)$ such that $\theta_1 \leq \theta_2$. Let $\theta_3 = \uparrow SuccessSet(\Sigma, Q)$. Therefore, $\theta_1 \leq \theta_3$. But, $SuccessSet(\Sigma, Q) \subseteq AnswerSet(\Sigma, Q)$, and thus, $\theta_3 = \uparrow SuccessSet(\Sigma, Q) \leq \uparrow AnswerSet(\Sigma, Q) = \theta_1$. Therefore, $Maxdeg(\Sigma, Q) = \uparrow AnswerSet(\Sigma, Q) = \theta_1 = \theta_3 = \uparrow SuccessSet(\Sigma, Q)$. Q.E.D.

Proposition 70 give us immediately a first method in order to compute the maximal degree of truth of a query Q w.r.t. a not recursive KB Σ.

In the following we will present an alternative method in order to determine whether $\Sigma \mid\!\!\approx_4 Q$. Essentially, we transform Σ and Q into a first-order logic program $\varphi(\Sigma)$ and a first-order query $\varphi(Q)$, respectively, in such a way that $\Sigma \mid\!\!\approx_4 Q$ can be decided in terms of $\varphi(\Sigma) \models_2 \varphi(Q)$. As a consequence, we can decide $\Sigma \mid\!\!\approx_4 Q$ by relying on a standard first-order prolog system.

For each letter $A \in \mathcal{L}$ consider an unary first-order predicate $A(\cdot)$ and let Σ be a horn HORN-$\mathcal{L}^f$ KB: φ is the following function.

1. Let R be a horn rule of the form

$$(A \geq V) \leftarrow (A_1 \geq V_1), \ldots, (A_n \geq V_n), \langle V, f(V_1, \ldots, V_n)\rangle.$$

Then

$$\varphi(R) \quad = \quad A(V) \leftarrow A_1(V_1), \ldots, A_n(V_n), V = f(V_1, \ldots, V_n).$$

2. Let γ be a horn fact (*i.e.* a fuzzy proposition) of the form $(p \geq n)$. Then

$$\varphi(\gamma) \quad = \quad P(n).$$

3. Let Q be a query of the form

$$\exists V_1, \ldots, V_n.(A_1 \geq V_1) \wedge \ldots \wedge (A_n \geq V_n).$$

Then

$$\varphi(Q) \quad = \quad \exists V_1, \ldots, V_n.A_1(V_1) \wedge \ldots \wedge A_n(V_n).$$

4. Let Σ be a horn HORN-$\mathcal{L}^f$ KB. Then

$$\begin{array}{lcl} \varphi(\Sigma_F) & = & \{\varphi(\gamma) \colon \gamma \in \Sigma_F\} \\ \varphi(\Sigma_R) & = & \{\varphi(R) \colon R \in \Sigma_R\} \\ \varphi(\Sigma) & = & \varphi(\Sigma_R) \cup \varphi(\Sigma_F). \end{array}$$

Essentially, φ transforms horn HORN-$\mathcal{L}^f$ KBs and queries into First-Order logic programs and logic program queries, respectively. Notice that we have to restrict the fuzzy degree function to those case for which $V = f(V_1, \ldots, V_n)$ can be expressed in logic programming.

Example 67 Consider Σ and Q in Example 64. By definition, $\varphi(\Sigma)$ and $\varphi(Q)$ are

$$\begin{array}{lcll} \varphi(\Sigma) & = & \{ & R_1 : \quad A_1(V_1) \leftarrow A_2(V_2), V_1 = \sqrt{V_2}, \\ & & & F_1 : \quad A_2(.4), \\ & & & F_2 : \quad A_2(.9)\} \end{array}$$

and

$$\varphi(Q) = \exists V.A_1(V).$$

The associated goal of $\varphi(Q)$ is

$$G_{\varphi(Q)} = \leftarrow A_1(V).$$

Below we show that there are two SLD-refutations for goal $G_{\varphi(Q)}$ in $\varphi(\Sigma)$.

(1)	$\leftarrow A_1(V)$	Associated goal $G_{\varphi(Q)}$
(2)	$\leftarrow A_2(V_2), V = \sqrt{V_2}$	$R_1, \theta_{1_1} = \{V_1/V\}$
(3)	$\leftarrow V = \sqrt{.4}$	$F_1, \theta_{1_2} = \{V_2/.4\}$
(4)	$\leftarrow \blacksquare$	$\theta_{1_3} = \{V/\sqrt{.4}\}$

(1)	$\leftarrow A_1(V)$	Associated goal $G_{\varphi(Q)}$
(2)	$\leftarrow A_2(V_2), V = \sqrt{V_2}$	$R_1, \theta_{2_1} = \{V_1/V\}$
(3)	$\leftarrow V = \sqrt{.9}$	$F_1, \theta_{2_2} = \{V_2/.9\}$
(4)	$\leftarrow \blacksquare$	$\theta_{2_3} = \{V/\sqrt{.9}\}$

The success set is $S = \{\theta_1, \theta_2\}$, where $\theta_1 = \{V/\sqrt{.4}\}$ and $\theta_2 = \{V/\sqrt{.9}\}$. Notice that each computed answer θ_i is also a computed answer in Example 64. This is justified by the fact that there is a easy to see correspondence between the SLD-refutations in Example 64 and the SLD-refutations above. ■

The following proposition shows that each computed answer θ w.r.t. SLD-derivation in horn HORN-$\mathcal{L}^f$ is also a computed answer w.r.t. to the transformation φ and vice-versa.

Proposition 71 *Let Σ be a horn* HORN-$\mathcal{L}^f$ *KB, let Q be a query. Then*

$$SuccessSet(\Sigma, Q) = SuccessSet(\varphi(\Sigma), \varphi(Q)).$$

⊣

Proof: (Sketch). The proof is given by induction on the length n of SLD-refutations. It is easily verified that there is a bijection between SLD-refutations for $\leftarrow G_Q$ w.r.t. Σ and SLD-refutations for $\leftarrow G_{\varphi(Q)}$ w.r.t. $\varphi(\Sigma)$. Q.E.D.

The above proposition has an enormous impact from an implementation point of view. It mainly says that rather than building a new engine, we can rely on an existing one: namely on a standard prolog system. It allows us also to compute $Maxdeg(\Sigma, Q)$ according to Proposition 70 and Proposition 71.

D.3.2 The case of generic HORN-$\mathcal{L}^f$ KBs

Suppose that Σ is a generic HORN-$\mathcal{L}^f$ KBs and Q is a query. How can we determine $Maxdeg(\Sigma, Q)$? We will see that we can apply Method 2 or Method 4 as in case of $\mathcal{L}$ (see Page 179), by relying on Proposition 52, Proposition 53, Proposition 25 and Proposition 26.

Proposition 72 *Let Σ be a* HORN-$\mathcal{L}^f$ *KB and let Q be a query. Then $\Sigma \mid\!\approx_4 Q$ if either*

1. *there is a SLD-refutation for goal G_Q in Σ; or*
2. *there are $n \geq 1$ SLD-derivations for goal G_Q in Σ ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, such that for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}}$ belonging to $Completions(\mathsf{T}\Sigma_F)$, $\mathcal{I}$ does satisfy some Q_i, for $1 \leq i \leq n$.*

If Σ is not recursive then the only if direction (completeness) holds too. ⊣

Combining Proposition 72 with Proposition 71, the following methods determine whether $\Sigma \mid\!\approx_4 Q$ (the methods are complete if Σ is not recursive).

Method 2: Collect $n \geq 1$ SLD-derivations of G_Q in Σ, ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, until (i) there is an empty goal; or, (ii) for all four-valued canonical models $\mathcal{I}$ of four-valued completions $\tilde{\mathcal{S}} \in Completions(\mathsf{T}\Sigma_F)$, $\mathcal{I}$ does satisfy some Q_i, for $1 \leq i \leq n$. In the worst case we have to compute all SLD-derivations.

Method 4.1: Compute $Completions(\mathsf{T}\Sigma_F)$. Determine whether for all $\tilde{\mathcal{S}} \in Completions(\mathsf{T}\Sigma_F)$, G_Q has a SLD-refutation in $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$, *i.e.* whether $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R \mid\!\approx_4 Q$. Note that $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$ is a horn HORN-$\mathcal{L}^f$ KB.

Method 4.2: Compute $Completions(\mathsf{T}\Sigma_F)$. Determine whether for all $\tilde{\mathcal{S}} \in Completions(\mathsf{T}\Sigma_F)$, $G_{\varphi}(Q)$ has a SLD-refutation in $\varphi(\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R)$, *i.e.* whether $\varphi(\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R) \models_2 \varphi(Q)$. Note that $\varphi(\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R)$ is a first-order horn KB.

Example 68 Consider

$$\begin{array}{rcll}\Sigma & = & \{ & R_1: \quad (A_1 \geq V_1) \leftarrow (A_2 \geq V_2), \langle V_1, V_2 \rangle, \\ & & & R_2: \quad (A_1 \geq V_3) \leftarrow (A_3 \geq V_4), \langle V_3, \sqrt{V_4} \rangle, \\ & & & F_1: \quad (A_2 \vee A_3 \geq .4)\}\end{array}$$

and query

$$Q = \exists V.(A_1 \geq V_1).$$

We would like to verify that $Maxdeg(\Sigma, Q) = \{V/.4\}$. It can be verified that $Completions(\mathsf{T}\Sigma_F) = \{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2\}$, where

$$\begin{array}{l}\tilde{\mathcal{S}}_1 = \{\mathsf{T}(A_2 \geq .4)\}, \\ \tilde{\mathcal{S}}_1 = \{\mathsf{NT}(A_2 \geq .4), \mathsf{T}(A_3 \geq .4)\}.\end{array}$$

It follows that $\Sigma^+_{\tilde{\mathcal{S}}_1} = \{(A_2 \geq .4)\}$, whereas $\Sigma^+_{\tilde{\mathcal{S}}_2} = \{(A_3 \geq .4)\}$.

As a consequence, the success set of Q w.r.t. $\Sigma^+_{\tilde{\mathcal{S}}_1} \cup \Sigma_R$ is $\{\theta_1\}$ ($\theta_1 = \{V/.4\}$), whereas the success set of Q w.r.t. $\Sigma^+_{\tilde{\mathcal{S}}_2} \cup \Sigma_R$ is $\{\theta_2\}$ ($\theta_2 = \{V/\sqrt{.4}\}$) (similarly for the success set of $\varphi(Q)$ w.r.t. $\varphi(\Sigma^+_{\tilde{\mathcal{S}}_i} \cup \Sigma_R)$, $i = 1, 2$). What is $Maxdeg(\Sigma, Q)$? Clearly,

$$Maxdeg(\Sigma, Q) = Maxdeg(\Sigma^+_{\tilde{\mathcal{S}}_1} \cup \Sigma_R, Q) \downarrow Maxdeg(\Sigma^+_{\tilde{\mathcal{S}}_2} \cup \Sigma_R, Q).$$

Since $\sqrt{.4} > .4$, it follows that $Maxdeg(\Sigma, Q) = \theta_1$

Finally, the following proposition shows us how to compute $Maxdeg(\Sigma, Q)$.

Proposition 73 *Let Σ be a not recursive* HORN-$\mathcal{L}^f$ *KB and let Q be a query. Let $Completions(\mathsf{T}\Sigma_F) = \{\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_k\}$. Then*

$$\begin{array}{rcl}Maxdeg(\Sigma, Q) & = & \downarrow_{1 \leq i \leq k} \{Maxdeg(\Sigma^+_{\tilde{\mathcal{S}}_i} \cup \Sigma_R, Q)\} \\ & = & \downarrow_{1 \leq i \leq k} \{Maxdeg(\varphi(\Sigma^+_{\tilde{\mathcal{S}}_i} \cup \Sigma_R), \varphi(Q))\}\end{array}$$

■

Notice that from an implementation point of view, certainly the last point is the easiest to implement (corresponds to Method 4.2).

D.4 Deciding entailment in fuzzy $\mathcal{ALC}$

The decision procedures we will see in this section are mainly a simple combination of those seen in Section D.2 about the four-valued fuzzy propositional logic $\mathcal{L}^f_+$ and those seen in Section C.4 about crisp four-valued $\mathcal{ALC}$. The decision procedure we will develop is restricted to the case of well formed KBs only. Hence, our purpose is to device a decision procedure for determining $\Sigma \mid\!\!\approx_4 (A \geq n)$, where A is an assertion an Σ is a well formed fuzzy KB. The procedure is an extension of the one for deciding entailment in $\mathcal{L}^f_+$ to the fuzzy $\mathcal{ALC}$ case.

In the following, *signed fuzzy formulae* are expressions of type

$$\begin{array}{l} \mathsf{T}(A \geq n), \\ \mathsf{NT}(A \geq n), \\ \mathsf{T}(A > n), \text{ and} \\ \mathsf{NT}(A > n), \end{array} \tag{D.37}$$

where A is an assertion and $n > 0$. Moreover, we will assume that each specialisation $C \to D$ is a signed fuzzy formula, too. We will use, as usual, σ as a metavariable for signed fuzzy formulae. In the following, we will use γ as metavariable for fuzzy expressions of type $(A \geq n)$ or $(A > n)$.

We extend interpretation functions in such a way that it has also to satisfy

$$\begin{array}{lll} t \in (A > n)^{\mathcal{I}} & \text{iff} & |A|^t > n \\ f \in (A > n)^{\mathcal{I}} & \text{iff} & |A|^f > n \end{array} \tag{D.38}$$

Of course, an interpretation $\mathcal{I}$ *satisfies* $(A > n)$ iff $t \in (A > n)^{\mathcal{I}}$. Now, an interpretation $\mathcal{I}$ *satisfies* a signed fuzzy formula σ iff

$$\begin{array}{ll} \mathcal{I} \text{ satisfies } \gamma & \text{if } \sigma = \mathsf{T}\gamma \\ \mathcal{I} \text{ does not satisfy } \gamma & \text{if } \sigma = \mathsf{NT}\gamma \end{array} \tag{D.39}$$

Just note that, *e.g.* if $\mathcal{I}$ satisfies $\mathsf{NT}(A > n)$ then $|A|^t \leq n$. Finally, $\mathcal{I}$ *satisfies* a set of signed fuzzy formulae S iff $\mathcal{I}$ satisfies every component of S.

Given a fuzzy KB Σ, let

$$\mathsf{T}\Sigma = \{\mathsf{T}\gamma : \gamma \in \Sigma_F\} \cup \{C \to D \in \Sigma_T\}. \tag{D.40}$$

As a consequence of the above definition, we have the well known condition

$$\Sigma \not\approx_4 (A \geq n) \text{ iff } \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq n)\} \text{ is not satisfiable} \tag{D.41}$$

reducing the entailment decision problem between a fuzzy $\mathcal{ALC}$ KB and an assertion to the non-satisfiability decision problem of a set of signed fuzzy formulae.

Now, we go on to define conjugated signed fuzzy formulae, signed fuzzy formulae of conjunctive type (of type α) and signed fuzzy formulae of disjunctive type (of type β), respectively.

With respect to conjugates we have the same table as Table D.7 for the $\mathcal{L}^f_+$ case. Each entry says us under which conditions the row-column pair of signed expressions are conjugated. A $\times$ symbol in an entry means that the pair cannot be a conjugated one.

	$\mathsf{T}(A \geq m)$	$\mathsf{T}(A > m)$	$\mathsf{NT}(A \geq m)$	$\mathsf{NT}(A > m)$
$\mathsf{T}(A \geq n)$	$\times$	$\times$	$n \geq m$	$n > m$
$\mathsf{T}(A > n)$	$\times$	$\times$	$n \geq m$	$n \geq m$
$\mathsf{NT}(A \geq n)$	$n \leq m$	$n \leq m$	$\times$	$\times$
$\mathsf{NT}(A > n)$	$n < m$	$n \leq m$	$\times$	$\times$

Table D.18: Conjugated signed fuzzy formulae in fuzzy $\mathcal{ALC}$.

Given two signed fuzzy formulae σ_1 and σ_2, we define $Cond(\sigma, \sigma_2)$ to be the condition, according to Table D.18, under which σ_1 and σ_2 are conjugated, *e.g.* $Cond(\mathsf{T}(A \geq n), \mathsf{NT}(A \geq m)) = n \geq m$ and $Cond(\mathsf{T}(A \geq n), \mathsf{T}(A \geq m)) = \times$. Given a signed fuzzy formulae σ with σ^c we indicate a conjugate of σ (if there exists one). Just notice that, as for $\mathcal{L}^f_+$, a conjugate of a signed fuzzy formula may be not unique, as there are could be infinitely many. With $\sigma^{c,max}$ we indicate the conjugate of σ obtained by exchanging the symbols T and NT in σ. Of course $\sigma^{c,max}$ is unique.

Signed formulae of type α (of conjunctive type) and β (of disjunctive type) and on their *components* are defined as follows (see crisp $\mathcal{ALC}$ case and Table D.8 for the $\mathcal{L}^f_+$ case):

1. With respect to the connectives $\sqcap, \sqcup$ we have:

α	α_1	α_2
$\mathsf{T}((C \sqcap D)(w) \geq n)$	$\mathsf{T}(C(w) \geq n)$	$\mathsf{T}(D(w) \geq n)$
$\mathsf{T}((C \sqcap D)(a) > n)$	$\mathsf{T}(C(w) > n)$	$\mathsf{T}(D(w) > n)$
$\mathsf{NT}((C \sqcup D)(a) \geq n)$	$\mathsf{NT}(C(w) \geq n)$	$\mathsf{NT}(D(w) \geq n)$
$\mathsf{NT}((C \sqcup D)(a) > n)$	$\mathsf{NT}(C(w) > n)$	$\mathsf{NT}(D(w) > n)$

β	β_1	β_2
$\mathsf{T}((C \sqcup D)(w) \geq n)$	$\mathsf{T}(C(w) \geq n)$	$\mathsf{T}(D(w) \geq n)$
$\mathsf{T}((C \sqcup D)(w) > n)$	$\mathsf{T}(C(w) > n)$	$\mathsf{T}(D(w) > n)$
$\mathsf{NT}((C \sqcap D)(w) \geq n)$	$\mathsf{NT}(C(w) \geq n)$	$\mathsf{NT}(D(w) \geq n)$
$\mathsf{NT}((C \sqcap D)(w) > n)$	$\mathsf{NT}(C(w) > n)$	$\mathsf{NT}(D(w) > n)$

2. With respect to fuzzy specializations we have:

β	β_1	β_2	Condition
$C \rightarrow D$	$\mathsf{NT}(C(w) \geq n)$	$\mathsf{T}(D(w) \geq n)$	for all objects w, for all $n \in [0, 1]$

   We will say that the specialisation σ has been *instantiated* with w and n, if w is the object and n is the real involved in σ's components.

   Signed formulae of this type are indicated also with $\beta^{\rightarrow}$ and their components with $\beta^{\rightarrow}_1$ and $\beta^{\rightarrow}_2$.

   Just notice here that the table for fuzzy specialization $C \rightarrow D$ is the result of viewing it according to Equation 8.47.

3. With respect to the $\forall$ and $\exists$ connectives we have:

α	α_1	α_2	Condition
$\mathsf{NT}((\forall R.C)(w) \geq n)$	$\mathsf{T}(R(w, x) > 1 - n)$	$\mathsf{NT}(C(x) \geq n)$	for a new variable x
$\mathsf{NT}((\forall R.C)(w) > n)$	$\mathsf{T}(R(w, x) \geq 1 - n)$	$\mathsf{NT}(C(x) > n)$	for a new variable x
$\mathsf{T}((\exists R.C)(w) \geq n)$	$\mathsf{T}(R(w, x) \geq n)$	$\mathsf{T}(C(x) \geq n)$	for a new variable x
$\mathsf{T}((\exists R.C)(w) > n)$	$\mathsf{T}(R(w, x) > n)$	$\mathsf{T}(C(x) > n)$	for a new variable x

   Signed fuzzy formulae of this type are indicated also with $\alpha^{\exists}$ and their components with $\alpha^{\exists}_1$ and $\alpha^{\exists}_2$, respectively. Moreover, we define $Ind(\alpha^{\exists}) = a$, $Ind(\alpha^{\exists}_i) = x$ and $Role(\alpha^{\exists}) = R$.

β	β_1	β_2	Condition
$\mathsf{T}((\forall R.C)(w) \geq n)$	$\mathsf{NT}(R(w,v) > 1-n)$	$\mathsf{T}(C(v) \geq n)$	for all objects v
$\mathsf{T}((\forall R.C)(w) > n)$	$\mathsf{NT}(R(w,v) \geq 1-n)$	$\mathsf{T}(C(v) > n)$	for all objects v
$\mathsf{NT}((\exists R.C)(w) \geq n)$	$\mathsf{NT}(R(w,v) \geq n)$	$\mathsf{NT}(C(v) \geq n)$	for all objects v
$\mathsf{NT}((\exists R.C)(w) > n)$	$\mathsf{NT}(R(w,v) > n)$	$\mathsf{NT}(C(v) > n)$	for all objects v

Signed fuzzy formulae of this type are indicated also with $\beta^{\forall}$ and their components with $\beta_1^{\forall}$ and $\beta_2^{\forall}$, respectively. Let $Ind(\beta^{\forall}) = a$, $Ind(\beta_i^{\forall}) = v$ and $Role(\beta^{\forall}) = R$.

Moreover, we will say that the above signed fuzzy formulae σ of type $\alpha^{\exists}$ ($\beta^{\forall}$) has been *instantiated* with x (v), if x is the new variable (object v is) involved in σ's components and given by the condition. For instance, if $\alpha^{\exists} = \mathsf{T}((\exists R.C)(a) \geq n)$ has been instantiated with x then $\alpha_1^{\exists}$ is $\mathsf{T}(R(a,x) \geq n)$, whereas $\alpha_2^{\exists}$ is $\mathsf{T}(C(x) \geq n)$.

In one shot, the α and β tables are shown in Table D.19 below.

α	α_1	α_2	Condition
$\mathsf{T}((C \sqcap D)(w) \geq n)$	$\mathsf{T}(C(w) \geq n)$	$\mathsf{T}(D(w) \geq n)$	
$\mathsf{T}((C \sqcap D)(a) > n)$	$\mathsf{T}(C(w) > n)$	$\mathsf{T}(D(w) > n)$	
$\mathsf{NT}((C \sqcup D)(a) \geq n)$	$\mathsf{NT}(C(w) \geq n)$	$\mathsf{NT}(D(w) \geq n)$	
$\mathsf{NT}((C \sqcup D)(a) > n)$	$\mathsf{NT}(C(w) > n)$	$\mathsf{NT}(D(w) > n)$	
$\mathsf{NT}((\forall R.C)(w) \geq n)$	$\mathsf{T}(R(w,x) > 1-n)$	$\mathsf{NT}(C(x) \geq n)$	for a new variable x
$\mathsf{NT}((\forall R.C)(w) > n)$	$\mathsf{T}(R(w,x) \geq 1-n)$	$\mathsf{NT}(C(x) > n)$	for a new variable x
$\mathsf{T}((\exists R.C)(w) \geq n)$	$\mathsf{T}(R(w,x) \geq n)$	$\mathsf{T}(C(x) \geq n)$	for a new variable x
$\mathsf{T}((\exists R.C)(w) > n)$	$\mathsf{T}(R(w,x) > n)$	$\mathsf{T}(C(x) > n)$	for a new variable x

β	β_1	β_2	Condition
$\mathsf{T}((C \sqcup D)(w) \geq n)$	$\mathsf{T}(C(w) \geq n)$	$\mathsf{T}(D(w) \geq n)$	
$\mathsf{T}((C \sqcup D)(w) > n)$	$\mathsf{T}(C(w) > n)$	$\mathsf{T}(D(w) > n)$	
$\mathsf{NT}((C \sqcap D)(w) \geq n)$	$\mathsf{NT}(C(w) \geq n)$	$\mathsf{NT}(D(w) \geq n)$	
$\mathsf{NT}((C \sqcap D)(w) > n)$	$\mathsf{NT}(C(w) > n)$	$\mathsf{NT}(D(w) > n)$	
$C \rightarrow D$	$\mathsf{NT}(C(w) \geq n)$	$\mathsf{T}(D(w) \geq n)$	for all objects w, for all $n \in [0,1]$
$\mathsf{T}((\forall R.C)(w) \geq n)$	$\mathsf{NT}(R(w,v) > 1-n)$	$\mathsf{T}(C(v) \geq n)$	for all objects v
$\mathsf{T}((\forall R.C)(w) > n)$	$\mathsf{NT}(R(w,v) \geq 1-n)$	$\mathsf{T}(C(v) > n)$	for all objects v
$\mathsf{NT}((\exists R.C)(w) \geq n)$	$\mathsf{NT}(R(w,v) \geq n)$	$\mathsf{NT}(C(v) \geq n)$	for all objects v
$\mathsf{NT}((\exists R.C)(w) > n)$	$\mathsf{NT}(R(w,v) > n)$	$\mathsf{NT}(C(v) > n)$	for all objects v

Table D.19: α and β table for fuzzy $\mathcal{ALC}$.

We extend the definition of closed branch ϕ by saying that a branch ϕ in a deduction tree is *closed* whenever it contains a conjugated pair or it contains $\mathsf{T}(A > 1)$.

The decision algorithm is a fuzzy extension of procedure $AcyclicSat_{DL}(S)$. At first, the rules are a combination of the rules for $\mathcal{L}_+^f$ (see Table D.3) and those for $\mathcal{ALC}$ in the case of well formed KBs (see Table C.15), and are shown in Table D.20.

The definitions of *deduction tree*, *closed tree*, *AB-analysed* formula in a branch ϕ, *AB-completed* branch ϕ and of *fulfilled* signed fuzzy formula is as for $\mathcal{ALC}$(see page 185).

Moreover, let S be a set of signed fuzzy formulae. With $S(\rightarrow)$ we indicate the set

$$S(\rightarrow) \quad = \quad \{C \rightarrow D \in S\}. \tag{D.42}$$

$$(A)\ \frac{\alpha}{\alpha_1, \alpha_2} \qquad \text{if } \alpha \text{ is not of type } \alpha^{\exists}$$

$$(B1)\ \frac{\beta, \beta_1^c}{\beta_2} \qquad \text{if } \beta \text{ is not of type } C \rightarrow A$$

$$(B2)\ \frac{\beta, \beta_2^c}{\beta_1} \qquad \text{if } \beta \text{ is neither of type } \beta^{\forall} \text{ nor of type } A \rightarrow C$$

$$(PB)\ \frac{\beta}{\beta_1 \mid \beta_1^{c,max}, \beta_2} \qquad \text{if } \beta \text{ is neither of type } \beta^{\forall} \text{ nor of type } \beta^{\rightarrow}$$

Table D.20: Semantic tableaux for well formed fuzzy $\mathcal{ALC}$ KBs.

The $\alpha^{\exists}$ expressions and $\beta^{\forall}$ expressions will be handled in a similar way as for $\mathcal{ALC}$. If there is a signed fuzzy formula $\alpha^{\exists} \in S$ then let $S(\alpha^{\exists})$ defined as follows (see $(C.31)$ and $(C.32)$ as a comparison to the crisp $\mathcal{ALC}$ case):

$$\begin{aligned} S(\mathsf{T}((\exists R.C)(w) \geq m)) \quad = \quad & \{\mathsf{T}(C(x) \geq m)\} \cup \\ & \{\mathsf{T}(D(x) \geq n) : \mathsf{T}((\forall R.D)(w) \geq n) \in S, m > 1 - n\} \cup \\ & \{\mathsf{T}(D(x) > n) : \mathsf{T}((\forall R.D)(w) > n) \in S, m \geq 1 - n\} \cup \\ & \{\mathsf{NT}(D(x) \geq n) : \mathsf{NT}((\exists R.D)(w) \geq n) \in S, m \geq n\} \cup \\ & \{\mathsf{NT}(D(x) > n) : \mathsf{NT}((\exists R.D)(w) > n) \in S, m > n\} \end{aligned} \tag{D.43}$$

$$\begin{aligned} S(\mathsf{T}((\exists R.C)(w) > m)) \quad = \quad & \{\mathsf{T}(C(x) > m)\} \cup \\ & \{\mathsf{T}(D(x) \geq n) : \mathsf{T}((\forall R.D)(w) \geq n) \in S, m \geq 1 - n\} \cup \\ & \{\mathsf{T}(D(x) > n) : \mathsf{T}((\forall R.D)(w) > n) \in S, m \geq 1 - n\} \cup \\ & \{\mathsf{NT}(D(x) \geq n) : \mathsf{NT}((\exists R.D)(w) \geq n) \in S, m \geq n\} \cup \\ & \{\mathsf{NT}(D(x) > n) : \mathsf{NT}((\exists R.D)(w) > n) \in S, m \geq n\} \end{aligned} \tag{D.44}$$

$$\begin{aligned} S(\mathsf{NT}((\forall R.C)(w) \geq m)) \quad = \quad & \{\mathsf{NT}(C(x) \geq m)\} \cup \\ & \{\mathsf{T}(D(x) \geq n) : \mathsf{T}((\forall R.D)(w) \geq n) \in S, m \leq n\} \cup \\ & \{\mathsf{T}(D(x) > n) : \mathsf{T}((\forall R.D)(w) > n) \in S, m \leq n\} \cup \\ & \{\mathsf{NT}(D(x) \geq n) : \mathsf{NT}((\exists R.D)(w) \geq n) \in S, 1 - m \geq n\} \cup \\ & \{\mathsf{NT}(D(x) > n) : \mathsf{NT}((\exists R.D)(w) > n) \in S, 1 - m \geq n\} \end{aligned} \tag{D.45}$$

$$\begin{aligned} S(\mathsf{NT}((\forall R.C)(w) > m)) \quad = \quad & \{\mathsf{NT}(C(x) > m)\} \cup \\ & \{\mathsf{T}(D(x) \geq n) : \mathsf{T}((\forall R.D)(w) \geq n) \in S, m < n\} \cup \\ & \{\mathsf{T}(D(x) > n) : \mathsf{T}((\forall R.D)(w) > n) \in S, m \leq n\} \cup \\ & \{\mathsf{NT}(D(x) \geq n) : \mathsf{NT}((\exists R.D)(w) \geq n) \in S, 1 - m \geq n\} \cup \\ & \{\mathsf{NT}(D(x) > n) : \mathsf{NT}((\exists R.D)(w) > n) \in S, 1 - m > n\}, \end{aligned} \tag{D.46}$$

where x is a new variable.

As for the crisp $\mathcal{ALC}$ case (see Equation $(C.33)$), the above four equations can be rewritten in the following compact way:

$$
\begin{array}{rl}
S(\alpha^{\exists}) = & \{\alpha_2^{\exists}\} \cup \\
& \{\beta_2^{\forall} : \ \beta^{\forall} \in S, \\
& \qquad Ind(\alpha^{\exists}) = Ind(\beta^{\forall}), \\
& \qquad Role(\alpha^{\exists}) = Role(\beta^{\forall}), \\
& \qquad Cond(\alpha_1^{\exists}, \beta_1^{\forall}) \neq \times\}
\end{array}
\tag{D.47}
$$

where both $\alpha^{\exists}$ and $\beta^{\forall}$ have been instantiated with the same new variable x ($Ind(\alpha_2^{\exists}) = Ind(\beta_2^{\forall}) = x$).

A short explanation of the above definition should take place. We will consider only the case of Equation (D.43) and the subset

$$S' = \{\mathsf{T}(D(x) \geq n) : \mathsf{T}((\forall R.D)(w) \geq n) \in S, m > 1 - n\}. \tag{D.48}$$

The other subsets and cases have similar explanations. Let $\mathcal{I}$ be a model of S. Hence $\mathcal{I}$ is a model of $\sigma = \mathsf{T}((\exists R.C)(w) \geq m) \in S$. From the α table of σ, it follows immediately that $\mathcal{I}$ has to satisfy both $\mathsf{T}(R(w, x) \geq m)$ and $\mathsf{T}(C(x) \geq m)$, for some new variable x. Now, suppose that there is $\mathsf{T}((\forall R.D)(w) \geq n) \in S$. Therefore, $\mathcal{I}$ satisfies $\mathsf{T}((\forall R.D)(w) \geq n)$ too. But, from Equation (8.53), it follows that $\mathcal{I}$ has to satisfy $\mathsf{T}(D(x) \geq n)$, if it is the case that $m > 1 - n$. Hence, if $\mathcal{I}$ satisfies S, then $\mathcal{I}$ has to satisfy S' too. In terms of an inference point of view, we have that from $\mathsf{T}((\exists R.C)(w) \geq m)$ we would infer $\mathsf{T}(R(w, x) \geq m)$ and $\mathsf{T}(C(x) \geq m)$. Now, ($B1$) can be applied according to

$$(B1) \quad \frac{\mathsf{T}(R(w, x) \geq m), \mathsf{T}((\forall R.D)(w) \geq n)}{\mathsf{T}(D(x) \geq n)}$$

if $m > 1 - n$.

Example 69 Let S be

$$
\begin{array}{rl}
S = \{ & \mathsf{T}((\exists R_1.C_1)(a) \geq .4), \\
& \mathsf{T}((\forall R_1.D_{1_1})(a) \geq .6), \\
& \mathsf{T}((\forall R_1.D_{1_2})(a) \geq .7), \\
& \mathsf{T}((\exists R_2.C_2)(b) \geq .8)\}
\end{array}
$$

then

$$S(\mathsf{T}((\exists R_1.C_1)(a) \geq .4)) = \{\mathsf{T}(C_1(x_1) \geq .4), \mathsf{T}(D_{1_2}(x_1) \geq .7)\},$$

whereas

$$S(\mathsf{T}((\exists R_2.C_2)(a) \geq .8)) = \{\mathsf{T}(C_2(x_2) \geq .8)\}.$$

It follows that S is satisfiable if and only if both the set $S(\mathsf{T}((\exists R_1.C_1)(a) \geq .4))$ and the set $S(\mathsf{T}((\exists R_2.C_2)(a) \geq .8))$ are satisfiable.

■

The procedure $AcyclicSat_{DL}(S)$ below determines whether a set of signed fuzzy formulae S is satisfiable or not. It works exactly as for the crisp $\mathcal{ALC}$ case: in order to determine whether S is satisfiable or not,we start with a root node labelled S. Each not closed branch, not being

AB-completed, will be expanded until it becomes AB-completed or closed. If it becomes closed then $AcyclicSat_{DL}(S)$ fails. Otherwise, if a signed formula of type $\alpha^{\exists}$ occurs in a branch ϕ and $S^{\phi}(\alpha^{\exists})$ has not yet been considered with respect to ϕ, *i.e.* $AcyclicSat_{DL}(S^{\phi}(\alpha^{\exists}))$ has not yet been tested, then we test $AcyclicSat_{DL}(S^{\phi}(\alpha^{\exists}) \cup S^{\phi}(\rightarrow))$. If for some of these $\alpha^{\exists}$ the test fails then, $AcyclicSat_{DL}(S)$ fails. Otherwise, we proceed by applying the principle of bivalence to a not fulfilled β (neither being of type $\beta^{\forall}$ nor of type $\beta^{\rightarrow}$).

Algorithm 18 ($AcyclicSat_{DL}(S)$)
Let S be a well formed set of signed fuzzy formulae. $AcyclicSat_{DL}(S)$ starts from the root labelled S and applies the following steps:

1. Select a not closed branch ϕ. If there is no such branch then return $false$ and exit.
2. If ϕ is not yet AB-completed then expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed. Update ϕ to the resulting branch;
3. If ϕ is AB-completed then
   (a) if for *some* signed fuzzy formula $\alpha^{\exists}$ in S^{ϕ}, $AcyclicSat_{DL}(S^{\phi}(\alpha^{\exists}) \cup S^{\phi}(\rightarrow)) = false$ holds, where $S(\alpha^{\exists})$ has not yet been tested with respect to ϕ, then close ϕ and go to Step 1. Otherwise,
   (b) select a signed fuzzy formula of type β, not being of type $\beta^{\forall}$, which is not yet fulfilled in the branch;
   (c) apply rule (PB) and go to Step 1. ■

Table D.21: Algorithm $AcyclicSat_{DL}(S)$ for fuzzy $\mathcal{ALC}$.

Example 70 Let Σ be the set

$$\begin{aligned}\Sigma = \{\ & (R(a,a) \geq .4)\\ & ((\forall R.D_{1_1})(a) \geq .8)\\ & ((\forall R.D_{1_2})(a) \geq .7)\\ & ((\forall R.D_{1_3})(a) \geq .6)\}\end{aligned}$$

Let A be the fuzzy assertion

$$((D_{1_2} \sqcap \forall R.(D_{1_1} \sqcap D_{1_2}))(a) \geq .7).$$

We will proof that $\Sigma \not\approx_4 A$ holds by showing that effectively $AcyclicSat_{DL}(\mathsf{T}\Sigma \cup \{\mathsf{NT}A\}) = false$. We will present the proof by means of the deduction tree in Figure D.6.
The first three nodes after the first one are obtained by a straightforward application of the $(B1)$ rule. Thereafter, Step 3a of $AcyclicSat_{DL}$ has been applied. It is easily verified that

$$AcyclicSat_{DL}(S(\mathsf{NT}((\forall R.(D_{1_1} \sqcap D_{1_2}))(a) \geq .7))) = false,$$

by observing that $S(\mathsf{NT}((\forall R_1.(D_{1_1} \sqcap D_{1_2}))(a) \geq .7))$ is

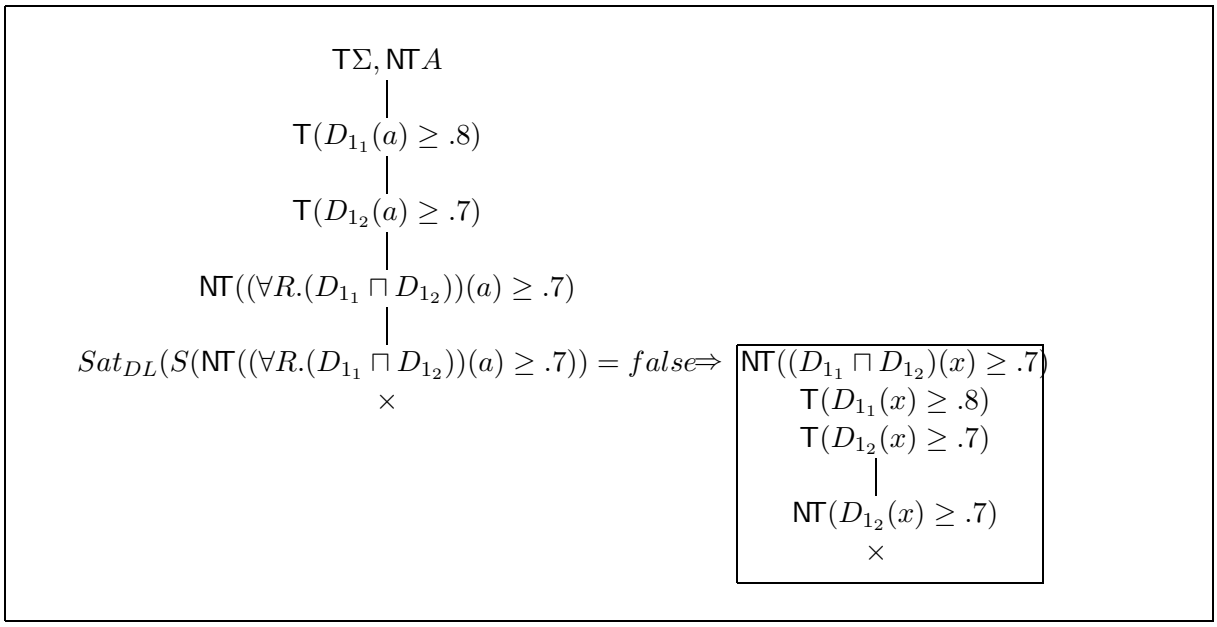


Figure D.6: Closed deduction tree with recursive call to $AcyclicSat_{DL}$ in fuzzy $\mathcal{ALC}$.

$$\{\mathsf{NT}((D_{1_1} \sqcap D_{1_2})(x) \geq .7), \mathsf{T}(D_{1_1}(x) \geq .8), \mathsf{T}(D_{1_2}(x) \geq .7)\}$$

according to Equation $(D.45)$. The proof of $AcyclicSat_{DL}(S(\mathsf{NT}((\forall R.(D_{1_1} \sqcap D_{1_2}))(a) \geq .7)))$ $= false$ is the small bounded deduction tree in Figure D.6. Therefore, $AcyclicSat_{DL}$ returns $false$ as expected. ■

Proposition 74 *Let S be a well formed set of signed fuzzy formulae in $\mathcal{ALC}$. It follows then that $AcyclicSat_{DL}(S)$ iff S is satisfiable.* ⊣

Proof: The proof is a combination of the proof of Proposition 28 for crisp $\mathcal{ALC}$ and the proof of Proposition 61 for $\mathcal{L}^f_+$.

At first, it can be easily verified that the rules (A), $(B1)$, $(B2)$ and (PB) in Table D.20 are correct, *i.e.* ϕ is a branch and S^ϕ is satisfiable iff there is a branch ϕ' as the result of the application of a rule to ϕ such that $S^{\phi'}$ satisfiable.

$\Rightarrow$.) Suppose $AcyclicSat_{DL}(S)$. We show by induction on the number n of occurrences of signed fuzzy formulae of type $\alpha^\exists \in S$ that there is (i) a finite deduction tree T build by $Sat(S)$; (ii) a not closed branch ϕ from S to a leaf in T; and a set $S(\phi)$ such that $S(\phi)$ is satisfiable and $S^\phi \subseteq S(\phi)$. As a consequence, $S \subseteq S^\phi$ is satisfiable.

Case $n = 0$ **:** Let T be the generated deduction tree and let ϕ be a not closed branch from S to a leaf in T. Such a branch has to exist, otherwise $AcyclicSat_{DL}(S) = false$. Let $S(\phi) = S^\phi$. Moreover, define

$$S^{\mathsf{T}}_{\geq} = \{\mathsf{T}(A \geq n) \in S(\phi)\},$$

$$S^{\mathsf{T}}_{>} = \{\mathsf{T}(A > n) \in S(\phi)\}, \text{ and}$$

$$S^{\mathsf{NT}} = \{\mathsf{NT}(A \geq n) \in S(\phi)\}.$$

Of course, $S(\phi) = S^{\mathsf{T}}_{\geq} \cup S^{\mathsf{T}}_{>} \cup S^{\mathsf{NT}}$. Define, for $\epsilon > 0$

$$n^{\geq}_A = \max\{n : \mathsf{T}(A \geq n) \in S^{\mathsf{T}}_{\geq}\}, \text{ and}$$

$$n^{>}_A = \max\{n : \mathsf{T}(A > n) \in S^{\mathsf{T}}_{>}\} + \epsilon.$$

Let $\mathcal{I}$ be a relation such that, the domain $\Delta^{\mathcal{I}}$ of $\mathcal{I}$ is the set of objects appearing in $S(\phi)$ and $w^{\mathcal{I}} = w$ for all $w \in \Delta^{\mathcal{I}}$. For each primitive concept A, for each role R, for all $w, v \in \Delta^{\mathcal{I}}$ define ($\max \emptyset = 0$)

$$\begin{array}{l} |A|^t(w) = \max\{n^{\geq}_{A(w)}, n^{>}_{A(w)}\}, \\ |A|^f(w) = 0, \\ |R|^t(w,v) = \max\{n^{\geq}_{R(w,v)}, n^{>}_{R(w,v)}\}, \text{ and} \\ |R|^f(w,v) = 0. \end{array}$$

Since, ϕ is completed and not closed, there is $\epsilon > 0$ such that $\mathcal{I}$ is a four-valued fuzzy interpretation and $\mathcal{I}$ satisfies all $\sigma \in S^{\mathsf{T}}_{\geq}$, $\mathcal{I}$ satisfies all $\sigma \in S^{\mathsf{T}}_{>}$ and $\mathcal{I}$ satisfies all $\sigma \in S^{\mathsf{NT}}$ and, thus, $\mathcal{I}$ satisfies $S(\phi)$. As a consequence, $S \subseteq S^{\phi}$ is satisfiable.

Case $n > 0$ **:** Let T be the generated deduction tree and let ϕ be a not closed and completed branch from S to a leaf in T. Such a branch has to exist, otherwise $AcyclicSat_{DL}(S) = false$ Now, Step 3. of the algorithm is applied. Suppose $\alpha^{\exists}_1, \ldots, \alpha^{\exists}_m \in S^{\phi}$ are the top level $\alpha^{\exists}_i$ for which $AcyclicSat_{DL}(S(\alpha^{\exists}_i))$ has been checked. Let x_i be the new variables introduced. Certainly, $m \leq n$ holds. Since $AcyclicSat_{DL}(S) = true$, it follows that $\bigwedge_{1 \leq i \leq m} Sat(S(\alpha^{\exists}_i))$ holds. Since, for all $1 \leq i \leq m$ the number of occurrences of signed fuzzy formulae of type $\alpha^{\exists}$ in $S(\alpha^{\exists}_i)$ is less than n, by induction there are (i) finite deduction trees T_i; (ii) not closed and completed branches ϕ_i from $S^{\phi}(\alpha^{\exists}_i)$ to a leaf in T_i; and (iii) sets $S(\phi_i)$ such that $S(\phi_i)$ is satisfiable and $S^{\phi}_i \subseteq S(\phi_i)$.

Let

$$S(\phi) = S^{\phi} \cup \bigcup_{1 \leq i \leq m} (S(\phi_i \cup \{\sigma_i\})$$

where (see the $\alpha^{\exists}$ table)

$$\sigma_i = \begin{cases} \mathsf{T}(R_i(w_i, x_i) > 1 - n) & \text{if } \alpha^{\exists}_i = \mathsf{NT}((\forall R_i.C_i)(w_i) \geq n) \\ \mathsf{T}(R_i(w_i, x_i) \geq 1 - n) & \text{if } \alpha^{\exists}_i = \mathsf{NT}((\forall R_i.C_i)(w_i) > n) \\ \mathsf{T}(R_i(w_i, x_i) \geq n) & \text{if } \alpha^{\exists}_i = \mathsf{T}((\exists R_i.C_i)(w_i) \geq n) \\ \mathsf{T}(R_i(w_i, x_i) > n) & \text{if } \alpha^{\exists}_i = \mathsf{T}((\exists R_i.C_i)(w_i) > n) \end{cases}$$

Define

$$S^{\mathsf{T}}_{\geq} = \{\mathsf{T}(A \geq n) \in S(\phi)\},$$
$$S^{\mathsf{T}}_{>} = \{\mathsf{T}(A > n) \in S(\phi)\}, \text{ and} \tag{D.49}$$
$$S^{\mathsf{NT}} = \{\mathsf{NT}(A \geq n) \in S(\phi)\}.$$

Of course, $S(\phi) = S^{\mathsf{T}}_{\geq} \cup S^{\mathsf{T}}_{>} \cup S^{\mathsf{NT}}$ and by definition $S^{\phi} \subseteq S(\phi)$. Define, for $\epsilon > 0$

$$n^{\geq}_A = \max\{n : \mathsf{T}(A \geq n) \in S^{\mathsf{T}}_{\geq}\}, \text{ and}$$
$$n^{>}_A = \max\{n : \mathsf{T}(A > n) \in S^{\mathsf{T}}_{>}\} + \epsilon.$$

Let $\mathcal{I}$ be a relation such that, the domain $\Delta^{\mathcal{I}}$ of $\mathcal{I}$ is the set of objects appearing in $S(\phi)$ and $w^{\mathcal{I}} = w$ for all $w \in \Delta^{\mathcal{I}}$. For each primitive concept A, for each role R, for all $w, v \in \Delta^{\mathcal{I}}$ define

$$\begin{array}{l} |A|^t(w) = \max\{n^{\geq}_{A(w)}, n^{>}_{A(w)}\}, \\ |A|^f(w) = 0, \\ |R|^t(w,v) = \max\{n^{\geq}_{R(w,v)}, n^{>}_{R(w,v)}\}, \text{ and} \\ |R|^f(w,v) = 0. \end{array}$$

Since, ϕ is completed and not closed, there is $\epsilon > 0$ such that $\mathcal{I}$ is a four-valued fuzzy interpretation and $\mathcal{I}$ satisfies all $\sigma \in S^{\mathsf{T}}_{\geq}$, $\mathcal{I}$ satisfies all $\sigma \in S^{\mathsf{T}}_{>}$ and $\mathcal{I}$ satisfies all $\sigma \in S^{\mathsf{NT}}$ and, thus, $\mathcal{I}$ satisfies $S(\phi)$. As a consequence, $S \subseteq S^{\phi}$ is satisfiable.

$\Leftarrow$.) Suppose S is satisfiable. Let T be the generated completed tree. From the correctness of the rules it follows that there is a completed branch ϕ in T such that S^{ϕ} is satisfiable. Therefore, $AcyclicSat_{DL}(S)$. Q.E.D.

Example 71 Let us see how the interpretation of the above proof is build. Let Σ be the set

$$\begin{array}{ll} \Sigma = \{ & (R(a,a) \geq .4) \\ & ((\forall R.D_{1_1})(a) \geq .8) \\ & ((\forall R.D_{1_2})(a) \geq .7)\} \end{array}$$

Let A be the fuzzy assertion

$$((D_{1_2} \sqcap \forall R.(D_{1_1} \sqcap D_{1_3}))(a) \geq .7).$$

It is easily verified that $\Sigma \models\!\!\!\not\models_4 A$ holds. In fact, $AcyclicSat_{DL}(\mathsf{T}\Sigma \cup \{\mathsf{NT}A\}) = true$, as shown in the not closed and completed deduction tree in Figure D.7.

Let $\alpha^{\exists}_1$ be $\mathsf{NT}((\forall R.(D_{1_1} \sqcap D_{1_3}))(a)) \geq .7)$. Just notice that $S(\alpha^{\exists}_1)$ is

$$S(\alpha^{\exists}_1) = \{\mathsf{NT}((D_{1_1} \sqcap D_{1_3})(x) \geq .7), \mathsf{T}(D_{1_1}(x) \geq .8), \mathsf{T}(D_{1_2}(x) \geq .7)\}$$

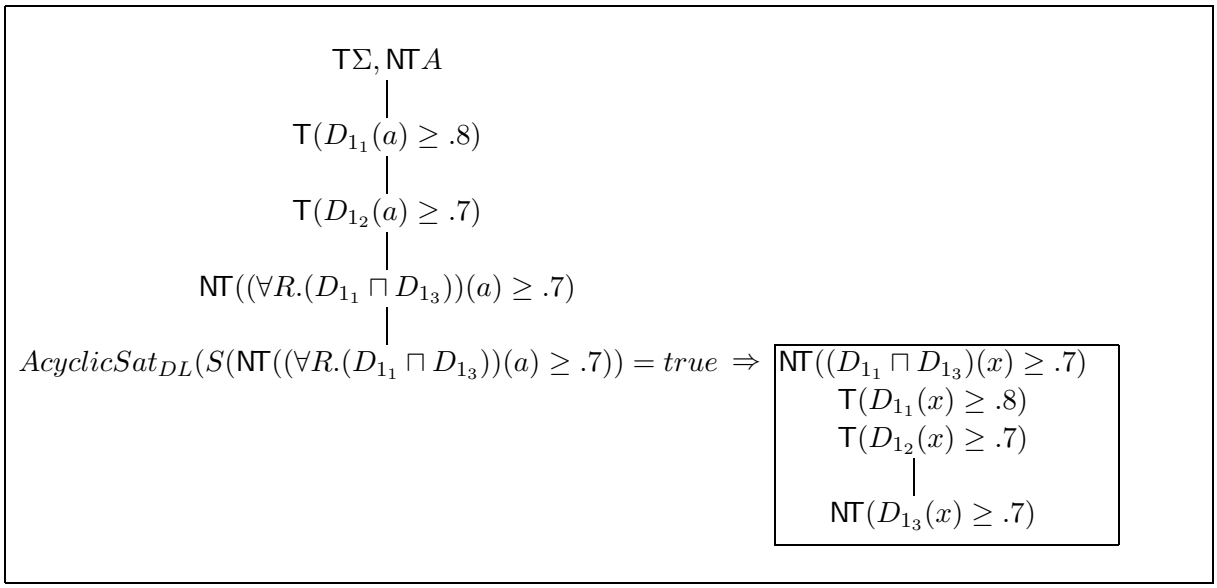


Figure D.7: Not closed and completed deduction tree with recursive call to $AcyclicSat_{DL}$ in fuzzy $\mathcal{ALC}$.

As shown in Figure D.7,

$$AcyclicSat_{DL}(\alpha_1^{\exists}) = true$$

According to the proof of Proposition 74 above, it follows that S^{ϕ_1} is

$$\begin{aligned} S(\phi_1) = \{ \quad & \mathsf{NT}((D_{1_1} \sqcap D_{1_3})(x) \geq .7) \\ & \mathsf{T}(D_{1_1}(x) \geq .8), \\ & \mathsf{T}(D_{1_2}(x) \geq .7), \\ & \mathsf{NT}(D_{1_3}(x) \geq .7)\} \end{aligned}$$

which is satisfiable. Finally, let

$$S(\phi) = S^{\phi} \cup S(\phi_1) \cup \{\mathsf{T}(R(a, x) > .3)\}$$

where

$$\begin{aligned} S^{\phi} = \mathsf{T}\Sigma \cup \{ \quad & \mathsf{NT}A, \\ & \mathsf{T}(D_{1_1}(a) \geq .8), \\ & \mathsf{T}(D_{1_2}(a) \geq .7), \\ & \mathsf{NT}((\forall R_1.(D_{1_1} \sqcap D_{1_3}))(a) \geq .7)\}. \end{aligned}$$

$\mathcal{I}$ is build as follows. The domain of $\mathcal{I}$ is

$$\Delta^{\mathcal{I}} = \{a, x\}$$

Of course, $a^{\mathcal{I}} = a$ and $x^{\mathcal{I}} = x$. Moreover, from the proof of Proposition 74 it follows that the interpretation $\mathcal{I}$ build from S' is such that for all $\epsilon > 0$

$$
\begin{array}{ll}
|D_{1_1}|^t(a) = .8, & |D_{1_1}|^f(a) = 0, \\
|D_{1_2}|^t(a) = .7, & |D_{1_2}|^f(a) = 0, \\
|D_{1_3}|^t(a) = 0, & |D_{1_3}|^f(a) = 0, \\
|D_{1_1}|^t(x) = .8, & |D_{1_1}|^f(x) = 0, \\
|D_{1_2}|^t(x) = .7, & |D_{1_2}|^f(x) = 0, \\
|D_{1_3}|^t(x) = 0, & |D_{1_3}|^f(x) = 0, \\
|R|^t(a,a) = .4, & |R|^f(a,a) = 0, \\
|R|^t(a,x) = .3 + \epsilon, & |R|^f(a,x) = 0, \\
|R|^t(x,a) = 0, & |R|^f(x,a) = 0, \\
|R|^t(b,b) = 0, & |R|^f(b,b) = 0.
\end{array}
$$

It follows that $\mathcal{I}$ satisfies $\mathsf{T}\Sigma \cup \{\mathsf{NT}A\}$. ■

Concerning the problem of determining $\Sigma \not\approx_2 (A \geq n)$, we can run $AcyclicSat_{DL}$ where the α table has been extended as follows:

α	α_1	α_2
$\mathsf{T}(\neg A \geq n)$	$\mathsf{NT}(A > 1-n)$	$\mathsf{NT}(A > 1-n)$
$\mathsf{T}(\neg A > n)$	$\mathsf{NT}(A \geq 1-n)$	$\mathsf{NT}(A \geq 1-n)$
$\mathsf{NT}(\neg A \geq n)$	$\mathsf{T}(A > 1-n)$	$\mathsf{T}(A > 1-n)$
$\mathsf{NT}(\neg A > n)$	$\mathsf{T}(A \geq 1-n)$	$\mathsf{T}(A \geq 1-n)$

It is straightforward to see that this yields to a correct and complete decision algorithm for two-valued fuzzy $\mathcal{ALC}$.

Concerning the computation of completions, we rely on the case of $\mathcal{ALC}$ with acyclic specialisations (see Section C.4.3).

Therefore, let S be a set of signed fuzzy formulae and define, as usual,

$$
\begin{array}{rcl}
\tilde{\mathcal{S}} & = & \{\mathsf{T}A \in S : A \text{ atomic fuzzy assertion}\} \cup \\
 & & \{\mathsf{NT}A \in S : A \text{ atomic fuzzy assertion}\}
\end{array}
\tag{D.50}
$$

It is worth noticing that in applying $AcyclicSat_{DL}(\mathsf{T}\Sigma)$, according to Equation D.49, $S(\phi)$ could contain signed atomic fuzzy assertions of type $\mathsf{T}(A > n)$. The question is: are these atomic fuzzy assertions necessary in order to build canonical models? The answer is no.

At first, note that starting with $\mathsf{T}\Sigma$ one could generate signed fuzzy assertions of the following form.

$$
\begin{array}{lll}
\mathsf{T}(C(w) \geq n) & \text{or} & \mathsf{NT}(C(w) \geq n), \\
\mathsf{T}(R(w,v) \geq n) & \text{or} & \mathsf{T}(R(w,v) > n).
\end{array}
\tag{D.51}
$$

This means that the only form of type $\mathsf{T}(A > n)$ is in fact $\mathsf{T}(R(w,v) > n)$. Just notice that *e.g.* $\mathsf{T}(R(w,v) > n)$ can be generated through the following steps: $\mathsf{T}(((\exists R.E) \sqcup D)(w) \geq n)$ may generate $\mathsf{T}((\exists R.E)(w) \geq n)$ and $\mathsf{NT}((\exists R.E)(w) \geq n)$, whereas $\mathsf{NT}((\exists R.E)(w) \geq n)$ may generate $T(R(w,x) > 1-n)$ and $\mathsf{NT}(E(w) \geq n)$.

If Σ is a set of fuzzy assertions of type $(A \geq n)$, then for all fuzzy assertions $(A > n)$ it follows that $\Sigma \not\approx_4 (A > n)$[3] iff $\Sigma \not\approx_4 (A \geq n)$. In fact, proceed in a similar way as in Section D.2.2, proof of Proposition 66: consider our set of rules

[3]For a moment suppose the definition of fuzzy entailment has been extended to the case $(A > n)$ too.

$$R_0 := \{(A), (B1), (B2), (PB)\}.$$

It is clear that R_0 can be replaced with the simple set

$$R_1 := \{(A), (B1), (B2), (PB_\beta)\},$$

where (PB_β) is

$$(PB_\beta)\ \frac{\beta}{\beta_1 \mid \beta_2} \quad \text{if } \beta \text{ is neither of type } \beta^\forall \text{ nor of type } \beta^\rightarrow$$

It is easily shown that R_1 is a correct and complete set of rules w.r.t. determining satisfiability, *i.e.* S satisfiable iff $AcyclicSat_{DL}(S)$, where the fuzzy assertions appearing in S are only of type $(A \geq n)$.

By considering the above set of rule R_1, it is easily verified that $AcyclicSat_{DL}(\mathsf{T}\Sigma)$ generates only signed fuzzy assertions of type

$$\mathsf{T}(C(w) \geq n) \quad \text{or} \quad \mathsf{T}(R(w,v) \geq n). \tag{D.52}$$

Note, from $\mathsf{T}\Sigma$, neither $\mathsf{NT}\gamma$ nor $\mathsf{T}(R(w,v) > n)$ can be generated. As R_1 is a correct and complete set of rules, this means that we can throw away from $S(\phi)$ all signed assertions of this form, *i.e.* in building completions $\tilde{\mathcal{S}}^\mathsf{T}$, we have to consider only signed fuzzy assertions of the form $\mathsf{T}(E \geq n)$, where A is an atomic assertion.

So, we define

$$\tilde{\mathcal{S}}^\mathsf{T} = \{\mathsf{T}(A \geq n) \in \tilde{\mathcal{S}}\}, \tag{D.53}$$

$$\tilde{\mathcal{S}}^+ = \{\mathsf{T}(A \geq n) \in \tilde{\mathcal{S}} : A \text{ primitive assertion}\} \tag{D.54}$$

$$\tag{D.55}$$

Moreover, let

$$\Sigma_{\tilde{\mathcal{S}}} = \{\gamma : \mathsf{T}\gamma \in \tilde{\mathcal{S}}^\mathsf{T}\}, \tag{D.56}$$

$$\Sigma^+_{\tilde{\mathcal{S}}} = \{\gamma : \mathsf{T}\gamma \in \tilde{\mathcal{S}}^+\}, \tag{D.57}$$

Similarly as for $\mathcal{L}^f$, given a set $\tilde{\mathcal{S}}$, we define the *canonical model* of $\tilde{\mathcal{S}}$ as follows. Let $\Delta^\mathcal{I}$ be the set of objects appearing in $\tilde{\mathcal{S}}$ and let $w^\mathcal{I} = w$, for all $w \in \Delta^\mathcal{I}$. For each primitive concept A, for each role R, for all objects $w, v \in \Delta^\mathcal{I}$, set $|R|^f(w,v) = 0$ and

$$\begin{array}{l} |A|^t(w) = \max\{n : \mathsf{T}(A(w) \geq n) \in S^\mathsf{T}\}, \\ |A|^f(w) = \max\{n : \mathsf{T}(\neg A(w) \geq n) \in S^\mathsf{T}\}, \text{ and} \\ |R|^t(w,v) = \max\{n : \mathsf{T}(R(w,v) \geq n) \in S^\mathsf{T}\} \end{array}$$

where, as usual, $\max \emptyset = 0$. It is easily verified that a canonical model of $\tilde{\mathcal{S}}$ is also a model of S. It is worth noticeable that given $\tilde{\mathcal{S}}$, the canonical model of $\tilde{\mathcal{S}}$ is unique.

The following definitions are an adaption of the ones defined in Section C.4.3 about well formed $\mathcal{ALC}$ KBs. Given a a well formed fuzzy KB Σ, define

$$Ext(\Sigma) \quad = \quad \{(A(a) \geq n) \colon A := C \in \Sigma_T, a \text{ occurs in } \Sigma_F, Maxdeg(\Sigma, A(a)) = n\}. \tag{D.58}$$

We extend the definition to sets S of signed fuzzy formulae as follows:

$$\begin{array}{rcl} Ext(S) & = & \{\mathsf{T}(A(a) \geq n) \colon \mathsf{T}A := C \in S, a \text{ occurs in } S, \\ & & n = max\{m \colon S \cup \{\mathsf{NT}(A(a) \geq m)\} \text{ not satisfiable}\}. \end{array} \tag{D.59}$$

The following algorithm in Table D.22 computes the four-valued completions of a well formed set S of signed formulae. Essentially it follows crisp $AcyclicCompl_{DL}(S)$.

Algorithm 19 ($AcyclicCompl_{DL}(S)$)
Essentially, the procedure proceeds in a similar way as $AcyclicSat_{DL}$.

1. Select a not closed branch ϕ. If there is no such ϕ then $AcyclicCompl_{DL}(S) := \emptyset$ and exit.
2. Let $Ext(S^\phi)$ as in Equation D.59. Assign $S^\phi := S^\phi \cup Ext(S^\phi)$.
3. If ϕ is not yet AB-completed then expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed. Update ϕ to the resulting branch;
4. If ϕ is AB-completed then
   (a) for *all* $\alpha^\exists \in S^\phi$, let $\tilde{\mathcal{S}}(\alpha^\exists) = AcyclicCompl_{DL}(S^\phi(\alpha^\exists) \cup S^\phi(\rightarrow))$, where $S^\phi(\alpha^\exists)$ has not yet been considered with respect to ϕ. If for *some* $\alpha^\exists \in S^\phi$, $\tilde{\mathcal{S}}(\alpha^\exists) = \emptyset$ then close ϕ and go to Step 1. Otherwise,
   (b) select a signed fuzzy formula of type β, neither of type $\beta^\forall$ nor of type $\beta^\rightarrow$ and not yet fulfilled in the branch;
   (c) apply rule (PB) and go to Step 1.
5. Let T be the generated deduction tree. Note that T at this point is not closed.
6. For all not closed branches ϕ from S to a leaf in T do:
   (a) let $\Psi = \{\alpha_1^\exists, \ldots, \alpha_m^\exists\}$ be the set of top level $\alpha_i^\exists \in S^\phi$ for which $S^\phi(\alpha_i^\exists)$ has been considered and let x_i be the new variables introduced.
   (b) for each tuple $(\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_m)$ such that $\tilde{\mathcal{S}}_j \in \tilde{\mathcal{S}}(\alpha_j^\exists)$, let $S(\phi) = S^\phi \cup \bigcup_{1 \leq j \leq m}(\tilde{\mathcal{S}}_j \cup \{\sigma_j\})$, where (see the $\alpha^\exists$ table)

$$\sigma_j = \begin{cases} \mathsf{T}(R_j(w_j, x_j) > 1 - n) & \text{if } \alpha_j^\exists = \mathsf{NT}((\forall R_j.C_j)(w_j) \geq n) \\ \mathsf{T}(R_j(w_j, x_j) \geq 1 - n) & \text{if } \alpha_j^\exists = \mathsf{NT}((\forall R_j.C_j)(w_j) > n) \\ \mathsf{T}(R_j(w_j, x_j) \geq n) & \text{if } \alpha_j^\exists = \mathsf{T}((\exists R_j.C_j)(w_j) \geq n) \\ \mathsf{T}(R_j(w_j, x_j) > n) & \text{if } \alpha_j^\exists = \mathsf{T}((\exists R_j.C_j)(w_j) > n) \end{cases}$$

   (c) define the $\tilde{S}$ from $S(\phi)$ according to $(D.50)$;
   (d) set $AcyclicCompl_{DL}(S) := AcyclicCompl_{DL}(S) \cup \{\tilde{S}\}$. ■

Table D.22: Algorithm $AcyclicCompl_{DL}(S)$ for fuzzy $\mathcal{ALC}$.

The analogue of Proposition 33 holds.

Proposition 75 *In $\mathcal{ALC}$, let Σ be a well formed fuzzy KB and let $(A(a) \geq n)$ be a fuzzy assertion such that A is a primitive concept. Then $\Sigma \mid\!\approx_4 (A(a) \geq n)$ iff a occurs in Σ_F and for all canonical models $\mathcal{I}$ of completions $\tilde{\mathcal{S}} \in AcyclicCompl_{DL}(\mathsf{T}\Sigma)$, $\mathcal{I}$ satisfies $(A(a) \geq n)$.* $\dashv$

Example 72 Consider

$$\begin{array}{rcl} \Sigma & = & \{A := C \sqcap E, B := D \sqcap F, G := A \sqcup B, \\ & & ((C \sqcup D)(a) \geq .6), (E(a) \geq .4), (F(a) \geq .7)\}. \end{array}$$

It can be verified that

$$\begin{array}{c} Maxdeg(\Sigma, A(a)) = 0, \\ Maxdeg(\Sigma, B(a)) = 0, \\ Maxdeg(\Sigma, G(a)) = .4. \end{array}$$

Let $S = \mathsf{T}\Sigma$. Therefore, Then $Ext(S) = \{\mathsf{T}(G(a) \geq .4)\}$. Let $S_1 = S \cup Ext(S)$, *i.e.*

$$S_1 = \{\mathsf{T}(E(a) \geq .4), \mathsf{T}(F(a) \geq .7), \mathsf{T}(G(a) \geq .4)\}.$$

Then $AcyclicCompl_{DL}(\mathsf{T}\Sigma) = \{\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_3\}$, such that

$$\begin{array}{rcl} \tilde{\mathcal{S}}_1^{\mathsf{T}} & = & S_1 \cup \{\mathsf{T}(A(a) \geq .4), \mathsf{T}(C(a) \geq .6)\} \\ \tilde{\mathcal{S}}_2^{\mathsf{T}} & = & S_1 \cup \{\mathsf{T}(A(a) \geq .4), \mathsf{T}(C(a) \geq .4), \mathsf{T}(D(a) \geq .6)\} \\ \tilde{\mathcal{S}}_3^{\mathsf{T}} & = & S_1 \cup \{\mathsf{T}(B(a) \geq .4), \mathsf{T}(D(a) \geq .6)\}. \end{array}$$

■

Example 73 Consider the fuzzy extension of Example 38. Let Σ defined as

$$\Sigma = \{A := C, B := D, ((C \sqcup D)(a) \geq .6)\}.$$

Let $S = \mathsf{T}\Sigma$. It follows that $Ext(S) = \emptyset$. Now, $AcyclicCompl_{DL}(S)$ applies rule (PB) to $((C \sqcup D)(a) \geq .6)$ creating

$$\begin{array}{rcl} S_1 & = & S \cup \{\mathsf{T}(C(a) \geq .6)\}, \\ S_2 & = & S \cup \{\mathsf{NT}(C(a) \geq .6), \mathsf{T}(D(a) \geq .6)\}. \end{array}$$

Now, $Ext(S_1) = \{\mathsf{T}(A(a) \geq .6)\}$ and $Ext(S_2) = \{\mathsf{T}(B(a) \geq .6)\}$, respectively. As a consequence,

$$AcyclicCompl_{DL}(S) = \{\tilde{\mathcal{S}}', \tilde{\mathcal{S}}''\},$$

where

$$\begin{array}{rcl} \tilde{\mathcal{S}}' & = & \{\mathsf{T}(C(a) \geq .6), \mathsf{T}(A(a) \geq .6)\}, \\ \tilde{\mathcal{S}}'' & = & \{\mathsf{NT}(C(a) \geq .6), \mathsf{T}(D(a) \geq .6), \mathsf{T}(B(a) \geq .6)\}. \end{array}$$

■

It is worth noting that the above algorithm requires that we are able to compute $Ext(S)$ and in particular $Maxdeg(\Sigma, A)$. The problem is the topic of the next section.

Algorithm 20 ($Max_{DL}(\Sigma, A)$)
Let Σ be a set of $\mathcal{ALC}$ fuzzy assertions and fuzzy specialisations, let A be an assertion or a specialisation. Set $Min = 0$, $Max = 2$.

1. Pick $n \in N_\Sigma \cup \{.5\}$ such that $Min < n < Max$. If there is no such n, then set $Maxdeg(\Sigma, A) := Min$ and exit.
2. Check if $\Sigma \not\approx_4 (A \geq n)$. If so, then set $Min = n$ and go to Step 1. If not so, then set $Max = n$ and go to Step 1.

■

Table D.23: Algorithm $Max_{DL}(\Sigma, A)$ for $\mathcal{ALC}$.

D.4.1 Determining the maximal degree of truth in fuzzy $\mathcal{ALC}$

Let us now consider the problem of determining $Maxdeg(\Sigma, A)$. We can apply an algorithm similar to $Max_+(\Sigma, A)$], as seen at Page 237 for the case $\mathcal{L}^f_+$, requiring several calls to *fuzzy entailment tests* an reported below (see Table D.23).

As already pointed out in Section D.2.2, the above method for computing $Maxdeg(\Sigma, A)$ requires $O(|\Sigma|)$ fuzzy entailment tests. As $O(|\Sigma|)$ can be very huge, this may be unfeasible.

So, in order to avoid this problem we will proceed as for $\mathcal{L}^f_+$ in Section D.2.2. Roughly speaking our algorithm for computing $Maxdeg(\Sigma, A)$ is similar to Algorithm 16 for the $\mathcal{L}^f_+$ case. Some difficulties arises from the fact that in $\mathcal{ALC}$ some recursive calls are required (see Step 3.a of $AcyclicSat_{DL}$). In order to handle these too, we simply record the "maximal value" returned from each of the recursive calls and combine them with the "maximal value" of the calling level. This will be clearer later on. As the following is mainly a tedious combination of the notions seen until now and those in Section D.2.2, we will only point out the main definitions. Most explanations can be found in Section D.2.2.

We generalise fuzzy assertions and specialisations to the form $(A \geq \lambda)$ and $(A > \lambda)$, where λ is a fuzzy extended value. In the following we will present definitions involving $(A \geq \lambda)$ only, as those involving $(A > \lambda)$ parallels those of $(A \geq \lambda)$. Interpretations are extended to λ and $(A \geq \lambda)$ as for the $\mathcal{L}^f_+$ case. Let σ be a signed fuzzy expression in $\mathcal{ALC}$ and $n \in (0, 1]$. A *conditioned signed fuzzy expression* is, as already seen, an expression of the form $\langle \sigma, v \in [0, n] \rangle$ with meaning as for the $\mathcal{L}^f_+$ case. The *satisfiability* condition of a conditioned signed fuzzy expression is: $\mathcal{I}$ satisfies $\langle \sigma, v \in [0, n] \rangle$ iff if $v^{\mathcal{I}} \in [0, n]$ then $\mathcal{I}$ satisfies $\sigma[v/v^{\mathcal{I}}]$, as for $\mathcal{L}^f_+$.

With respect to *α and β tables in the conditioned case*, these are a straightforward extension of those in Table D.19 (see also Table D.12 for the $\mathcal{L}^f_+$ case). We omit these here.

For the definition of conjugated signed fuzzy propositions, we rely on Table D.18, where n and m are replaced with λ_1 and λ_2, respectively. The definition of *conditioned conjugated* signed fuzzy expressions is as for the $\mathcal{L}^f_+$ case. We will write σ^c for a c-conjugate of σ and $\sigma^{c,max}$ for the the c-conjugate of σ obtained by exchanging T and NT in σ. The definition of $Sol(\cdot)$ and $Cond(\cdot)$ are as for the $\mathcal{L}^f_+$ case.

Concerning the deduction rules, these are a combination of those for $\mathcal{L}^f_+$ (see Table D.13) and those for fuzzy $\mathcal{ALC}$ and are described in Table D.24 below.

$$(A)\ \frac{\langle \alpha, v \in r\rangle}{\langle \alpha_1, v \in r\rangle, \langle \alpha_2, v \in r\rangle} \qquad \text{if } \alpha \text{ is not of type } \alpha^{\exists}$$

$$(B1)\ \frac{\langle \beta, v \in r_1\rangle, \langle \beta_1^c, v \in r_2\rangle}{\langle \beta_2, v \in [0,k]\rangle} \qquad \begin{array}{l}\text{if } \beta \text{ is not of type } C \to A \text{ and}\\ [0,k] = r_1 \cap r_2 \cap Sol(Cond(\beta_1, \beta_1^c))\end{array}$$

$$(B2)\ \frac{\langle \beta, v \in r_1\rangle, \langle \beta_2^c, v \in r_2\rangle}{\langle \beta_1, v \in [0,k]\rangle} \qquad \begin{array}{l}\text{if } \beta \text{ is neither of type } \beta^{\forall}\\ \text{nor of type } A \to C, \text{ and}\\ [0,k] = r_1 \cap r_2 \cap Sol(Cond(\beta_2, \beta_2^c))\end{array}$$

$$(PB)\ \frac{\langle \beta, v \in r\rangle}{\langle \beta_1, v \in r\rangle \mid \langle \beta_1^{c,max}, v \in r\rangle, \langle \beta_2, v \in r\rangle} \qquad \begin{array}{l}\text{if } \beta \text{ is neither of type } \beta^{\forall}\\ \text{nor of type } \beta^{\rightarrow}\end{array}$$

Table D.24: Semantic tableaux inference rules for conditioned singed fuzzy expressions in $\mathcal{ALC}$.

The rules are correct.

Proposition 76 *The rules in Table D.24 are correct.* ⊣

Proof: The proof is as for Proposition 65. Q.E.D.

Example 74 Consider the following signed expressions (see Example 58 as a comparison to $\mathcal{L}_+^f$):

$$\begin{array}{l}\sigma_1 = \mathsf{T}((A \vee B)(a) \geq .9),\\ \sigma_2 = \mathsf{NT}(A(a) \geq .6),\\ \sigma_3 = \mathsf{T}((A \vee B)(a) \geq 1 - v).\end{array}$$

The following instances of rule $(B1)$ can be applied.

$$(B1)\ \frac{\langle \sigma_1, v \in [0,.7]\rangle, \langle \sigma_2, v \in [0,.6]\rangle}{\langle \mathsf{T}(B \geq .9), v \in [0,.6]\rangle}$$

$$(B1)\ \frac{\langle \sigma_3, v \in [0,.7]\rangle, \langle \sigma_2, v \in [0,.6]\rangle}{\langle \mathsf{T}(B \geq 1 - v), v \in [0,.4]\rangle}$$

The last inference is motivated by observing that $Sol(Cond(\sigma_3, \sigma_2)) = [0,.4]$ and that $[0,.7] \cap [0,.6] \cap [0,.4] = [0,.4]$.

■

Concerning closed branches we recall the following definitions from the $\mathcal{L}_+^f$ case: a branch ϕ *is conditioned closed* (*c-closed*) iff S^ϕ contains $\rho_1 = \langle \sigma_1, v \in r_1\rangle$ and $\rho_2 = \langle \sigma_2, v \in r_2\rangle$ such that ρ_1 and ρ_2 are c-conjugated; ϕ *is closed* iff $r_1 \cap r_2 \cap Sol(Cond(\sigma_1, \sigma_2)) = [0,1]$.

Before we go on to present the algorithm for determining $Maxdeg(\Sigma, A)$, we have to show how the $\alpha^{\exists}$ and $\beta^{\forall}$ will be handled in the conditioned case. Essentially, we extend the Equations $(D.43)$–$(D.46)$ according to the following observation. Suppose that α and β are

$$\alpha = \mathsf{T}((\exists R.C)(a) \geq \lambda_1)$$
$$\beta = \mathsf{T}((\forall R.D)(a) \geq \lambda_2)$$

and that S contains

$$\rho_1 = \langle \alpha, v \in r_1 \rangle$$
$$\rho_1 = \langle \beta, v \in r_2 \rangle$$

Consider α's components and β's components, *i.e.*

$$\alpha_1 = \mathsf{T}(R(a, x) \geq \lambda_1)$$
$$\alpha_2 = \mathsf{T}(C(x) \geq \lambda_1)$$
$$\beta_1 = \mathsf{NT}(R(a, x) > 1 - \lambda_2)$$
$$\beta_2 = \mathsf{T}(D(x) \geq \lambda_2)$$

Suppose r_3 is

$$r_3 = r_1 \cap r_2 \cap Sol(Cond(\alpha_1, \beta_1)) = [0, k]$$

that is, r_3 is the condition under which α_1 and β_1 are conjugated and thus rule $(B1)$ can be applied according to the schema

$$(B1) \quad \frac{\langle \alpha_1, v \in r_1 \rangle, \langle \beta, v \in r_2 \rangle}{\langle \beta_2, v \in r_3 \rangle}$$

It follows that if S is satisfiable, then

$$S' = \{\langle \alpha_2, v \in r_1 \rangle, \langle \beta_2, v \in r_3 \rangle\}$$

has to be be satisfiable too. This leads to the following definition of $S(\alpha^{\exists})$ in case we are considering conditioned signed fuzzy formulae. Let S be a set of conditioned signed fuzzy formulae. If there is a conditioned signed fuzzy formula $\alpha^{\exists} \in S$ then let $S(\alpha^{\exists})$ defined as follows (see $(D.43)$–$(D.46)$ as a comparison to the non conditioned case). We will use the compact definition (see Equation $(D.47)$).

$$\begin{array}{lll} S(\langle \alpha^{\exists}, v \in r_1 \rangle) = & \{\langle \alpha_2^{\exists}, v \in r_1 \rangle\} \cup & \\ & \{\langle \beta_2^{\forall}, v \in r_3 \rangle : & \langle \beta^{\forall}, v \in r_2 \rangle \in S, \\ & & Ind(\alpha^{\exists}) = Ind(\beta^{\forall}), \\ & & Role(\alpha^{\exists}) = Role(\beta^{\forall}), \\ & & r_3 = r_1 \cap r_2 \cap Sol(Cond(\alpha_1^{\exists}, \beta_1^{\forall})) = [0, k]\} \end{array} \tag{D.60}$$

where both $\alpha^{\exists}$ and $\beta^{\forall}$ have been instantiated with the same new variable x ($Ind(\alpha_2^{\exists}) = Ind(\beta_2^{\forall}) = x$).

Example 75 Let S be

$$\begin{array}{rl} S = \{ & \langle \mathsf{T}((\exists R_1.C_1)(a) \geq .4), v \in [0,.5]\rangle, \\ & \langle \mathsf{T}((\forall R_1.D_{1_1})(a) \geq v), v \in [0,.6]\rangle, \\ & \langle \mathsf{T}((\forall R_1.D_{1_2})(a) \geq .7), v \in [0,.3]\rangle, \\ & \langle \mathsf{T}((\exists R_2.C_2)(b) \geq 1-v), v \in [0,.8]\rangle, \\ & \langle \mathsf{NT}((\exists R_2.D_{2_1})(b) \geq .6), v \in [0,.7]\rangle\}. \end{array}$$

It follows that

$$\begin{array}{rl} S(\langle \mathsf{T}((\exists R_1.C_1)(a) \geq .4), v \in [0,.5]\rangle) = \{ & \langle \mathsf{T}(C_1(x_1) \geq .4), v \in [0,.5]\rangle, \\ & \langle \mathsf{T}(D_{1_2}(x_1) \geq .7), v \in [0,.3]\rangle\}. \end{array}$$

Notice here that $\langle \mathsf{T}(D_{1_1}(x_1) \geq v), v \in r\rangle$ does not belong to $S(\langle \mathsf{T}((\exists R_1.C_1)(a) \geq .4), v \in [0,.5]\rangle)$, as

$$Sol(Cond(\mathsf{T}(R_1(a,x_1) \geq .4), \mathsf{NT}(R_1(a,x_1) > 1-v))) = (.6,1].$$

Similarly, we obtain that

$$\begin{array}{rl} S(\langle \mathsf{T}((\exists R_2.C_2)(b) \geq 1-v), v \in [0,.8]\rangle) = \{ & \langle \mathsf{T}(C_2(x_2) \geq 1-v), v \in [0,.8]\rangle, \\ & \langle \mathsf{T}(D_{2_1}(x_2) \geq .6), v \in [0,.4]\rangle\}. \end{array}$$

In fact,

$$Sol(Cond(\mathsf{T}(R_2(a,x_2) \geq 1-v), \mathsf{NT}(R_2(a,x_2) \geq .6))) = [0,.4].$$

and

$$[0,.4] = [0,.8] \cap [0,.7] \cap [0,.4].$$

■

In the following, given a branch ϕ and $\alpha^{\exists} \in S$, let $r_{\phi}^{\alpha^{\exists}} \in [0,1]$ be an interval. The interval $r_{\phi}^{\alpha^{\exists}}$ will be setup during the deduction process in such a way that it will be the maximal range $[0,k]$ such that $S(\alpha^{\exists})[v/n]$ is not satisfiable, for all $n \in [0,k]$. Therefore, unlike Equation $(D.33)$, $SOL(\phi)$ –the set of intervals r such that for all $n \in r$, $\phi[v/n]$ is closed– is defined as

$$\begin{array}{l} SOL(\phi) = \\ \{r_1 \cap r_2 \cap Sol(Cond(\sigma_1,\sigma_2)) : \langle \sigma_i, v \in r_i\rangle \in S^{\phi}, \sigma_1, \sigma_2 \text{ c-conjugated}\} \cup \\ \bigcup_{\alpha^{\exists} \in S^{\phi}} \{r_{\phi}^{\alpha^{\exists}}\}. \end{array} \tag{D.61}$$

As for $\mathcal{L}_{+}^{f}$, given the solutions $SOL(\phi)$, we are looking for the maximal one (see Equation $(D.34)$).

$$Max(SOL(\phi)) = \bigcup_{r_i \in SOL(\phi)} r_i. \tag{D.62}$$

We are ready now to describe our procedure for determining $Maxdeg(\Sigma, A)$ for fuzzy $\mathcal{ALC}$. Let T be a deduction tree and consider Equation D.35 for min max.

$$\min\max(T_{\Sigma,A}) = \bigcap_{\phi_i \in T_{\Sigma,A}} Max(SOL(\phi_i)).$$

The algorithm for determining $Maxdeg(\Sigma, A)$ is described in Table D.25 below.

Algorithm 21 (**$MaxVal_{DL}(\Sigma, A)$**)
$MaxVal(\Sigma, A)$ takes as input a well formed fuzzy $\mathcal{ALC}$ KB Σ and an $\mathcal{ALC}$ assertion A. $MaxVal(\Sigma, A)$ returns n iff $Maxdeg(\Sigma, A) = n$. Let $S_{\Sigma,A} = \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq v)\}$. Let $r = MaxAcyclicSat_{DL}(S_{\Sigma,A}) = [0, l]$. Return $MaxVal(\Sigma, A) = l$ and exit.

$MaxAcyclicSat_{DL}(S)$ **:**
Let the root node be labelled with S. The algorithm applies the rules of Table D.24 until each branch in the resulting tree T_S is either closed or completed. At each step of the construction of a deduction tree the following steps are performed:

1. select a branch ϕ which is neither completed nor closed. If there is no such branch go to Step 4.;
2. If ϕ is not yet AB-completed then expand ϕ by means of the rules (A), $(B1)$ and $(B2)$ until it becomes AB-completed. Update ϕ to the resulting branch;
3. If ϕ is neither closed nor completed then
   (a) for all signed fuzzy formula $\alpha^{\exists}$ in S^{ϕ}, let $r_{\phi}^{\alpha^{\exists}} := MaxAcyclicSat_{DL}(S(\alpha^{\exists}) \cup S(\rightarrow))$;
   (b) select a signed fuzzy formula of type β, neither being of type $\beta^{\forall}$ not of type $\beta^{\rightarrow}$, which is not yet fulfilled in the branch;
   (c) apply rule (PB) and go to Step 1.
4. let T_S be the resulting tree and let $r_S = \min\max(T_S)$. Return $MaxAcyclicSat_{DL}(S) = r_S$ and exit.

Table D.25: Algorithm $MaxVal_{DL}(\Sigma, A)$ in fuzzy $\mathcal{ALC}$.

Explanation: Let Σ be a well formed fuzzy $\mathcal{ALC}$ KB and let A be an $\mathcal{ALC}$ assertion. In order to determine $Maxdeg(\Sigma, A)$ we are looking for the maximal value n in place of v such that

$$S_{\Sigma,A} = \mathsf{T}\Sigma \cup \{\mathsf{NT}(A \geq v)\}$$

is not satisfiable. The above subprocedure $MaxAcyclicSat_{DL}(S)$ does this exactly. In fact, $MaxAcyclicSat_{DL}(S)$ tries to find the maximal range $[0, n]$ such that $S[v/n]$ is not satisfiable. $MaxAcyclicSat_{DL}(S)$ is essentially a combination of the algorithms $AcyclicSat_{DL}(S)$ and $MaxVal(\Sigma, A)$ for $\mathcal{L}_{+}^{f}$. The first two steps are quite obvious. Step 3b. in $MaxAcyclic$-$Sat_{DL}(S)$ is a generalisation of Step 3b. in $AcyclicSat_{DL}(S)$. In $AcyclicSat_{DL}(S)$ we are looking for *some not satisfiable* set $S(\alpha^{\exists})$. Consequently, we are looking for the *maximal* range $r_{\phi}^{\alpha^{\exists}} = [0, n]$ for which the set $S(\alpha^{\exists})[v/n]$ is not satisfiable. This motivates the assignment

$$r_{\phi}^{\alpha^{\exists}} := MaxAcyclicSat_{DL}(S(\alpha^{\exists}) \cup S(\rightarrow)).$$

Any such $r_{\phi}^{\alpha^{\exists}} = [0, n]$ is a candidate for being the maximal range such that $\phi[v/n]$ is closed.

Example 76 Let us consider the following KB

$$\Sigma = \{((\exists R.(C \sqcap D))(a) \geq .7), ((\exists R.(C \sqcap E))(a) \geq .8)\},$$

and the assertion $A = (\exists R.C)(a)$. We are looking for $Maxdeg(\Sigma, A)$. Indeed, it is easily verified that $Maxdeg(\Sigma, A) = .8$. We will show that $MaxVal_{DL}(\Sigma, A) = .8$. For simplicity, let[4]

$$\begin{array}{l} \alpha_1^{\exists} = \mathsf{T}((\exists R.(C \sqcap D))(a) \geq .7) \\ \alpha_2^{\exists} = \mathsf{T}((\exists R.(C \sqcap E))(a) \geq .8) \\ \beta^{\forall} = \mathsf{NT}((\exists R.C)(a) \geq v) \end{array}$$

Let $S_{\Sigma,A}$ be

$$S_{\Sigma,A} = \{\alpha_1^{\exists}, \alpha_2^{\exists}, \beta^{\forall}\}$$

From Equation $(D.47)$ it follows that

$$\begin{array}{l} S(\alpha_1^{\exists}) = \{\mathsf{T}((C \sqcap D)(x_1) \geq .7), \langle \mathsf{NT}(C(x_1) \geq v), v \in [0, .7]\rangle\} \\ S(\alpha_2^{\exists}) = \{\mathsf{T}((C \sqcap E)(x_2) \geq .8), \langle \mathsf{NT}(C(x_2) \geq v), v \in [0, .8]\rangle\} \end{array}$$

Consider the function call $MaxAcyclicSat_{DL}(S_{\Sigma,A})$. The deduction tree is shown in Figure D.8.
Since S is AB-completed, Step 3a. will be executed. It immediately follows from the small deduction trees that

$$\begin{array}{l} r_{\phi_1}^{\alpha_1^{\exists}} := MaxAcyclicSat_{DL}(S(\alpha_1^{\exists})) = [0, .7] \\ r_{\phi_2}^{\alpha_2^{\exists}} := MaxAcyclicSat_{DL}(S(\alpha_1^{\exists})) = [0, .8] \end{array}$$

As a consequence, $Sol(\phi)$ is

$$Sol(\phi) = \{r_{\phi_1}^{\alpha_1^{\exists}}, r_{\phi_2}^{\alpha_2^{\exists}}\}$$

from which

$$Max(Sol(\phi)) = r_{\phi_1}^{\alpha_1^{\exists}} \cup r_{\phi_2}^{\alpha_2^{\exists}} = [0, .8]$$

follows. Finally, we have

$$r_S = \min \max(T_S) = [0, .8]$$

and, thus, $MaxVal(\Sigma, A) = .8$. ■

[4]Remember that a signed expression σ is also used as an abbreviation of $\langle \sigma, v \in [0, 1]\rangle$.

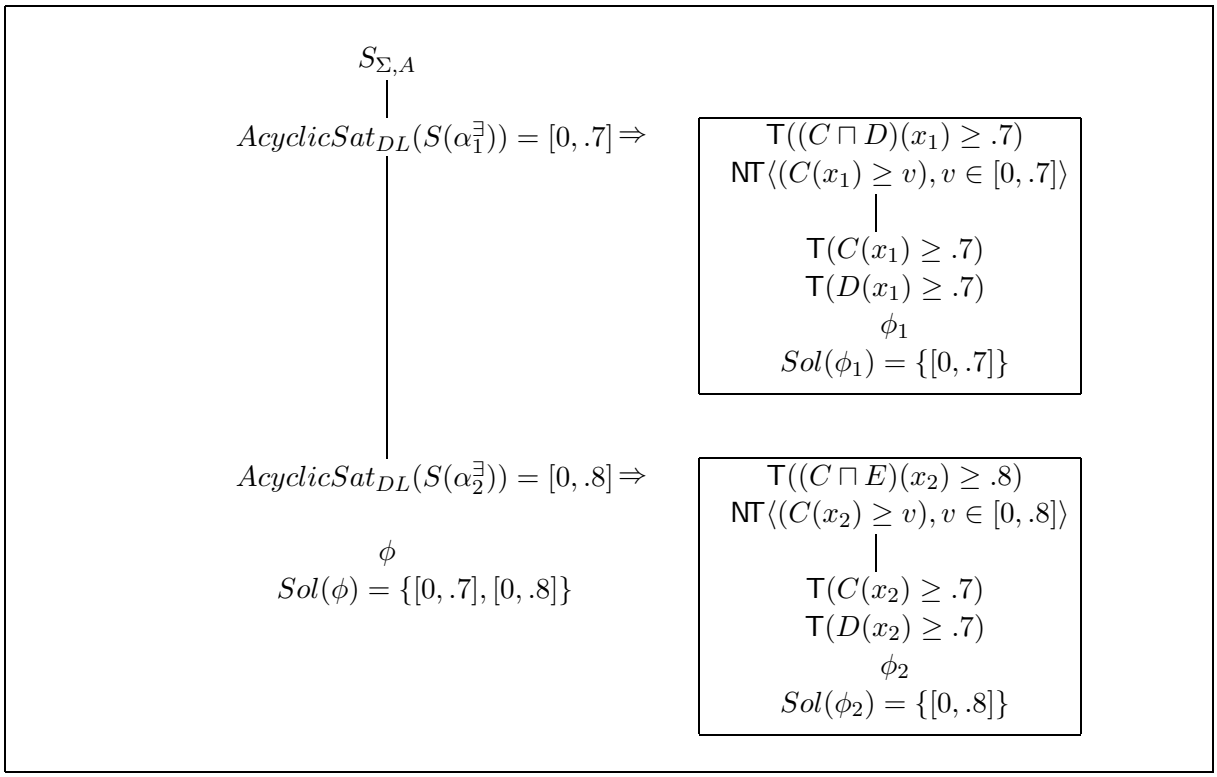


Figure D.8: Example of $MaxVal_{DL}(\Sigma, A)$ execution in fuzzy $\mathcal{ALC}$.

Practical consideration: From Step 3a. in algorithm $MaxVal(\Sigma, A)$ it seems that a recursive call to $MaxAcyclicSat_{DL}(S(\alpha^\exists))$ is necessary for all $\alpha^\exists \in S^\phi$. This is indeed not the case. In fact, let σ be a signed fuzzy formula $(A \geq h)$ or $(A > h)$ and let $\rho = \langle \sigma, v \in [0,k]\rangle$ be a conditioned signed fuzzy formula, then we define $val(\cdot)$ such that:

$$\begin{array}{rcl} val(\sigma) & = & h \\ val(\rho) & = & \min\{h,k\}. \end{array} \tag{D.63}$$

Now, given a branch ϕ let

$$n^{\alpha^\exists} = \max\{val(\rho) : \rho \in S(\alpha^\exists)\}$$

It is easily verified that $MaxAcyclicSat_{DL}(S(\alpha^\exists) \cup S(\rightarrow)) \leq n^{\alpha^\exists}$. Suppose we order the $\alpha^\exists \in S^\phi$ according $n^{\alpha^\exists}$, *i.e.* $\alpha_i^\exists \succeq \alpha_j^\exists$ iff $n^{\alpha_i^\exists} \geq n^{\alpha_j^\exists}$. We proceed as follow: let $Max = 0$ and $S_\exists = \{\alpha_1^\exists, \ldots, \alpha_l^\exists\}$

1. select greatest $\alpha^\exists \in S_\exists$ according $\succeq$, such that $n^{\alpha^\exists} \geq Max$. If there is no such $\alpha^\exists$ then exit;

2. compute $r_\phi^{\alpha^\exists} = MaxAcyclicSat_{DL}(S(\alpha^\exists) \cup S(\rightarrow)) = [0,k]$. If $k > Max$ then set $Max := k$ and let

$$S_\exists := S_\exists \setminus (\{\alpha^\exists\} \cup \{\alpha^\exists \in S_\exists : n^{\alpha^\exists} \leq Max\})$$

and goto Step 1. If $k \leq Max$ then let

$$S_\exists := S_\exists \setminus \{\alpha^\exists\}$$

and goto Step 1.

This reduces the number of recursive calls. For instance, in Example 76, we first compute $r_{\phi_2}^{\alpha_2^\exists} := MaxAcyclicSat_{DL}(S(\alpha_2^\exists) \cup S(\rightarrow))$, as $\alpha_2^\exists \succeq \alpha_1^\exists$ ($n^{\alpha_2^\exists} = .8 \geq .7 = n^{\alpha_1^\exists}$). Since $r_{\phi_2}^{\alpha_2^\exists}$ is $[0, .8]$, we set $Max := .8$. From $Max = .8 \geq .7 = n^{\alpha_1^\exists}$ it follows certainly that $MaxAcyclicSat_{DL}(S(\alpha_1^\exists) \cup S(\rightarrow)) \leq Max$. Therefore, in this case we need not to compute $MaxAcyclicSat_{DL}(S(\alpha_1^\exists) \cup S(\rightarrow))$.

Proposition 77 *Let Σ be a well formed fuzzy $\mathcal{ALC}$ KB, let A be an assertion and $n \geq 0$. Then $Maxdeg(\Sigma, A) = n$ iff $MaxVal_{DL}(\Sigma, A) = n$.* ⊣

Proof: The proof is a straightforward adaption of the proof for Proposition 66. It can easily be shown by induction on the number n of occurrences of conditioned signed fuzzy formulae of type $\alpha^\exists \in S$ that $MaxAcyclicSat_{DL}(S)$ returns the maximal range r such that $S[v/k]$ is not satisfiable, for all $k \in r$. The case $n = 0$ is shown as for Proposition 66, whereas for the case $n > 0$, we first uses induction on the $\alpha^\exists \in S$ yielding some ranges $r^\exists$ for which the actual branch $\phi[v/k]$ becomes closed (for $k \in r$) and from which we can compute $Sol(\phi)$ and, thus, finally $\min\max(T_S)$, yielding the maximal range for which $T_S[v/k]$ becomes closed (for $k \in r$). Q.E.D.

Finally, concerning the computation of $Ext(S)$ (see Equation ($D.59$)) in $AcyclicCompl_{DL}$, it is easily verified that this can be done by invoking subprocedure $MaxAcyclicSat_{DL}(S)$ in algorithm $MaxVal_{DL}$.

Proposition 78 *Let ϕ be a branch computed during the execution of $AcyclicCompl_{DL}(S)$. Then $Ext(S)$ can be computed by relying on subprocedure $MaxAcyclicSat_{DL}(S)$ in algorithm $MaxVal_{DL}$,* i.e.

$$\begin{aligned} Ext(S^\phi) \quad = \quad & \{\mathsf{T}(A(a) \geq n) \colon \mathsf{T}A \colon = C \in S^\phi, a \textit{ occurs in } S^\phi, \\ & n = MaxAcyclicSat_{DL}(S^\phi \cup \{\mathsf{NT}(A(a) \geq v)\})\}. \end{aligned}$$

⊣

D.4.2 Short remarks on computational complexity

From a computational point of view, essentially all results are trivially inherited from crisp $\mathcal{ALC}$. Hence, we have in *the case without specialisations*

Proposition 79 *Let Σ be a set of fuzzy assertions and let A be an assertion. Determining whether $\Sigma \approx_4 (A \geq n)$ is a PSPACE-complete problem for fuzzy $\mathcal{ALC}$.* ⊣

Proof: PSPACE-hardness follows from the following reduction. Let Σ' be a crisp $\mathcal{ALC}$ KB and let A be an assertion. Then $\Sigma' \models_4 A$ iff $\Sigma \approx_4 (A \geq 1)$, where Σ is obtained from Σ' as

$$\Sigma = \{(A \geq 1) : A \in \Sigma'\}$$

(see Proposition 14 and Proposition 15, and observe that in Σ all numbers are 1). PSPACE-completeness holds, as $AcyclicSat_{DL}$ runs in polynomial space as for the crisp case (see Proposition 36). Q.E.D.

In the *case we are considering specialisations*, from [74] it follows immediately that

Proposition 80 *Let Σ be a well formed set of fuzzy assertions and fuzzy specialisations, A be an assertion. Determining whether $\Sigma \models\!\!\approx_4 (A \geq n)$ is a PSPACE-complete problem for fuzzy $\mathcal{ALC}$.* ⊣

Finally, concerning the problem of determining $Maxdeg(\Sigma, A)$ we have.

Proposition 81 *Let Σ be a well formed fuzzy $\mathcal{ALC}$ KB, let A be an assertion and $n \in [0,1]$. Then Then checking $Maxdeg(\Sigma, A) \geq n$ is a PSPACE-complete problem.* ⊣

Proof: Let $n \in [0,1]$. Since $\Sigma \models\!\!\approx_4 (A \geq n)$ iff $Maxdeg(\Sigma, A) \geq n$, and (see Proposition 79) for all n, and deciding $\Sigma \models\!\!\approx_4 (A \geq n)$ is a PSPACE-complete problem, PSPACE-hardness of the $Maxdeg(\Sigma, A) \geq n$ decision problem follows. Furthermore, it is easily verified that $MaxAcyclicSat_{DL}$ runs in polynomial space (by induction on the number n of occurrences of conditioned signed fuzzy formulae of type $\alpha^{\exists} \in S$). Q.E.D.

D.5 Deciding entailment in fuzzy HORN-$\mathcal{ALC}$

The decision procedure determining whether $\Sigma \models\!\!\approx_4 Q$, where Σ is a well formed fuzzy HORN-$\mathcal{ALC}$ KB and Q is a query, is a natural combination of the decision procedure developed for HORN-$\mathcal{L}^f$ in Section D.3 and for the decision procedure developed for HORN-$\mathcal{ALC}$ in Section C.5.4 (case well formed KBs). As we will see later on, Method 2, Method 4.1 and Method 4.2, defined for HORN-$\mathcal{L}^f$ (see Page 259) are applicable in the case of fuzzy HORN-$\mathcal{ALC}$ too.

At first, we define the notions of SLD-derivation and SLD-refutation in fuzzy HORN-$\mathcal{ALC}$. Let G be a goal of the form

$$\begin{array}{rl} \leftarrow & (P_1(\vec{X}_1) \geq V_1), \ldots, (P_n(\vec{X}_n) \geq V_n), \\ & \langle V_{f^1}, f^1(\vec{V}_{f^1}, \vec{Y}_{f^1})\rangle, \ldots, \langle V_{f^k}, f^k(\vec{V}_{f^k}, \vec{Y}_{f^k})\rangle : VR. \end{array}$$

Let E be a horn rule R of the form

$$(P'(\vec{Y}) \geq W) \leftarrow (P'_1(\vec{Y}_1) \geq W_1), \ldots, (P'_m(\vec{Y}_m) \geq W_m), \langle W, f(\vec{W}, \vec{Z})\rangle,$$

or a horn fact $(P(\vec{w}) \geq n)$.

1. If $(P_i(\vec{X}_i) \geq V_i)$ is the selected atom in G and if there is a most general unifier (mgu) θ of $(P'(\vec{Y}) \geq W)$ and $(P_i(\vec{X}_i) \geq V_i)$ (*i.e.* $(P'(\vec{Y}) \geq W)\theta = (P_i(\vec{X}_i) \geq V_i)\theta$), then the *resolvent* of the goal G and the horn rule R using θ is the goal

$$\begin{array}{rl} \leftarrow & ((P_1(\vec{X}_1) \geq V_1), \ldots, (P_{i-1}(\vec{X}_{i-1}) \geq V_{i-1}), \\ & (P'_1(\vec{Y}_1) \geq W_1), \ldots, (P'_m(\vec{Y}_m) \geq W_m), \\ & (P_{i+1}(\vec{X}_{i+1}) \geq V_{i+1}), \ldots, (P_n(\vec{X}_n) \geq V_n), \\ & \langle V_{f^1}, f^1(\vec{V}_{f^1}, \vec{Y}_{f^1})\rangle, \ldots, \langle V_{f^k}, f^k(\vec{V}_{f^k}, \vec{Y}_{f^k})\rangle, \\ & \langle W, f(\vec{W}, \vec{Z})\rangle)\theta : VR\theta \cup \{V_i \leq W\}. \end{array}$$

2. If $(P_i(\vec{X}_i) \geq V_i)$ is the selected atom in G and if there is a mgu θ of $(P(\vec{w}) \geq n)$ and $(P_i(\vec{X}_i) \geq V_i)$ (*i.e.* (*i.e.* $(P(\vec{w}) \geq n) = (P_i(\vec{X}_i) \geq V_i)\theta)$), then the *resolvent* of the goal G and the horn fact E is the goal

$$\begin{array}{rl} \leftarrow & ((P_1(\vec{X}_1) \geq V_1), \ldots, (P_{i-1}(\vec{X}_{i-1}) \geq V_{i-1}), \\ & (P_{i+1}(\vec{X}_{i+1}) \geq V_{i+1}), \ldots, (P_n(\vec{X}_n) \geq V_n), \\ & \langle V_{f^1}, f^1(\vec{V}_{f^1}, \vec{Y}_{f^1})\rangle, \ldots, \langle V_{f^k}, f^k(\vec{V}_{f^k}, \vec{Y}_{f^k})\rangle)\theta : VR\theta \cup \{V_i \leq n\}. \end{array}$$

3. If $\langle V_{f^i}, f^i(\vec{V}_{f^i}, \vec{Y}_{f^i})\rangle$ is the selected atom in G and, if it is of the form $\langle V_{f^i}, n\rangle$, where $n \in [0,1]$, then for $\theta = \{V_{fi}/n\}$ the *resolvent* of the goal G is the goal

$$\begin{array}{rl} \leftarrow & (P_1(\vec{X}_1) \geq V_1), \ldots, (P_n(\vec{X}_n) \geq V_n), \\ & \langle V_{f^1}, f^1(\vec{V}_{f^1}, \vec{Y}_{f^1})\rangle, \ldots, \\ & \langle V_{f^{i-1}}, f^{i-1}(\vec{V}_{f^{i-1}}, \vec{Y}_{f^{i-1}})\rangle, \\ & \langle V_{f^{i+1}}, f^{i+1}(\vec{V}_{f^{i+1}}, \vec{Y}_{f^{i+1}})\rangle, \ldots, \\ & \langle V_{f^k}, f^k(\vec{V}_{f^k}, \vec{Y}_{f^k})\rangle)\theta : VR\theta \cup \{V_{fi} \leq n\}. \end{array}$$

A *SLD-derivation* for a goal G_0 in a fuzzy HORN-$\mathcal{ALC}$ KB Σ is a derivation constituted by:

1. a sequence of horn rules and horn facts $E_1, \ldots, E_n$ in Σ;
2. a sequence of mgu's $\theta_1, \ldots, \theta_n$;
3. a sequence of goals $G_0, \ldots, G_n$ such that for each $i \in \{0, \ldots, n-1\}$, G_{i+1} is the resolvent of G_i and E_{i+1} using θ_{i+1}.

A SLD-derivation may terminate with an empty goal in which case the derivation is a *SLD-refutation*. Let Q be a query $\exists\vec{X}\exists\vec{V}.(P_1(\vec{X}_1) \geq V_1) \wedge \ldots \wedge (P_n(\vec{X}_n) \geq V_n)$.

An answer θ to a query Q w.r.t. a fuzzy HORN-$\mathcal{ALC}$ KB Σ is called a *computed answer* if the goal associated with $Q\theta$ has a SLD-refutation in Σ, *i.e.* if θ is the restriction to the variables $\vec{X}, \vec{V}$ in Q of the composition $\theta_1\theta_2 \ldots \theta_n$, where $\theta_1, \ldots, \theta_n$ are the mgu's used in the SLD-refutation. The *success set* of Q w.r.t. Σ is defined as

$$SuccessSet(\Sigma, Q) \quad = \quad \{\theta : \theta \text{ computed answer of } Q \text{ w.r.t. } \Sigma\}. \tag{D.64}$$

D.5.1 The case of horn fuzzy HORN-$\mathcal{ALC}$ KBs

At first, we will concentrate our attention to those fuzzy HORN-$\mathcal{ALC}$ KBs which are horn and establish correctness and completeness of SLD-refutation.

Let us consider the following example.

Example 77 Let Σ be the following fuzzy HORN-$\mathcal{ALC}$ KB which is "equivalent" to the KB given in Example 24.

$$\begin{array}{rcl} \Sigma & = & \{(A(a) \geq .2), (B(a) \geq .7), \\ & & (A(a) \geq .1), (B(a) \geq .5), \\ & & (A(b) \geq .4), (B(b) \geq .6)\} \end{array}$$

and consider the query Q

$$Q = \exists X \exists V_1, V_2.(A(X) \geq V_1) \wedge (B(X) \geq V_2).$$

and its associated goal G_Q

$$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2), \langle V_0, 1\rangle : \emptyset.$$

We have already seen that the answer set is

$$AnswerSet(\Sigma, Q) \quad = \quad \{\theta : \theta \leq \theta'\} \cup \{\theta : \theta \leq \theta''\}.$$

where

$$\begin{array}{rcl} \theta' & = & \{X/a, V_1/.2, V_2/.7\} \\ \theta'' & = & \{X/b, V_1/.4, V_2/.6\}. \end{array}$$

and thus

$$Maxdeg(\Sigma, Q) =\uparrow AnswerSet(\Sigma, Q) = \{\theta', \theta''\}.$$

Below we show that there are five simple SLD-refutations for goal G_Q in Σ.

(1)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2), \langle V_0, 1\rangle : \emptyset$	Associated goal G_Q
(2)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2) : \{V_0 \leq 1\}$	$\theta_{1_1} = \{V_0/1\}$
(3)	$\leftarrow (B(a) \geq V_2) : \{V_0 \leq 1, V_1 \leq .2\}$	$\theta_{1_2} = \{X/a, V_1/.2\}$
(4)	$\leftarrow \blacksquare : \{V_0 \leq 1, V_1 \leq .2, V_2 \leq .7\}$	$\theta_{1_3} = \{V_2/.7\}$

(1)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2), \langle V_0, 1\rangle : \emptyset$	Associated goal G_Q
(2)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2) : \{V_0 \leq 1\}$	$\theta_{2_1} = \{V_0/1\}$
(3)	$\leftarrow (B(a) \geq V_2) : \{V_0 \leq 1, V_1 \leq .2\}$	$\theta_{2_2} = \{X/a, V_1/.2\}$
(4)	$\leftarrow \blacksquare : \{V_0 \leq 1, V_1 \leq .2, V_2 \leq .5\}$	$\theta_{2_3} = \{V_2/.5\}$

(1)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2), \langle V_0, 1\rangle : \emptyset$	Associated goal G_Q
(2)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2) : \{V_0 \leq 1\}$	$\theta_{3_1} = \{V_0/1\}$
(3)	$\leftarrow (B(a) \geq V_2) : \{V_0 \leq 1, V_1 \leq .1\}$	$\theta_{3_2} = \{X/a, V_1/.1\}$
(4)	$\leftarrow \blacksquare : \{V_0 \leq 1, V_1 \leq .1, V_2 \leq .7\}$	$\theta_{3_3} = \{V_2/.7\}$

(1)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2), \langle V_0, 1\rangle : \emptyset$	Associated goal G_Q
(2)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2) : \{V_0 \leq 1\}$	$\theta_{4_1} = \{V_0/1\}$
(3)	$\leftarrow (B(a) \geq V_2) : \{V_0 \leq 1, V_1 \leq .1\}$	$\theta_{4_2} = \{X/a, V_1/.1\}$
(4)	$\leftarrow \blacksquare : \{V_0 \leq 1, V_1 \leq .1, V_2 \leq .5\}$	$\theta_{4_3} = \{V_2/.5\}$

(1)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2), \langle V_0, 1\rangle : \emptyset$	Associated goal G_Q
(2)	$\leftarrow (A(X) \geq V_1), (B(X) \geq V_2) : \{V_0 \leq 1\}$	$\theta_{5_1} = \{V_0/1\}$
(3)	$\leftarrow (B(b) \geq V_2) : \{V_0 \leq 1, V_1 \leq .4\}$	$\theta_{5_2} = \{X/b, V_1/.4\}$
(4)	$\leftarrow \blacksquare : \{V_0 \leq 1, V_1 \leq .4, V_2 \leq .6\}$	$\theta_{5_3} = \{V_2/.6\}$

Therefore, the success set is

$$SuccessSet(\Sigma, Q) = \{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5\},$$

where

$$\begin{array}{rcl} \theta_1 & = & \{X/a, V_1/.2, V_2/.7\} \\ \theta_2 & = & \{X/a, V_1/.2, V_2/.5\} \\ \theta_3 & = & \{X/a, V_1/.1, V_2/.7\} \\ \theta_4 & = & \{X/a, V_1/.1, V_2/.5\} \\ \theta_5 & = & \{X/b, V_1/.4, V_2/.6\}. \end{array}$$

It is easily verified that each computed answer θ_i is a correct answer, *i.e.* $\Sigma \mid\!\approx_4 Q\theta_i$. Moreover, from $\theta_2 \leq \theta_1$, $\theta_3 \leq \theta_1$ and $\theta_4 \leq \theta_3 \leq \theta_1$ it is easily verified that

$$\uparrow SuccessSet(\Sigma, Q) = \{\theta_1, \theta_5\} = Maxdeg(\Sigma, Q).$$

■

Since HORN-$\mathcal{L}^f$ can be reduced to HORN-$\mathcal{ALC}$, the nondecreasing condition for fuzzy degree function is necessary in order to preserve correctness of the calculus (see Example 65) and some termination problems could arise with recursive horn fuzzy HORN-$\mathcal{ALC}$ KBs (see Example 66).

The following proposition can be shown in a similar way as for the HORN-$\mathcal{L}^f$ case.

Proposition 82 *Let Σ be a horn fuzzy* HORN-$\mathcal{ALC}$ *KB and let Q be a query:*

1. *every computed answer θ of Q w.r.t. Σ is a correct answer (correctness),* i.e.

$$SuccessSet(\Sigma, Q) \subseteq AnswerSet(\Sigma, Q);$$

2. *if Σ_R is not recursive then for every correct answer θ_1 of Q w.r.t. Σ there is a computed answer θ_2 of Q w.r.t. Σ such that $\theta_1 \leq \theta_2$ (completeness).* ⊣

Proof: (Sketch) The proof can be given in a similar way as for Proposition 69. Correctness proof is as for Proposition 69. For the completeness part, just define a *fuzzy Herbrand base* w.r.t. Σ as a set of ground facts $(P(\vec{w}) \geq n)$, where both P and all objects in $\vec{w}$ occur in Σ and proceed similarly as in proof of Proposition 69. Q.E.D.

The following proposition follows immediately.

Proposition 83 *Let Σ be a horn fuzzy* HORN-$\mathcal{ALC}$ *KB and let Q be a query. If Σ_R is not recursive then*

$$Maxdeg(\Sigma, Q) = \uparrow SuccessSet(\Sigma, Q).$$

⊣

Proposition 83 give us a first method in order to compute the maximal degree of truth of a query Q w.r.t. a KB Σ and is based on SLD-refutations in fuzzy HORN-$\mathcal{ALC}$(completeness holds, if Σ_R is not recursive).

As for HORN-$\mathcal{L}^f$, in order to determine whether $\Sigma \approx_4 Q$ we can rely on current standard first-order prolog systems. That is, we can transform Σ and Q into a first-order logic program $\varphi(\Sigma)$ and a first-order query $\varphi(Q)$, respectively, in such a way that $\Sigma \approx_4 Q$ can be decided in terms of $\varphi(\Sigma) \models_2 \varphi(Q)$.

For each n-ary predicate $P(X_1, \ldots, X_n)$ consider a new $n+1$-ary first-order predicate $P(X_1, \ldots, X_n, X_{n+1})$ and let Σ be a horn fuzzy HORN-$\mathcal{ALC}$ KB: φ is the following function.

1. Let R be a horn rule of the form

$$(P(\vec{X}) \geq V) \leftarrow (P_1(\vec{X}_1) \geq V_1), \ldots, (P_n(\vec{X}_n) \geq V_n), \langle V, f(\vec{V}, \vec{Y}) \rangle,$$

   Then

$$\varphi(R) \quad = \quad P(\vec{X}, V) \leftarrow P_1(\vec{X}_1, V_1), \ldots, P_n(\vec{X}_n, V_n), V = f(\vec{V}, \vec{Y}).$$

2. Let γ be a horn fact, *i.e.* an expression of the form $(P(\vec{w}) \geq n)$. Then

$$\varphi(\gamma) \quad = \quad P(\vec{w}, n).$$

3. Let Q be a query of the form

$$\exists \vec{X} \exists \vec{V}.(P_1(\vec{X}_1) \geq V_1) \wedge \ldots \wedge (P_n(\vec{X}_n) \geq V_n).$$

   Then

$$\varphi(Q) \quad = \quad \exists \vec{X} \exists \vec{V}.P_1(\vec{X}_1, V_1) \wedge \ldots \wedge P_n(\vec{X}_n, V_n).$$

4. Let Σ be a horn fuzzy HORN-$\mathcal{ALC}$ KB. Then

$$\begin{array}{lcl} \varphi(\Sigma_F) & = & \{\varphi(\gamma) \colon \gamma \in \Sigma_F\} \\ \varphi(\Sigma_R) & = & \{\varphi(R) \colon R \in \Sigma_R\} \\ \varphi(\Sigma) & = & \varphi(\Sigma_R) \cup \varphi(\Sigma_F). \end{array}$$

As for HORN-$\mathcal{L}^f$, φ transforms horn fuzzy HORN-$\mathcal{ALC}$ KBs and queries into First-Order logic programs and a logic program queries, respectively. Of course, we have to restrict the fuzzy degree function to those cases for which $V = f(\vec{V}, \vec{Y})$ can be expressed in logic programming.

Example 78 Consider Σ and Q in Example 77. By definition, $\varphi(\Sigma)$ and $\varphi(Q)$ are

$$\begin{array}{lcl} \varphi(\Sigma) & = & \{A(a, .2), B(a, .7), \\ & & A(a, .1), B(a, .5), \\ & & A(b, .4), B(b, .6)\} \end{array}$$

and

$$\varphi(Q) = \exists X \exists V_1, V_2 . A(X, V_1) \wedge B(X, V_2).$$

The associated goal of $\varphi(Q)$ is

$$G_{\varphi(Q)} = \leftarrow A(X, V_1), B(X, V_2).$$

As in in Example 77, there are five SLD-refutations for $G_{\varphi(Q)}$ in $\varphi(\Sigma)$. It is easily verified that the set of computed answers is as in Example 77

$$SuccessSet(\varphi(\Sigma), \varphi(Q)) = \{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5\},$$

where

$$\begin{array}{rcl} \theta_1 & = & \{X/a, V_1/.2, V_2/.7\} \\ \theta_2 & = & \{X/a, V_1/.2, V_2/.5\} \\ \theta_3 & = & \{X/a, V_1/.1, V_2/.7\} \\ \theta_4 & = & \{X/a, V_1/.1, V_2/.5\} \\ \theta_5 & = & \{X/b, V_1/.4, V_2/.6\}. \end{array}$$

and thus

$$\uparrow SuccessSet(\varphi(\Sigma), \varphi(Q)) = \{\theta_1, \theta_5\} = Maxdeg(\Sigma, Q).$$

■

The following proposition shows that each computed answer θ w.r.t. SLD-derivation in horn fuzzy HORN-$\mathcal{ALC}$ is also a computed answer w.r.t. to the transformation φ and vice-versa.

Proposition 84 *Let Σ be a horn fuzzy* HORN-$\mathcal{ALC}$ *KB, let Q be a query. Then*

$$SuccessSet(\Sigma, Q) = SuccessSet(\varphi(\Sigma), \varphi(Q)).$$

⊣

Proof: (Sketch). The proof is given by induction on the length n of SLD-refutations. It is easily verified that there is a bijection between SLD-refutations for $\leftarrow G_Q$ w.r.t. Σ and SLD-refutations for $\leftarrow G_{\varphi(Q)}$ w.r.t. $\varphi(\Sigma)$. Q.E.D.

As for horn HORN-$\mathcal{L}^f$, the above proposition has an enormous impact from an implementation point of view. It mainly says that rather than building a new engine, we can rely on existing ones: namely on standard prolog systems. They allow us also to compute $Maxdeg(\Sigma, Q)$ according to Proposition 83 and Proposition 84.

D.5.2 The case of generic fuzzy HORN-$\mathcal{ALC}$ KBs

Finally, Suppose that Σ is a generic fuzzy HORN-$\mathcal{ALC}$ KB and Q is a query. How can we determine $Maxdeg(\Sigma, Q)$? It is not difficult to see that we are able to device a decision procedure for determining $\Sigma \mid\!\approx_4 Q$, by relying on a combination of decision procedure for HORN-$\mathcal{L}^f$ (see Section D.3.2) and for HORN-$\mathcal{ALC}$ (see Section C.5.4).

By combining Proposition 72 and Proposition 47 we get

Proposition 85 *Let Σ be a fuzzy* HORN-$\mathcal{ALC}$ *KB and let Q be a query. Then $\Sigma \mid\!\approx_4 Q$ if either*

1. *there is a SLD-refutation for goal G_Q in Σ; or*
2. *there are $n \geq 1$ SLD-derivations for goal G_Q in Σ ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, such that for all canonical models $\mathcal{I}$ of completions $\tilde{\mathcal{S}} \in AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$, $\mathcal{I}$ does satisfy some Q_i, for $1 \leq i \leq n$.*

If Σ_R is not recursive then the only if direction (completeness) holds too. ⊣

Combining Proposition 85 with Proposition 84, the following methods determine whether $\Sigma \mid\!\approx_4 Q$ (completeness holds, if Σ_R is not recursive).

Method 2: Collect $n \geq 1$ SLD-derivations of G_Q in Σ, ending with goals $G_{Q_1}, \ldots, G_{Q_n}$, until (i) there is an empty goal; or, (ii) for all canonical models $\mathcal{I}$ of completions $\tilde{\mathcal{S}} \in AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$, $\mathcal{I}$ does satisfy some Q_i, for $1 \leq i \leq n$. In the worst case we have to compute all SLD-derivations.

Method 4.1: Compute $AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$. Determine whether for all $\tilde{\mathcal{S}} \in AcyclicCompl_{DL}(\ \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$, G_Q has a SLD-refutation in $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$, *i.e.* whether $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R \mid\!\approx_4 Q$. Note that $\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R$ is a horn HORN-$\mathcal{L}^f$ KB.

Method 4.2: Compute $AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$. Determine whether for all $\tilde{\mathcal{S}} \in AcyclicCompl_{DL}(\ \mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F)$, $G_{\varphi}(Q)$ has a SLD-refutation in $\varphi(\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R)$, *i.e.* whether $\varphi(\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R) \models_2 \varphi(Q)$. Note that $\varphi(\Sigma^+_{\tilde{\mathcal{S}}} \cup \Sigma_R)$ is a first-order horn KB.

Finally, the following proposition is the fuzzy HORN-$\mathcal{ALC}$ analogue of Proposition 73 and shows us how to determine $Maxdeg(\Sigma, Q)$.

Proposition 86 *Let Σ be a fuzzy* HORN-$\mathcal{ALC}$ *KB and let Q be a query. Suppose that Σ_R is not recursive and*

$$AcyclicCompl_{DL}(\mathsf{T}\Sigma_T \cup \mathsf{T}\Sigma_F) = \{\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_k\}.$$

Then

$$\begin{aligned} Maxdeg(\Sigma, Q) &= \downarrow_{1 \leq i \leq k} \{Maxdeg(\Sigma^+_{\tilde{\mathcal{S}}_i} \cup \Sigma_R, Q)\} \\ &= \downarrow_{1 \leq i \leq k} \{Maxdeg(\varphi(\Sigma^+_{\tilde{\mathcal{S}}_i} \cup \Sigma_R), \varphi(Q))\} \end{aligned}$$

■

Notice that from an implementation point of view, certainly the last point is the easiest to implement (corresponds to Method 4.2).

Example 79 Consider the following KB Σ.

$$\begin{array}{rcl}\Sigma & = & \{(H \geq V) \leftarrow (D \geq V_1), \langle V, V_1\rangle,\\ & & (H \geq V) \leftarrow (C \geq V_2), \langle V, V_2\rangle,\\ & & \\ & & A{:=}\, C \sqcap E,\\ & & B{:=}\, D \sqcap F,\\ & & G{:=}\, A \sqcup B,\\ & & \\ & & ((C \sqcup D)(a) \geq .6), (E(a) \geq .4), (F(a) \geq .7), (E(b) \geq .8), (C(b) \geq .9)\}.\end{array}$$

Notice that $\Sigma_T \cup \Sigma_F$ is a superset of the KB Example 72. Now, let S_1 be

$$S_1 = \{\mathsf{T}(E(a) \geq .4), \mathsf{T}(F(a) \geq .7), \mathsf{T}(E(b) \geq .8), \mathsf{T}(C(b) \geq .9), \mathsf{T}(G(a) \geq .4), \mathsf{T}(A(b) \geq .8), \mathsf{T}(G(b) \geq .8)\}.$$

From the fact that

$$Ext(\mathsf{T}\Sigma) = \{\mathsf{T}(G(a) \geq .4), \mathsf{T}(A(b) \geq .8), \mathsf{T}(G(b) \geq .8)\},$$

it can easily be verified (see Example 72) that $AcyclicCompl_{DL}(\Sigma_T \cup \Sigma_F) = \{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2, \tilde{\mathcal{S}}_3\}$, where

$$\begin{array}{rcl}\tilde{\mathcal{S}}_1^{\mathsf{T}} & = & S_1 \cup \{\mathsf{T}(A(a) \geq .4), \mathsf{T}(C(a) \geq .6)\}\\ \tilde{\mathcal{S}}_2^{\mathsf{T}} & = & S_1 \cup \{\mathsf{T}(A(a) \geq .4), \mathsf{T}(C(a) \geq .6), \mathsf{T}(D(a) \geq .6)\}\\ \tilde{\mathcal{S}}_3^{\mathsf{T}} & = & S_1 \cup \{\mathsf{T}(B(a) \geq .4), \mathsf{T}(D(a) \geq .6)\}.\end{array}$$

Let Q be the query

$$Q = \exists X \exists V_1, V_2.(G(X) \geq V_1) \wedge (H(X) \geq V_2).$$

It follows that

$$\begin{array}{rcl}SuccessSet(\Sigma^+_{\tilde{\mathcal{S}}_1} \cup \Sigma_R, Q) & = & \{\theta_1, \theta_2\}\\ SuccessSet(\Sigma^+_{\tilde{\mathcal{S}}_2} \cup \Sigma_R, Q) & = & \{\theta_1, \theta_2\}\\ SuccessSet(\Sigma^+_{\tilde{\mathcal{S}}_3} \cup \Sigma_R, Q) & = & \{\theta_1, \theta_2\}\end{array}$$

where

$$\begin{array}{rcl}\theta_1 & = & \{X/a, V_1/.4, V_2/.6\}\\ \theta_2 & = & \{X/b, V_1/.8, V_2/.9\}\end{array}$$

It follows that

$$\begin{array}{rcl}Maxdeg(\Sigma^+_{\tilde{\mathcal{S}}_1} \cup \Sigma_R, Q) & = & \{\theta_1, \theta_2\}\\ Maxdeg(\Sigma^+_{\tilde{\mathcal{S}}_2} \cup \Sigma_R, Q) & = & \{\theta_1, \theta_2\}\\ Maxdeg(\Sigma^+_{\tilde{\mathcal{S}}_4} \cup \Sigma_R, Q) & = & \{\theta_1, \theta_2\}\end{array}$$

and thus

$$Maxdeg(\Sigma, Q) = \{\theta_1, \theta_2\}.$$

■

Example 80 Consider the fuzzy extension of Example 73 and Example 45. Let Σ defined as

$$\begin{array}{rcl}\Sigma & = & \{E(X) \leftarrow A(X),\\ & & E(X) \leftarrow B(X),\\ & & \\ & & A{:=}\, C,\\ & & B{:=}\, D,\\ & & \\ & & ((C \sqcup D)(a) \geq .6)\}.\end{array}$$

and consider the query

$$Q = \exists X \exists V.(E(X) \geq V).$$

Let us verify that $\Sigma \not\approx_4 Q$.

From Example 73, we already know that $AcyclicCompl_{DL}(S) = \{\tilde{\mathcal{S}}', \tilde{\mathcal{S}}''\}$, where

$$\begin{array}{rcl}\tilde{\mathcal{S}}' & = & \{\mathsf{T}(C(a) \geq .6), \mathsf{T}(A(a) \geq .6)\},\\ \tilde{\mathcal{S}}'' & = & \{\mathsf{NT}(C(a) \geq .6), \mathsf{T}(D(a) \geq .6), \mathsf{T}(B(a) \geq .6)\}.\end{array}$$

It follows that

$$\begin{array}{rcl}\Sigma^+_{\tilde{\mathcal{S}}'} & = & \{(C(a) \geq .6), (A(a) \geq .6)\},\\ \Sigma^+_{\tilde{\mathcal{S}}''} & = & \{(D(a) \geq .6), (B(a) \geq .6)\}.\end{array}$$

Now, it is easily verified that G_Q has a SLD-refutation in $\Sigma^+_{\tilde{\mathcal{S}}'} \cup \Sigma_R$ and in $\Sigma^+_{\tilde{\mathcal{S}}''} \cup \Sigma_R$, confirming $\Sigma \models_4 Q$. In particular we have that

$$\begin{array}{rcl}Maxdeg(\Sigma, Q) & = & \{\theta\}, \text{ where}\\ \theta & = & \{X/a, V/.6\}.\end{array}$$

■

Part VI

Bibliography

Bibliography

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison Wesley Publ. Co., Reading, Massachussetts, 1995.

[2] Wasfi Al-Khatib, Y. Francis Day, Arif Ghafoor, and P. Bruce Berra. Semantic modeling and knowledge representation in multimedia databases. *IEEE Transactions on Knowledge and Data Engineering*, 11(1):64–80, 1999.

[3] Y. Alp Aslandogan, C. Thier, C.T. Yu, J. Zou, and N. Rishe. Using semantic contents and WordNet in image retrieval. In *Proceedings of the 20th International Conference on Research and Development in Information Retrieval (SIGIR-97)*, pages 286–295, Philadelphia,PA, July 1997.

[4] P. Alshuth, Th. Hermes, Ch. Klauck, J. Kryß, and M. Röper. Iris: A system for image and video retrieval. In *Proceedings of the CASCON ConferencE (CASCON-96)*, Toronto, Canada, 1996.

[5] Gianni Amati, Fabio Crestani, and Flavio Ubaldini. A learning system for selective dissemination of information. In *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI-97)*, pages 764–769, Nagoya, Japan, 1997.

[6] Giuseppe Amato, Gianni Mainetto, and Pasquale Savino. An approach to content-based retrieval of multimedia data. Technical Report B4-36-12-96, Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy, 1996.

[7] Giuseppe Amato, Gianni Mainetto, and Pasquale Savino. A model for content-based retrieval of multimedia data. *Multimedia Tools and Applications*, 7:9–36, 1998.

[8] Alan R. Anderson and Nuel D. Belnap. *Entailment - the logic of relevance and necessity*. Princeton University Press, Princeton, NJ, 1975.

[9] T.M. Anwar, H.W. Beck, and S.B. Navathe. Knowledge mining by imprecise querying: A classification-based approach. In *Proceedings of the 8th International Conference on Data Engineering*, pages 622–630, 1992.

[10] Edoardo Ardizzone and Mohand-Said Hacid. A Semantic Modeling Approach for Video Retrieval by Content. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems, Florence, Italy*, June 1999.

[11] Alessandro Artale and Enrico Franconi. A computational account for a description logic of time and action. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Proc. of the*

4th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-94), pages 3–14, Bonn, 1994. Morgan Kaufmann, Los Altos.

[12] Alessandro Artale and Enrico Franconi. Hierarchical plans in a description logic of time and action. In Borgida A., Lenzerini M., Nardi D., and Nebel B., editors, *International Workshop on Description Logics*, pages 1–5, Rome, Italy, 1995.

[13] Alessandro Artale and Enrico Franconi. A temporal description logic for reasoning about actions and plans. *Journal of Artificial Intelligence Research*, 9:463–506, 1998.

[14] M.P. Atkinson, F. Bancilhon, D. DeWitt, K. Dittrich, D. Maier, and S. Zdonik. The object-oriented database system manifesto. In *Proc. of the 1st Int. Conf. on Deductive and Object-Oriented Databases (DOOD-89)*, pages 40–47, Japan, 1989.

[15] Giuseppe Attardi. An analysis of taxonomic reasoning. In Maurizio Lenzerini, Daniele Nardi, and Maria Simi, editors, *Inheritance hierarchies in knowledge representation and programming languages*, pages 29–49. Wiley, Chichester, GB, 1991.

[16] F. Baader. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91)*, Sydney, Australia, 1991.

[17] F. Baader. Computing a minimal representation of the subsumption lattice of all conjunctions of concepts defined in a terminology. In *Proceedings of the International Symposium on Knowledge Retrieval, Use, and Storage for Efficiency, KRUSE 95*, Lecture Notes in Artificial Intelligence, Santa Cruz, USA, 1995. Springer Verlag.

[18] F. Baader, E. Franconi, B. Hollunder, B. Nebel, and H.J. Profitlich. An empirical analysis of optimization techniques for terminological representation systems or: Making KRIS get a move on. *Applied Artificial Intelligence. Special Issue on Knowledge Base Management*, 4:109–132, 1994.

[19] F. Baader and P. Hanschke. Extensions of concept languages for a mechanical engineering application. In *Proceedings of the 16th German AI-Conference, GWAI-92*, volume 671 of *Lecture Notes in Computer Science*, pages 132–143, Bonn (Germany), 1993. Springer–Verlag.

[20] F. Baader and B. Hollunder. KRIS: Knowledge representation and inference system, system description. *ACM SIGART Bulletin*, 2:8–14, 1991.

[21] F. Baader and U. Sattler. Description logics with aggregates and concrete domains. In *Proceedings of the International Workshop on Description Logics*, Gif sur Yvette, France, 1997. Université Paris-Sud. An extended version has appeared as Technical Report LTCS-97-01.

[22] Franz Baader. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. Technical Report RR-90-13, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Kaiserslautern, Germany, 1990. An abridged version appeared in *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence IJCAI-91*, pp. 446–451.

[23] Franz Baader. A formal definition for the expressive power of knowledge representation languages. In *Proceedings of ECAI-90, 9th European Conference on Artificial Intelligence*, pages 53–58, Stockholm, Sweden, 1990.

[24] Franz Baader. A formal definition for the expressive power of knowledge representation languages. Research Report RR-90-05, Deutsches Forschungszentrum für Künstliche Intelligenz, Kaiserslautern, FRG, 1990.

[25] Franz Baader. Terminological cycles in KL-ONE-based knowledge representation languages. In *Proceedings of AAAI-90, 8th Conference of the American Association for Artificial Intelligence*, pages 621–626, Boston, MA, 1990.

[26] Franz Baader. Terminological cycles in KL-ONE-based knowledge representation languages. Research Report RR-90-01, Deutsches Forschungszentrum für Künstliche Intelligenz, Kaiserslautern, FRG, 1990.

[27] Franz Baader, Martin Buchheit, and Bernhard Hollunder. Cardinality restrictions on concepts. *Artificial Intelligence Journal*, 88:195–213, 1996.

[28] Franz Baader, H.-J. Bürckert, Bernhard Hollunder, Werner Nutt, and J.H. Siekman. Concept logics. Research Report RR-90-10, Deutsches Forschungszentrum für Künstliche Intelligenz, Kaiserslautern, FRG, 1990.

[29] Franz Baader, Hans-Jürgen Bürckert, Bernhard Hollunder, Werner Nutt, and Jörg H. Siekmann. Concept logics. In John W. Lloyd, editor, *Computational logic*, pages 177–201. Springer, Heidelberg, FRG, 1990.

[30] Franz Baader, Hans-Jürgen Bürkert, Jochen Heinsohn, Bernhard Hollunder, Jürgen Müller, Bernard Nebel, Werner Nutt, and Hans-Jürge Profitlich. Terminological knowledge representation: A proposal for a terminological logic. Technical Report TM-90-04, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Kaiserslautern, Germany, 1991.

[31] Franz Baader and Philipp Hanschke. A schema for integrating concrete domains into concept languages. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91)*, pages 452–457, Sydney, 1991.

[32] Franz Baader and Bernhard Hollunder. A terminological knowledge representation system with complete inference algorithm. In *Proc. of the Workshop on Processing Declarative Knowledge, PDK-91*, number 567 in Lecture Notes In Artificial Intelligence, pages 67–86. Springer-Verlag, 1991.

[33] Franz Baader and Bernhard Hollunder. Embedding defaults into terminological knowledge representation formalisms. In *Proc. of the 3rd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-92)*, pages 306–317. Morgan Kaufmann, Los Altos, 1992.

[34] Franz Baader and Bernhard Hollunder. How to prefer more specific defaults in terminological default logic. In *Proc. of the 13th Int. Joint Conf. on Artificial Intelligence (IJCAI-93)*, pages 669–674, Chambery, France, 1993. Morgan Kaufmann, Los Altos.

[35] Franz Baader, Bernhard Hollunder, Bernhard Nebel, and Hans-Jürgen Profitlich. An empirical analysis of optimization techniques for teminological representationsystems. Technical report, DFKI Saarbrücken, Germany, 1992.

[36] Franz Baader, Bernhard Hollunder, Bernhard Nebel, Hans-Jürgen Profitlich, and Enrico Franconi. An empirical analisys of optimization techniques for terminological representation systems. In *Proc. of the 3rd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-92)*, pages 270–281. Morgan Kaufmann, Los Altos, 1992.

[37] Franz Baader and Armin Laux. Terminological logics with modal operators. In Borgida A., Lenzerini M., Nardi D., and Nebel B., editors, *International Workshop on Description Logics*, pages 6–12, Rome, Italy, 1995.

[38] Franz Baader and Armin Laux. Terminological logics with modal operators. In *Proc. of the 14th Int. Joint Conf. on Artificial Intelligence (IJCAI-95)*, pages 808–814, Montreal, Canada, 1995. MK.

[39] Franz Baader, Maurizio Lenzerini, Werner Nutt, and Peter F. Patel-Schneider, editors. *Working Notes of the 1994 Description Logics Workshop*, Bonn, Germany, 1994.

[40] Franz Baader and Paliath Narendran. Unification of concept terms in description logics. In *Proceedings of the International Workshop on Description Logics*, pages 34–38, LRI, Universit PARIS-SUD, Cente d'Orsay, 1997.

[41] Franz Baader and Werner Nutt. Are complete and expressive terminological systems feasible? – position paper –. In *Working Notes of the AAAI Fall Symposium on Issues on Description Logics: Users meet Developers*, pages 1–5, 1992.

[42] Franz Baader and Ulrike Sattler. Description logics with symbolic number restrictions. In *Proc. of the 12th European Conf. on Artificial Intelligence (ECAI-96)*, pages 283–287, Budapest, 1996.

[43] Franz Baader and Ulrike Sattler. Number restrictions on complex roles in description logics: A preliminary report. In *Proc. of the 6th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-96)*, 1996.

[44] Jeffrey R. Bach, Charles Fuller, Amarnath Gupta, Arun Hampapur, Bradley Horowitz, Rich Humphrey, Ramesh Jain, and Chiao-fe Shu. The virage image search engine: An open framework for image management. In *Proceedings of the SPIE Conference on Storage and Retrieval for Still Images and Video Databases IV (SPIE-96)*, pages 76–87, San Jose, CA, February 1996.

[45] N.J. Belkin. Ineffable concepts in information retrieval. In K.SparckJones, editor, *Information retrieval experiment*, pages 44–58. Butterworths, London, UK, 1981.

[46] Richard Bellman and Magnus Giertz. On the analytic formalism of the theory of fuzzy sets. *Information Systems*, 5:149–156, 1973.

[47] Nuel D. Belnap. How a computer should think. In Gilbert Ryle, editor, *Contemporary aspects of philosophy*, pages 30–56. Oriel Press, Stocksfield, GB, 1977.

[48] Nuel D. Belnap. A useful four-valued logic. In Gunnar Epstein and J. Michael Dunn, editors, *Modern uses of multiple-valued logic*, pages 5–37. Reidel, Dordrecht, NL, 1977.

[49] D. Beneventano, S. Bergamaschi, S. Lodi, and C. Sartori. Using subsumption in semantic query optimization. In A. Napoli, editor, *IJCAI Workshop on Object-Based Representation Systems*, 1993.

[50] Krishna Bharat, Tomonari Kamba, and Michael Albers. Personalized, interactive news on the web. *Multimedia Systems*, (6):349–358, 1998.

[51] T. Bollinger and U. Pletat. The LILOG knowledge representation system. *ACM SIGART Bulletin*, 2(3):22–27, 1991.

[52] Abraham Bookstein. Fuzzy request: an approach to weighted boolean searches. *Journal of the American Society for Information Science*, (31):240–247, 1980.

[53] G. Bordogna, P Carrara, and G. Pasi. Query term weights as constraints in fuzzy information retrieval. *Information Processing and Management*, 27(1):15–26, 1991.

[54] Alexander Borgida. Structural subsumption: What is it and why is it important? In *AAAI Fall Symposium:Issues in Description Logics*, pages 14–18, 1992.

[55] Alexander Borgida. Description logics in data management. *Data and Knowledge Engineering*, 7(5):671–682, 1995.

[56] Alexander Borgida. On the relative expressiveness of description logics and predicate logics. *Artificial Intelligence Journal*, 82:353–367, 1996.

[57] Alexander Borgida and Peter F. Patel-Schneider. A semantics and complete algorithm for subsumption in the CLASSIC description logic. *Journal of Artificial Intelligence Research*, 1:277–308, 1994.

[58] R. J. Brachman, D .L. McGuinness, P. F. Patel-Schneider, L. Alperin Resnick, and A. Borgida. Living with CLASSIC: when and how to use a KL-ONE-like language. In John F. Sowa, editor, *Principles of Semantic Networks*, pages 401–456. Morgan Kaufmann, Los Altos, 1991.

[59] Ronald J. Brachman, Alexander Borgida, Deborah L. McGuinness, and Lori Alperin Resnick. The CLASSIC knowledge representation system, or, KL-ONE: the next generation. Preprints of the Workshop on Formal Aspects of Semantic Networks, Two Harbors, Cal., 1989.

[60] Ronald J. Brachman, Alexander Borgida, Deborah L. McGuinness, and Lori A. Resnick. CLASSIC: a structural data model for objects. In *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*, pages 59–67, Portland, OR, 1989.

[61] Ronald J. Brachman and Hector J. Levesque. Competence in knowledge representation. In *Proceedings AAAI-82*, pages 189–192. American Association for Artificial Intelligence, August 1982.

[62] Ronald J. Brachman and Hector J. Levesque. The tractability of subsumption in frame-based description languages. In *Proceedings of AAAI-84, 4th Conference of the American Association for Artificial Intelligence*, pages 34–37, Austin, TX, 1984. [a] An extended version appears as [184].

[63] Ronald J. Brachman and Hector J. Levesque, editors. *Readings in knowledge representation*. Morgan Kaufmann, Los Altos, CA, 1985.

[64] Ronald J. Brachman and James G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, 1985.

[65] Ronald J. Brachman, P. G. Selfridge, L. G. Terveen, B. Altman, Alex Borgida, F. Halpern, T. Kirk, A. Lazar, Deborah L. McGuinness, and Lori Alperin Resnick. Knowledge representation support for data archeology. In Y. Yesha, editor, *Proc. of the Int. Conf. on Information and Knowledge Management (CIKM-92)*, pages 457–464, 1992.

[66] Anne Brink, Sherry Marcus, and V.S. Subrahmanian. Heterogeneous multimedia reasoning. *IEEE Computer*, pages 33–39, September 1995.

[67] Martin Buchheit, Francesco M. Donini, Werner Nutt, and Andrea Schaerf. Refining the structure of terminological systems: Terminology = schema + views. In *Proc. of the 11th Nat. Conf. on Artificial Intelligence (AAAI-94)*, pages 199–204, 1994.

[68] Martin Buchheit, Francesco M. Donini, and Andrea Schaerf. Decidable reasoning in terminological knowledge representation systems. In *Proc. of the 13th Int. Joint Conf. on Artificial Intelligence (IJCAI-93)*, pages 704–709, Chambery, France, 1993. Morgan Kaufmann, Los Altos.

[69] Martin Buchheit, Francesco M. Donini, and Andrea Schaerf. Decidable reasoning in terminological knowledge representation systems. *Journal of Artificial Intelligence Research*, 1:109–138, 1993.

[70] Martin Buchheit, A. Manfred Jeusfeld, Werner Nutt, and Martin Staudt. Subsumption between queries to Object-Oriented databases. Technical Report RR-93-44, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Saarbrücken, Germany, 1993.

[71] Martin Buchheit, A. Manfred Jeusfeld, Werner Nutt, and Martin Staudt. Subsumption between queries to object-oriented databases. *Information Systems*, 19(1):33–54, 1994. Special issue on Extending Database Technology, EDBT'94.

[72] Hans-Jürgen Bürkert. A resolution principle for clauses with constraints. In *Proc. of the 10th Conf. on Automated Deduction (CADE-90)*, number 449 in Lecture Notes In Artificial Intelligence, pages 178–192. Springer-Verlag, 1990.

[73] Hans-Jürgen Bürkert. A resolution principle for constrained logics. *Artificial Intelligence Journal*, 66:235–271, 1994.

[74] Diego Calvanese. Reasoning with inclusion axioms in description logics: Algorithms and complexity. In *Proc. of the 12th European Conf. on Artificial Intelligence (ECAI-96)*, pages 303–307, Budapest, 1996.

[75] Diego Calvanese, Giuseppe De Giagomo, and Maurizio Lenzerini. Structured objects: Modelling and reasoning. In *Proc. of the 4th Int. Conf. on Deductive and Object-Oriented Databases (DOOD-95)*, Lecture Notes in Computer Science, pages 229–246. Springer-Verlag, 1995.

[76] Diego Calvanese, Giuseppe De Gioacomo, and Maurizio Lenzerini. What can knowledge representation do for semi-structured data? In *Proc. of the 15th Nat. Conf. on Artificial Intelligence (AAAI-98)*, pages 205–210, 1998.

[77] W. Alexandre Carnielli, Luis Fariñas del Cerro, and Mamede Lima Marques. Contextual negations adn reasoning with contradictions. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91)*, pages 532–537, Sydney, Australia, 1991.

[78] Jianhua Chen and Sukhamany Kundu. A sound and complete fuzzy logic system using Zadeh's implication operator. In Zbigniew W. Ras and Michalewicz Maciek, editors, *Proc. of the 9th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-96)*, number 1079 in Lecture Notes In Artificial Intelligence, pages 233–242. Springer-Verlag, 1996.

[79] Patrick S. Chen. On inference rules of logic-based information retrieval systems. *Information Processing and Management*, 30(1):43–59, 1994.

[80] Wesley W. Chu, Alfonso F. Cardenas, and Ricky K. Taira. Knowledge-based image retrieval with spatial and temporal constructs. In Zbigniew W. Ras and Andrzej Skowron, editors, *Proc. of the 10th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-97)*, number 1325 in Lecture Notes In Artificial Intelligence, pages 17–34. Springer-Verlag, 1997.

[81] Wesley W. Chu, Alfonso F. Cardenas, and Ricky K. Taira. Knowledge-based image retrieval with spatial and temporal constructs. *IEEE Transactions on Knowledge and Data Engineering*, 10(6):872–888, 1998.

[82] Anthony G. Cohn. Calculi for qualitative spatial reasoning. In *Proceedings of AISMC-3*, Lecture Notes in Computer Science. Springer-Verlag, 1996.

[83] Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew McCallum, Tom Mitchell, Kamal Nigam, and Seán Slattery. Learning to extract symbolic knowledge from the worl wide web. In *Proc. of the 15th Nat. Conf. on Artificial Intelligence (AAAI-98)*, pages 509–516, 1998.

[84] Rita Maria da Silva, Antonio Eduardo C. Pereira, and Marcio Andrade Netto. A system of knowledge representation based on formulae of predicate calculus whose variables are annotated by expressions of a fuzzy terminological logic. In *Proc. of the 5th Int. Conf. on Information Processing and Managment of Uncertainty in Knowledge-Based Systems, (IPMU-94)*, number 945 in Lecture Notes in Computer Science. Springer-Verlag, 1994.

[85] Marcello D'Agostino and Marco Mondadori. The taming of the cut. Classical refutations with analytical cut. *Journal of Logic and Computation*, 4(3):285–319, 1994.

[86] Czelasw Danilowicz. Modellling of user preferences and needs in boolean retrieval systems. *Information Processing and Management*, 30:363–378, 1994.

[87] M. Davis and H. Putnam. A computing procedure for quantificatio theory. *Journal of the ACM*, 7:201–215, 1960.

[88] Giuseppe De Giacomo and Maurizio Lenzerini. Making $\mathcal{CATS}$ out of kittens: description logics with aggregates. In Borgida A., Lenzerini M., Nardi D., and Nebel B., editors, *International Workshop on Description Logics*, pages 85–88, Rome, Italy, 1995.

[89] Cyril Decleir, Mohand-Saïd Hacid, and Jacques Kouloumdjian. A Database Approach for Modeling and Querying Video Data. In Masaru Kitsuregawa, Leszek Maciaszek, and Mike Papazoglou, editors, *Proceedings of the 15th International Conference on Data Engineering, Sydney, Australia*, pages 6–13. IEEE Computer Society, March 1999.

[90] Premkumar Devanbu, Ronald J. Brachman, Peter J. Selfridge, and Bruce W. Ballard. LASSIE: A knowledge-based software information system. *Communications of the ACM*, 34(5):36–49, 1991.

[91] DL. Description Logic Web Home Page: `http://dl.kr.org/dl`, WWW.

[92] Francesco M. Donini, Bernhard Hollunder, Maurizio Lenzerini, A. Marchetti Spaccamela, Daniele Nardi, and Werner Nutt. The complexity of existential quantification in concept languages. *Artificial Intelligence*, 2-3:309–327, 1992.

[93] Francesco M. Donini, Maurizio Lenzerini, and Daniele Nardi. Using terminological reasoning in hybrid systems. *AI Communications—The European Journal for Artificial Intelligence*, 3(3):128–138, 1990.

[94] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Werner Nutt. The complexity of concept languages. In *Proceedings of KR-91, 2nd International Conference on Principles of Knowledge Representation and Reasoning*, pages 151–162, Cambridge, MA, 1991.

[95] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Werner Nutt. Tractable concept languages. In *Proceedings of IJCAI-91, 12th International Joint Conference on Artificial Intelligence*, pages 458–463, Sidney, Australia, 1991.

[96] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, Werner Nutt, and Andrea Schaerf. Adding epistemic operators to concept languages. In *Proc. of the 3rd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-92)*, pages 342–353. Morgan Kaufmann, Los Altos, 1992.

[97] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. A hybrid system integrating datalog and concept languages. In *Proc. of the 2nd Conf. of the Italian Association for Artificial Intelligence (AI*IA-91)*, number 549 in Lecture Notes In Artificial Intelligence. Springer-Verlag, 1991. An extended version appeared also in the Working Notes of the AAAI Fall Symposium "Principles of Hybrid Reasoning".

[98] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. From subsumption to instance checking. Technical Report 15.92, Università degli studi di Roma "La Sapienza". Dipartimento di informatica e sistemistica, Rome, Italy, 1992.

[99] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. Deduction in concept languages: From subsumption to instance checking. *Journal of Logic and Computation*, 4(4):423–452, 1994.

[100] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. Reasoning in taxonomic logics. In Gerhard Brewka, editor, *Foundation of Knowledge Representation*. Cambrige University Press, 1994.

[101] Jon Doyle and Ramesh S. Patil. Two thesis of knowledge representation: Language restrictions, taxonomic classification, and the utility of representation services. *Artificial Intelligence Journal*, 48:261–297, 1991.

[102] Didier Dubois. Operations in a fuzzy-valued logic. *Information and Control*, 43:224–240, 1979.

[103] Didier Dubois, Jérome Lang, and Henri Prade. Theorem porving under uncertainty - a possibilistic theory-based approach. In *Proceedings of IJCAI-87, 10th International Joint Conference on Artificial Intelligence*, pages 984–986, Milano, Italy, 1987.

[104] Didier Dubois, Jérome Lang, and Henri Prade. Towards possibilistic logic programming. In *Proc. of the 8th Int. Conf. on Logic Programming (ICLP-91)*, pages 581–595. The MIT Press, 1991.

[105] Didier Dubois and Henri Prade. *Fuzzy Sets and Systems*. Academic Press, New York, NJ, 1980.

[106] Didier Dubois and Henri Prade. Possibilistic logic. In Dov M. Gabbay and C. J. Hogger, editors, *Handbook of Logic in Artificial Intelligence*, volume 3, pages 439–513. Clarendon Press, Oxford, Dordrecht, NL, 1986.

[107] Didier Dubois and Henri Prade. Weighted minimum and maximum operations in fuzzy set theory. *Information Systems*, 39:205–210, 1986.

[108] Didier Dubois and Henri Prade. Necessity measures and the resolution principle. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(3):474–478, 1987.

[109] Didier Dubois and Henri Prade. Approximate and commonsense reasoning: From theory to practice. In Zbigniew W. Ras and Michalewicz Maciek, editors, *Proc. of the 9th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-96)*, number 1079 in Lecture Notes In Artificial Intelligence, pages 19–33. Springer-Verlag, 1996.

[110] J. Michael Dunn. Intuitive semantics for first-degree entailments and coupled trees. *Philosophical Studies*, 29:149–168, 1976.

[111] J. Michael Dunn. Relevance logic and entailment. In Dov M. Gabbay and Franz Guenthner, editors, *Handbook of Philosophical Logic*, volume 3, pages 117–224. Reidel, Dordrecht, NL, 1986.

[112] M. Olivier Duschka and Y. Alon Levy. Recursive plans for information gathering. In *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI-97)*, pages 778–784, Nagoya, Japan, 1997.

[113] Charles Elkan. The paradoxical success of fuzzy logic. In *Proc. of the 10th Nat. Conf. on Artificial Intelligence (AAAI-93)*, pages 698–703, 1993.

[114] Gonzalo Escalada-Imaz and Felip Manyà. Efficient interpretation of propositional multiple-valued logic programs. In *Proc. of the 5th Int. Conf. on Information Processing and Managment of Uncertainty in Knowledge-Based Systems, (IPMU-94)*, number 945 in Lecture Notes in Computer Science, pages 428–439. Springer-Verlag, 1994.

[115] Ronald Fagin. Combining fuzzy information from multiple systems. In *Proceedings of 15th ACM Symposium on Principles of Database Systems*, pages 216–226, Montreal, 1996.

[116] Ronald Fagin and Joseph Y Halpern. Uncertainty, belief, and probability. In *Proceedings of IJCAI-89, 11th International Joint Conference on Artificial Intelligence*, pages 1104–1105, Detroit, MI, 1989.

[117] Ronald Fagin and Edward L. Wimmers. Incorporating user preferences in multimedia queries. In *Proc. 6th International Conference on Database Theory*, number 1186 in Lecture Notes in Computer Science, Delphi, 1997.

[118] Ronald Fagin and S. Marek Yoëlle. Allowing users to weight search terms in information retrieval. IBM Research Report RJ 10108, jan 1998.

[119] Melvin Fitting. *First-Order Logic and Automated Theorem Proving.* Springer-Verlag, 1990.

[120] Melvin Fitting. bilattices and the semantics of logic programming. *Journal of Logic Programming*, 11:91–116, 1991.

[121] Larry Fitzpatrick and Mei Dent. Automatic feedback using past queries: Social searching? In *Proceedings of the 20th International Conference on Research and Development in Information Retrieval (SIGIR-97)*, pages 306–313, Philadelphia,PA, July 1997.

[122] Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, and Dom Byron. Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23–32, 1995.

[123] Enrico Franconi. Crack. In J. W. Schmidt and A. A. Stogny, editors, *Proceedings of the International Workshop on Description Logics - DL-98*, pages 58–59, 1998.

[124] Norbert Fuhr. Probabilistic datalog - a logic for powerful retrieval methods. In *Proceedings of SIGIR-95, 18th International Conference on Research and Development in Information Retrieval*, pages 282–290, Seattle, WA, 1995.

[125] Norbert Fuhr, Norbert Gövert, and Thomas Rölleke. DOLORES: A system for logic-based retrieval of multimedia objects. In *Proceedings of the 21th International Conference on Research and Development in Information Retrieval (SIGIR-98)*, pages 257–265, Melbourne, Australia, 1998.

[126] Norbert Fuhr and Thomas Rölleke. Hyspirit –a probabilistic inference engine for hypermedia retrieval in large databases. In *Proceedings of the 6th International Conference*

on Extending Database Technology (EDBT),, Lecture Notes in Computer Science, pages 24–38. Springer-Verlag, 1997.

[127] Borko Fuhrt. *Multimedia Systems and Techniques*. Kluwer Academic Publishers, London, GB, 1996.

[128] Brian R. Gaines. Fuzzy and probability uncertainty logics. *Information and Control*, 38:154–169, 1978.

[129] Michael R. Garey and David S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness*. Freeman and Company, New York, NY, 1979.

[130] D. James Gemmell, Harrick M. Vin, Dilip D. Kandlur, P. Venkat Rangan, and Lawrence A. Rowe. Multimedia storage servers: A tutorial. *IEEE Computer*, pages 40–49, May 1995.

[131] Giangiacomo Gerla. Inferences in probability logic. *Artificial Intelligence Journal*, 70:33–52, 1994.

[132] Fausto Giunchiglia and Roberto Sebastiani. Building decision procedures for modal logics from propositional decision procedures - the case study of modal K. In *Proc. of the 13th Conf. on Automated Deduction (CADE-96)*, number 449 in Lecture Notes In Artificial Intelligence. Springer-Verlag, 1996.

[133] Franco Giunchiglia and Roberto Sebastiani. A SAT-based decision procedure for ALC. In *Proc. of the 6th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-96)*, 1996.

[134] C. A. Gobel, C. Haul, and S. Bechhofer. Describing and classifying multimedia using the description logic GRAIL. In *Proceedings of the SPIE Conference on Storage and Retrieval for Still Images and Video Databases IV (SPIE-96)*, pages 132–143, San Jose, CA, February 1996.

[135] A. Goker and T.L. McCluskey. Towards an adaptive information retrieval system. In Zbigniew W. Ras and Maria Zemenkova, editors, *Proc. of the 6th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-91)*, number 542 in Lecture Notes In Artificial Intelligence, pages 349–357. Springer-Verlag, 1991.

[136] Adam J. Grove, Joseph Y. Halpern, and Daphne Koller. Asymptotic conditional probabilities for first-order logic. In *Proc. of the 24th ACM SIGACT Sym. on Theory of Computing (STOC-92)*, pages 294–305, Victoria, Canada, 1992.

[137] E. J. Guglielmo and N. C. Rowe. Natural-language retrieval of images based on descriptive captions. *ACM Transaction on Information Systems*, 14(3):237–267, July 1996.

[138] Amarnath Gupta, Terry Weymouth, and Ramesh Jain. Semantic queries with pictures: The VIMSYS model. In *Proc. of the 17th Int. Conf. on Very Large Data Bases (VLDB-91)*, pages 69–79, Barcelona, 1991.

[139] Susan Haack. *Philosophy of logics*. Cambridge University Press, Cambridge, GB, 1978.

[140] M-S. Hacid and Christophe Rigotti. Representing and reasoning on conceptual queries over image databases. In *Proc. of the 12th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-99)*, Lecture Notes In Artificial Intelligence, pages –, Warsaw, Poland, 1999.

[141] Alan Hájek. Probabilities of conditionals - revisited. *Journal of Philosophical Logic*, 18(4):423–428, 1989.

[142] Joseph Y. Halpern. An analysis of first-order logics of probability. In *Proceedings of IJCAI-89, 11th International Joint Conference on Artificial Intelligence*, pages 1375–1381, Detroit, MI, 1989.

[143] Joseph Y. Halpern. An analysis of first-order logics of probability. *Artificial Intelligence Journal*, 46:311–350, 1990.

[144] Joseph Y. Halpern and Yoram O. Moses. A guide to the modal logic of knowledge and belief. Technical Report 4753 (50521), IBM Research Center, 1985.

[145] Pierre Hansen, Brgitte Jaumard, Guy-Blaise Nguetsé, and Marcus Poggi de Aragão. Models and algorithms for probabilistic and bayesian logic. In *Proc. of the 14th Int. Joint Conf. on Artificial Intelligence (IJCAI-95)*, pages 1862–1868, Montreal, Canada, 1995. MK.

[146] Jochen Heinsohn. Probabilistic description logics. In R. Lopez de Mantara and D. Pool, editors, *Proceedings of the 10th Conference on Uncertainty in Artificila Intelligence*, pages 311–318, 1994.

[147] Th. Hermes, Ch. Lauck, Kreyß, and J. Zhang. Image retrieval for information systems. In *Proceedings of the SPIE Conference on Storage and Retrieval for Still Images and Video Databases IV (SPIE-95)*, San Jose, CA, February 1995.

[148] Bernhard Hollunder. Hybrid inferences in KL-ONE-based knowledge representation systems. In *Proceedings of the 14th German Workshop on Artificial Intelligence*, pages 38–47, Eringerfeld, FRG, 1990.

[149] Bernhard Hollunder. An alternative proof method for possibilistic logic and its application to terminological logics. In *10th Annual Conference on Uncertainty in Artificial Intelligence*, Seattle, Washington, 1994. R. Lopez de Mantaras and D. Pool.

[150] Bernhard Hollunder, Werner Nutt, and Manfred Schmidt-Schauß. Subsumption algorithms for concept description languages. In *Proceedings of ECAI-90, 9th European Conference on Artificial Intelligence*, pages 348–353, Stockholm, Sweden, 1990.

[151] Ian Horrocks. Using an expressive description logic: Fact or fiction? In *Proc. of the 8th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-98)*, 1998.

[152] Ian Horrocks and Peter F. Patel-Schneider. Optimising propositional modal satisfiability for description logic subsumption. In *Proc. of the Int. Conf. on Artificial Intelligence and Symbolic Computation*, number 1476 in Lecture Notes In Artificial Intelligence. Springer-Verlag, 1998.

[153] Yen-Teh Hsia. A possibility-based propositional logic of conditionals. In Zbigniew W. Ras and Maria Zemenkova, editors, *Proc. of the 8th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-94)*, number 869 in Lecture Notes In Artificial Intelligence, pages 551–560. Springer-Verlag, 1994.

[154] Scott B. Huffman and Catherine Baudin. Toward structured retrieval in semi-structured information spaces. In *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI-97)*, pages 751–756, Nagoya, Japan, 1997.

[155] George E. Hughes and M. J. Cresswell. *A Companion to Modal Logic*. Methuen, London, 1984.

[156] Mitsuru Ishizuka and Naoki Kanai. Prolog-ELF: incorporating fuzzy logic. In *Proc. of the 9th Int. Joint Conf. on Artificial Intelligence (IJCAI-85)*, pages 701–703, Los Angeles, CA, 1985.

[157] Manfred Jäger. Probabilistic reasoning in terminological logics. In *Proceedings of KR-94, 5-th International Conference on Principles of Knowledge Representation and Reasoning*, pages 305–316, Bonn, FRG, 1994.

[158] Ramesh Jain. Infoscopes: Multimedia Information Systems. In Borko Furht, editor, *Multimedia Systems and Techniques*, pages 217–253. Kluwer Academic Publishers, 1996.

[159] M. Kamel and B. Hadfield. Fuzzy query processing using clustering techniques. *Information Processing and Management*, 26(2):279–293, 1990.

[160] M. Kifer. A logic for reasoning with inconsitency. In *Proc. of the 4th IEEE Sym. on Logic in Computer Science (STACS-95)*, pages 253–262, 1989.

[161] M. Kifer and Ai Li. On the semantics of rule-based expert systems with uncertainty. In *Proc. of the Int. Conf. on Database Theory (ICDT-88)*, number 326 in Lecture Notes in Computer Science, pages 102–117. Springer-Verlag, 1988.

[162] Michael Kifer, Georg Lausen, and James Wu. Logical foundations of Object-Oriented and frame-based languages. *Journal of the ACM*, 42(4):741–843, 1995.

[163] Michael Kifer and V.S. Subrahmanian. Theory of generalized annotaded logic programming and its applications. *Journal of Logic Programming*, 12:335–367, 1992.

[164] Toshiki Kindo, Hideyuki Yoshida, Tetsuro Morimoto, and Taisuke Watanabe. Adaptive personal information filtering system that organizes personal profiles automatically. In *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI-97)*, pages 716–721, Nagoya, Japan, 1997.

[165] Craig A. Knoblock, Steven Minton, Ambite Jose Luis, Naveen Ashish, Pragnesh Jay Modi, Muslea Ion, Andrew G. Philpot, and Sheila Tejada. Modeling web sources for information integration. In *Proc. of the 15th Nat. Conf. on Artificial Intelligence (AAAI-98)*, pages 211–218, 1998.

[166] Daphne Koller, Alon Levy, and Avi Pfeffer. P-CLASSIC: A tractable probabilistic description logic. In *Proc. of the 14th Nat. Conf. on Artificial Intelligence (AAAI-97)*, pages 390–397, 1997.

[167] Donald H. Kraft and Duncan Buel. Fuzzy sets and generalised boolean retrieval systems. *Int. J. Man-Machine Studies*, 19:45–56, 1983.

[168] R. Kruse, E. Schwecke, and J. Heinsohn. *Uncertainty and Vagueness in Knowledge Based Systems*. Springer-Verlag, Berlin, Germany, 1991.

[169] J. Kryß, M. Röper, P. Alshuth, Th. Hermes, and O. Herzog. Video retrieval by still image analysis with imageminer™. In *Proceedings of the SPIE Conference on Storage and Retrieval for Still Images and Video Databases IV (SPIE-97)*, 1997.

[170] Sukhamay Kundu and Jianhua Chen. Fuzzy logic or Lukasiewicz logic: A clarification. In Zbigniew W. Ras and Maria Zemenkova, editors, *Proc. of the 8th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-94)*, number 869 in Lecture Notes In Artificial Intelligence, pages 56–64. Springer-Verlag, 1994.

[171] Laks Lakshmanan. An epistemic foundation for logic programming with uncertainty. In *Foundations of Software Technology and Theoretical Computer Science*, number 880 in Lecture Notes in Computer Science, pages 89–100. Springer-Verlag, 1994.

[172] Laks V.S. Lakshmanan and Nematollaah Shiri. A parametric approach to deductive databases with uncertainty. In *Logic in Databases*, number 1154 in Lecture Notes in Computer Science, pages 61–81. Springer-Verlag, 1996.

[173] Mounia Lalmas. Dempster-Shafer's theory of evidence applied to structured documents: modelling uncertainty. In *Proceedings of the 20th International Conference on Research and Development in Information Retrieval (SIGIR-97)*, pages 110–118, Philadelphia,PA, July 1997.

[174] Wai Lam, Kon Fan Low, and Chao Yang Ho. Using a bayesian network induction approach for text categorization. In *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI-97)*, pages 744–750, Nagoya, Japan, 1997.

[175] P Lambrix and J. Maleki. Learning composite concepts in description logics: A first step. In *Proceedings of the 9th International Symposium on Methodologies for Intelligent Systems - ISMIS 96, LNAI 1079*, pages 68–77, Zakopane, Poland, 1996.

[176] Patrick Lambrix and Lin Padgham. A knowledge base for structured documents. In *In Proceedings of the 1st Australian Document Computing Symposium*, Melbourne, Australia, 1996.

[177] Leslie Lamport. *LaTeX: document Preparation System*. Addison Wesley Publ. Co., Reading, Massachussetts, 1985.

[178] Jérôme Lang. Semantic evaluation in possibilistic logic. In *Proc. of the 3th Int. Conf. on Information Processing and Managment of Uncertainty in Knowledge-Based Systems, (IPMU-90)*, number 521 in Lecture Notes in Computer Science. Springer-Verlag, 1990.

[179] Richard C. T. Lee. Fuzzy logic and the resolution principle. *Journal of the ACM*, 19(1):109–119, January 1972.

[180] Maurizio Lenzerini and Andrea Schaerf. Concept languages as query languages. In *Proc. of the 8th Nat. Conf. on Artificial Intelligence (AAAI-91)*, pages 471–476, 1991.

[181] Hector J. Levesque. A fundamental tradeoff in knowledge representation and reasoning. In *Proceedings of CSCSI/SCEIO-84, 5th Biennial Conference of the Canadian Society for Computational Studies of Intelligence*, pages 141–152, London, Ontario, 1984. [a] An extended version appears, authored by Levesque, Hector J. and Brachman, Ronald J., in [63], pp. 41–70.

[182] Hector J. Levesque. A logic of implicit and explicit belief. In *Proc. of the 3th Nat. Conf. on Artificial Intelligence (AAAI-84)*, pages 198–202, Austin, TX, 1984.

[183] Hector J. Levesque. Logic and the complexity of reasoning. *Journal of Philosophical Logic*, 17:355–389, 1988.

[184] Hector J. Levesque and Ronald J. Brachman. Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, 3:78–93, 1987.

[185] Alon Y. Levy and Marie-Christine Rousset. CARIN: A representation language combining horn rules and description logics. In Borgida A., Lenzerini M., Nardi D., and Nebel B., editors, *International Workshop on Description Logics*, pages 44–51, Rome, Italy, 1995.

[186] Alon Y. Levy and Marie-Christine Rousset. CARIN: A representation language combining horn rules and description logics. In *Proc. of the 12th European Conf. on Artificial Intelligence (ECAI-96)*, Budapest, Hungary, 1996.

[187] Alon Y. Levy and Marie-Christine Rousset. The limits on combining recursive horn rules with descripion logics. In *Proc. of the 13th Nat. Conf. on Artificial Intelligence (AAAI-96)*, pages 577–584, Portland, Oregon, 1996.

[188] Alon Y. Levy and Marie-Christine Rousset. Combining horn rules and description logics in CARIN. *Artificial Intelligence*, 104:165–209, 1998.

[189] David D. Lewis, Robert E. Schapire, James P. Callan, and Ron Papka. Training algorithms for linear text classifiers. In *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval*, pages 298–306, Zürich, CH, 1996.

[190] Fangzhen Lin. Reasoning in the presence of inconsistency. In *Proc. of the 5th Nat. Conf. on Artificial Intelligence (AAAI-87)*, Seattle, WA, 1987.

[191] Jinxin Lin. A semantics for reasoning consistently in the presence of inconsistency. *Artificial Intelligence Journal*, 86:75–95, 1996.

[192] T.Y. Lin, Qing Liu, and Y.Y. Yao. Logic systems for approximate reasoning: via rough sets and topology. In Zbigniew W. Ras and Maria Zemenkova, editors, *Proc. of the 8th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-94)*, number 869 in Lecture Notes In Artificial Intelligence, pages 65–74. Springer-Verlag, 1994.

[193] Z.Q. Liu and J.P. Sun. Structured image retrieval. *Journal of Visual Languages and Computing,*, 8:333–357, June 1997.

[194] John W. Lloyd. *Foundations of Logic Programming.* Springer, Heidelberg, RG, 1987.

[195] James J. Lu, Neil V. Murray, and Erik Rosenthal. Signed formulas and fuzzy operator logics. In Zbigniew W. Ras and Maria Zemenkova, editors, *Proc. of the 8th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-94)*, number 869 in Lecture Notes In Artificial Intelligence, pages 75–84. Springer-Verlag, 1994.

[196] Robert MacGregor. Inside the LOOM description classifier. *SIGART Bulletin*, 2(3):88–92, 1991.

[197] Thomas Mantay and Ralf Möller. Content-based information retrieval by computation of least common subsumers in probabilistic descritpion logic. In *Proc. International Workshop on Intelligent Information Integration, ECAI'98*, Brighton, UK, 1998.

[198] Sherry Marcus and V. S. Subrahmanian. Foundations of multimedia database systems. *Journal of the ACM*, 3(43):474–523, 1996.

[199] Eugene Margulis. Modelling documents with multiple poisson distributions. *Information Processing and Management*, 29(2):215–227, 1993.

[200] Wolfgang May, Betram Ludäscher, and Georg Lausen. Well-founded semantics for deductive object-oriented database languages. In *Proc. of the 5th Int. Conf. on Deductive and Object-Oriented Databases (DOOD-97)*, number 1341 in Lecture Notes in Computer Science, pages 320–335, 1997.

[201] Carlo Meghini, Fabrizio Sebastiani, Umberto Straccia, and Costantino Thanos. A model of information retrieval based on a terminological logic. In *Proceedings of SIGIR-93, 16th International Conference on Research and Development in Information Retrieval*, pages 298–307, Pittsburgh, PA, 1993.

[202] Carlo Meghini and Umberto Straccia. Extending a description logic to cope with the completeness of multimedia documents. In *Proc. of the 12th European Conf. on Artificial Intelligence (ECAI-96): Workshop on Knowledge Representation for Interactive Multimedia Systems*, pages 42–50, Budapest, Hungary, 1996.

[203] Carlo Meghini and Umberto Straccia. A relevance terminological logic for information retrieval. In *Proceedings of SIGIR-96, 19th International Conference on Research and Development in Information Retrieval*, pages 197–205, Zurich, Switzerland, 1996.

[204] Amihai Motro. Management of uncertainty in database systems. In Won Kim, editor, *Modern Database Systems*, pages 457–476. ACM Press and Addison Wesley, 1995.

[205] Karen L. Myers. Hybrid reasoning using universal attachment. *Artificial Intelligence Journal*, (67):329–375, 1994.

[206] Bernhard Nebel. Computational complexity of terminological reasoning in BACK. *Artificial Intelligence*, 34:371–383, 1988.

[207] Bernhard Nebel. *Reasoning and revision in hybrid representation systems*. Springer, Heidelberg, FRG, 1990.

[208] Bernhard Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43:235–249, 1990.

[209] C.V. Negoita and P. Flondor. On fuzziness in information retrieval. *Int. J. Man-Machine Studies*, 8:711–716, 1976.

[210] E.J. Nelson. On three logical principles in intension. *The Monist*, 43, 1933.

[211] Raymond Ng and V.S. Subrahmanian. Stable model semantics for probabilistic deductive databases. In Zbigniew W. Ras and Maria Zemenkova, editors, *Proc. of the 6th Int. Sym. on Methodologies for Intelligent Systems (ISMIS-91)*, number 542 in Lecture Notes In Artificial Intelligence, pages 163–171. Springer-Verlag, 1991.

[212] Raymond Ng and V.S. Subrahmanian. Probabilistic logic programming. *Information and Computation*, 101(2):150–201, 1993.

[213] Raymond Ng and V.S. Subrahmanian. A semantical framework for supporting subjective and conditional probabilities in deductive databases. *Journal of Automated Reasoning*, 10(3):191–235, 1993.

[214] H. Niemann, V. Fischer, D. Paulus, and J. Fischer. Knowledge based image understanding by iterative optimization. In *20th Annual German Conference on Artificial Intelligence*, number 1137 in Lecture Notes In Artificial Intelligence, pages 287–301. Springer-Verlag, 1996.

[215] Eitetsu Oomoto and Katsumi Tanaka. OVID: Designing and implementation of a video-object database system. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):629–643, 1993.

[216] Peter F. Patel-Schneider. A decidable first-order logic for knowledge representation. In *Proceedings of IJCAI-85, 9th International Joint Conference on Artificial Intelligence*, pages 455–458, Los Angeles, CA, 1985.

[217] Peter F. Patel-Schneider. A four-valued semantics for frame-based description languages. In *Proceedings of AAAI-86, 5th Conference of the American Association for Artificial Intelligence*, pages 344–348, Philadelphia, PA, 1986.

[218] Peter F. Patel-Schneider. *Decidable First-Order Logic for Knowledge Representation.* PhD thesis, University of Toronto, May 1987.

[219] Peter F. Patel-Schneider. Decidable, logic-based knowledge representation. Technical Report 201/87, Department of Computer Science, University of Toronto, Toronto, Ontario, 1987.

[220] Peter F. Patel-Schneider. A hybrid, decidable, logic-based knowledge representation system. *Computational Intelligence*, 3:64–77, 1987.

[221] Peter F. Patel-Schneider. Adding number restrictions to a four-valued terminological logic. In *Proc. of the 6th Nat. Conf. on Artificial Intelligence (AAAI-88)*, pages 485–490, 1988.

[222] Peter F. Patel-Schneider. A four-valued semantics for terminological logics. *Artificial Intelligence*, 38:319–351, 1989.

[223] Peter F. Patel-Schneider. A decidable first-order logic for knowledge representation. *Journal of automated reasoning*, 6:361–388, 1990.

[224] Peter F. Patel-Schneider, D. L. McGuinness, Ronald J. Brachman, Lori Alperin Resnick, and A. Borgida. The CLASSIC knowledge representatin system: Guiding principles and implementation rational. *SIGART Bulletin*, 2(3):108–113, 1991.

[225] Zdzislaw Pawlak. Rough sets. *Iternational journal of Computer and Information Sciences*, 11(5):341–356, 1982.

[226] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, Los Altos, 1988.

[227] C. Peltason. The BACK system – an overview. *SIGART Bulletin*, 2(3):114–119, 1991.

[228] A. Pentland, R.W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. Technical Report 255, M.I.T. Media Laboratory Perceptual Computing, 1993.

[229] A. Petrakis, R.W. Picard, and S. Sclaroff. Similarity searching in large image databases. *IEEE Transactions on Knowledge and Data Engineering*, 1996.

[230] David A. Randell, Cui Zhan, and Anthony G. Cohn. A spatial logic based on regions and connection. In *Proceedings of KR-92, 3rd International Conference on Knowledge Representation and Reasoning*, 1992. Revised version.

[231] R. Reiter. On asking what a database knows. In J. W. Lloyd, editor, *Symposium on Computational Logics*, pages 96–113. Springer-Verlag, ESPRIT Basic Research Action Series, 1990.

[232] Raymond Reiter. On closed world data bases. In Hervé Gallaire and Jack Minker, editors, *Logic and data bases*, pages 55–76. Plenum Press, New York, NY, 1978.

[233] Raymond Reiter. Towards a logical reconstruction of relational database theory. In Michael L. Brodie, John Mylopoulos, and Joachim W. Schmidt, editors, *On conceptual modelling*, pages 191–233. Springer, Heidelberg, FRG, 1984.

[234] R. Bruce Roberts and Ira P. Goldstein. The FRL manual. Technical Report 409, The Artificial Intelligence Laboratory, MIT, Cambridge, MA, 1977.

[235] Thomas Rölleke and Norbert Fuhr. Retrieval of complex objects using a four-valued logic. In *Proceedings of SIGIR-96, 19th International Conference on Research and Development in Information Retrieval*, pages 206–214, Zurich, Switzerland, 1996.

[236] Thomas Rölleke and Norbert Fuhr. Retrieving complex objects with hyspirit. In *Proceedings of the 19th annual BCS-IRSG Colloquium on IR Research*, Aberdeen, Scotland, 1997.

[237] Thomas Rölleke and Norbert Fuhr. Information retrieval with probabilistic datalog. In F. Crestani, M. Lalmas, and C.J. van Rijsbergen, editors, *Logic and Uncertainty in Information Retrieval: Advanced models for the representation and retrieval of information*, volume 4 of *The Kluwer International Series On Information Retrieval*, chapter 9, pages 221–245. Kluwer Academic Publishers, Boston, MA, 1998.

[238] Dan Roth. On the hardness of approximate reasoning. *Artificial Intelligence Journal*, 82:273–302, 1996.

[239] Enrique H. Ruspini. On the semantics of fuzzy logic. Technical Note 475, SRI International, 1989.

[240] Alessandro Saffiotti. A hybrid framework for representing uncertain knowledge. In *Proc. of the 7th Nat. Conf. on Artificial Intelligence (AAAI-90)*, pages 653–658, Boston, MA, 1990.

[241] Alessandro Saffiotti, Enrique H. Ruspini, and Kurt Konolige. Using fuzzy logic for mobile robot control. In D. Dubois, H. Prade, and H.J. Zimmermann, editors, *International Handbook of Fuzzy Sets and Possibility Theory*. Kluwer Academic Publishers, 1997.

[242] Gerard Salton. *Automatic Text Processing: The Tranformation, Analysis, and Retrieval of Information by Computer*. McGraw-Hill, New York, 1983.

[243] Gerard Salton, Edward Fox, and Harry Wu. Extended boolean information retrieval. *Communications of the ACM*, 26(12):1022–1036, 1983.

[244] Gerard Salton and J. Michael McGill. *Introduction to Modern Information Retrieval*. Addison Wesley Publ. Co., Reading, Massachussetts, 1983.

[245] Gerard Salton and A. Wong. A vector space model for automatic indexing. *Communications of the ACM*, 18:613–620, 1975.

[246] T. Saracevic. Relevance: a review of and a framework for thinking on the notion of information science. *Journal of the American Society for Information Science*, (26):321–343, 1975.

[247] Andrea Schaerf. Reasoning with individuals in concept languages. *Data and Knowledge Engineering*, 13(2):141–176, 1994.

[248] Klaus Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91)*, pages 466–471, Sydney, 1991.

[249] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48:1–26, 1991.

[250] Albrecht Schmiedel. A temporal terminological logic. In *Proceedings of AAAI-90, 8th Conference of the American Association for Artificial Intelligence*, pages 640–645, Boston, MA, 1990.

[251] James G. Schmolze. Terminological knowledge representation systems supporting n-ary terms. In *Proceedings of KR-89, 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 432–443, Toronto, Ontario, 1989.

[252] Fabrizio Sebastiani. A probabilistic terminological logic for modelling information retrieval. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 122–130, Dublin, IRL, 1994. Published by Springer Verlag, Heidelberg, FRG.

[253] Fabrizio Sebastiani and Umberto Straccia. A computationally tractable terminological logic. In *Proceedings of SCAI-91, 3rd Scandinavian Conference on Artificial Intelligence*, pages 307–315, Roskilde, Denmark, 1991.

[254] Ehud Y. Shapiro. Logic programs with uncertainties: A tool for implementing rule-based systems. In *Proc. of the 8th Int. Joint Conf. on Artificial Intelligence (IJCAI-83)*, pages 529–532, 1983.

[255] R.K. Shirari. Automatic indexing and content-based retrieval of captioned images. *IEEE Computer*, 28:49–56, Sept. 1995.

[256] Amit Singhal and Mandar Mitra Buckley. Learning routing queries in a query zone. In *Proceedings of the 20th International Conference on Research and Development in Information Retrieval (SIGIR-97)*, pages 25–32, Philadelphia,PA, July 1997.

[257] A. F. Smeaton and I. Quigley. Experiments on using semantic distances between words in image caption retrieval. In *Proceedings of the 19th International Conference on Research and Development in Information Retrieval (SIGIR-96)*, pages 174–180, Zurich, CH, August 1996.

[258] John R. Smith and Shih-Fu Chang. Tools and techniques for color image retrieval. In *Proceedings of the SPIE Conference on Storage and Retrieval for Still Images and Video Databases IV (SPIE-96)*, 1996.

[259] Roger R. Smith, Dorota Kieronska, and Svetha Venkatesh. Media-indipendent knowledge representation via UMART: Unified mental annotation and retrieval tool. In *Proceedings of the SPIE Conference on Storage and Retrieval for Still Images and Video Databases IV (SPIE-96)*, pages 96–107, San Jose, CA, February 1996.

[260] R. M. Smullyan. *First Order Logic*. Springer, Berlin, 1968.

[261] John F. Sowa, editor. *Principles of Semantic Networks*. Morgan Kaufmann, Los Altos, 1991.

[262] Karen Sparck Jones. Search term relevance weighting given little relevance information. *Journal of Documentation*, 35:30–48, 1979.

[263] Viggo Stoltenberg-Hansen, Ingrid Lindström, and Edward R. Griffor. *Mathematical theory of Domains*. Cambridge University Press, Cambridge, UK, 1994.

[264] Umberto Straccia. Default inheritance reasoning in hybrid KL-ONE-style logics. In *Proceedings of IJCAI-93, 13th International Joint Conference on Artificial Intelligence*, pages 676–681, Chambery, France, 1993.

[265] Umberto Straccia. A four-valued fuzzy propositional logic. In *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI-97)*, pages 128–133, Nagoya, Japan, 1997.

[266] Umberto Straccia. A sequent calculus for reasoning in four-valued description logics. In *Proc. of the Int. Conf. on Analytic Tableaux and Related Methods (TABLEAUX-97)*, number 1227 in Lecture Notes In Artificial Intelligence, pages 343–357, Pont-à-Mousson, France, 1997.

[267] Umberto Straccia. A fuzzy description logic. In *Proc. of the 15th Nat. Conf. on Artificial Intelligence (AAAI-98)*, pages 594–599, 1998.

[268] C. Tresp and R. Molitor. A description logic for vague knowledge. In *Proc. of the 13th European Conf. on Artificial Intelligence (ECAI-98)*, Brighton (England), August 1998.

[269] C. Tresp and U. Tüben. Medical terminology processing for a tutoring system. In *International Conference on Computational Intelligence and Multimedia Applications (ICCIMA98)*, Monash Univ. (Australien), Februar 1998.

[270] C. B. Tresp and R. Molitor. A Description Logic for Vague Knowledge. LTCS-Report 98-01, RWTH Aachen, 1998.

[271] J. D. Ullman. *Principles of Database and Knowledge Base Systems*, volume 1,2. Computer Science Press, Potomac, Maryland, 1988.

[272] M.H. van Emden. Quantitative deduction and its fixpoint theory. *Journal of Philosophical Logic*, (1):37–53, 1986.

[273] Cornelis J. van Rijsbergen. A new theoretical framework for information retrieval. In *Proceedings of the 1986 ACM Conference on Research and Development in Information Retrieval*, pages 194–200, Pisa, Italy, 1986.

[274] Cornelis J. van Rijsbergen. A non-classical logic for information retrieval. *The Computer Journal*, 29:481–485, 1986.

[275] Cornelis J. van Rijsbergen. Towards an information logic. In *Proceedings of SIGIR-89, 12th Conference of the ACM Special Interest Group on Information Retrieval*, pages 77–86, Cambridge, MA, 1989.

[276] Bienvenido Vélez, Ron Weiss, Mark Sheldon, and David K. Gifford. Fast and effective query refinement. In *Proceedings of the 20th International Conference on Research and Development in Information Retrieval (SIGIR-97)*, pages 6–15, Philadelphia,PA, July 1997.

[277] Marc Vilain. Deduction as parsing: Tractable classification in the KL-ONE framework. In *Proc. of the 8th Nat. Conf. on Artificial Intelligence (AAAI-91)*, pages 464–470, 1991.

[278] Peter Vojtáš and Leonard Paulík. Soundness and completeness of non-classical extended sld-resolution. In *5th International Workshop on Extensions of Logic Programming (ELP'96)*, number 1050 in Lecture Notes In Artificial Intelligence, pages 289–301, Leipzig, Germany, 1996.

[279] W3C. World Wide Web Consortium Home Page: `http://www.w3.org`.

[280] Gerd Wagner. Ex contradictione nihil sequitur. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91)*, pages 538–543, Sydney, Australia, 1991.

[281] Gerd Wagner. A logical reconstruction of fuzzy inference in databases and logic programs. In *Proc. of the 7th Int. Fuzzy Set Association world Congress (IFSA-97)*, Prague, 1997.

[282] Gerd Wagner. Negation in fuzzy and possibilistic logic programs. In T. Martin and F. Arcelli, editors, *Logic programming and Soft Computing*, pages –. Research Studies Press, 1998.

[283] Robert Weida. Closing the terminology. In Borgida A., Lenzerini M., Nardi D., and Nebel B., editors, *International Workshop on Description Logics*, pages 25–31, Rome, Italy, 1995.

[284] Robert Weida and Diane Litman. Terminological reasoning with constraint networks and an application to plan recognition. In *Proc. of the 3rd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-92)*, pages 282–293. Morgan Kaufmann, Los Altos, 1992.

[285] Cristopher Welty. Description logics for digital libraries. In *International Workshop on Description Logics*, Trento, Italy, 1998.

[286] William A. Woods. Understanding subsumption and taxomony: A framework for progress. In J. F. Sowa, editor, *Principles of Semantic Networks*, pages 45–94. Morgan Kaufmann, Los Altos, 1991.

[287] J.K. Wu and A. Desai Narasimhalu. Fuzzy content-based retrieval in image databases. *Information Processing and Management*, 34(5):513–534, 1998.

[288] J.K. Wu, A. Desai Narasimhalu, B. M. Mehtre, C.P. Lam, and J. Gao, Y. CORE: a content-based retrieval engine for multimedia information systems. *Multimedia Systems*, 1(3):25–31, 1995.

[289] Beat Wüttrich. Probabilistic knowledge bases. *IEEE Transactions on Knowledge and Data Engineering*, 7(5):691–698, 1995.

[290] Cheng Xiachun, Jiang Yunfei, and Liu Xuhua. The rationality and decidability of fuzzy implications. In *Proc. of the 14th Int. Joint Conf. on Artificial Intelligence (IJCAI-95)*, pages 1910–1911, Montreal, Canada, 1995. MK.

[291] R. Yager, Ronald. Inference in multivalued logic system. *Int. J. Man-Machine Studies*, 23:27–44, 1985.

[292] Ronald R. Yager. Fuzzy sets as a tool for modeling. In Jan van Leeuwen, editor, *Computer Science Today*, number 1000 in Lecture Notes in Computer Science, pages 536–548. Springer-Verlag, 1995.

[293] R.R. Yager. Possibilistic qualification and default rules. In B. Bouchan and R.R. Yager, editors, *Uncertainty in Knowledge-Based Systems*, pages 41–57. Springer, Berlin, FRG, 1987.

[294] R.R. Yager. Using approximate reasoning to represent default knowledge. *Artificial Intelligence*, 31:99–112, 1987.

[295] John Yen. Generalizing term subsumption languages to fuzzy logic. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91)*, pages 472–477, Sydney, Australia, 1991.

[296] A. Yoshitaka, S. Kishida, M. Hirakawa, and T. Ichikawa. Knowledge-assisted content-based retrieval for multimedia databases. *IEEE Multimedia*, pages 12–20, 1994.

[297] Atsuo Yoshitaka and Tadao Ichikawa. A survey on content-based retrieval for multimedia databases. *IEEE Transactions on Knowledge and Data Engineering*, 11(1):81–93, 1999.

[298] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[299] L. A. Zadeh. A theory of approximate reasoning. *Machine Intelligence*, 9:149–194, 1979.

[300] A. Zeller. Versioning system models through description logic. In *Proc. 8th International Symposium on System Configuration Management (SCM-8)*, Brussels, 1988.

Part VII

Index

Index